KL28Z Reference Manual

MKL28Z512VDC7, MKL28Z512VLL7

Document Number: MKL28ZRM
Rev. 4, 06/2016

Contents

Section number **Title** **Page**

Chapter 1
About This Document

Chapter 2
Introduction

Chapter 3
Chip Configuration

Chapter 4
Memory Map

Chapter 5
Clock Distribution

Chapter 6
Reset and Boot

Chapter 7
Power Management

Chapter 8
Security

Chapter 9
Debug

Chapter 10
Signal Multiplexing and Signal Descriptions

Chapter 11
Analog-to-Digital Converter (ADC)

Chapter 12
Crossbar Switch Lite (AXBS-Lite)

Chapter 13
Bit Manipulation Engine2 (BME2)

Chapter 14
Kinetis ROM Bootloader

Chapter 15
Cryptographic Acceleration Unit (CAU)

Chapter 16
Comparator (CMP)

Chapter 17
Cyclic Redundancy Check (CRC)

Chapter 18
12-bit Digital-to-Analog Converter (DAC)

Chapter 19
Direct Memory Access Multiplexer (DMAMUX)

Chapter 20

Enhanced Direct Memory Access (eDMA)

Chapter 21
Smart Card Interface Module (EMV SIM)

Chapter 22
Flexible I/O (FlexIO)

Chapter 23
Flash Memory Controller (FMC)

Chapter 24
Flash Memory Module (FTFA)

Chapter 25
Interrupt Multiplexer (INTMUX)

Chapter 26
Low-Leakage Wakeup Unit (LLWU)

Chapter 27
Low Power Inter-Integrated Circuit (LPI2C)

Chapter 28
Low Power Interrupt Timer (LPIT)

Chapter 29
Low Power Serial Peripheral Interface (LPSPI)

Chapter 30
Low-Power Timer (LPTMR)

Chapter 31
Low Power Universal Asynchronous Receiver/Transmitter (LPUART)

Chapter 32
Memory-Mapped Divide and Square Root (MMDVSQ)

Chapter 33
Miscellaneous Control Module (MCM)

Chapter 34
Miscellaneous System Control Module (MSCM)

Chapter 35
Micro Trace Buffer (MTB)

Chapter 36
Peripheral Clock Control (PCC)

Chapter 37
Power Management Controller (PMC)

Chapter 38
Port Control and Interrupts (PORT)

Chapter 41
Real Time Clock (RTC)

Chapter 42
Integrated Interchip Sound (I2S) / Synchronous Audio Interface (SAI)

Chapter 43
System Clock Generator (SCG)

Chapter 44
System Integration Module (SIM)

Chapter 45
System Mode Controller (SMC)

Chapter 46
System Register File

Chapter 47
Timer/PWM Module (TPM)

Chapter 48
Trigger MUX Control (TRGMUX)

Chapter 49
SA-TRNG Standalone

Chapter 50
Touch Sensing Input (TSI)

Chapter 51
Time Stamp Timer Module (TSTMR)

Chapter 52
Universal Serial Bus Full Speed OTG Controller (USBFSOTG)

Chapter 53
USB Voltage Regulator (VREG)

Chapter 54
Voltage Reference (VREF)

Chapter 55
Watchdog timer (WDOG32)

Chapter 1
About This Document

1.1 Overview

1.1.1 Purpose

This document describes the features, architecture, and programming model of the Freescale microcontroller.

1.1.2 Audience

A reference manual is primarily for system architects and software application developers who are using or considering using a Freescale product in a system.

1.2 Conventions

1.2.1 Numbering systems

The following suffixes identify different numbering systems:

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is written 101b. In some cases, binary numbers are shown with the prefix *0b*.
d	Decimal number. Decimal numbers are followed by this suffix only when the possibility of confusion exists. In general, decimal numbers are shown without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is written 3Ch. In some cases, hexadecimal numbers are shown with the prefix *0x*.

1.2.2 Typographic notation

The following typographic notation is used throughout this document:

Example	Description
placeholder, x	Items in italics are placeholders for information that you provide. Italicized text is also used for the titles of publications and for emphasis. Plain lowercase letters are also used as placeholders for single letters and numbers.
`code`	Fixed-width type indicates text that must be typed exactly as shown. It is used for instruction mnemonics, directives, symbols, subcommands, parameters, and operators. Fixed-width type is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.
SR[SCM]	A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets and separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus.

1.2.3 Special terms

The following terms have special meanings:

Term	Meaning
asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
reserved	Refers to a memory space, register, or field that is either reserved for future use or for which, when written to, the module or chip behavior is unpredictable.

Chapter 2
Introduction

2.1 Overview

Information found here provides an overview of the KL28Z MCU part belonging to the Kinetis L series of MCUs based on the ARM® Cortex®-M0+ processor. It also presents high-level descriptions of the modules available on the devices covered by this document.

2.2 KL28Z sub-family introduction

The KL28Z is a sub-family of 32-bit Kinetis microcontrollers based on the ARM® Cortex®-M0+ processor. It is an extension of the existing L series MCU family, with additional memory. The KL28Z sub-family provides up to 96 MHz (HSRUN mode) CPU performance. It supports ultra low power, up to 512 KB Flash with 128 KB SRAM and integrates functions such as USB with crystal-less recovery. The features of the KL28Z sub-family derivatives are:

- Core - One ARM® Cortex®-M0+ core functional operational up to 96 MHz.
- Memory option is up to 512 KB Flash and 128 KB RAM, partitioned to provide optimum performance.
- Wide operating voltage range from 1.71–3.6 V with fully functional flash program/erase/read operations
- Multiple package options including 100 and 121-pin
- Ambient operating temperature ranges from –40 °C to 105 °C.

NOTE

The 121-pin XFBGA package for this product is not yet available. However, it is included in a Package Your Way program for Kinetis MCUs. Visit nxp.com/KPYW for more details.

2.3 Feature Summary

The following table lists the features of this device.

Table 2-1. Feature Summary

Feature	Details
Hardware Characteristics	
Package	100-pin LQFP 121-pin XFBGA (Package Your Way)
Voltage range	1.71 V to 3.6 V
Temperature range (T_A)	-40°C to 105°C
Temperature range (T_J)	-40°C to 125°C
System	
Central processing unit (Core)	ARM Cortex M0+ core
Max. Core frequency	72 MHz
Max. Bus frequency	24 MHz
Nested vectored Interrupt controller (NVIC0)	32 vectored interrupts 4 programmable interrupt priority levels
Interrupt Mux (INTMUX0)	32 input interrupt mux to 4 outputs assign to NVIC0 with OR and AND interrupt functionality
Direct memory access (DMA0)	8 channels
(Memory-Mapped) Divide and Square Root module (DVSQ0)	Yes
Non-maskable interrupt (NMI0)	Yes
Software Watch Dog (WDOG0)	Yes
Low-leakage Wakeup Unit (LLWU)	external wake-up pins with digital glitch filter and internal wake-up sources (see LLWU section for details)
Debug and Trace	2-pin serial wire debug (SWD) [Micro trace buffers (MTB) + Data Watchpoints and Traces (DWT)] Simple Cross Triggering Interface
Unique Identification (ID) Number	80-bit wide
AXBS Crossbar	4 Master 4 Slave crossbar Core/DMA0/USB0
Peripheral Bridge (AIPS0)	Peripheral bridge interface
Peripheral Clock Control (PCC)	Peripheral Clock Control module that provides asynchronous clock options and gating function for each peripheral
Crypto Acceleration Unit (CAU0)	CAU0
Cyclic Redundancy Check module (CRC)	Yes
System Tick Timer (SYSTIK)	one 24-bit counter

Table continues on the next page...

Table 2-1. Feature Summary (continued)

Feature	Details
Cyclic Redundancy Check module (CRC)	Yes
Memory	
Flash memory	512 KB flash
Random-access memory (RAM)	128 KB
Read-Only Memory (ROM)	32 KB
System Register File	32 bytes
Clocks	
System Clock Generator (SCG)	SCG module with following modules.
Internal clock references	48 to 60 MHz internal IRC (FIRC) with 1.5% max. deviation across temperature 8 MHz / 2 MHz internal IRC (SIRC) with 3% max. deviation across temperature External Crystal oscillator module XOSC 1 kHz oscillator
External crystal oscillator or resonator	Low range, low power or full-swing: 32 -40 kHz High range, low power or full-swing: 3 - 32 MHz
External square wave input clock	DC to 48 MHz
Phased-locked loop (PLL)	Up to 288 MHz VCO
Human-Machine Interface (HMI)	
General-purpose input/output (GPIO)	1.71-V to 3.3-V Default to disabled (no leakage) Hysteresis and configurable pull direction on all input pins Configurable slew rate and fixed drive strength on all output pins 8 pins with configurable 20 mA high current drive ability Single cycle GPIO control via IOPORT
Touch sensor inputs (TSI)	16-channel Selectable Single channel wakeup source available in all modes DMA support
Pin interrupt	All input Port Pins
Analog	
Power management controller (PMC)	Low and high voltage warning and detect with selectable trip points 1kHz LPO
16-bit analog-to-digital SAR converter (ADC)	>24 single-ended channels for larger packages 4 differential pair (muxed) 2 "status & control", and "result" registers DMA support
1.2-V and 2.1-V Voltage References	High accuracy configurable voltage reference to provide a stable reference for ADC
High speed comparator (CMP) with internal 6-bit digital-to-analog converter (DAC)	2 Up to 6 input channels

Table continues on the next page...

Table 2-1. Feature Summary (continued)

Feature	Details
12-bit DAC	1 DMA support 16 x 12-bit data buffer, true FIFO mode available
Timers	
16-bit timer/pwm module 0 (TPM0)	6 channels, basic TPM function supporting input filter, dead-time insertion and quadrature decode functional in Stop/VLPS mode
16-bit timer/pwm module 1 (TPM1)	2 channels, basic TPM function supporting input filter, dead-time insertion and quadrature decode functional in Stop/VLPS mode
16-bit timer/pwm module 2 (TPM2)	2 channels, basic TPM function supporting input filter, dead-time insertion and quadrature decode functional in Stop/VLPS mode
32-bit Programmable interrupt timer (PIT)	one 4-ch LPIT
Real-time clock (RTC)	Yes
Low-power Timer (LPTMR0/1)	Two 16-bit pulse counter or periodic interrupt functional in all power modes
TSTMR	56-bit software timestamp timer incrementing at 1 MHz
Communication Interfaces	
Universal Serial Bus (USB) 2.0 controller	Supports USB FS device (crystal-less) or USB Host/OTG Supports up to 16 endpoints
USB Regulator	Yes (5V input, 3.3V output, up to 120 mA)
Low Power Serial peripheral interface (LPSPI0/1/2)	3 LPSPIs Functional in Stop/VLPS DMA support, 4-word FIFO support on all 3 LPSPIs
Low Power Inter-Integrated Circuit (LPI^2C0/1/2)	3 LPI2C standard SMBUS compatible I2C 4-word FIFO support on all 3 LPI2Cs DMA support Supports standard, fast and ultra-fast modes Fast+ mode and slave HS-mode are supported on high drive pads Functional in Stop/VLPS modes
Low Power Universal asynchronous receiver/transmitter (LPUART0/1/2)	3 LPUART Standard features Tx pin pseudo open drain with enable/disable programmable configurable from x4 to x32 oversampling Functional in STOP/VLPS modes 32-bit data width

Table continues on the next page...

Table 2-1. Feature Summary (continued)

Feature	Details
	DMA support, 4-word FIFO support on all 3 LPUARTs
Euro, Mastercard, Visa Serial Interface Module (EMVSIM)	1 standalone module supporting ISO 7816 smart card communications
Synchronous Audio Interface (SAI)	1 SAI Standard features support I2S DMA support, 4-word FIFO support
FlexIO	1 FlexIO Support a wide range of protocols including, but not limited to: UART, I2C, SPI, I2S, Camera IF, LCD RGB, PWM/Waveform generation 8 Shift registers and 8 Timers 8/16/24-bit smart LCD drive Functional in STOP/VLPS modes DMA support

2.4 Block Diagram

The below figure shows a top-level block diagram of the superset device.

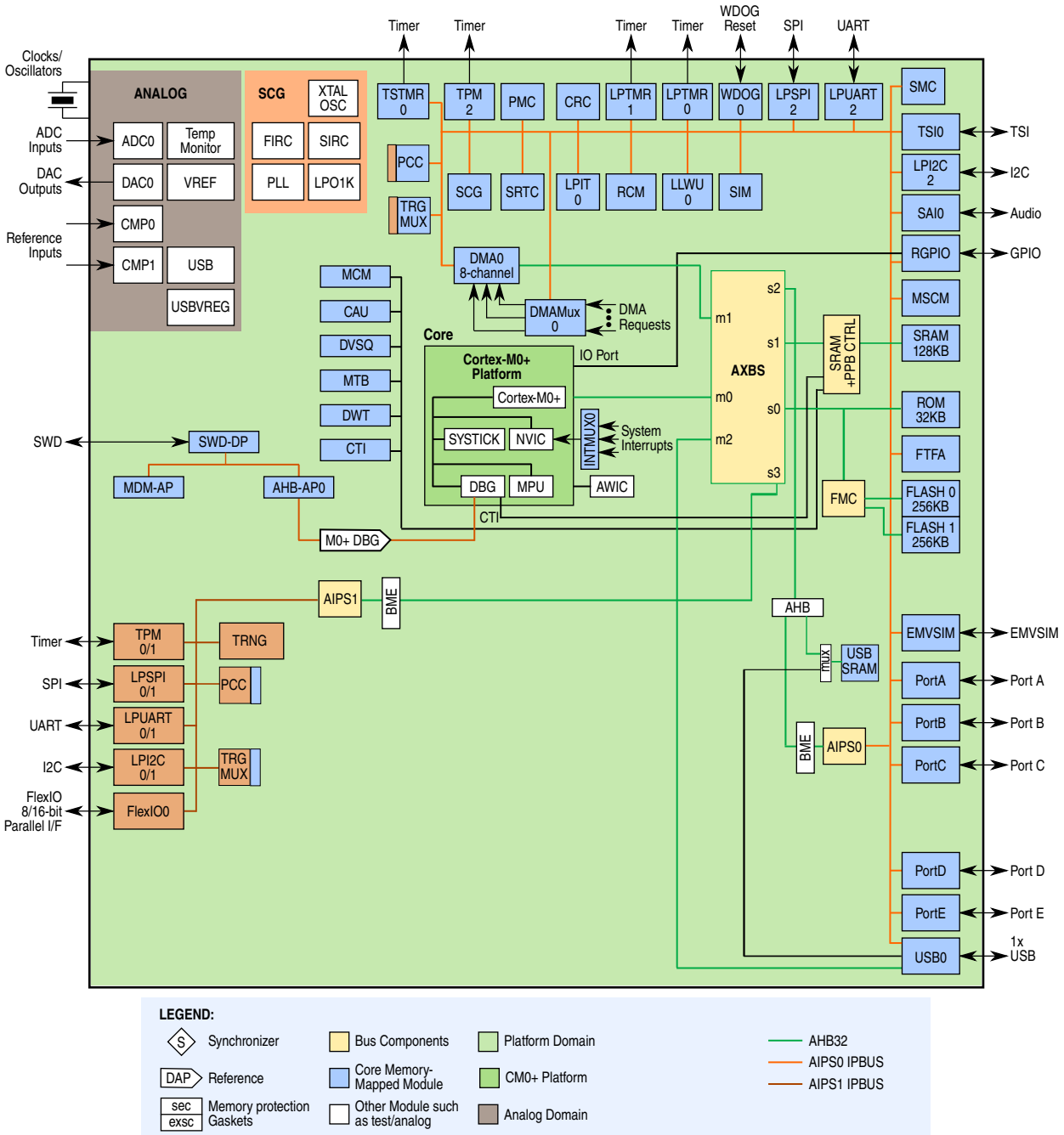


Figure 2-1. Block Diagram

Chapter 3
Chip Configuration

3.1 Introduction

This chapter provides details on the individual modules of the microcontroller.

It includes:

- Module block diagrams showing immediate connections within the device
- Specific module-to-module interactions not necessarily discussed in the individual module chapters
- Links for more information

In more detail:

- The device has 1 CPU (also called a core)
- Peripherals attached to the AIPS0 bridge are SoC-specific.
- The core and the DMAs have the capability to access peripherals on the AIPS0 and AIPS1 bridges.
- The core and DMA0 can freely access all peripherals.

3.2 Clock gating

Peripherals on the AIPS0/AIPS1 bridges have clock gating capability via a Peripheral Clock Control module (PCC).

3.3 Module to Module Interconnects

3.3.1 Interconnection overview

NOTE

Many of the module to module interconnects are handled by the Trigger Mux module (TRGMUX). See the TRGMUX for more information.

The following table captures the module to module interconnections for this device that are not handled by the TRGMUX.

Table 3-1. Module-to-module interconnects

Peripheral	Signal	—	to Peripheral	Use Case	Control
CMP0	CMP0_OUT	to	LPTMR0	Count CMP events	LPTMR0_CSR[TPS]
LPTMRx	Hardware trigger	to	CMPx	Low power triggering of the comparator	CMPx_CR1[TRIGM]
LPTMR0	LPTMR0 Trigger	to	LPTMR1	Count LPTMR0 trigger events	LPTMR1_CSR[TPS]
LPTMR0	Hardware trigger	to	TSI0	TSI triggering	TSI0_GENCS[STM]

3.3.2 Analog reference options

Several analog blocks have selectable reference voltages as shown in Table 3-2. These options allow analog peripherals to share or have separate analog references. Care should be taken when selecting analog references to avoid cross talk noise.

Table 3-2. Analog reference options

Module	Reference option	Comment/ Reference selection
16-bit SAR ADC	1 - VREFH/VREFO 2 - VDDA 3 - Reserved	Selected by ADCx_SC2[REFSEL]
12-bit DAC	1 - VDDA 2 - VREFH/VREFO	Selected by DACx_C0[DACRFS] bit
CMP with 6-bit DAC	Vin1 - VDD Vin2 - VREFH/VREFO	Selected by CMPx_DACCR[VRSEL]

3.4 Core Modules

3.4.1 Introduction

This device has one ARM Cortex M0+ CPU, called the Core.

- The enhanced ARM Cortex M0+ is the member of the Cortex-M Series of processors, with microcontroller cores focused on very cost sensitive, low power applications.
- Has a single 32-bit AMBA AHB-Lite interface and includes an NVIC component.
- Has hardware debug functionality, including support for simple program trace capability.
- The processor supports the ARMv6-M instruction set (Thumb) architecture, including all but three 16-bit Thumb opcodes (52 total) plus seven 32-bit instructions.
- Upward compatible with other Cortex-M profile processors.

3.4.2 ARM Cortex M0+ core

ARM Cortex M0+ parameter settings are:

Table 3-3. ARM Cortex-M0+ parameter settings

Parameter	Verilog name	Value	Description
Arch Clock Gating	ACG	1 = Present	Implements architectural clock gating
DAP Slave Port Support	AHBSLV	1	Supports any AHB debug access port (like the CM0+ DAP)
DAP ROM Table Base	BASEADDR	0xF000_2003	Base address for DAP ROM table
Endianess	BE	0	Little endian control for data transfers
Breakpoints	BKPT	2	Implements 2 breakpoints
Debug Support	DBG	1 = Present	—
Halt Event Support	HALTEV	1 = Present	—
I/O Port	IOP	1 = Present	Implements single-cycle ld/st accesses to special address space
IRQ Mask Enable	IRQDIS	0x00000000	All 32 IRQs are used (set if IRQ is disabled)
Debug Port Protocol	JTAGnSW	0 = SWD	SWD protocol, not JTAG
Core Memory Protection	MPU	1 = Present	Includes ARM based MPU with 8 regions implemented.[1]
Number of IRQs	NUMIRQ	32	Full NVIC request vector
Reset all registers	RAR	0 = Standard	Do not force all registers to be async reset
Multiplier	SMUL	0 = Fast Mul	Implements single-cycle multiplier

Table continues on the next page...

Table 3-3. ARM Cortex-M0+ parameter settings (continued)

Parameter	Verilog name	Value	Description
Multi-drop Support	SWMD	0 = Absent	Do not include serial wire support for multi-drop
System Tick Timer	SYST	1 = Present	Implements system tick timer (for CM4 compatibility)
DAP Target ID	TARGETID	0	—
User/Privileged	USER	1 = Present	Implements processor operating modes
Vector Table Offset Register	VTOR	1 = Present	Implements relocation of exception vector table
WIC Support	WIC	1 = Present	Implements WIC interface
WIC Requests	WICLINES	34	Exact number of wake-up IRQs is 34
Watchpoints	WPT	2	Implements two watchpoints

1. Documentation for this feature can be found at the ARM website

For details on the ARM Cortex-M0+ processor core, see the ARM website:**arm.com**.

3.4.3 Debug facilities

This device supports standard ARM 2-pin SWD debug port.

3.4.4 Buses, interconnects, and interfaces

The ARM Cortex-M0+ core has two bus interfaces:

- Single 32-bit AMBA-3 AHB-Lite system interface that provides connections to peripherals and all system memory, which includes flash memory and RAM
- Single 32-bit I/O port bus interfacing to the GPIO with 1-cycle loads and stores

3.4.5 System tick timer

The CLKSOURCE field in SysTick Control and Status register selects either the core clock (when CLKSOURCE = 1) or a divide-by-16 of the core clock (when CLKSOURCE = 0). Because the timing reference is a variable frequency, the TENMS field in the SysTick Calibration Value Register is always 0.

3.4.6 Caches

This device does not include any processor cache memories.

3.4.7 Interrupt connections

The device has one NVIC module (NVIC0), which has peripheral interrupts assigned to it.

There is one INTMUX module (INTMUX0), which has 4 outputs that each are given a fixed NVIC0 vector space. INTMUX0-ch0-3 has 4 dedicated vector slots on NVIC0.

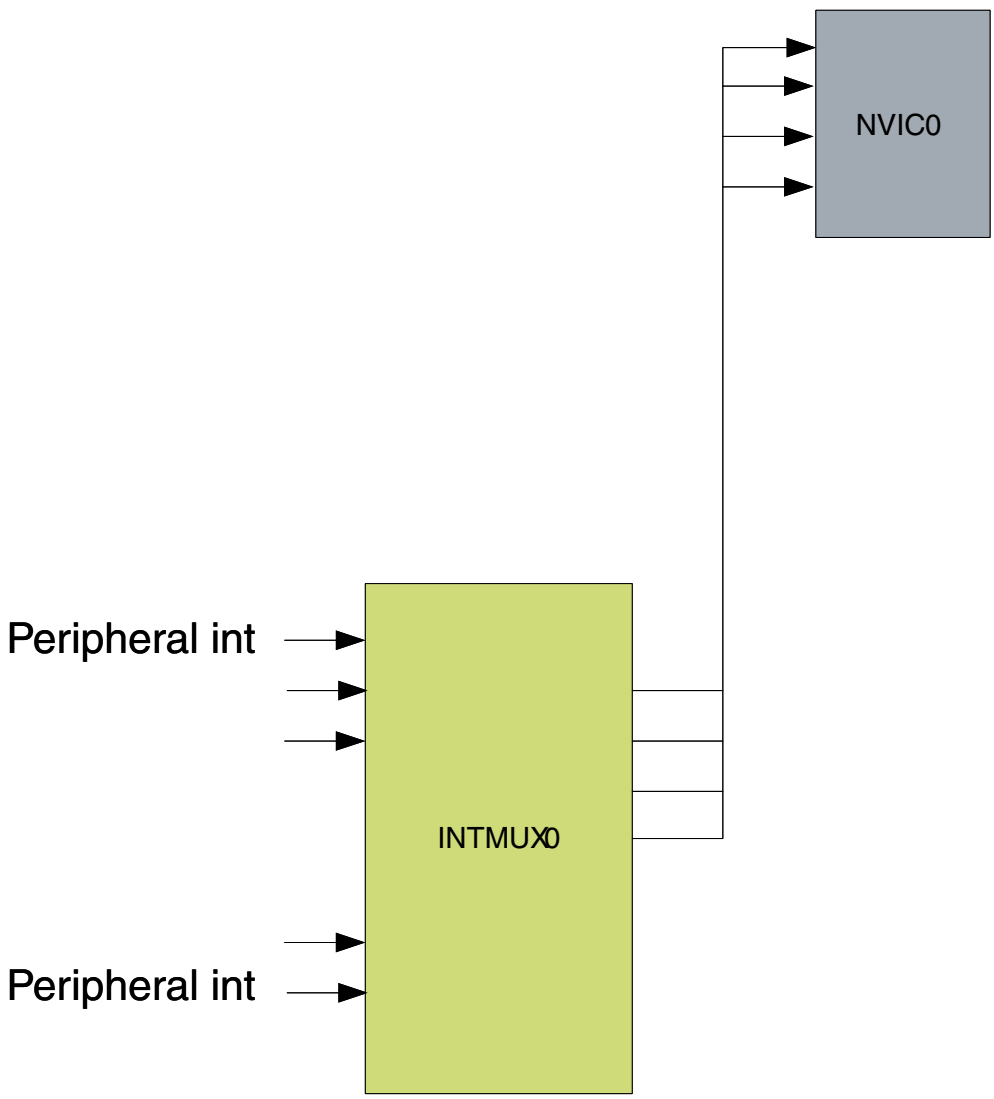


Figure 3-1. Interrupt Routing Block Diagram

The CPU can mask off any interrupt connected to the 32-slot NVIC0 module. The INTMUX0 module allows:

- the ORing of selectable interrupt sources to one specific NVIC0 vector interrupt slot of the CPU.
- the ANDing of selectable interrupt sources to one specific NVIC0 vector interrupt slot of the CPU.

3.4.7.1 NVIC0 Interrupt assignments

The interrupt vector assignments for NVIC0 are defined in the following table.

Table 3-4. NVIC0

Address	Vector	IRQ	Source Module	Source Description
0x0000_0000	0	—	ARM core	Initial Stack Pointer
0x0000_0004	1	—	ARM core	Initial Program Counter
0x0000_0008	2	—	ARM core	Non-maskable interrupt (NMI)
0x0000_000C	3	—	ARM core	Hard Fault (MPU)
0x0000_0010	4	—	—	Reserved
0x0000_0014	5	—	—	Reserved
0x0000_0018	6	—	—	Reserved
0x0000_001C	7	—	—	Reserved
0x0000_0020	8	—	—	Reserved
0x0000_0024	9	—	—	Reserved
0x0000_0028	10	—	—	Reserved
0x0000_002C	11	—	ARM core	Supervisor call (SVCall)
0x0000_0030	12	—	—	Reserved
0x0000_0134	13	—	—	Reserved
0x0000_0038	14	—	ARM core	Pendable request for system service (PendableSrvReq)
0x0000_003C	15	—	ARM core	System tick timer (SysTick)
On Platform vectors	—	—	—	—
0x0000_0040	16	0	DMA0	DMA0 channel 0 or 4 transfer complete
0x0000_0044	17	1	DMA0	DMA0 channel 1 or 5 transfer complete
0x0000_0048	18	2	DMA0	DMA0 channel 2 or 6 transfer complete
0x0000_004C	19	3	DMA0	DMA0 channel 3 or 7 transfer complete
0x0000_0050	20	4	CTI0, DMA0_error	Cross Trigger Interface 0 and DMA0 channel errors
Off Platform vectors	—	—	—	—

Table continues on the next page...

Table 3-4. NVIC0 (continued)

Address	Vector	IRQ	Source Module	Source Description
0x0000_0054	21	5	FlexIO0	Flexible IO
0x0000_0058	22	6	TPM0	Low Power Timer PWM module 0
0x0000_005C	23	7	TPM1	Low Power Timer PWM module 1
0x0000_0060	24	8	TPM2	Low Power Timer PWM module 2
0x0000_0064	25	9	LPIT0	Low Power Periodic Interrupt Timer 0
0x0000_0068	26	10	LPSPI0	Low Power Serial Peripheral Interface 0
0x0000_006C	27	11	LPSPI1	Low Power Serial Peripheral Interface 1
0x0000_0070	28	12	LPUART0	Low Power UART 0
0x0000_0074	29	13	LPUART1	Low Power UART 1
0x0000_0078	30	14	LPI2C0	Low Power IIC module 0
0x0000_007C	31	15	LPI2C1	Low Power IIC module 1
0x0000_0080	32	16	—	Reserved
0x0000_0084	33	17	PortA	Port A
0x0000_0088	34	18	PortB	Port B
0x0000_008C	35	19	PortC	Port C
0x0000_0090	36	20	PortD	Port D
0x0000_0094	37	21	PortE	Port E
0x0000_0098	38	22	LLWU0	Low Leakage Wake Up
0x0000_009C	39	23	SAI0	Serial Audio Interface 0
0x0000_00A0	40	24	USB0	Universal Serial Bus 0
0x0000_00A4	41	25	ADC0	Analog to Digital Convertor 0
0x0000_00A8	42	26	LPTMR0	Low Power Timer 0
0x0000_00AC	43	27	RTC Secs	Real Time Clock Seconds
0x0000_00B0	44	28	INTMUX0-0	Selectable peripheral interrupt INTMUX0-0
0x0000_00B4	45	29	INTMUX0-1	Selectable peripheral interrupt INTMUX0-1
0x0000_00B8	46	30	INTMUX0-2	Selectable peripheral interrupt INTMUX0-2
0x0000_00BC	47	31	INTMUX0-3	Selectable peripheral interrupt INTMUX0-3

3.4.7.2 INTMUX0 Input Mux Assignment

Table 3-5. INTMUX0 Peripheral Interrupt assigment

INTMUX0 input	Peripheral Interrupt Source
0	LPTMR1
1	Reserved
2	Reserved
3	Reserved
4	LSPI2
5	LPUART2
6	EMVSIM
7	LPI2C2
8	TSI
9	PMC
10	FTFA
11	SCG
12	WDOG0
13	DAC0
14	TRNG
15	RCM
16	CMP0
17	CMP1
18	RTC Alarm
19	Reserved
20	Reserved
21	Reserved
22	Reserved
23	Reserved
24	Reserved
25	Reserved
26	Reserved
27	Reserved
28	Reserved
29	Reserved
30	Reserved
31	Reserved

3.4.7.3 Interrupt priority levels

This device supports 4 priority levels for interrupts. Therefore, in the NVIC, each source in the IPR registers contains 2 bits. For example, IPR0 is shown below:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IRQ3		0	0	0	0	0	0	IRQ2		0	0	0	0	0	0	IRQ1		0	0	0	0	0	0	IRQ0		0	0	0	0	0	0
W																																

3.4.7.4 Non-maskable interrupt

The non-maskable interrupt request to the NVIC is controlled by the external $\overline{\text{NMI}}$ signal. The pin that the $\overline{\text{NMI0}}$ signal is multiplexed on, must be configured for the $\overline{\text{NMI}}$ function, to generate the non-maskable interrupt request.

- The Core enables the NMI in NVIC0 by default, and the $\overline{\text{NMI0}}$ pin is also enabled by default.

3.4.8 Asynchronous Wake-up Interrupt Controller (AWIC)

3.4.8.1 AWIC overview

The primary function of the AWIC block is to detect asynchronous wake-up events in stop modes, and then signal to the clock control logic to resume system clocking. After the clock restarts, the NVIC observes the pending interrupt and performs the normal interrupt or event processing.

3.4.8.2 Wake-up sources

The device uses the following internal and external inputs to the AWIC module.

Table 3-7. AWIC stop wake-up sources

Wake-up source	Description
Available system resets	RESET_b pin when LPO is its clock source
Low-voltage detect	Power management controller—functional in Stop mode
Low-voltage warning	Power management controller—functional in Stop mode
Pin interrupts	Port control module—any enabled pin interrupt is capable of waking the system.
ADC	The ADC is functional when using internal clock source.
CMP	Interrupt in normal or trigger mode

Table continues on the next page...

Table 3-7. AWIC stop wake-up sources (continued)

Wake-up source	Description
LPI2C	Any enabled interrupt can be a source as long as the module remains clocked.
LPUART	Any enabled interrupt can be a source as long as the module remains clocked.
RTC	Alarm or seconds interrupt
TSI	Any enabled interrupt can be a source as long as the module remains clocked.
NMI	NMI_b pin
TPM	Any enabled interrupt can be a source as long as the module remains clocked.
LPTMRx	Any enabled interrupt can be a source as long as the module remains clocked.
LPSPI	Any enabled interrupt can be a source as long as the module remains clocked.
LPIT	Any enabled interrupt can be a source as long as the module remains clocked.
FlexIO	Any enabled interrupt can be a source as long as the module remains clocked.
USB	Wake-up

3.5 System Modules

3.5.1 Crossbar Switch

3.5.1.1 Crossbar switch master assignments

The masters connected to the crossbar switch are assigned as follows:

AXBS0 Master module	Master port number
Core 0- ARM core unified bus	0
DMA0	1
USB0	2
Reserved	3

3.5.1.2 Crossbar switch slave assignments

The slaves connected to the crossbar switch are assigned as follows:

AXBS0 Slave module	Slave port number
Flash memory controller and Boot ROM	0
Core 0 SRAM controller	1

Table continues on the next page...

AXBS0 Slave module	Slave port number
Peripheral bridge 0	2
Peripheral bridge 1	3

3.5.2 Low-Leakage Wake-up Unit (LLWU)

3.5.2.1 Wake-up Sources

The device uses the following internal peripheral and external pin inputs as wakeup sources to the LLWU module. LLWU_Px are external pin inputs, and LLWU_M0IF-M7IF are connections to the internal peripheral interrupt flags.

NOTE

In addition to the LLWU wakeup sources, the device also wakes from low power modes when NMI or RESET pins are enabled and the respective pin is asserted.

Table 3-8. LLWU Wakeup Sources

IRQ	Module source or pin name
LLWU_P0	PTE1
LLWU_P1	PTE2
LLWU_P2	PTE4
LLWU_P3	PTA4
LLWU_P4	PTA13
LLWU_P5	PTB0
LLWU_P6	PTC1
LLWU_P7	PTC3
LLWU_P8	PTC4
LLWU_P9	PTC5
LLWU_P10	PTC6
LLWU_P11	PTC11
LLWU_P12	PTD0
LLWU_P13	PTD2
LLWU_P14	PTD4
LLWU_P15	PTD6
LLWU_P16	PTE6
LLWU_P17	Reserved
LLWU_P18	Reserved
LLWU_P19	PTE17

Table continues on the next page...

Table 3-8. LLWU Wakeup Sources (continued)

IRQ	Module source or pin name
LLWU_P20	PTE18
LLWU_P21	PTE25
LLWU_P22	PTA10
LLWU_P23	PTA11
LLWU_P24	PTD8
LLWU_P25	PTD11
LLWU_P26	VREGIN
LLWU_P27	USB0_DM
LLWU_P28	USB0_DP
LLWU_P29 - LLWU_P31	Reserved
LLWU_M0IF	LPTMR0
LLWU_M1IF	CMP0
LLWU_M2IF	CMP1
LLWU_M3IF	LPTMR1 asynchronous interrupt
LLWU_M4IF	TSI0
LLWU_M5IF	RTC Alarm
LLWU_M6IF	Reserved
LLWU_M7IF	RTC Seconds
LLWU_M0DR	LPTMR0 asynchronous DMA
LLWU_M3DR	LPTMR1 asynchronous DMA
LLWU_M4DR	TSI asynchronous DMA
LLWU_M6DR	LPTMR0 trigger
LLWU_M7DR	LPTMR1 trigger

3.5.3 DMAMUXs

3.5.3.1 DMAMUX0

DMAMUX0 is a DMA request mux that allows up to 63 DMA request signals to be mapped to any of the 8 DMA channels of DMA0. Because of the mux, there is no hard correlation between any of the DMA request sources and a specific DMA channel. Some of the modules support asynchronous DMA operation, as shown in the DMA source assignment table.

Table 3-9. DMAMUX0 source assignments

Source No	Source module	Source description	Async DMA capable
0	—	Channel disabled	—
1	FlexIO0	Shifter 0	Yes
2	FlexIO0	Shifter 1	Yes
3	FlexIO0	Shifter 2	Yes
4	FlexIO0	Shifter 3	Yes
5	FlexIO0	Shifter 4	Yes
6	FlexIO0	Shifter 5	Yes
7	FlexIO0	Shifter 6	Yes
8	FlexIO0	Shifter 7	Yes
9	LPI2C0	Master/Slave Receive	Yes
10	LPI2C0	Master/Slave Transmit	Yes
11	LPI2C1	Master/Slave Receive	Yes
12	LPI2C1	Master/Slave Transmit	Yes
13	LPI2C2	Master/Slave Receive	Yes
14	LPI2C2	Master/Slave Transmit	Yes
15	LPUART0	Receive	Yes
16	LPUART0	Transmit	Yes
17	LPUART1	Receive	Yes
18	LPUART1	Transmit	Yes
19	LPUART2	Receive	Yes
20	LPUART2	Transmit	Yes
21	LPSPI0	Receive	Yes
22	LPSPI0	Transmit	Yes
23	LPSPI1	Receive	Yes
24	LPSPI1	Transmit	Yes
25	LPSPI2	Receive	Yes
26	LPSPI2	Transmit	Yes

Table continues on the next page...

Table 3-9. DMAMUX0 source assignments (continued)

Source No	Source module	Source description	Async DMA capable
27	TPM0	Channel 0	Yes
28	TPM0	Channel 1	Yes
29	TPM0	Channel 2	Yes
30	TPM0	Channel 3	Yes
31	TPM0	Channel 4	Yes
32	TPM0	Channel 5	Yes
33	Reserved	—	—
34	Reserved	—	—
35	TPM0	Overflow	Yes
36	TPM1	Channel 0	Yes
37	TPM1	Channel 1	Yes
38	TPM1	Overflow	Yes
39	TPM2	Channel 0	Yes
40	TPM2	Channel 1	Yes
41	TPM2	Overflow	Yes
42	Reserved	—	—
43	EMVSIM	Receive	Yes
44	EMVSIM	Transmit	Yes
45	SAI0	Receive	Yes
46	SAI0	Transmit	Yes
47	PortA	Port Control module	Yes
48	PortB	Port Control module	Yes
49	PortC	Port Control module	Yes
50	PortD	Port Control Module	Yes
51	PortE	Port Control module	Yes
52	ADC0	Conversion complete	Yes
53	Reserved	—	—
54	DAC0	Digital-to-Analog Converter	Yes
55	Reserved	—	—
56	CMP0	Comparator	Yes
57	CMP1	Comparator	Yes
58	Reserved	—	—
59	Reserved	—	—
60	TSI	Touch Sensor Interface	Yes
61	LPTMR0	Low Power Timer 0	Yes
62	LPTMR1	Low Power Timer 1	Yes
63	DMA MUX	Always enabled	

3.5.4 Watchdog (WDOG)

3.5.4.1 WDOG clocks

The multiple clock inputs for the WDOG are:

- 1 kHz clock
- bus clock
- 8 MHz or 2 MHz internal reference clock
- external crystal

3.5.5 System Register File Configuration

This section summarizes how the module has been configured in the chip.

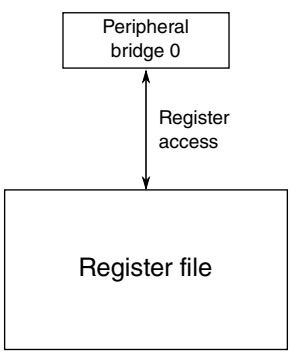


Figure 3-2. System Register file configuration

Table 3-10. Reference links to related information

Topic	Related module	Reference
Full description	Register file	Register file
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management

3.5.5.1 System Register file

This device includes a 32-byte register file that is powered in all power modes.

Also, it retains contents during low-voltage detect (LVD) events and is only reset during a power-on reset.

Base address of this register file is 0x4007_C000. (see Table "Peripheral bridge 0 slot assignment").

3.5.6 Peripheral Clock Control (PCC) Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

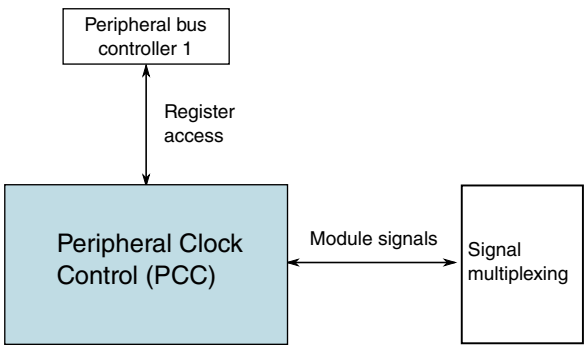


Figure 3-3. PCC configuration

3.5.6.1 Overview

The device deploys two Peripheral Clock Control modules that allow clock gating, and clock source selection to each peripheral. Each AIPS bridge has its own corresponding PCC module.

In addition to this clock, there are optional peripheral clock sources that may be asynchronous to the CPU/Platform clock: SCGIRCLK, SCGFIRCLK, SCGPCLK, SCGFCLK, LPO, OSCCLK, SCGSPCLK.

3.5.7 System Register File Configuration

This section summarizes how the module has been configured in the chip.

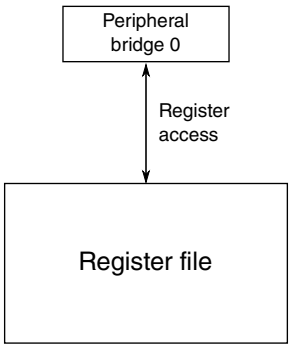


Figure 3-4. System Register file configuration

Table 3-11. Reference links to related information

Topic	Related module	Reference
Full description	Register file	Register file
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management

3.5.7.1 System Register file

This device includes a 32-byte register file that is powered in all power modes.

Also, it retains contents during low-voltage detect (LVD) events and is only reset during a power-on reset.

Base address of this register file is 0x4007_C000. (see Table "Peripheral bridge 0 slot assignment").

3.6 Security

3.6.1 CAU Configuration

There is 1 CAU module: CAU0, which is connected to the Core PPB.

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

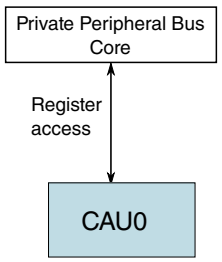


Figure 3-5. CAU configuration

3.6.1.1 CAU Overview

The CAU provides security encrypt/decrypt acceleration to allow users to share secure data with external devices via a serial communication port (such as an SAI or LPUART module).

The CAU clock source is the CPU/platform clock.

3.7 Analog

3.7.1 16-bit SAR ADC configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

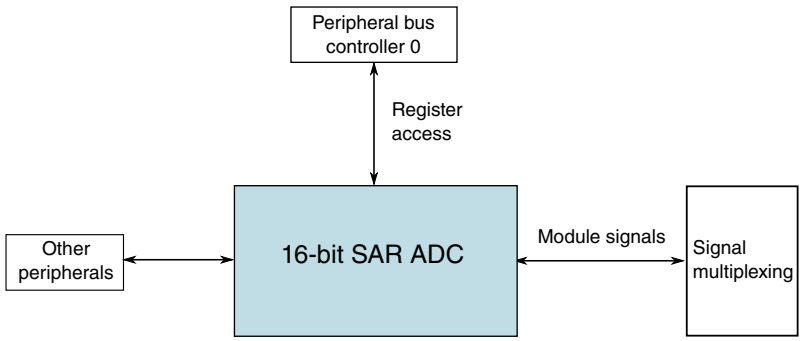


Figure 3-6. 16-bit SAR ADC configuration

Table 3-12. Reference links to related information

Topic	Related module	Reference
Full description	16-bit SAR ADC	16-bit SAR ADC
System memory map	—	System memory map
Clocking	—	Clock distribution
Power management	—	Power management
Signal multiplexing	Port control	Signal multiplexing

3.7.1.1 ADC instantiation information

This device contains one 16-bit successive approximation ADC with up to ~30 to 38 channels.

The ADC supports both software and hardware triggers.

The number of ADC channels present on the device is determined by the pinout of the specific device package.

3.7.1.2 DMA Support on ADC

Applications may require continuous sampling of the ADC that may have considerable load on the CPU. The ADC supports DMA request functionality for higher performance when the ADC is sampled at a very high rate. The ADC can trigger the DMA (via DMA req) on conversion completion.

3.7.1.3 ADC0 connections/channel assignments

NOTE

As indicated by the following sections, each ADCx_DPx input and certain ADCx_DMx inputs may operate as single-ended ADC channels in single-ended mode.

ADC channel (SC1n[ADCH])	Channel	Input signal (SC1n[DIFF]= 1)	Input signal (SC1n[DIFF]= 0)
00000	DAD0	ADC0_DP0 and ADC0_DM0	ADC0_DP0/ADC0_SE0
00001	DAD1	ADC0_DP1 and ADC0_DM1	ADC0_DP1/ADC0_SE1
00010	DAD2	ADC0_DP2 and ADC0_DM2	ADC0_DP2/ADC0_SE2
00011	DAD3	ADC0_DP3 and ADC0_DM3	ADC0_DP3/ADC0_SE3
00100[1]	AD4a	Reserved	ADC0_DM0/ADC0_SE4a
00101[1]	AD5a	Reserved	ADC0_DM1/ADC0_SE5a
00110[1]	AD6a	Reserved	ADC0_DM2/ADC0_SE6a
00111[1]	AD7a	Reserved	ADC0_DM3/ADC0_SE7a
00100[1]	AD4b	Reserved	ADC0_SE4b
00101[1]	AD5b	Reserved	ADC0_SE5b
00110[1]	AD6b	Reserved	ADC0_SE6b
00111[1]	AD7b	Reserved	ADC0_SE7b
01000	AD8	Reserved	ADC0_SE8
01001	AD9	Reserved	ADC0_SE9
01010	AD10	Reserved	Reserved
01011	AD11	Reserved	ADC0_SE11
01100	AD12	Reserved	ADC0_SE12
01101	AD13	Reserved	ADC0_SE13
01110	AD14	Reserved	ADC0_SE14
01111	AD15	Reserved	ADC0_SE15
10000	AD16	Reserved	Reserved
10001	AD17	Reserved	Reserved
10010	AD18	Reserved	Reserved
10011	AD19	Reserved	Reserved
10100	AD20	Reserved	Reserved
10101	AD21	Reserved	Reserved
10110	AD22	Reserved	Reserved
10111	AD23	Reserved	12-bit DAC0 Output/ ADC0_SE23
11000	AD24	Reserved	Reserved
11001	AD25	Reserved	Reserved
11010	AD26	Temperature Sensor (Diff)	Temperature Sensor (S.E)
11011	AD27	Bandgap (Diff)[2]	Bandgap (S.E)[2]
11100	AD28	Reserved	Reserved

Table continues on the next page...

ADC channel (SC1n[ADCH])	Channel	Input signal (SC1n[DIFF]= 1)	Input signal (SC1n[DIFF]= 0)
11101	AD29	-VREFH (Diff)	VREFH (S.E)
11110	AD30	Reserved	VREFL
11111	AD31	Module Disabled	Module Disabled

1. ADCx_CFG2[MUXSEL] bit selects between ADCx_SEn channels a and b. Refer to MUXSEL description in ADC chapter for details.
2. This is the PMC bandgap 1V reference voltage. Prior to reading from this ADC channel, ensure that you enable the bandgap buffer by setting the PMC_REGSC[BGBE] bit. Refer to the device data sheet for the bandgap voltage (V_{BG}) specification.

3.7.1.4 ADC analog supply and reference connections

See Analog reference options.

- This device includes dedicated VDDA and VSSA pins.
- This device contains dedicated VREFH and VREFL pins in all packages.
- The output of the on-chip high precision VREF module is shared with VREFH. When VREF is enabled, this pin needs to connect a capacitor to ground.

The ADC supports the following references:

- VREFH is connected to VREF_OUT as the external reference.
- VDDA is connected as the V_{ALTH} reference.

3.7.1.5 ADC clock source

The ADC module clock is sourced from the Peripheral Clock Control (PCC) module.

3.7.2 CMP configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

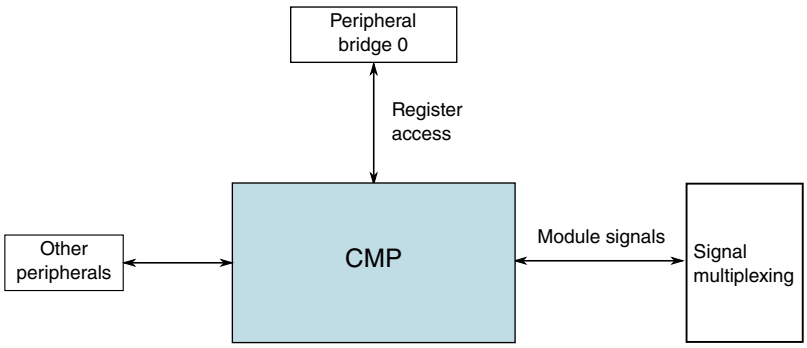


Figure 3-7. CMP configuration

Table 3-13. Reference links to related information

Topic	Related module	Reference
Full description	Comparator (CMP)	Comparator
System memory map	—	System memory map
Clocking	—	Clock distribution
Power management	—	Power management
Signal multiplexing	Port control	Signal multiplexing

3.7.2.1 CMP instantiation information

The device includes two high-speed comparators each with two 8-input multiplexers for both the inverting and non-inverting inputs of the comparator. Each CMP input channel connects to both muxes. Two of the channels are connected to internal sources, leaving resources to support up to 6 input pins. See the channel assignment table for a summary of CMP input connections for this device.

The CMPs also include one 6-bit DAC with a 64-tap resistor ladder network, which provides a selectable voltage reference for applications where voltage reference is needed for an internal connection to the CMP.

The CMPs can be optionally on in all modes except VLLS0.

3.7.2.2 CMP connections

The following table shows the fixed internal connections to the CMPs.

Table 3-14. CMP input connections

CMP inputs	CMP0	CMP1
IN0	PTC6	PTC2
IN1	PTC7	PTC3
IN2	PTC8	Reserved
IN3	PTC9	PTE30/12-bit DAC0 reference
IN4	PTE30/12-bit DAC0 reference	Reserved
IN5	PTE29	PTE29
IN6	Bandgap[1]	Bandgap[1]
IN7	6-bit DAC0 reference	6-bit DAC1 reference

1. This is the PMC bandgap 1V reference voltage. Prior to using as CMP input, ensure that you enable the bandgap buffer by setting PMC_REGSC[BGBE]. See the device data sheet for the bandgap voltage (V_{BG}) specification.

The following table shows the external output pin options for the CMPs.

Table 3-15. CMP external output pin options

CMP outputs	External pin options
CMP0_OUT	PTE0, PTB20, PTC0, PTC5
CMP1_OUT	PTB21, PTC4

3.7.3 VREF

3.7.3.1 VREF Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

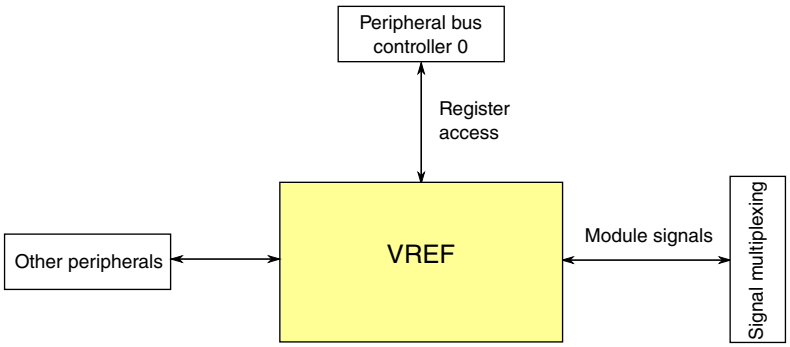


Figure 3-8. VREF configuration

Table 3-16. Reference links to related information

Topic	Related module	Reference
Full description	VREF	VREF Analog reference options
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.7.3.2 VREF Overview

This device includes a voltage reference (VREF) to supply an accurate 1.2 V or 2.1 V voltage output.

The voltage reference can provide a reference voltage to external peripherals or a reference to analog peripherals, such as the ADC, DAC, or CMP.

NOTE

PMC_REGSC[BGEN] bit must be set if the VREF regulator is required to remain operating in VLPx modes.

NOTE

For either an internal or external reference if the VREF_OUT functionality is being used, VREF_OUT signal must be connected to an output load capacitor. Refer to the device data sheet for more details.

3.7.3.3 Clock Gating

The clock to the VREF module can be turned on or off using the PCC_VREF[CGC] bit in the PCC module. This bit is cleared after any reset, which disables the clock to the VREF module to conserve power. Before initializing the VREF module, set PCC_VREF[CGC] to enable the clock. Before turning off the clock, make sure to disable the VREF module. For more details, see the clock distribution chapter.

3.7.4 12-bit DAC configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

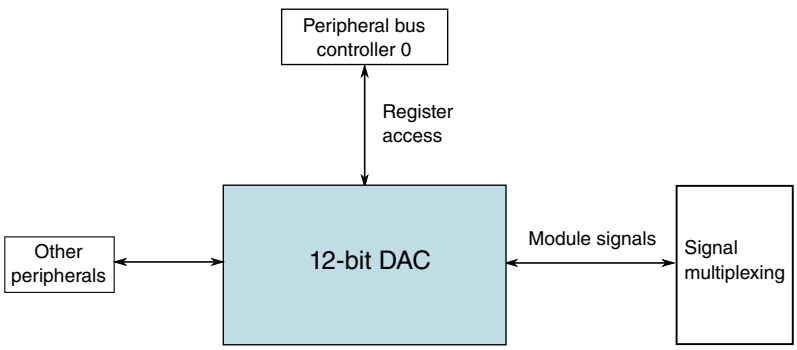


Figure 3-9. 12-bit DAC configuration

Table 3-17. Reference links to related information

Topic	Related module	Reference
Full description	12-bit DAC	12-bit DAC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.7.4.1 12-bit DAC instantiation information

This device contains one 12-bit digital-to-analog converter (DAC) with programmable reference generator output. The DAC includes a 16-word FIFO for DMA support.

3.7.4.2 12-bit DAC Analog Supply and Reference Connections

This device:

- includes dedicated VDDA and VSSA pins.
- contains dedicated VREFH and VREFL pins in all packages.

3.7.4.3 12-bit DAC reference

The 12-bit DAC has a selectable reference voltage. See Analog Reference Options for more information.

NOTE

If the DAC and ADC use the same reference simultaneously, then some degradation of ADC accuracy can be expected (because of DAC switching).

3.8 Timers

3.8.1 Timer/PWM module configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

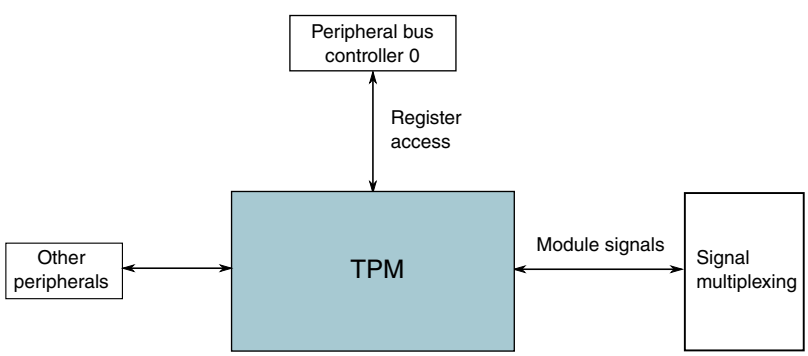


Figure 3-10. TPM configuration

Table 3-18. Reference links to related information

Topic	Related module	Reference
Full description	Timer/PWM module	Timer/PWM module

Table continues on the next page...

Table 3-18. Reference links to related information (continued)

Topic	Related module	Reference
System memory map	—	System memory map
Clocking	—	Clock distribution
Power management	—	Power management
Signal multiplexing	Port control	Signal multiplexing

3.8.1.1 TPM instantiation information

This device contains 3 low-power Timer/PWM modules (TPM), which can all be functional in Stop/VLPS mode. In Stop/VLPS mode, the clock source is either external or internal. The following table shows how these modules are configured.

Table 3-19. TPM configuration

TPM instance	Number of channels	Features/usage
TPM0	6	Basic TPM, support for input filters, dead-time insertion and quadrature decode, functional in Stop/VLPS mode
TPM1	2	Basic TPM, support for input filters, dead-time insertion and quadrature decode, functional in Stop/VLPS mode
TPM2	2	Basic TPM, support for input filters, dead-time insertion and quadrature decode, functional in Stop/VLPS mode

- TPM0 and TPM1 are accessed via the AIPS1 peripheral bridge.
- TPM2 is accessed via the AIPS0 peripheral bridge.
- TPM0 and TPM2 can be optionally clocked from the TPM1 clock, thereby providing synchronization for all 3 timers.

3.8.1.2 Clock options

Each TPM module has its own TPM clock. The selected source is controlled by a PCC module.

Each TPM also supports an external clock mode (TPMx_SC[CMOD]=1x) in which the counter increments after a rising edge detect of an external clock input (synchronized to the TPM clock source). The external clock can be either TPM0_CLKIN or TPM1_CLKIN.

NOTE

The selected external clock must be less than half the frequency of the TPM clock source.

3.8.1.3 Trigger options

Each TPM has a selectable external trigger input source controlled by TPMx_CONF[TRGSEL] to use for starting the counter and/or reloading the counter. The options available are shown in the following table.

External trigger input source is enabled by clearing TPMx_CONF[TRGSRC] bit.

Table 3-20. TPM trigger options

TPMx_CONF[SRC]	TPMx_CONF[TRGSEL]	Selected source
0	00	TRGMUX_TPMx_trigger0
0	01	TRGMUX_TPMx_trigger1
0	10	TRGMUX_TPMx_trigger2
0	11	Reserved
1	00	Reserved
1	01	Channel 0 pin input capture
1	10	Channel 1 pin input capture
1	11	Channel 0 or Channel 1 pin input capture

3.8.1.4 TPM interrupts

The TPM has multiple sources of interrupt. However, these sources are OR'd together to generate a single interrupt request to the interrupt controller. When a TPM interrupt occurs, read the TPM status registers to determine the exact interrupt source.

3.8.2 LPIT

3.8.2.1 LPIT/DMA periodic trigger assignments

The LPIT generates periodic trigger events to the DMA channel mux. The LPIT to the DMAMUX trigger is configured in the Trigger MUX module (TRGMUX).

Table 3-21. LPIT0 channel assignments for periodic DMA triggering

LPIT0 channel	DMA channel number
LPIT0 Channel 0	DMA0 Channel 0
LPIT0 Channel 1	DMA0 Channel 1

3.8.2.2 PIT/ADC triggers

PIT triggers can be selected as ADCx trigger sources. For more details, see the Trigger Mux.

3.8.2.3 LPIT/TPM Triggers

LPIT triggers are selected as TPMx trigger sources in the TRGMUX module. For more details, refer to TRGMUX chapter.

3.8.3 Low Power Timer (LPTMR)

3.8.3.1 LPTMR instantiation information

This device has two low-power timers: LPTMR0 and LPTMR1. Both allow operation during all power modes. LPTMR0 is accessed via AIPS0. LPTMR1 is accessed via AIPS1.

The LPTMR can operate as a real-time interrupt or pulse accumulator. It includes a 2^N prescaler (real-time interrupt mode) or a glitch filter (pulse accumulator mode).

An LPTMR can be clocked from the internal reference clock, the internal 1 kHz LPO, ERCLK, or an external 32.768 kHz crystal. In VLLS0 mode, the clocking option is limited to an external pin with the OSC configured for bypass (external clock) operation.

An interrupt is generated (and the counter may reset) when the counter equals the value in the 16-bit compare register.

3.8.3.2 LPTMR pulse counter input options

LPTMRx_CSR[TPS] configures the input source used in pulse counter mode. The following table shows the chip-specific input assignments for this field.

LPTMRx_CSR[TPS]	Pulse counter input number	LPTMR0 input source	LPTMR1 input source
00	0	CMP0 output	LPTMR0 trigger
01	1	LPTMR0_ALT1 pin	LPTMR1_ALT1 pin
10	2	LPTMR0_ALT2 pin	LPTMR1_ALT2 pin
11	3	LPTMR0_ALT3 pin	LPTMR1_ALT3 pin

NOTE

LPTMR0_ALT*n* is on the same pin as the corresponding LPTMR1_ALT*n*.

3.8.3.3 LPTMR prescaler/glitch filter clocking options

The prescaler and glitch filter of the LPTMR module can be clocked from one of four sources determined by LPTMRx_PSR[PCS]. The following table shows the chip-specific clock assignments for this field.

NOTE

The chosen clock must remain enabled if the LPTMR is to continue operating in all required low-power modes.

LPTMR0_PSR[PCS], LPTMR1_PSR[PCS]	Prescaler/glitch filter clock number	Chip clock
00	0	SIRCCLK—internal reference clock (not available in LLS and VLLS modes)
01	1	LPO—1 kHz clock (not available in VLLS0 mode)
10	2	OSC32KCLK (not available in VLLS0 mode when using 32 kHz oscillator)
11	3	ERCLK—external reference clock (not available in VLLS0 mode)

See Clock distribution for more details on these clocks.

3.8.4 RTC configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

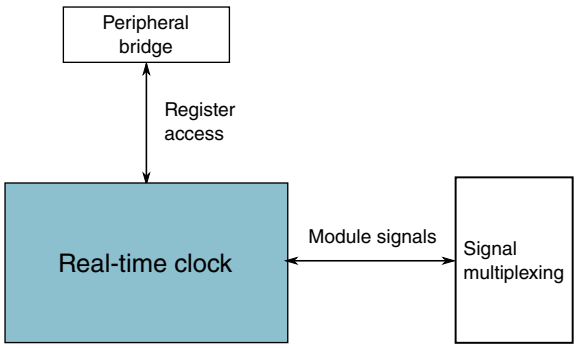


Figure 3-11. RTC configuration

The device has one secure Real Time Clock module (RTC). The RTC module is accessed via AIPS0 peripheral bridge.

3.9 Communication interfaces

3.9.1 Universal Serial Bus (USB) FS Subsystem

The USB FS subsystem includes these components:

- Dual-role USB OTG-capable (On-The-Go) controller that supports a full-speed (FS) device or FS/LS host. The module complies with the USB 2.0 specification.
- USB transceiver that includes internal 15 kΩ pulldowns on the D+ and D- lines for host mode functionality and a 1.5 kΩ pullup on the D+ line for device mode functionality.
- A 3.3 V regulator (See Introduction).
- Status detection and wakeup functions for USB data pins, VBUS pin, and OTG ID pin.
- IRC48 with clock recovery block to eliminate the 48MHz crystal. This is available for USB device mode only.

NOTE

USB OTG is not functional in VLPx, VLLSx, and any Stop modes.

NOTE

For the USB FS OTG controller to operate, the minimum system clock frequency is 20 MHz.

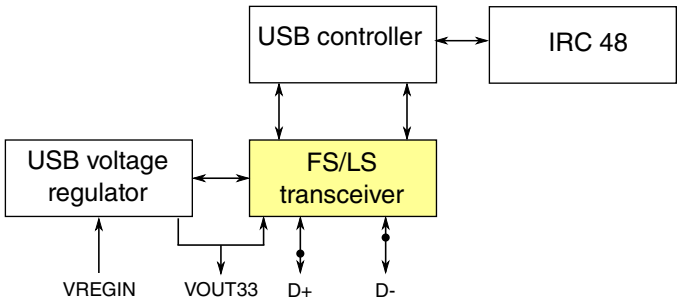


Figure 3-12. USB FS/LS Subsystem Overview

NOTE

Use the following code sequence to select USB clock source, USB clock divide ratio, and enable its clock gate to avoid potential clock glitches which may result in USB enumeration stage failure.

1. Select the USB clock source by configuring PCC_USB0FS[PCS]. Ensure that PCC_USB0FS[CGC] is cleared.
2. Select the desired clock divide ratio by configuring PCC_USB0FS[FRAC] and PCC_USB0FS[PCD].
3. Enable USB clock gate by setting PCC_USB0FS[CGC].

3.9.1.1 USB and VREGIN pin status detection and wakeup interrupt features

This device does not have a dedicated VBUS detect pin. For VBUS detection, use a GPIO pin for both bus-powered and self-powered USB cases. Because the GPIO pins on this device do not directly support a 5V input, use an external resistive voltage divider to keep the input voltage within the valid range if a GPIO pin is used for VBUS detection.

This device does not have a dedicated OTG ID detect pin. For OTG ID pin detection, if needed, use a GPIO configured as an input pin with pullup enabled.

When the USB detects that there is no activity on the USB bus for more than 3 ms, the USBx_ISTAT[SLEEP] bit is set. This bit can cause an interrupt and software decides the appropriate action.

Waking from a low power mode (except in LLS/VLLS mode where USB is not powered) occurs through an asynchronous interrupt triggered by activity on the USB bus. Setting the USBx_USBTRC0[USBRESMEN] bit enables this function.

The following wakeup feature is also supported:

- In LLS/VLLS, if the GPIO pins chosen to detect VBUS and OTG ID are selected from those pins which are inputs to the LLWU, transitions on them can generate a wakeup.

3.9.1.2 USB Power Distribution

This chip includes an internal 5 V to 3.3 V USB regulator that powers the USB transceiver or the MCU (depending on the application).

3.9.1.2.1 AA/AAA cells power supply

The chip can be powered by two AA/AAA cells. In this case, the MCU is powered through VDD which is within the 1.8 to 3.0 V range. After USB cable insertion is detected, the USB regulator is enabled to power the USB transceiver.

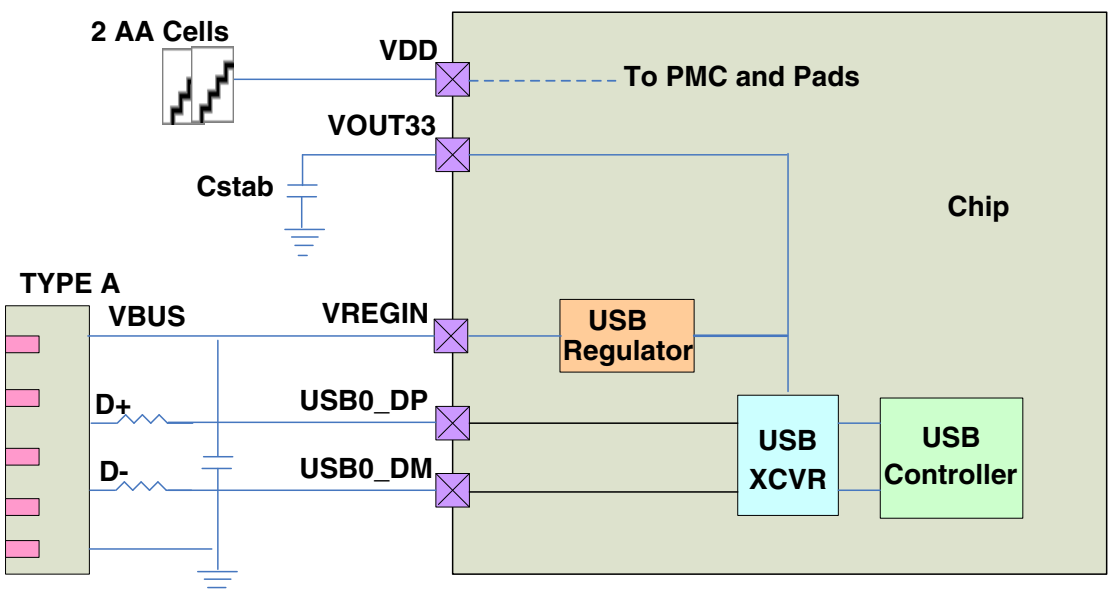


Figure 3-13. USB regulator AA cell usecase

3.9.1.2.2 Li-Ion battery power supply

The chip can also be powered by a single Li-ion battery. In this case, VOUT33 is connected to VDD. The USB regulator must be enabled by default to power the MCU. When connected to a USB host, the input source of this regulator is switched to the USB bus supply from the Li-ion battery. To charge the battery, the MCU can configure the battery charger according to the charger detection information.

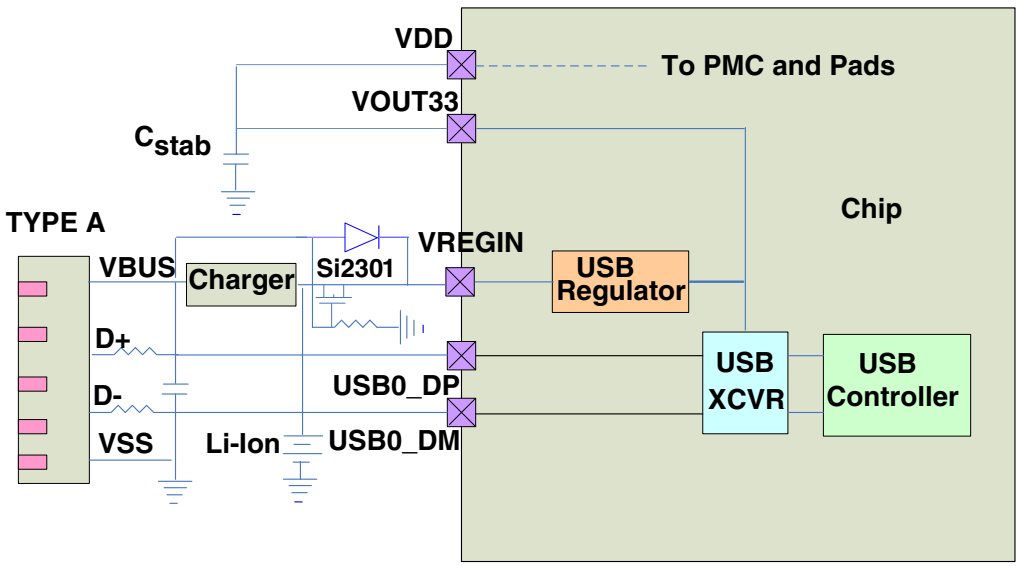


Figure 3-14. USB regulator Li-ion usecase

3.9.1.2.3 USB bus power supply

The chip can also be powered by the USB bus directly. In this case, VOUT33 is connected to VDD. The USB regulator must be enabled by default to power the MCU, then to power USB transceiver or external sensor.

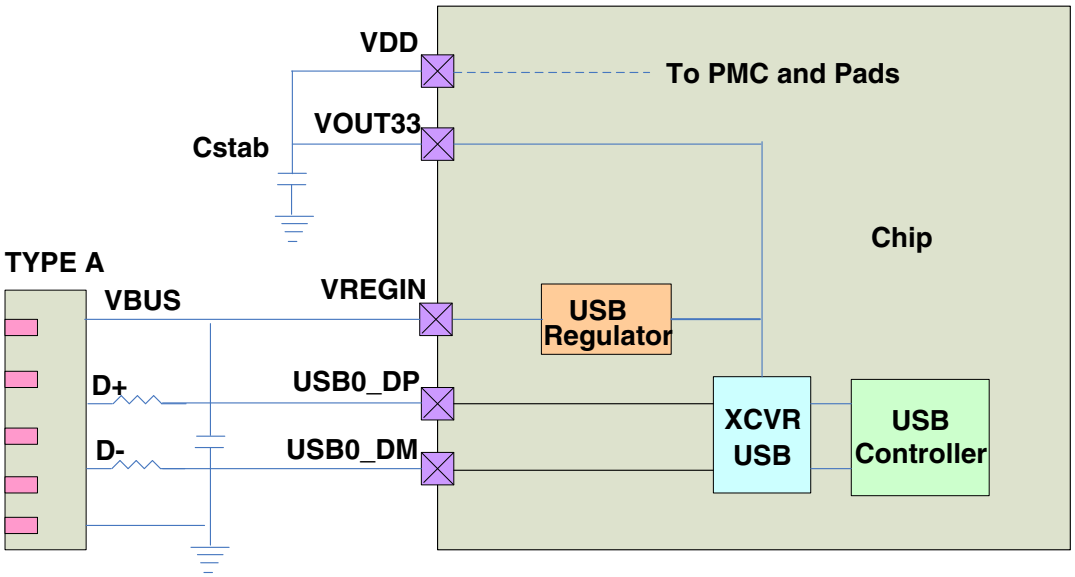


Figure 3-15. USB regulator bus supply

3.9.1.3 USB power management

The regulator should be put into STANDBY mode whenever the chip is in Stop mode and the USB bus is not in use. The regulator must remain in RUN mode for all USB communication including Suspend, Resume, and Reset signaling.

3.9.1.4 VBUS detect

The USB FS controller does not include a dedicated VBUS detect pin. However, a GPIO pin with interrupt capability can be used to detect the presence of VBUS in device mode. If a GPIO pin with LLWU capability is used, VBUS can be detected in all low power modes, including LLS and VLLS.

Software is responsible for detecting VBUS via a GPIO pin and configuring the USB controller appropriately in response to the rising or falling edge of VBUS.

NOTE

An appropriately sized resistive voltage divider must be used if the GPIO pin does not directly support a 5V input.

3.9.1.5 USB controller configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

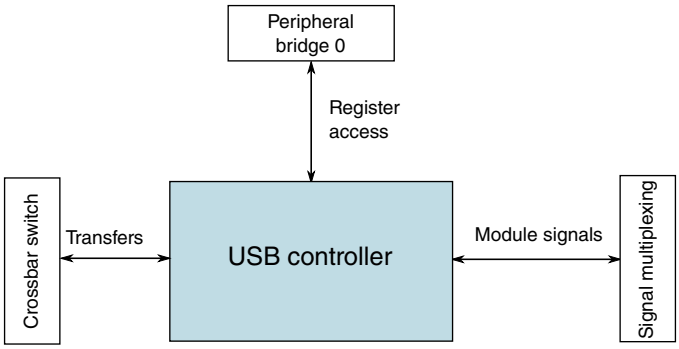


Figure 3-16. USB controller configuration

Table 3-22. Reference links to related information

Topic	Related module	Reference
Full description	USB controller	USB controller
System memory map		System memory map
Clocking		Clock Distribution
Transfers	Crossbar switch	Crossbar switch
Signal Multiplexing	Port control	Signal Multiplexing

NOTE

When USB is not used in the application, it is recommended that the USB regulator VREGIN and VOUT33 pins remain floating.

3.9.1.6 USB Voltage Regulator Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

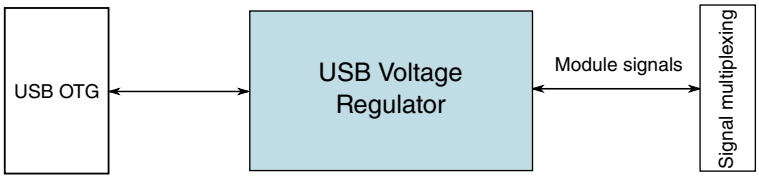


Figure 3-17. USB Voltage Regulator configuration

Table 3-23. Reference links to related information

Topic	Related module	Reference
Full description	USB Voltage Regulator	USB Voltage Regulator
System memory map		System memory map
Clocking		Clock Distribution
	USB controller	USB controller
Signal Multiplexing	Port control	Signal Multiplexing

NOTE

When USB is not used in the application, it is recommended that the USB regulator VREGIN and VOUT33 pins remain floating.

3.9.1.7 USB SRAM

In all VLLS power modes, USB SRAM is NOT retained, but in all the other power modes USB SRAM is retained. As a result, USB SRAM cannot be used in all VLLS modes.

In Run modes, the USB SRAM can be used. For example, the Buffer Descriptor Table (BDT) and USB buffers can be placed in USB SRAM in Run modes.

See System Memory Map for the location of the USB SRAM.

3.9.2 LPSPI configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

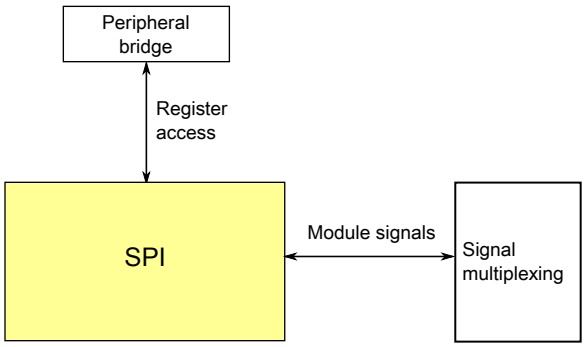


Figure 3-18. LPSPI configuration

Table 3-24. Reference links to related information

Topic	Related module	Reference
Full description	LPSPI	LPSPI
System memory map	—	System memory map
Clocking	—	Clock distribution
Signal multiplexing	Port control	Signal multiplexing

3.9.3 LPI2C

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

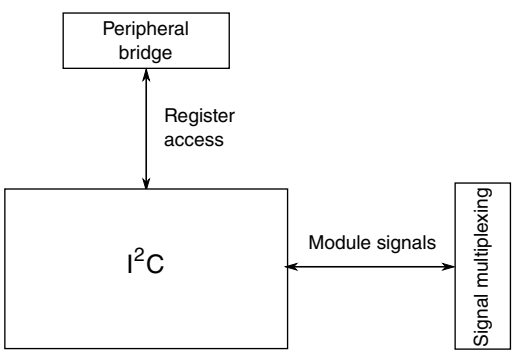


Figure 3-19. LPI2C configuration

Table 3-25. Reference links to related information

Topic	Related module	Reference
Full description	LPI2C	LPI2C
System memory map	—	System memory map
Clocking	—	Clock distribution
Signal multiplexing	Port control	Signal multiplexing

3.9.3.1 LPI2C instantiation information

This device has three LPI2C modules. The LPI2C module provides a low power IIC module that can operate in low power stop modes if required.

Each of the three LPI2C modules will have a 4 word (32-bit) FIFO for transmit and receive of messages.

3.9.4 LPUART configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

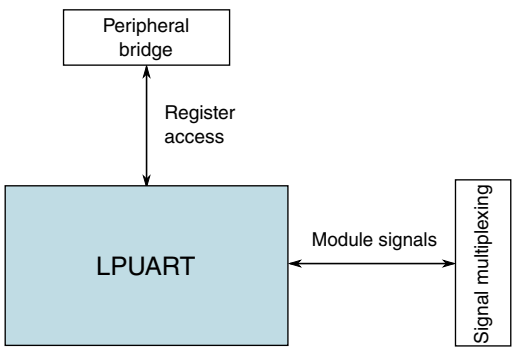


Figure 3-20. LPUART configuration

Table 3-26. Reference links to related information

Topic	Related module	Reference
Full description	LPUART	LPUART
System memory map	—	System memory map
Clocking	—	Clock distribution
Power management	—	Power management
Signal multiplexing	Port control	Signal multiplexing

3.9.4.1 LPUART overview

The LPUART modules support the basic UART with DMA interface function and x4 to x32 oversampling of baud-rate.

This module supports LIN slave operation.

The module can remain functional in VLPS mode provided the clock it is using remains enabled.

3.9.5 EMVSIM Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

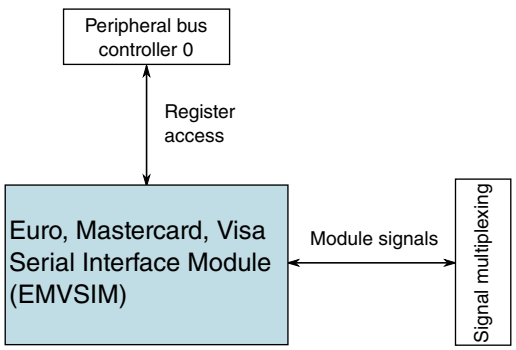


Figure 3-21. EMVSIM configuration

3.9.5.1 Overview

The EMVSIM (Euro/Mastercard/Visa/SIM Serial Interface Module) is a standalone ISO 7816 module that is connected to the AIPS0 Peripheral Bridge. The EMVSIM module's clock source is the CPU/platform clock.

3.9.6 FlexIO

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

Table 3-27. Reference links to related information

Topic	Related module	Reference
Full description	FlexIO	FlexIO
System memory map	—	System memory map
Clocking	—	Clock distribution
Trigger Input Selection	TRGMUX	TRGMUX
Signal multiplexing	Port control	Signal multiplexing

3.9.6.1 FlexIO overview

The FlexIO module can remain functional in VLPS mode provided the clock it is using remains enabled.

3.9.7 I²S configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

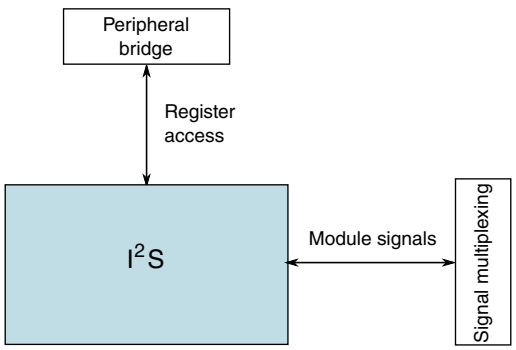


Figure 3-22. I²S configuration

Table 3-28. Reference links to related information

Topic	Related module	Reference
Full description	I²S	I2S
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.9.7.1 Instantiation information

This device contains one I²S module.

As configured on the device, module features include:

- TX data lines: 1
- RX data lines: 1
- FIFO size (words): 4
- Maximum words per frame: 16
- Maximum bit clock divider: 512

3.9.7.2 I²S/SAI clocking

3.9.7.2.1 Audio Master Clock

The audio master clock (MCLK) is used to generate the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The audio master clock can also be output to or input from a pin. The transmitter and receiver have the same audio master clock inputs.

3.9.7.2.2 Bit Clock

The I²S/SAI transmitter and receiver support asynchronous bit clocks (BCLKs) that can be generated internally from the audio master clock or supplied externally. The module also supports the option for synchronous operation between the receiver and transmitter or between two separate I²S/SAI peripherals.

3.9.7.2.3 Bus Clock

The bus clock is used by the control registers and to generate synchronous interrupts and DMA requests.

3.9.7.2.4 I²S/SAI clock generation

The master clock selection/direction is configured using the PCC.

3.9.7.3 I²S/SAI operation in low power modes

3.9.7.3.1 Very low power modes

In VLPS mode, the module behaves as it does in stop mode if VLPS mode is entered from run mode. However, if VLPS mode is entered from VLPR mode, then the FIFO might underflow or overflow before wakeup from stop mode due to the limits in bus bandwidth.

In VLPW and VLPR modes, the module is limited by the maximum bus clock frequencies.

3.10 Human-machine interfaces (HMI)

3.10.1 GPIO configuration

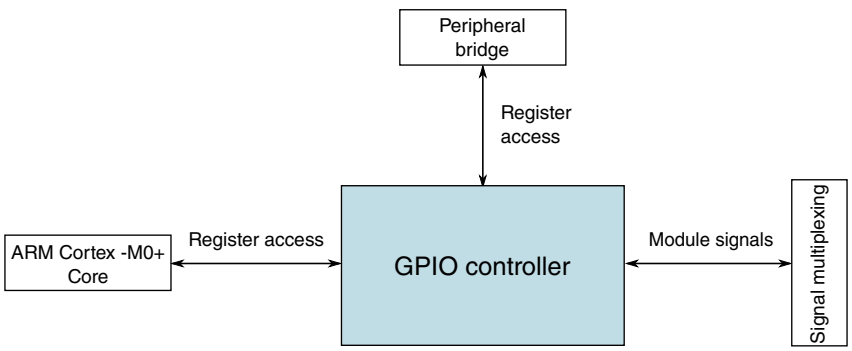


Figure 3-23. GPIO configuration

Table 3-29. Reference links to related information

Topic	Related module	Reference
Full description	GPIO	GPIO
System memory map	—	System memory map
Clocking	—	Clock distribution
Power management	—	Power management
Crossbar switch	Crossbar switch	Crossbar switch
Signal multiplexing	Port control	Signal multiplexing

3.10.1.1 GPIO instantiation information

The device includes pins PTB0, PTB1, PTC3, PTC4, PTD4, PTD5, PTD6, and PTD7 with high current drive capability. These pins can be used to drive LED or power MOSFETs directly. The high drive capability applies to all functions which are multiplexed on these pins.

3.10.1.1.1 Pull devices and directions

The pull devices are enabled out of POR only for RESET_b, NMI_b, and the SWD signals. Coming out of POR:

- RESET_b, NMI_b and SWD_DIO are pulled up.
- SWD_CLK is pulled down.

The pull devices for other pins are not enabled out of POR. They can be enabled by writing to PORTx_PCRn[PE].

All the pins have controllable pull direction using the PORTx_PCRn[PS] field. All the pins default to pullup except for SWD_CLK, when enabled.

3.10.1.2 GPIO accessibility in the memory map

The GPIO is multi-ported and can be accessed directly by the core with zero wait states at base address 0xF800_0000. It can also be accessed by the core and DMA masters through the cross bar/AIPS interface address 0x4000_F000. All BME operations to the GPIO space can be accomplished referencing the address 0x4000_F000.

3.10.2 TSI configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

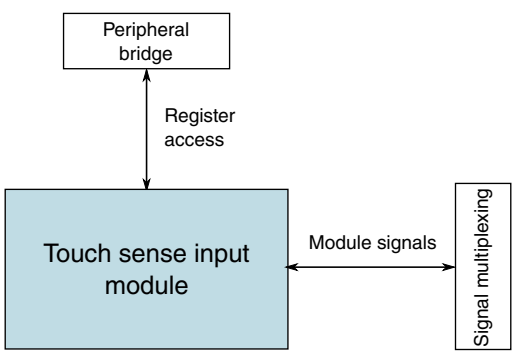


Figure 3-24. TSI configuration

Table 3-30. Reference links to related information

Topic	Related module	Reference
Full description	TSI	TSI
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal Multiplexing	Port control	Signal multiplexing

3.10.2.1 TSI instantiation information

This device includes one TSI module containing the channels as shown in the following table. In Stop, VLPS, LLS, and VLLSx modes, any one channel can be enabled to be the wake-up source.

TSI hardware trigger is from the LPTMR0. For more about the LPTMR0 module interconnects, see the Module-to-Module section.

The number of TSI channels present on the device is determined by the pinout of the specific device package, and is shown in the following table.

Table 3-31. Number of TSI channels

Device	TSI channels
100 LQFP, 121 XFBGA[1]	16

1. The 121-pin XFBGA package for this product is not yet available. However, it is included in a Package Your Way program for Kinetis MCUs. Visit nxp.com/KPYW for more details.

3.10.2.2 TSI Interrupts

The TSI has multiple sources of interrupt requests. However, these sources are OR'd together to generate a single interrupt request. When a TSI interrupt occurs and you want to determine the exact interrupt source, read the TSI status register.

3.11 Signal multiplexing integration

3.11.1 Signal multiplexing configuration

Information found here summarizes how the module is integrated into the device. For a comprehensive description of the module itself, see the module's dedicated chapter.

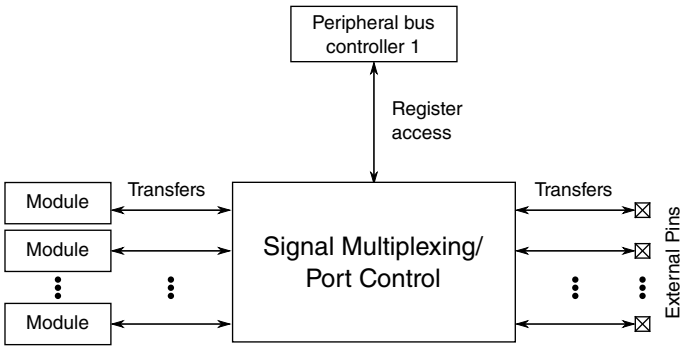


Figure 3-25. Signal multiplexing integration

Table 3-32. Reference links to related information

Topic	Related module	Reference
Full description	Port control	Port control
System memory map		System memory map
Clocking		Clock Distribution
Register access	Peripheral bus controller	Peripheral bridge

3.11.1.1 Port control and interrupt summary

The following table provides more information regarding the Port Control and Interrupt configurations .

Table 3-33. Ports summary

Feature	Port A	Port B	Port C	Port D	Port E
Pull select control	Yes	Yes	Yes	Yes	Yes
Pull select after reset	PTA0=Pull down, Others=Pull up	Pull up	Pull up	Pull up	Pull up
Pull enable control	Yes	Yes	Yes	Yes	Yes
Pull enable after reset	PTA0/PTA3/PTA4/ RESET_b=Enabled ; Others=Disabled	Disabled	Disabled	Disabled	Disabled
Slew rate enable control	Yes	Yes	Yes	Yes	Yes
Slew rate enable after reset	PTA3=Disabled; Others=Enabled	Enabled	Enabled	Enabled	Enabled
Passive filter enable control	PTA4 and PTA20/ RESET_b only	PTB19 only	No	No	No
Passive filter enable after reset	RESET_b=Enabled ; Others=Disabled	Disabled	Disabled	Disabled	Disabled

Table continues on the next page...

Table 3-33. Ports summary (continued)

Feature	Port A	Port B	Port C	Port D	Port E
Open drain enable control	Yes	Yes	Yes	Yes	Yes
Open drain enable after reset	Disabled	Disabled	Disabled	Disabled	Disabled
Drive strength enable control	No	PTB0/PTB1 only	PTC3/PTC4 only	PTD4/PTD5/PTD6/ PTD7 only	No
Drive strength enable after reset	Disabled	Disabled	Disabled	Disabled	Disabled
Pin mux control	Yes	Yes	Yes	Yes	Yes
Pin mux after reset	PTA0/PTA3/ PTA4=ALT7; Others=ALT0	ALT0	ALT0	ALT0	ALT0
Lock bit	Yes	Yes	Yes	Yes	Yes
Interrupt and DMA request	Yes	Yes	Yes	Yes	Yes
Digital glitch filter	No	No	No	No	No

3.11.1.2 Clock gating

See the Clock distribution chapter.

3.11.1.3 Signal multiplexing constraints

1. A given peripheral function must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
2. To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.

Chapter 4
Memory Map

4.1 Memory Map

This device contains various memories and memory-mapped peripherals which are located in a 4 GB memory space.

This chapter describes the memory and peripheral locations within that memory space.

4.2 SRAM sizes

This device contains 128 KB SRAM that can be accessed by bus masters through the cross-bar switch.

The amount of System SRAM retained for the different power modes is shown in the following table.

Table 4-1. System SRAM Retained in Power Modes

Power Mode	System SRAM Retained(KB)
VLLS2/LLS2	64 KB located at 0x2000_0000-0x2000_FFFF
VLLS0/VLLS1	None - All System SRAM powered down
All other power modes	All 128 KB retained

4.3 System Memory Map

There is 1 memory map:

- Memory Map

4.3.1 Memory Map

Table 4-2. Memory Map

Type	START ADDRESS	END ADDRESS	Function
Code	0x0000_0000	0x0007_FFFF	Program Flash
	0x0008_0000	0x1CFF_FFFF	Reserved
	0x1C00_0000	0x1C00_7FFF	Boot ROM
	0x1C00_8000	0x1CFF_FFFF	Reserved
	0x1D00_0000	0x1D01_FFFF	Reserved
	0x1D02_0000	0x1D03_FFFF	Reserved
	0x1D04_0000	0x1D1F_FFFF	Reserved
	0x1D20_0000	0x1D21_FFFF	Reserved
	0x1D22_0000	0x1D23_FFFF	Reserved
	0x1D24_0000	0x1FFF_7FFF	Reserved
	0x1FFF_8000	0x1FFF_9FFF	SRAM
	0x1FFF_A000	0x1FFF_FFFF	SRAM
SRAM	0x2000_0000	0x2001_1FFF	SRAM
	0x2001_2000	0x2001_7FFF	SRAM
	0x2001_8000	0x2CFF_FFFF	Reserved
	0x2D00_0000	0x2D01_FFFF	Reserved
	0x2D02_0000	0x2D03_FFFF	Reserved
	0x2D04_0000	0x2D0F_FFFF	Reserved
	0x2D10_0000	0x2D10_7FFF	Reserved
	0x2D10_4000	0x2D1F_FFFF	Reserved
	0x2D20_0000	0x2D21_FFFF	Reserved
	0x2D22_0000	0x2D23_FFFF	Reserved
	0x2D24_0000	0x2D2F_FFFF	Reserved
	0x2D30_0000	0x2D30_7FFF	Reserved
	0x2D30_4000	0x3FFF_FFFF	Reserved
Peripheral	0x4000_0000	0x4007_FFFF	AIPS0
	0x4008_0000	0x400F_FFFF	AIPS1
	0x4010_0000	0x4107_FFFF	USB SRAM
	0x4108_0000	0x410F_FFFF	Reserved
	0x4110_0000	0x4117_FFFF	Reserved
	0x4118_0000	0x411F_FFFF	Reserved
	0x4120_0000	0x43FF_FFFF	Reserved
	0x4400_0000	0x5FFF_FFFF	BME (AIPS0 and AIPS1) (448 MB)
External RAM	0x6000_0000	0xDFFF_FFFF	Reserved
	0xE000_0000	0xE0FF_FFFF	PPB - ARM SYS Modules
	0xE100_0000	0xE1FF_FFFF	

Table continues on the next page...

Table 4-2. Memory Map (continued)

Type	START ADDRESS	END ADDRESS	Function
	0xE200_0000	0xE2FF_FFFF	
	0xE300_0000	0xE3FF_FFFF	
System	0xE400_0000	0xE4FF_FFFF	PPB - ARM DBG Modules
	0xE500_0000	0xE5FF_FFFF	
	0xE600_0000	0xE6FF_FFFF	
	0xE700_0000	0xE7FF_FFFF	
	0xE800_0000	0xE8FF_FFFF	PPB - Freescale Modules
	0xE900_0000	0xE9FF_FFFF	
	0xEA00_0000	0xEAFF_FFFF	
	0xEB00_0000	0xEBFF_FFFF	
	0xEC00_0000	0xEFFF_FFFF	Reserved
	0xF000_0000	0xF00F_FFFF	PPB
	0xF010_0000	0xF0FF_FFFF	Reserved
	0xF100_0000	0xF10F_FFFF	Reserved
	0xF110_0000	0xF11F_FFFF	Reserved
	0xF120_0000	0xF7FF_FFFF	Reserved
	0xF800_0000	0xF80F_FFFF	SYS IOPORT
	0xF810_0000	0xF8FF_FFFF	Reserved
	0xF900_0000	0xF90F_FFFF	Reserved
	0xF910_0000	0xFFFF_FFFF	Reserved

4.4 Flash Memory Maps

There is 1 flash memory map:

- Flash Memory Map

4.4.1 Flash Memory Map

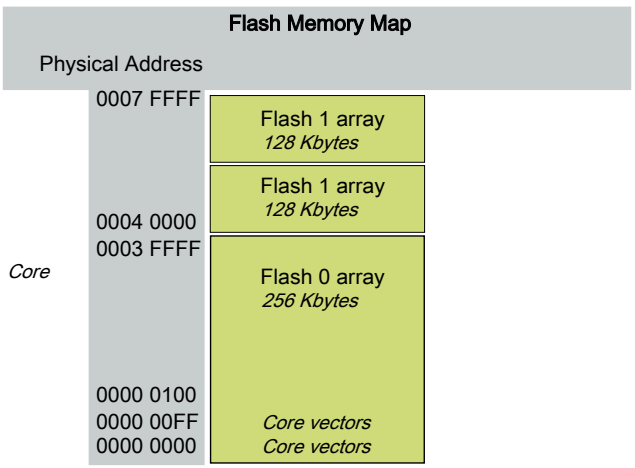


Figure 4-1. Flash Memory Map

4.5 SRAM Memory Map

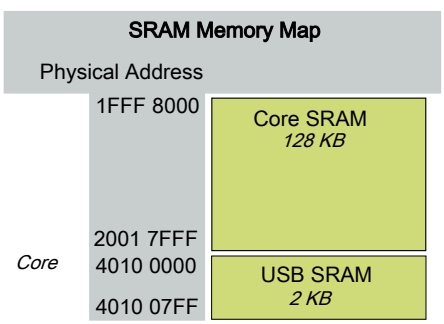


Figure 4-2. SRAM Memory Map

4.6 Bit Manipulation Engine

The Bit Manipulation Engine (BME2) provides hardware support for atomic read-modify-write memory operations to the peripheral address space.

By combining the basic load and store instruction support in the Cortex-M instruction set architecture with the concept of decorated storage provided by the BME2, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. See the Bit Manipulation Engine Block Guide (BME2) for a detailed description of BME2 functionality.

4.7 Peripheral bridge (AIPS-Lite) memory map

The peripheral memory map is accessible via 2 slave ports that are on a crossbar.

- AIPS0 is on crossbar AXBS.
- AIPS1 is on crossbar AXBS.

Each AIPSx bridge has 2 regions associated with this space:

- A 128 KB region: partitioned as 32 spaces, each 4 KB in size and reserved for on-platform peripheral devices. The AIPS controller generates unique module enables for all 32 spaces.
- A 384 KB region: partitioned as 96 spaces, each 4 KB in size and reserved for off-platform modules. The AIPS controller generates unique module enables for all 96 spaces.

Modules that are disabled (via their clock gate control bits in the PCC registers) disable the associated AIPS slots. Access to any address within an unimplemented or disabled peripheral bridge slot causes a transfer error termination.

For programming model accesses via the peripheral bridges, there is generally only a small range within the 4 KB slots that is implemented. Accessing an address that is not implemented in the peripheral causes a transfer error termination.

4.7.1 AIPS0 Peripheral Slot Assignments

Table 4-3. AIPS0

AIPS0 (no overlap of peripherals)		
System 32-bit base address	**Slot**	**Source Module**
0x4000_0000	0	Reserved
0x4000_1000	1	Miscellaneous System Control Module (MSCM)
0x4000_2000	2	Reserved
0x4000_3000	3	Reserved
0x4000_4000	4	Reserved
0x4000_5000	5	Reserved

Table continues on the next page...

Table 4-3. AIPS0 (continued)

AIPS0 (no overlap of peripherals)		
System 32-bit base address	**Slot**	**Source Module**
0x4000_6000	6	Reserved
0x4000_7000	7	Reserved
0x4000_8000	8	DMA0 controller
0x4000_9000	9	DMA0 controller transfer control descriptors
0x4000_A000	10	Reserved
0x4000_B000	11	Reserved
0x4000_C000	12	Reserved
0x4000_D000	13	Reserved
0x4000_E000	14	Reserved
0x4000_F000	15	RGPIO controller (GPIOA-GPIOE)
0x4001_0000	16	Reserved
0x4001_1000	17	Reserved
0x4001_2000	18	Reserved
0x4001_3000	19	Reserved
0x4001_4000	20	Reserved
0x4001_5000	21	Reserved
0x4001_6000	22	Reserved
0x4001_7000	23	Reserved
0x4001_8000	24	Reserved
0x4001_9000	25	Reserved
0x4001_A000	26	Reserved
0x4001_B000	27	Reserved
0x4001_C000	28	Reserved
0x4001_D000	29	Reserved
0x4001_E000	30	Reserved
0x4001_F000	31	Reserved
Off-platform		
0x4002_0000	32	Flash Memory Unit
0x4002_1000	33	DMA0 channel multiplexer 0 (DMAMUX0)
0x4002_2000	34	Reserved
0x4002_3000	35	Reserved
0x4002_4000	36	INTMUX0
0x4002_5000	37	Reserved
0x4002_6000	38	Reserved
0x4002_7000	39	TRGMUX
0x4002_8000	40	Reserved
0x4002_9000	41	Reserved

Table continues on the next page...

Table 4-3. AIPS0 (continued)

AIPS0 (no overlap of peripherals)		
System 32-bit base address	**Slot**	**Source Module**
0x4002_A000	42	Reserved
0x4002_B000	43	Reserved
0x4002_C000	44	Reserved
0x4002_D000	45	Reserved
0x4002_E000	46	TPM2
0x4002_F000	47	Reserved
0x4003_0000	48	LPIT0
0x4003_1000	49	Reserved
0x4003_2000	50	Reserved
0x4003_3000	51	Reserved
0x4003_4000	52	Low-power Timer 0 (LPTMR0)
0x4003_5000	53	Reserved
0x4003_6000	54	Reserved
0x4003_7000	55	Reserved
0x4003_8000	56	Real-time Clock (RTC)
0x4003_9000	57	Reserved
0x4003_A000	58	Reserved
0x4003_B000	59	Reserved
0x4003_C000	60	Reserved
0x4003_D000	61	Reserved
0x4003_E000	62	LPSPI2
0x4003_F000	63	Reserved
0x4004_0000	64	Reserved
0x4004_1000	65	Reserved
0x4004_2000	66	LPI2C2
0x4004_3000	67	Reserved
0x4004_4000	68	Reserved
0x4004_5000	69	Reserved
0x4004_6000	70	LPUART2
0x4004_7000	71	Reserved
0x4004_8000	72	Reserved
0x4004_9000	73	Reserved
0x4004_A000	74	Reserved
0x4004_B000	75	Reserved
0x4004_C000	76	SAI0
0x4004_D000	77	Reserved
0x4004_E000	78	EVMSIM0
0x4004_F000	79	Reserved

Table continues on the next page...

Table 4-3. AIPS0 (continued)

AIPS0 (no overlap of peripherals)		
System 32-bit base address	Slot	Source Module
0x4005_0000	80	Reserved
0x4005_1000	81	Reserved
0x4005_2000	82	Reserved
0x4005_3000	83	Reserved
0x4005_4000	84	Reserved
0x4005_5000	85	USB0 FS
0x4005_6000	86	Reserved
0x4005_7000	87	Reserved
0x4005_8000	88	Reserved
0x4005_9000	89	Reserved
0x4005_A000	90	PortA multiplex control
0x4005_B000	91	PortB multiplex control
0x4005_C000	92	PortC multiplex control
0x4005_D000	93	PortD multiplex control
0x4005_E000	94	PortE multiplex control
0x4005_F000	95	Reserved
0x4006_0000	96	Reserved
0x4006_1000	97	Low-leakage Wake-up Unit 0 (LLWU0)
0x4006_2000	98	TSI0
0x4006_3000	99	Reserved
0x4006_4000	100	Reserved
0x4006_5000	101	Reserved
0x4006_6000	102	ADC0
0x4006_7000	103	Reserved
0x4006_8000	104	Reserved
0x4006_9000	105	Reserved
0x4006_A000	106	DAC0
0x4006_B000	107	Reserved
0x4006_C000	108	Reserved
0x4006_D000	109	Reserved
0x4006_E000	110	CMP0
0x4006_F000	111	Reserved
0x4007_0000	112	Reserved
0x4007_1000	113	Reserved
0x4007_2000	114	VREF
0x4007_3000	115	Reserved
0x4007_4000	116	SIM
0x4007_5000	117	SIM / TSTMR0

Table continues on the next page...

Table 4-3. AIPS0 (continued)

AIPS0 (no overlap of peripherals)		
System 32-bit base address	**Slot**	**Source Module**
0x4007_6000	118	WDOG0
0x4007_7000	119	Reserved
0x4007_8000	120	CRC
0x4007_9000	121	Reserved
0x4007_A000	122	PCC
0x4007_B000	123	SCG
0x4007_C000	124	System Register File
0x4007_D000	125	Power Management Controller (PMC)
0x4007_E000	126	System Mode Controller (SMC)
0x4007_F000	127	Reset Control Module (RCM)

4.7.2 AIPS1 Peripheral Slot Assignments

Table 4-4. AIPS1 Peripheral Slot Assignments

AIPS1		
System 32-bit base address	**Slot**	**Source Module**
0x4008_0000	0	Reserved
0x4008_1000	1	Reserved
0x4008_2000	2	Reserved
0x4008_3000	3	Reserved
0x4008_4000	4	Reserved
0x4008_5000	5	Reserved
0x4008_6000	6	Reserved
0x4008_7000	7	Reserved
0x4008_8000	8	Reserved
0x4008_9000	9	Reserved
0x4008_A000	10	Reserved
0x4008_B000	11	Reserved
0x4008_C000	12	Reserved
0x4008_D000	13	Reserved
0x4008_E000	14	Reserved
0x4008_F000	15	Reserved
0x4009_0000	16	Reserved
0x4009_1000	17	Reserved
0x4009_2000	18	Reserved
0x4009_3000	19	Reserved

Table continues on the next page...

Table 4-4. AIPS1 Peripheral Slot Assignments (continued)

AIPS1		
System 32-bit base address	**Slot**	**Source Module**
0x4009_4000	20	Reserved
0x4009_5000	21	Reserved
0x4009_6000	22	Reserved
0x4009_7000	23	Reserved
0x4009_8000	24	Reserved
0x4009_9000	25	Reserved
0x4009_A000	26	Reserved
0x4009_B000	27	Reserved
0x4009_C000	28	Reserved
0x4009_D000	29	Reserved
0x4009_E000	30	Reserved
0x4009_F000	31	Reserved
Off-platform		
0x400A_0000	32	Reserved
0x400A_1000	33	Reserved
0x400A_2000	34	Reserved
0x400A_3000	35	Reserved
0x400A_4000	36	Reserved
0x400A_5000	37	Random Number Generator (TRNG)
0x400A_6000	38	Reserved for Crypto
0x400A_7000	39	TRGMUX
0x400A_8000	40	Reserved
0x400A_9000	41	Reserved
0x400A_A000	42	Reserved
0x400A_B000	43	Reserved
0x400A_C000	44	TPM0
0x400A_D000	45	TPM1
0x400A_E000	46	Reserved
0x400A_F000	47	Reserved
0x400B_0000	48	Reserved
0x400B_1000	49	Reserved
0x400B_2000	50	Reserved
0x400B_3000	51	Reserved
0x400B_4000	52	Reserved
0x400B_5000	53	Low-power Timer 1 (LPTMR1)
0x400B_6000	54	Reserved
0x400B_7000	55	Reserved

Table continues on the next page...

Table 4-4. AIPS1 Peripheral Slot Assignments (continued)

AIPS1		
System 32-bit base address	Slot	Source Module
0x400B_8000	56	Reserved
0x400B_9000	57	Reserved
0x400B_A000	58	Reserved
0x400B_B000	59	Reserved
0x400B_C000	60	LPSPI0
0x400B_D000	61	LPSPI1
0x400B_E000	62	Reserved
0x400B_F000	63	Reserved
0x400C_0000	64	LPI2C0
0x400C_1000	65	LPI2C1
0x400C_2000	66	Reserved
0x400C_3000	67	Reserved
0x400C_4000	68	LPUART0
0x400C_5000	69	LPUART1
0x400C_6000	70	Reserved
0x400C_7000	71	Reserved
0x400C_8000	72	Reserved
0x400C_9000	73	Reserved
0x400C_A000	74	FlexIO0
0x400C_B000	75	Reserved
0x400C_C000	76	Reserved
0x400C_D000	77	Reserved
0x400C_E000	78	Reserved
0x400C_F000	79	Reserved
0x400D_0000	80	Reserved
0x400D_1000	81	Reserved
0x400D_2000	82	Reserved
0x400D_3000	83	Reserved
0x400D_4000	84	Reserved
0x400D_5000	85	Reserved
0x400D_6000	86	Reserved
0x400D_7000	87	Reserved
0x400D_8000	88	Reserved
0x400D_9000	89	Reserved
0x400D_A000	90	Reserved
0x400D_B000	91	Reserved
0x400D_C000	92	Reserved

Table continues on the next page...

Table 4-4. AIPS1 Peripheral Slot Assignments (continued)

AIPS1		
System 32-bit base address	Slot	Source Module
0x400D_D000	93	Reserved
0x400D_E000	94	Reserved
0x400D_F000	95	Reserved
0x400E_0000	96	Reserved
0x400E_1000	97	Reserved
0x400E_2000	98	Reserved
0x400E_3000	99	Reserved
0x400E_4000	100	Reserved
0x400E_5000	101	Reserved
0x400E_6000	102	Reserved
0x400E_7000	103	Reserved
0x400E_8000	104	Reserved
0x400E_9000	105	Reserved
0x400E_A000	106	Reserved
0x400E_B000	107	Reserved
0x400E_C000	108	Reserved
0x400E_D000	109	Reserved
0x400E_E000	110	Reserved
0x400E_F000	111	CMP1
0x400F_0000	112	Reserved
0x400F_1000	113	Reserved
0x400F_2000	114	Reserved
0x400F_3000	115	Reserved
0x400F_4000	116	Reserved
0x400F_5000	117	Reserved
0x400F_6000	118	Reserved
0x400F_7000	119	Reserved
0x400F_8000	120	Reserved
0x400F_9000	121	Reserved
0x400F_A000	122	PCC
0x400F_B000	123	Reserved
0x400F_C000	124	Reserved
0x400F_D000	125	Reserved
0x400F_E000	126	Reserved
0x400F_F000	127	Reserved

Chapter 5
Clock Distribution

5.1 Introduction

This chapter provides an overview of the clock architecture: clock usage, clock multiplexing, and clock gating.

The heart of the clocking architecture is the System Clock Generator (SCG) module. The SCG controls four clock sources that can then be distributed to the main core platform, the memory modules and the peripherals. The peripherals in general have two clocks, one being the peripheral interface clock which is used by the core/DMA to interface with the peripheral's registers, the second being the peripheral functional clock which is used for the main timing function of the peripheral (e.g. sources the baud rate for a serial communications peripheral, or is the input clock to the counter of a timer peripheral).

The peripheral interface clock generally comes directly from the SCG to the peripheral. The second peripheral functional clock is selected via the Peripheral Clock Control (PCC) module. The SCG provides additional peripheral functional clocks with optional dividers via the PCC module

The following diagram shows the various clock sources and clock trees for KL28Z.

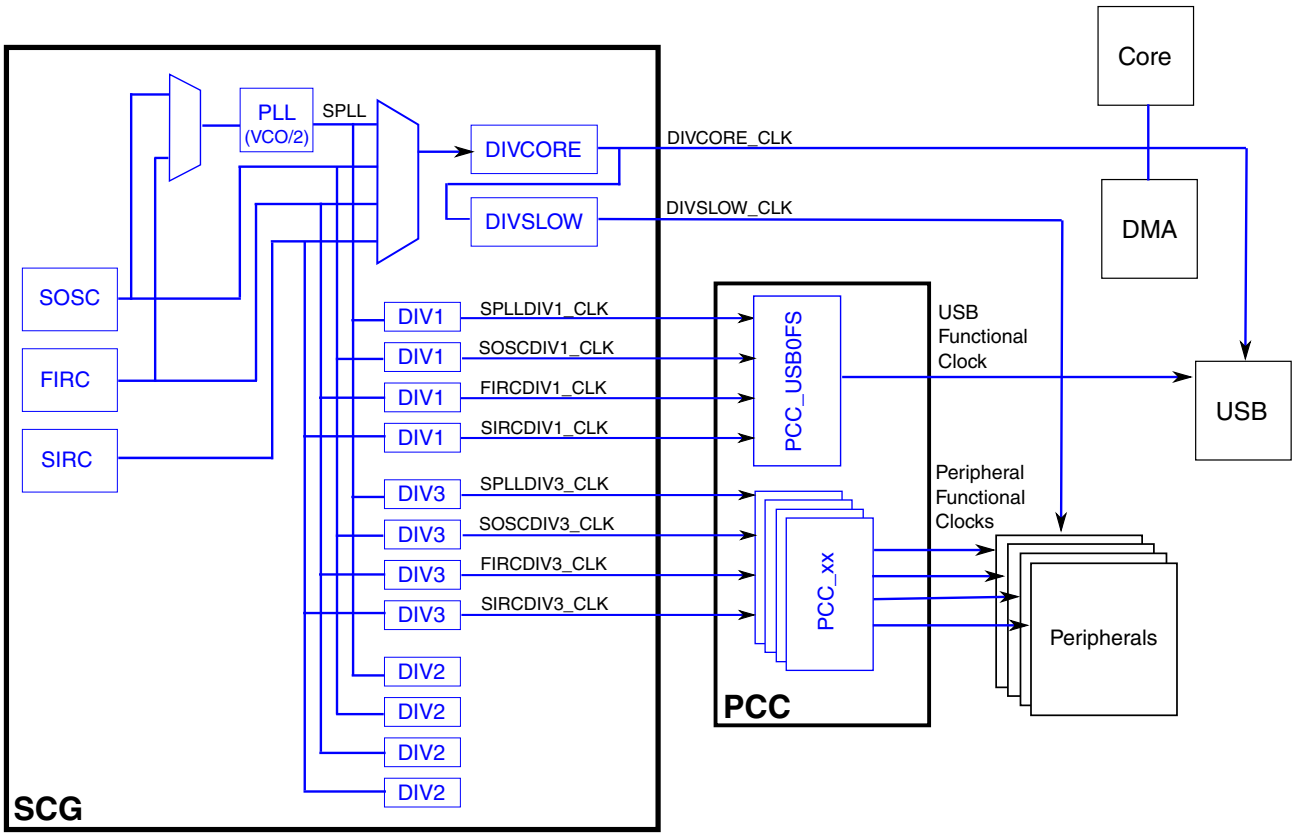


Figure 5-1. KL28Z High Level Clock Distribution

5.2 Clock Sources

The SCG provides four clock sources that are then distributed and optionally divided to the CPU, memory and various peripherals.

The four clock sources available to the SCG are:

- SOSC – output of the external oscillator (a crystal or externally applied clock input).
- SIRC – output of the slow (2/8Mhz) internal RC oscillator
- FIRC – output of the fast (48-60MHZ) internal RC oscillator
- SPLL – output of the PLL, which is multiple of the SOSC or FIRC clock source.

The configuration of these clock sources are detailed in the SCG chapter.

5.3 SCG Output Clocks

The SCG has several output clock trees. All of the four clock sources, SOSC, SIRC, FIRC and SPLL each can feed these output clock trees with optional dividers.

5.3.1 DIVCORE_CLK

The main system clock that feeds the CPU, DMA, tightly coupled memories and platform is provided via a divider called DIVCORE. The SCG can select one of the four clock sources to feed DIVCORE. This is generally the fastest clock in the system but can also run slower than peripheral timing clocks.

5.3.2 DIVSLOW_CLK

The DIVSLOW clock is used as the main “peripheral interface” clock and is referenced by the CPU and the DMA when accessing the peripheral registers.

The DIVSLOW_CLK is also used to clock the flash memory module. This clock is fed directly from DIVCORE clock, and has optional dividers to ensure the frequency of this clock is <25MHZ, for reliable flash operation.

NOTE

Throughout this manual, DIVSLOW_CLK may also be referred to as the bus clock.

5.3.3 Peripheral functional clocks

The SCG provides additional output clock trees that are used as the “peripheral functional” clocks. Peripheral functional clocks are used as the fundamental “timing” of the peripheral, providing asynchronous clocking from the CPU main clock. For example the peripheral functional clock for a timer will clock the reference counter that is used for all timings, and for a serial communications peripherals will source the baud rate for the external communication.

This allows peripherals to operate either slower or faster than the CPU platform.

Each of the four clock sources can provide three additional output clocks that are generally used as alternative peripheral functional clocks. This provides flexible clocking in a SoC, and allows groups of peripherals to be clocked by one specific functional clock,

and another group of peripherals to be clocked by another specific functional clock. E.g. a group of low power peripherals would be clocked at a lower frequency than high resolution timers or fast serial communication peripherals.

Each of these peripheral functional clocks has a programmable divider of the source input, that is controlled within the SCG. Each of the peripheral functional clocks outputs, after the SCG divider, are selectable via a peripheral's PCCn register.

5.3.3.1 SOSCDIV1_CLK

This output clock is fed from the SOSC, and is an optional functional clock for peripherals.

For KL28 the SOSCDIV1_CLK is an optional peripheral functional clock for the USB0 module only. It is selectable via the PCCUSB0 register.

5.3.3.2 SOSCDIV2_CLK

This output clock is fed from the SOSC, and is an optional functional clock for peripherals.

For KL28 the SOSCDIV2_CLK is not implemented.

5.3.3.3 SOSCDIV3_CLK

This output clock is fed from the SOSC, and is an optional functional clock for peripherals.

For KL28 the SOSCDIV3_CLK is an optional peripheral functional clock for various timers and serial communication peripherals. It is selectable via the peripherals PCCn register.

5.3.3.4 SIRCDIV1_CLK

This output clock is fed from the SIRC, and is an optional functional clock for peripherals.

For KL28 the SIRCDIV1_CLK is an optional peripheral functional clock for the USB0 module only. It is selectable via the PCCUSB0 register.

5.3.3.5 SIRCDIV2_CLK

This output clock is fed from the SIRC, and is an optional functional clock for peripherals.

For KL28 the SIRCDIV2_CLK is not implemented.

5.3.3.6 SIRCDIV3_CLK

This output clock is fed from the SIRC, and is an optional functional clock for peripherals.

For KL28 the SIRCDIV3_CLK is an optional peripheral functional clock for various timers and serial communication peripherals. It is selectable via the peripherals PCCn register.

5.3.3.7 FIRCDIV1_CLK

This output clock is fed from the FIRC, and is an optional functional clock for peripherals.

For KL28 the FIRCDIV1_CLK is an optional peripheral functional clock for the USB0 module only. It is selectable via the PCCUSB0 register.

5.3.3.8 FIRCDIV2_CLK

This output clock is fed from the FIRC, and is an optional functional clock for peripherals.

For KL28 the FIRCDIV2_CLK is not implemented.

5.3.3.9 FIRCDIV3_CLK

This output clock is fed from the FIRC, and is an optional functional clock for peripherals.

For KL28 the FIRCDIV3_CLK is an optional peripheral functional clock for various timers and serial communication peripherals. It is selectable via the peripherals PCCn register.

5.3.3.10 SPLLDIV1_CLK

This output clock is fed from the SPLL, and is an optional functional clock for peripherals.

For KL28 the SPLLDIV1_CLK is an optional peripheral functional clock for the USB0 module only. It is selectable via the PCCUSB0 register.

5.3.3.11 SPLLDIV2_CLK

This output clock is fed from the SPLL, and is an optional functional clock for peripherals.

For KL28 the FIRCDIV2_CLK is not implemented.

5.3.3.12 SPLLDIV3_CLK

This output clock is fed from the SPLL, and is an optional functional clock for peripherals.

For KL28 the SPLLDIV3_CLK is an optional peripheral functional clock for various timers and serial communication peripherals. It is selectable via the peripherals PCCn register.

5.4 Peripheral Clock Summary

The SCG module supplies peripheral interface and functional clocks to the PCC, depending on the peripheral. The peripheral interface clock can be either the DIVCORE_CLK or the DIVSLOW_CLK, depending on the peripheral. The peripheral functional clocks and the peripheral interface clocks for each module/peripheral are detailed in table below.

Table 5-1. Peripheral Clocks

Source Module	Module Reset	Bus Interface Clock	Bus Interface Clock Gating Control	Peripheral Functional Clock	PCCn Functional Clock Divider
System					
Coretex M0+ Core controller	Chip Reset	DIVCORE_CLK			
DMA controller	Chip Reset	DIVCORE_CLK	Yes		

Table continues on the next page...

Table 5-1. Peripheral Clocks (continued)

Source Module	Module Reset	Bus Interface Clock	Bus Interface Clock Gating Control	Peripheral Functional Clock	PCCn Functional Clock Divider
DMA channel multiplexer (DMAMUX)	Chip Reset	DIVSLOW_CLK	Yes		
AXBS	Chip Reset	DIVCORE_CLK			
AIPS	Chip Reset	DIVCORE_CLK			
INTMUX	Chip Reset	DIVSLOW_CLK			
LLWU	Chip Reset not VLLS	DIVSLOW_CLK		LPO	
SCG	Chip Reset	DIVSLOW_CLK			
PCC	Chip Reset	DIVSLOW_CLK			
PORT multiplex control	Chip Reset	DIVSLOW_CLK	Yes		
SIM	Chip Reset	DIVSLOW_CLK			
Power Management Controller (PMC)	POR	DIVSLOW_CLK			
System Mode Controller (SMC)	Chip Reset not VLLS	DIVSLOW_CLK			
Reset Control Module (RCM)	POR/LVD/VLLS	DIVSLOW_CLK		LPO	
Security/Integrity					
WDOG[1]	Chip Reset	DIVSLOW_CLK		LPO, ERCLK, SIRC	
TRNG	Chip Reset	DIVSLOW_CLK	Yes		
CAU	Chip Reset	DIVCORE_CLK			
CRC	Chip Reset	DIVSLOW_CLK	Yes		
Memory					
Flash Memory Unit	Early Reset	DIVSLOW_CLK	Yes		
Flash Memory Controller	Chip Reset	DIVCORE_CLK			
SRAM	Chip Reset	DIVCORE_CLK			
System Register File	POR	DIVSLOW_CLK			
Analog					
TSI	Chip Reset not VLLS	DIVSLOW_CLK	Yes		
ADC	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
VREF	Chip Reset	DIVSLOW_CLK	Yes		
DAC	Chip Reset	DIVSLOW_CLK	Yes		
CMP	Chip Reset	DIVSLOW_CLK	Yes		
Timers					

Table continues on the next page...

Table 5-1. Peripheral Clocks (continued)

Source Module	Module Reset	Bus Interface Clock	Bus Interface Clock Gating Control	Peripheral Functional Clock	PCCn Functional Clock Divider
TPM	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
LPIT	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
Low-Power Timer (LPTMR)[3]	POR/LVD	DIVSLOW_CLK	Yes	LPO, ERCLK[2], SIRC, OSC32KCLK	
Real-time Clock (RTC)[4]	POR	DIVSLOW_CLK	Yes	OSC32KCLK, LPO	
TSTMR	Chip Reset	DIVSLOW_CLK		SIRCLK @ 1 MHz	
Communications					
USB	Chip Reset	DIVCORE_CLK[5]	Yes	SCG DIV1, USB_CLKIN [6]	3-bit Divide, 1-bit Fraction [7]
LPSPI	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
LPI2C	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
LPUART	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
EMVSIM	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
SAI	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3, SAI_MCLK [8]	
FlexIO	Chip Reset	DIVSLOW_CLK	Yes	SCG DIV3	
GPIO controller	Chip Reset	DIVCORE_CLK			

1. Watchdog clock sources are selected by WDOGx_CS[CLK]
2. ERCLK is either from an external pin or from the SCG Internal OSC (SOSC) and configured with the SCG_SOSCCFG[EREFS] bit.
3. LPTMR clock sources are selected by LPTMR_PSR[PCS]
4. RTC clock sources are selected by RTC_CR[LPOS]
5. For the USB FS OTG controller to operate, the minimum required DIVCORE_CLK is 24 MHz (half of USB functional clock)
6. An additional external clock USB_CLKIN, can also selected via PCCUSB0FS, and device's PORTx MUX field
7. Additional 3bit Divide field and 1-bit Fractional field reside in PCCUSB0FS register
8. An additional external clock SAI_MCLK, can also selected via PCCSAI, and device's PORTx MUX field

5.5 DIV3 Peripheral Clocking

For KL28 some timer peripherals and serial communication peripherals make use of the peripheral functional clocks using the xxDIV3 clock trees.

The example below shows the main peripheral interface clock come from the DIVSLOW_CLK output. The PCC can then choose one of four optional peripheral functional clocks from the existing four clock sources within the SCG. The PCC can also provide inputs for clock sources external to the SCG.

Example Peripheral Clocking - TPM

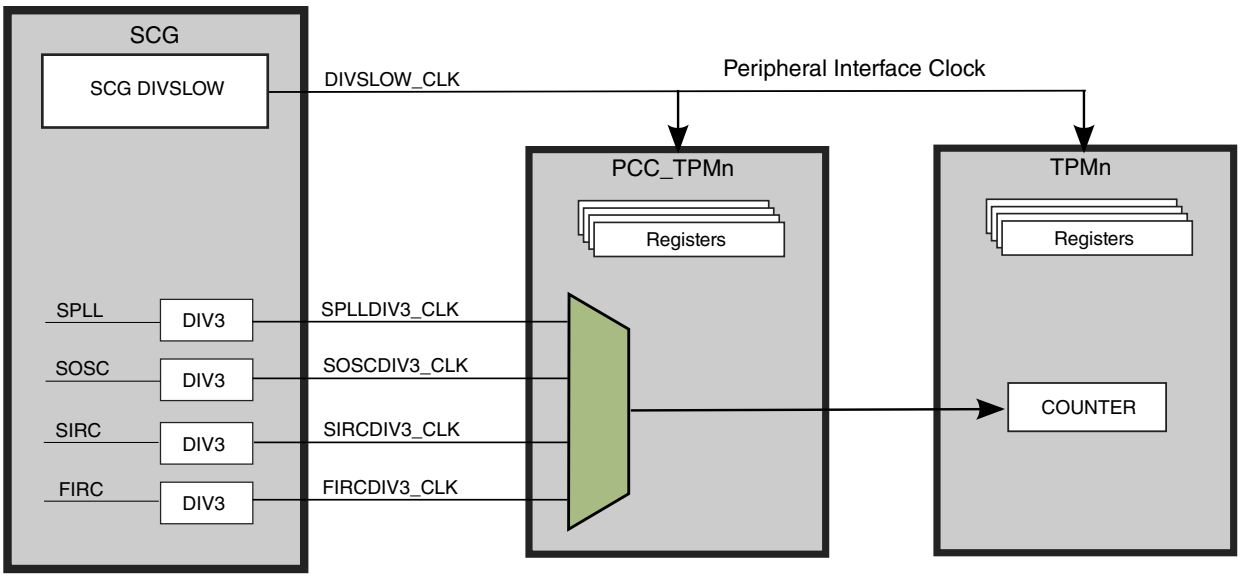


Figure 5-2. Example Peripheral Clocking (TPM)

5.6 DIV1 Peripheral Clocking

For KL28 only the USB0 module makes use of the *xx*DIV1 peripheral functional clocks. This allows the USB0 module to have its functional clock frequency independent of other timer and serial communication peripherals.

The example below shows the main Peripheral interface clock come from the DIVCORE_CLK output providing a direct 1:1 interface to the CPU. The PCC can then choose one of four optional clocks from the existing four clock sources within the SCG. The PCC can also provide inputs for clock sources external to the SCG.

Example Peripheral Clocking - USB

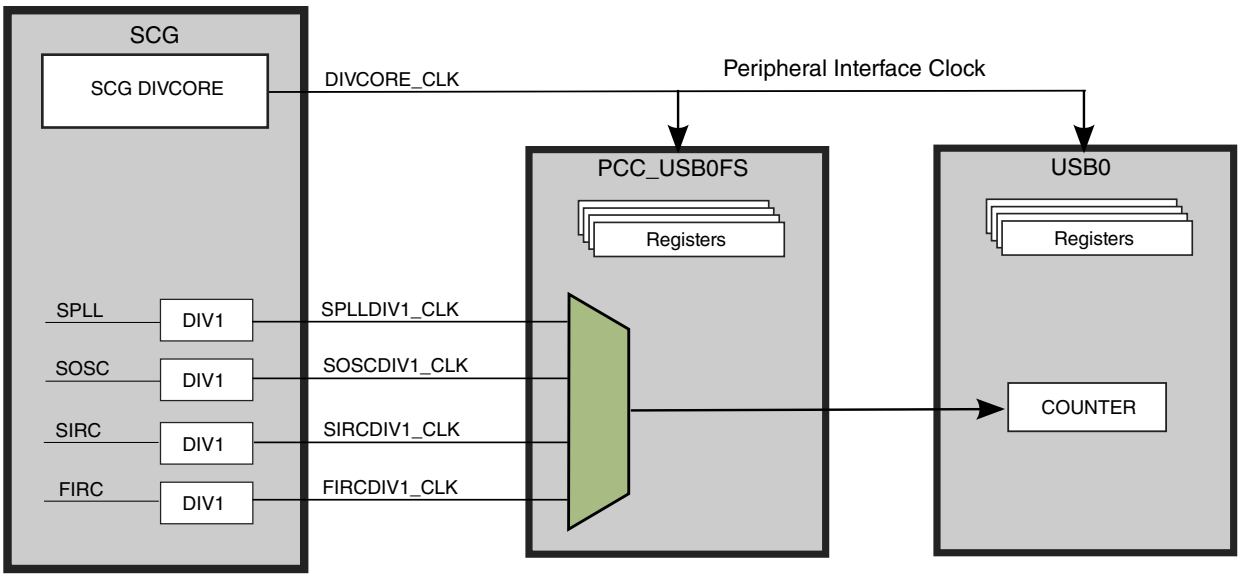


Figure 5-3. Example Peripheral Clocking (USB)

5.7 Programming model

The selection and multiplexing of system clock sources is controlled and programmed via the System Clock Generation (SCG) module. First, the SCG selects the clock source for driving the main CPU and the peripheral interface clocks, DIVCORE_CLK and DIVSLOW_CLK respectively.

The DIVCORE_CLK and the DIVSLOW_CLK clocks support three different modes of operation, normal RUN mode, HSRUN mode and Low Power RUN mode. Each of these run modes can have different clock sources and dividers options and are automatically selected via the options provided in SCG_xCCR registers. Each specific run mode is configured via the SMC_PMCTRL register.

Secondly the SCG can provide additional peripheral functional clocks to the various peripherals, via 3 optional output clock trees from each of the four sources. Each of the peripheral functional clock outputs has a programmable divider within the SCG. These *xxxx*DIV*y*_CLK outputs are fed to the PCC module where the peripheral can select one of these alternative clock sources to clock the timing function of that peripheral.

- SPLL provides SPLLDIV1_CLK, SPLLDIV2_CLK and SPLLDIV3_CLK
- SOSC provides SOSCDIV1_CLK, SOSCDIV2_CLK and SOSCDIV3_CLK
- FIRC provides FIRCDIV1_CLK, FIRCDIV2_CLK and FIRCDIV3_CLK
- SIRC provides SIRCDIV1_CLK, SIRCDIV2_CLK and SIRCDIV3_CLK

NOTE

The twelve additional clock trees may not all be implemented on every device.

Each peripheral that can support alternative clock inputs, will have it's own dedicated PCCn register which allows selection of one of the above 12 clock inputs. The PCCn will also provide a clock gate enable/disable to the peripheral. The CPU will gain access to these peripherals referencing either the DIVSLOW_CLK or DIVCORE_CLK with the peripheral.

See SCG and PCC chapters for further details.

5.8 Other Clock Sources

5.8.1 OSC32KCLK

The SCG also provide an additional output from the SOSC, named OSC32KCLK, which is used when the OSC is working from an external crystal of 32KHz or external 32KHz clock. This signal is fed directly to some low power peripherals to allow slower clocking in low power modes.

5.8.2 LPO Low Power Oscillator

The Low Power Oscillator (LPO) is an internal 1KHz RC oscillator and is not part of the SCG. It is controlled via the PMC module. It provides an optional clock input for peripherals that are required to operate in low power modes

5.9 Clock definitions

The following table describes the clocks in this device.

Table 5-2. Clock Definitions

Clock name	Description
Clock name	Description
SOSC	System Oscillator clock from the OSC module or an external squarewave input on EXTAL pin. One of the four clock sources. References to ERCLK are to this SOSC clock
SIRC	Slow Internal RC oscillator clock (2 or 8 MHz). One of the four clock sources.
FIRC	Fast Internal RC oscillator clock (48-60 MHz). One of the four clock sources.
SPLL	System PLL clock, that is a multiple of the FIRC or SOSC. One of the four clock sources.
DIVCORE_CLK	Main clock for CPU platform and closely tighly coupled peripherals/modules. This clock is derived from one of the four clock sources
DIVSLOW_CLK	Main clock used for peripheral/CPU interfacing and clocking the flash module. This clock is derived from DIVCORE_CLK.
SOSCDIV1_CLK	Alternative clock for clocking peripherals tightly coupled with CPU, sourced from SOSC. Selected via PCCn register of peripheral
SIRCDIV1_CLK	Alternative clock for clocking peripherals tightly coupled with CPU, sourced from SIRC. Selected via PCCn register of peripheral
FIRCDIV1_CLK	Alternative clock for peripherals tightly coupled with CPU, sourced from FIRC. Selected via PCCn register of peripheral
SPLLDIV1_CLK	Alternative clock for clocking peripherals tightly coupled with CPU, sourced from SPLL. Selected via PCCn register of peripheral
SOSCDIV3_CLK	Alternative clock for clocking low power peripherals, sourced from SOSC. Selected via PCCn register of peripheral
SIRCDIV3_CLK	Alternative clock for clocking low power peripherals, sourced from SIRC. Selected via PCCn register of peripheral
FIRCDIV3_CLK	Alternative clock for clocking low power peripherals, sourced from FIRC. Selected via PCCn register of peripheral
SPLLDIV3_CLK	Alternative clock for clocking low power peripherals, sourced from SPLL. Selected via PCCn register of peripheral
OSC32KCLK	System oscillator output clock when either a 32 kHz crystal or 32 kHz external clock source is provide to the EXTAL pin.
LPO	PMC 1kHz output. Used to clock some periodic timers when in low power stop modes to provide periodic wake-up or periodic interrupts. The LPO is independent of the SCG and is controlled via the PMC module.

5.10 Clocking details

The following table provide more information regarding the on-chip clocks.

Table 5-3. Detailed Clock Summary

Clock name	Run mode - Clock Frequency	VLPR mode - Clock Frequency	HSRUN mode - Clock Frequency	Clock source	Clock can be disabled when…
DIVCORE_CLK	Up to 72 MHz	Up to 8 MHz	Up to 96 MHz	SCG	When both CPUs are in any stop

Table continues on the next page...

Table 5-3. Detailed Clock Summary (continued)

Clock name	Run mode - Clock Frequency	VLPR mode - Clock Frequency	HSRUN mode - Clock Frequency	Clock source	Clock can be disabled when…
					modes except for partial stop modes.
DIVSLOW_CLK	Up to 24 MHz	DIVCORE_CLK divide by 4 or more.	Up to 24 MHz	SCG	When both CPUs are in any stop modes except for partial stop modes.
SOSCDIV1_CLK, SOSCDIV3_CLK	Up to 48 MHz	Up to 8 MHz	Up to 48 MHz	SCG	If not being used by any peripheral and/or DIVCORE_CLK or feeding the PLL
SPLLDIV1_CLK, SPLLDIV3_CLK	Up to 72 MHz	PLL disabled	Up To 96 MHz	SCG	If not being used by any peripheral and/or DIVCORE_CLK
FIRCDIV1_CLK, FIRCDIV3_CLK	Up to 60 MHz	FIRC is disabled	Up to 60 MHz	SCG	If not being used by any peripheral and/or DIVCORE_CLK or feeding the PLL
SIRCDIV1_CLK, SIRCDIV3_CLK	8 MHz	8 MHz	8 MHz	SCG	If not being used by any peripheral and/or DIVCORE_CLK
OSC32KCLK	30 kHz to 40 kHz	30-40 kHz low-range crystal	OSC32KCLK is disabled	external crystal oscillator connected to EXTAL/XTAL or ext clock on EXTAL pin	If FIRC or SIRC is being used by SCG, and if no peripherals are using OSC32KCLK
LPO	1 kHz	1 kHz	LPO is disabled	PMC's Low Power Oscillator	Can be disabled in VLPSx Stop modes to save sub uA current consumption if not used by LPTMRx or WDOGs.

5.11 Internal Clocking Requirements

The following requirements must be met when configuring the clocks for this device:

- The DIVCORE_CLK should not exceed 72MHz in normal RUN mode. The DIVSLOW_CLK should not exceed 24 MHz for reliable flash memory operation
- The DIVCORE_CLK should not exceed 96MHz in High Speed RUN mode. The DIVSLOW_CLK should not exceed 24 MHz for reliable flash memory operation.
- The DIVCORE_CLK should not exceed 8MHz in Low Power RUN mode. The DIVSLOW_CLK should not exceed 1MHz.

- Peripheral functional clocks from *xxx*DIV3_CLK or *xxx*DIV1_CLK in normal RUN mode should be limited to 72 MHz.
- Peripheral functional clocks from *xxx*DIV3_CLK or *xxx*DIV1_CLK in high speed RUN mode should be limited to 96MHz.
- When switching from normal RUN mode to HSRUN mode, clock sources should be enabled prior to changing run modes. Once the Core is in HSRUN mode, if the normal RUN mode source oscillator is not being used then it can then be disabled.
- When switching from normal RUN mode to VLPR mode, clock sources should be enabled prior to changing run modes. Once the Core is in VLPR mode, if the normal RUN mode source oscillator is not being used then it can then be disabled.
- When switching from VLPR mode to normal RUN mode, clock sources should be enabled prior to changing run modes. Once the Core is in normal RUN mode, if the VLPR mode source oscillator is not being used then it can then be disabled.
- When switching from HSRUN mode to normal RUN mode, clock sources should be enabled prior to changing run modes. Once the Core is in normal RUN mode, if the HSRUN mode source oscillator is not being used then it can then be disabled.
- Switching from VLPR to HSRUN mode, and vice versa is not supported. Users must switch to normal RUN mode first then into either HSRUN or VLPR mode.

Refer to the SMC chapter for further power mode information.

5.12 Clock divider values after reset

The device resets to either normal RUN mode, or VLP Run mode and the SIRC clock source is used to drive the DIVCORE_CLK. The specific run mode is selected via NVM (IFR) options.

In addition to the run mode when the device resets, the DIVCORE_CLK divide ratio is determined by the NVM (IFR) options. The IFR bits set the DIVCORE field in the SCG's Run Clock Control Register (SCG_RCCR). Similarly the DIVSLOW_CLK divide ratio is also determined by the NVM (IFR) options.

For KL28, out of reset, the SIRC module is enabled and used as the main clock source out of reset. This provides an 8MHz source clock.

The PLL, FIRC and SOSC are disabled out of reset.

All peripheral clocks *xxx*DIV*y*_CLK's are disabled out of reset.

The speed of the DIVCORE_CLK and DIVSLOW_CLK out of reset can be modified by programming NVM(IFR) register bits.

Table 5-4. DIVCORE_CLK and DIVSLOW_CLK out-of-reset configuration

FTFA_FOPT[4:0]	DIVCORE_CLK	DIVSLOW_CLK	Execution mode
00	0x7 (divide by 8)	0x1 (divide by 2)	VLPR
01	0x3 (divide by 4)	0x1 (divide by 2)	VLPR
10	0x1 (divide by 2)	0x1 (divide by 2)	RUN
11	0x0 (divide by 1)	0x1 (divide by 2)	RUN

This gives the user flexibility in selecting between a lower frequency, low-power boot option and higher frequency, higher power during and after reset. The flash erased state defaults to fast clocking mode, since these bits reside in flash, which is logic 1 in the flash erased state. To enable a lower power boot option, program the appropriate bits in FTFA_FOPT. During the reset sequence, if either of the control bits is cleared, the system is in a slower clock configuration. Upon any system reset, the clock dividers return to this configurable reset state. The factory default reset clock for core/system clock is 8MHz from SIRC.

5.13 Clock gating

Clock gating of modules helps users to only consume power for modules that are needed for their end application. The clock to each module will have the capability to be gated on or off via a programmable register. Each peripheral has independent clock gating for both the peripheral interface clock and the peripheral functional clock. This clock gating is controlled via the peripheral's PCCn register. Clearing PCCn[CGC] disables the peripheral interface clock, and clearing of PCCn[PCS] disables the peripheral functional clock. The SCG also has clock gating functionality of the peripheral functional clocks, which can be used to gate several peripheral functional clocks via the *xxx*DIV[*xxx*DIV*y*] fields. The DIVCORE_CLK and DIVSLOW_CLK are always enabled in the various RUN and Wait modes. If these are not used in low power stop modes, then they will be disabled via the SMC and PMC modules.

Both the Core Clock and Slow (Flash) Clocks have associated clock gates in the SCG.

5.14 Flash Memory Clock

The embedded Flash memory has a maximum operating frequency of 25Mhz. The KL28 uses the DIVSLOW_CLK from the SCG. This clock output should not exceed 25MHz for reliable flash operation.

Chapter 6
Reset and Boot

6.1 Introduction

Reset sources supported in this microcontroller are listed here.

Table 6-1. Reset sources

Reset sources	Description
POR reset	• Power-on reset (POR)
External global system resets	• External pin reset (PIN) • Low-voltage detect (LVD) • Low leakage wakeup (LLWU) reset • Clock monitor reset sources • Stop mode acknowledge error (SACKERR)
Internal Core-generated resets	• Watchdog timer reset • Software reset (SW) • Lockup reset (LOCKUP) • MDM DAP system reset request
Debug reset	• Debug resets

Each of the system reset sources has an associated bit in the System Reset Status (SRS) registers. See the Reset Control Module for register details.

The microcontroller has 1 CPU (called the Core).

- By default, resets initiated externally from the Core (like External Global System resets) force a reset to the the Core.
- On exit from any reset, the Core starts to execute.
- The Core is always enabled after a global external reset.

The microcontroller can exit and reset in functional mode where the Core is executing code (default) or the Core is in a debug halted state. There are several boot options that can be configured. See Boot for more details.

6.2 Reset

The information found here discusses basic reset mechanisms and sources.

Some modules that cause resets can be configured to cause interrupts instead. Consult the individual peripheral chapters for more information.

6.2.1 Power-on reset (POR)

When power is initially applied to the MCU or when the supply voltage drops below the power-on reset re-arm voltage level (V_{POR}), the POR circuit causes a POR reset condition.

As the supply voltage rises, the LVD circuit holds the MCU in reset until the supply has risen above the LVD low threshold (V_{LVDL}). The POR and LVD fields in the Reset Status Register are set following a POR.

6.2.2 System reset sources

Resetting the MCU provides a way to start processing from a known set of initial conditions. System reset begins with the on-chip regulator in full regulation and system clocking generation from an internal reference. When the processor exits reset, it performs the following:

- Reads the start SP (SP_main) from vector-table offset 0
- Reads the start program counter (PC) from vector-table offset 4
- Link register (LR) is set to 0xFFFF_FFFF.

The on-chip peripheral modules are disabled and the non-analog I/O pins are initially configured as disabled. The pins with analog functions assigned to them default to their analog function after reset.

During and following a reset, the SWD pins have their associated input pins configured as:

- SWD_CLK in pulldown (PD)
- SWD_DIO in pullup (PU)

6.2.2.1 External pin reset (RESET_b)

This pin is open drain and has an internal pullup device. Asserting RESET_b wakes the device from any mode.

The RESET_b pin can be disabled by programming RESET_PIN_CFG option bit to 0. When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pin-out low prior to establishing the setting of this option and releasing the reset function on the pin. When the RESET pin is disabled and configured as a GPIO output, it operates as a pseudo open drain output.

6.2.2.1.1 $\overline{\text{RESET}}$ pin filter

The $\overline{\text{RESET}}$ pin filter supports filtering from both the 1 kHz LPO clock and the bus clock. RCM_RPC[RSTFLTSS], RCM_RPC[RSTFLTSRW], and RCM_RPC[RSTFLTSEL] control this functionality; see the RCM chapter. The filters are asynchronously reset by Chip POR. The reset value for each filter assumes the $\overline{\text{RESET}}$ pin is negated.

For all stop modes where LPO clock is still active (Stop, VLPS, LLS, VLLS3, VLLS2 and VLLS1), the only filtering option is the LPO-based digital filter. The filtering logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected. When entering VLLS0, the $\overline{\text{RESET}}$ pin filter is disabled and bypassed.

The LPO filter has a fixed filter value of 3. Due to a synchronizer on the input data, there is also some associated latency (2 cycles). As a result, 5 cycles are required to complete a transition from low to high or high to low.

6.2.2.2 Low-voltage detect (LVD)

The chip includes a system for managing low-voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system consists of a power-on reset (POR) circuit and an LVD circuit with a user-selectable trip voltage. The LVD system is always enabled in Normal Run, Wait, or Stop mode. The LVD system is disabled when entering VLPx, LLS, or VLLSx modes.

The LVD can be configured to generate a reset upon detection of a low-voltage condition by setting PMC_LVDSC1[LVDRE] to 1. The low-voltage detection threshold is determined by PMC_LVDSC1[LVDV]. After an LVD reset has occurred, the LVD system holds the MCU in reset until the supply voltage has risen above the low voltage detection threshold. RCM_SRS[LVD] is set following either an LVD reset or POR.

6.2.2.3 High-voltage detect (HVD)

The chip includes a system to guard against high-voltage conditions. See:

- High-voltage detect (HVD) system
- HVD reset operation
- HVD interrupt operation

6.2.2.4 Watchdog timer (WDOG)

The watchdog timer monitors the operation of the system by expecting periodic communication from the software. This communication is generally known as servicing (or refreshing) the watchdog. If this periodic refreshing does not occur, then the watchdog issues a system reset, which causes RCM_SRS[WDOG] to set.

6.2.2.5 Low leakage wakeup (LLWU)

The LLWU module provides the means for a number of external pins and a number of internal peripherals to wake the MCU from low leakage power modes. It also allows for asynchronous DMA wakeup from LLS for certain peripherals.

The LLWU module is functional only in low leakage power modes. In VLLSx modes, all enabled inputs to the LLWU can generate a system reset.

After a system reset, the LLWU retains the flags indicating the input source of the last wakeup until the user clears them.

NOTE

Some flags are cleared in the LLWU and some flags are required to be cleared in the peripheral module. Refer to the individual peripheral chapters for more information.

6.2.2.6 Clock monitor reset sources

The SCG module contains a clock monitor with reset and interrupt request capability for SPLL and SOSC clocks.

NOTE

To prevent unexpected reset events, all clock monitors must be disabled before entering any low-power modes, including VLPR and VLPW.

6.2.2.7 Stop mode acknowledge error (SACKERR)

This reset is generated if the core attempts to enter Stop mode or Compute Operation, but not all modules acknowledge Stop mode within 1025 cycles of the 1 kHz LPO clock.

A module might not acknowledge the entry to Stop mode if an error condition occurs. The error can be caused by a failure of an external clock input to a module.

6.2.2.8 Software reset (SW)

The SYSRESETREQ field in the NVIC Application Interrupt and Reset Control register can be set to force a software reset on the device. (See ARM's NVIC documentation for the full description of the register fields, especially the VECTKEY field requirements.) Setting SYSRESETREQ generates a software reset request. This reset forces a system reset of all major components except for the debug module. A software reset causes RCM_SRS[SW] to set.

6.2.2.9 Lockup reset (LOCKUP)

The LOCKUP gives immediate indication of a seriously errant kernel software. This is the result of the core being locked, because of an unrecoverable exception following the activation of the processor’s built-in system state protection hardware.

The LOCKUP condition causes a system reset and also causes RCM_SRS[LOCKUP] to set.

6.2.2.10 MDM-AP system reset request

To initiate a system reset, set the System Reset Request field in the MDM-AP control register. This is the primary method for resets via the SWD interface. The system reset is held until this field is cleared.

To hold the core in reset as the rest of the chip comes out of system reset, set the Core Hold Reset field in the MDM-AP control register.

6.2.3 MCU resets

A variety of resets are generated by the MCU to reset different modules.

Table 6-2. MCU Resets

Reset	Assertion	Resets What?	Other Actions
POR Only	POR Only reset asserts on the POR reset source only.	It resets the PMC and RTC.	The POR Only reset also causes all other reset types to occur.
Chip POR not VLLS	Chip POR not VLLS reset asserts on POR and LVD reset sources.	It resets parts of the SMC and SIM, and also resets the LPTMR.	The Chip POR not VLLS reset also causes these other resets to occur: • Chip POR • Chip Reset not VLLS • Chip Reset (including Early Chip Reset)
Chip POR	Chip POR asserts on POR, LVD, and VLLS Wakeup reset sources.	It resets the RCM registers and parts of the SIM and SCG.	The Chip POR also causes the Chip Reset (including Early Chip Reset) to occur.
Chip Reset not VLLS	Chip Reset not VLLS reset asserts on all reset sources except a VLLS Wakeup that does not occur via the RESET_b pin.	It resets parts of the SMC, LLWU, and other modules that remain powered during VLLS mode.	The Chip Reset not VLLS reset also causes the Chip Reset (including Early Chip Reset) to occur.
Early Chip Reset	Early Chip Reset asserts on all reset sources.	It resets only the flash memory module.	It negates before flash memory initialization begins ("earlier" than when the Chip Reset negates).
Chip Reset	Chip Reset asserts on all reset sources.	It resets the remaining modules (the modules not reset by other reset types).	It only negates after flash initialization has completed and the RESET_b pin has also negated.

6.2.4 RESET_b pin

For all reset sources except a VLLS Wakeup that does not occur via the RESET_b pin, the RESET_b pin is driven low by the MCU for at least 128 bus clock cycles and until flash initialization has completed.

After flash initialization has completed, the RESET_b pin is released, and the internal Chip Reset negates after the RESET_b pin is pulled high. Keeping the RESET_b pin asserted externally delays the negation of the internal Chip Reset.

The RESET_b pin can be disabled by programming FTFA_FOPT[RESET_PIN_CFG] option bit to 0 (See Table 6-3). When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pin low prior to establishing the setting of this option and releasing the reset function on the pin. When the RESET pin is disabled and configured as a GPIO output, it operates as a pseudo open drain output.

6.2.5 Debug resets

The following sections detail the debug resets available on the device.

6.2.5.1 Resetting the Debug subsystem

Use the CDBGRSTREQ field within the DP CTRL/STAT register to reset the debug modules. However, as explained below, using the CDBGRSTREQ field does not reset all debug-related registers.

CDBGRSTREQ resets the debug-related registers within the following modules:

- SW-DP
- AHB-AP
- MDM-AP (MDM control and status registers)

CDBGRSTREQ does not reset the debug-related registers within the following modules:

- CM0+ core (core debug registers: DHCSR, DCRSR, DCRDR, DEMCR)
- BPU
- DWT
- NVIC
- Crossbar bus switch[1]
- Private peripheral bus[1]

6.3 Boot

The information found here describes the boot sequence, including sources and options.

Some configuration information such as clock trim values stored in factory programmed flash locations is autoloaded.

6.3.1 Boot sources

The CM0+ core adds support for a programmable Vector Table Offset Register (VTOR) to relocate the exception vector table. This device supports booting from internal flash and RAM.

1. CDBGRSTREQ does not affect AHB resources so that debug resources on the private peripheral bus are available during System Reset.

This device supports booting from internal flash with the reset vectors located at addresses 0x0 (initial SP_main), 0x4 (initial PC), and RAM with relocating the exception vector table to RAM.

6.3.2 FOPT boot options

The Flash Option (FOPT) register in the Flash Memory module (FTFA_FOPT) allows the user to customize the operation of the MCU at boot time. The register contains read-only bits that are loaded from the NVM's option byte in the flash configuration field. The default setting for all values in the FTFA_FOPT register is logic 1 since it is copied from the option byte residing in flash, which has all bits as logic 1 in the flash erased state. To configure for alternate settings, program the appropriate bits in the NVM option byte. The new settings will take effect on subsequent POR, VLLSx recoveries, and any system reset. For more details on programming the option byte, see the flash memory chapter.

The MCU uses FTFA_FOPT to configure the device at reset as shown in the following table.

NOTE

An FTFA_FOPT value of 0x00 is invalid and will be ignored. The FOPT register is written to 0xFF if the contents of the Flash nonvolatile option are 0x00.

Table 6-3. Flash Option Register (FTFA_FOPT) definition

Bit Num	Field	Value	Definition
7-6	BOOTSRC_SEL	Boot Source Selection: these bits select the boot sources if boot pin option bit BOOTPIN_OPT = 1	
		00	Boot from Flash
		01	Reserved
		10	Reserved
		11	Boot from ROM
5	FAST_INIT	Selects initialization speed on POR, VLLSx, and any system reset.	
		0	Slower initialization: The flash initialization will be slower with the benefit of reduced average current during this time. The duration of the recovery will be controlled by the clock divider selection determined by the LPBOOT setting.
		1	Fast Initialization: The flash has faster recoveries at the expense of higher current during these times.
3	RESET_PIN_CFG	Enables/disables control for the RESET pin.	
		0	RESET_b pin is disabled following a POR and cannot be enabled as reset function. When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pin low prior to establishing the

Table continues on the next page...

Table 6-3. Flash Option Register (FTFA_FOPT) definition (continued)

Bit Num	Field	Value	Definition
			setting of this option and releasing the reset function on the pin. When the RESET pin is disabled and configured as a GPIO output, it operates as a pseudo open drain output. This bit is preserved through system resets and low-power modes. When RESET_b pin function is disabled, it cannot be used as a source for low-power mode wake-up. **NOTE:** When the reset pin has been disabled and security has been enabled by means of the FSEC register, a mass erase can be performed only by setting both the Mass Erase and System Reset Request fields in the MDM-AP register.
		1	RESET_b pin is dedicated. The port is configured with pullup enabled, open drain, passive filter enabled.
2	NMI_DIS	Enables/disables control for the NMI function.	
		0	NMI interrupts are always blocked. The associated pin continues to default to NMI_b pin controls with internal pullup enabled. When NMI_b pin function is disabled, it cannot be used as a source for low-power mode wake-up. If the NMI function is not required, either for an interrupt or wake up source, it is recommended that the NMI function be disabled by clearing NMI_DIS.
		1	NMI_b pin/interrupts reset default to enabled.
1	BOOTPIN_OPT	External pin selects boot options	
		0	Force Boot from ROM if BOOTCFG0 asserted, where BOOTCFG0 is the boot config function which is muxed with NMI_b pin. RESET_b pin must be enabled (FOPT[RESET_PIN_CFG] = 1) when this option is selected. NMI_b pin is sampled at the end of reset (when reset pin negates). If BOOTCFG0 pin is not asserted, Boot source configured by FOPT[7:6] (BOOTSRC_SEL) bits.
		1	Boot source configured by FOPT[7:6] (BOOTSRC_SEL) bits
4,0	LPBOOT	Controls the reset value of DIVCORE in the SCG_RCCR and SCG_VCCR registers, and the state of the RUNM field in the SMC_PMCTRL register. Larger divide value selections produce lower average power consumption during POR, VLLSx recoveries and reset sequencing and after reset exit. The recovery times are also extended if the FAST_INIT option is not selected.	
		00	Core and system clock divider (SCG_VCCR[DIVCORE]) is 0x7 (divide by 8). Device is configured for VLPR mode on exit from reset.
		01	Core and system clock divider (SCG_VCCR[DIVCORE]) is 0x3 (divide by 4). Device is configured for VLPR mode on exit from reset.
		10	Core and system clock divider (SCG_RCCR[DIVCORE]) is 0x1 (divide by 2). Device is configured for RUN mode on exit from reset.
		11	Core and system clock divider (SCG_RCCR[DIVCORE]) is 0x0 (divide by 1). Device is configured for RUN mode on exit from reset.

6.3.3 Boot sequence

At power up, the on-chip regulator holds the system in a POR state until the input supply exceeds the POR threshold. The system continues to be held in this static state until the internally regulated supplies have reached a safe operating voltage as determined by the LVD. The Reset Controller logic then controls a sequence to exit reset.

1. A system reset is held on internal logic, the RESET pin is driven out low, and the SCG is enabled in its default clocking mode.
2. Required clocks are enabled (system clock, flash clock, and any bus clocks that do not have clock gate control reset to disabled).
3. The system reset on internal logic continues to be held, but the Flash Controller is released from reset and begins initialization operation while the Reset Control logic continues to drive the RESET pin out low.
4. Early in reset sequencing, the NVM option byte is read and stored to the FOPT register of the Flash Memory module (FTFA_FOPT). If the bits associated with FTFA_FOPT[LPBOOT] are programmed for an alternate clock divider reset value, the system/core clock is switched to a slower clock speed. If FTFA_FOPT[FAST_INIT] is programmed clear, the flash initialization switches to slower clock resulting longer recovery times.
5. When flash Initialization completes, the RESET pin is released. If RESET continues to be asserted (an indication of a slow rise time on the RESET pin or external drive in low), the system continues to be held in reset. Once the RESET pin is detected high, the core clock is enabled and the system is released from reset.
6. When the system exits reset, the processor sets up the stack, program counter (PC), and link register (LR). The processor reads the start SP (SP_main) from vector-table offset 0. The core reads the start PC from vector-table offset 4. LR is set to 0xFFFF_FFFF. The next sequence of events depends on the NMI/BOOTCFG0 input and FTFA_FOPT[NMI_DIS] and FTFA_FOPT[BOOTSRC_SEL] and FTFA_FOPT[BOOTPIN_OPT] as well as RCM_MR[BOOTROM](See Table 6-3 and RCM block guide) :
   - If the NMI/BOOTCFG0 input is high or the NMI function is disabled in FTFA_FOPT, the CPU fetches the reset vector from flash address 0x0000_0004 and code execution begins from this address.
   - If the NMI/BOOTCFG0 input is low and the NMI function is enabled in FTFA_FOPT, this results in an NMI interrupt. The processor executes an Exception Entry and reads the NMI interrupt handler address from vector-table offset 8. The CPU begins execution at the NMI interrupt handler.
   - When FTFA_FOPT[BOOTPIN_OPT] = 0, it forces boot from ROM if NMI/BOOTCFG0 pin asserted.

NOTE

If the NMI function is not required, either for an interrupt or wake up source, it is recommended that the NMI function be disabled by clearing NMI_DIS in the FOPT register.

Subsequent system resets follow this same reset flow.

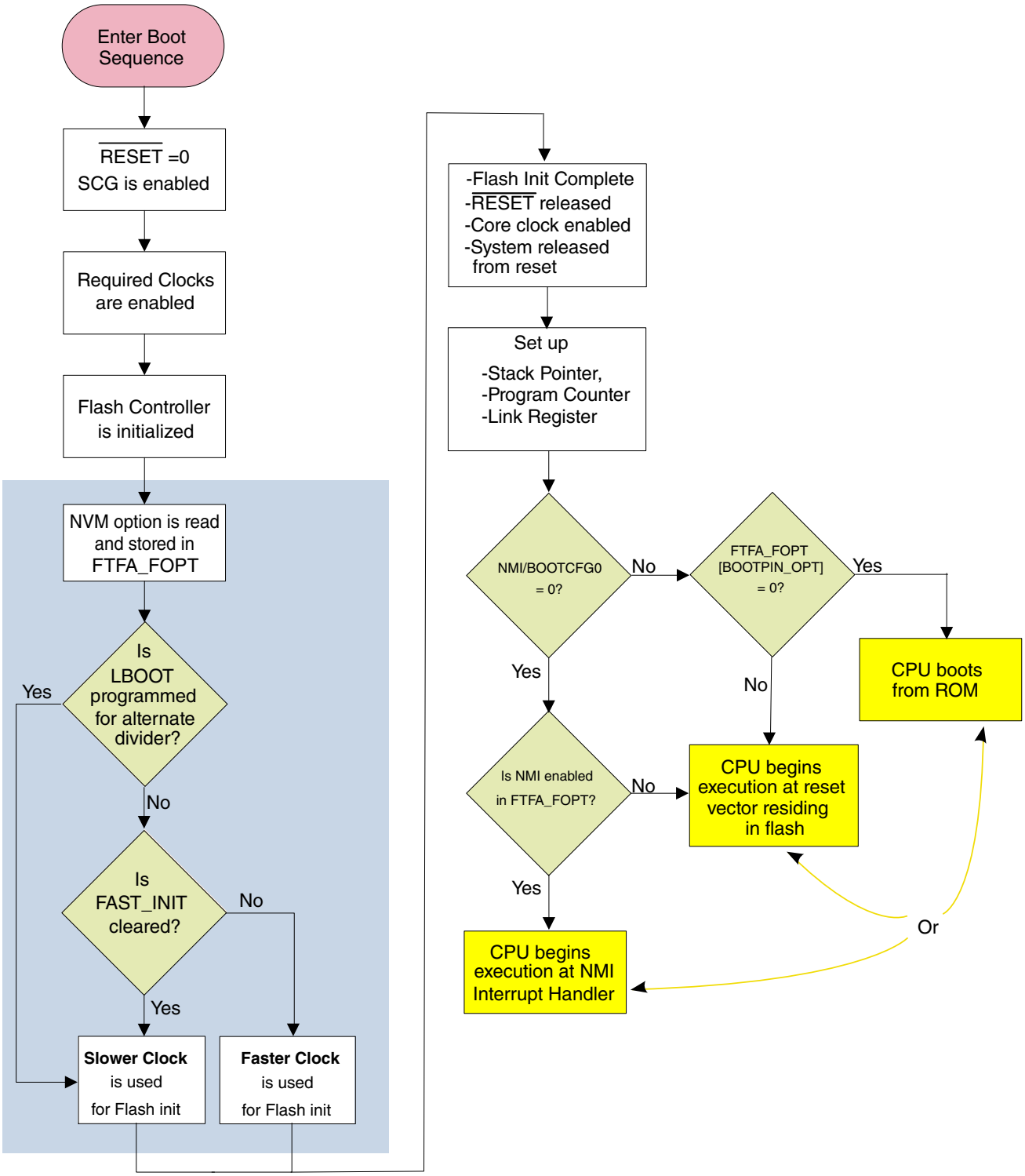


Figure 6-1. Boot Sequence

Chapter 7
Power Management

7.1 Introduction

Information about the various chip power modes and functionality of the individual modules in these modes can be found here.

7.2 Clocking modes

Information found here describes the various clocking modes supported on this device.

7.2.1 Partial Stop

Partial Stop is a clocking option that can be taken instead of entering Stop mode and is configured in the SMC Stop Control Register (SMC_STOPCTRL). The Stop mode is only partially entered, which leaves some additional functionality alive at the expense of higher power consumption. Partial Stop can be entered from either Run mode or VLP Run mode.

When configured for PSTOP2, only the core and system clocks are gated and the bus clock remains active. The bus masters and bus slaves clocked by the system clock enter Stop mode, but the bus slaves clocked by bus clock remain in Run (or VLP Run) mode. The clock generators in the SCG and the on-chip regulator in the PMC also remain in Run (or VLP Run) mode. Exit from PSTOP2 can be initiated by a reset, an asynchronous interrupt from a bus master or bus slave clocked by the system clock, or a synchronous interrupt from a bus slave clocked by the bus clock. If configured, a DMA request (using the asynchronous DMA wakeup) can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP2.

When configured for PSTOP1, both the system clock and bus clock are gated. All bus masters and bus slaves enter Stop mode, but the clock generators in the SCG and the on-chip regulator in the PMC remain in Run (or VLP Run) mode. Exit from PSTOP1 can be initiated by a reset or an asynchronous interrupt from a bus master or bus slave. If configured, an asynchronous DMA request can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP1.

PSTOP1 is functionally similar to Stop mode, but offers faster wake-up at the expense of higher power consumption. Another benefit is that it keeps all of the SCG clocks enabled, which can be useful for some of the asynchronous peripherals that can remain functional in Stop modes.

7.2.2 DMA Wakeup

The DMA can be configured to wake the device on a DMA request whenever it is placed in Stop mode. The wake-up is configured per DMA channel and is supported in Compute Operation, PSTOP, STOP, and VLPS low power modes.

When a DMA wake-up is detected in PSTOP, STOP or VLPS then the device will initiate a normal exit from the low power mode. This can include restoring the on-chip regulator and internal power switches, enabling the clock generators in the SCG, enabling the system and bus clocks (but not the core clock) and negating the stop mode signal to the bus masters and bus slaves. The only difference is that the CPU will remain in the low power mode with the CPU clock disabled.

During Compute Operation, a DMA wake-up will initiate a normal exit from Compute Operation. This includes enabling the clocks and negating the stop mode signal to the bus masters and bus slaves. The core clock always remains enabled during Compute Operation.

Since the DMA wakeup will enable the clocks and negate the stop mode signals to all bus masters and slaves, software needs to ensure that bus masters and slaves that are not involved with the DMA wake-up and transfer remain in a known state. That can be accomplished by disabling the modules before entry into the low power mode or by setting the Doze enable bit in selected modules.

Once the DMA request that initiated the wake-up negates and the DMA completes the current transfer, the device will transition back to the original low-power mode. This includes requesting all non-CPU bus masters to enter Stop mode and then requesting bus slaves to enter Stop mode. In STOP and VLPS modes the SCG and PMC would then also enter their appropriate modes.

NOTE

If the requested DMA transfer cannot cause the DMA request to negate then the device will remain in a higher power state until the low power mode is fully exited.

An enabled DMA wake-up can cause an aborted entry into the low power mode, if the DMA request asserts during the stop mode entry sequence (or reentry if the request asserts during a DMA wakeup) and can cause the SMC to assert its Stop Abort flag. Once the DMA wake-up completes, entry into the low power mode will restart.

An interrupt that occurs during a DMA wake-up will cause an immediate exit from the low power mode (this is optional for Compute Operation) without impacting the DMA transfer.

A DMA wake-up can be generated by either a synchronous DMA request or an asynchronous DMA request. Not all peripherals can generate an asynchronous DMA request in stop modes, although in general if a peripheral can generate synchronous DMA requests and also supports asynchronous interrupts in stop modes, then it can generate an asynchronous DMA request.

7.2.3 Compute Operation

Compute Operation is an execution or compute-only mode of operation that keeps the CPU enabled with full access to the SRAM and Flash read port, but places all other bus masters and bus slaves into their stop mode. Compute Operation can be enabled in either Run mode or VLP Run mode.

NOTE

Do not enter any Stop mode without first exiting Compute Operation.

Because Compute Operation reuses the Stop mode logic (including the staged entry with bus masters disabled before bus slaves), any bus master or bus slave that can remain functional in Stop mode also remains functional in Compute Operation, including generation of asynchronous interrupts and DMA requests. When enabling Compute Operation in Run mode, module functionality for bus masters and slaves is the equivalent of STOP mode. When enabling Compute Operation in VLP Run mode, module functionality for bus masters and slaves is the equivalent of VLPS mode. The SCG, PMC, SRAM, and Flash read port are not affected by Compute Operation, although the Flash register interface is disabled.

During Compute Operation, the AIPS peripheral space is disabled and attempted accesses generate bus errors. The private peripheral space remains accessible during Compute Operation, including the MCM, NVIC, IOPORT, and SysTick. Although access to the GPIO registers via the IOPORT is supported, the GPIO Port Data Input registers do not return valid data since clocks are disabled to the Port Control and Interrupt modules. By writing to the GPIO Port Data Output registers, it is possible to control those GPIO ports that are configured as output pins.

Compute Operation is controlled by the CPO register in the MCM (MCM_CPO), which is only accessible to the CPU. Setting or clearing MCM_CPO[CPOREQ] initiates entry or exit into Compute Operation. Compute Operation can also be configured to exit automatically on detection of an interrupt, which is required in order to service most interrupts. Only the core system interrupts (exceptions, including NMI and SysTick) and any edge-sensitive interrupts can be serviced without exiting Compute Operation.

- When entering Compute Operation, the CPOACK status field in the CPO register of MCM module (MCM_CPO[CPOACK]) indicates when entry has completed.
- When exiting Compute Operation in Run mode, MCM_CPO[CPOACK] negates immediately.
- When exiting Compute Operation in VLP Run mode, the exit is delayed to allow the PMC to handle the change in power consumption. This delay means that MCM_CPO[CPOACK] is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

The DMA wake-up is also supported during Compute Operation and causes MCM_CPO[CPOACK] to clear and the AIPS peripheral space to be accessible for the duration of the DMA wakeup. At the completion of the DMA wake-up, the device transitions back into Compute Operation.

7.2.4 Peripheral Doze

Several peripherals support a Peripheral Doze mode, where a register bit can be used to disable the peripheral for the duration of a low-power mode. The flash memory can also be placed in a low-power state during Peripheral Doze via a register bit in the SIM.

Peripheral Doze is defined to include all of the modes of operation listed below.

- The CPU is in Wait mode.
- The CPU is in Stop mode, including the entry sequence and for the duration of a DMA wakeup.
- The CPU is in Compute Operation, including the entry sequence and for the duration of a DMA wakeup.

Peripheral Doze can therefore be used to disable selected bus masters or slaves for the duration of WAIT or VLPW mode. It can also be used to disable selected bus slaves immediately on entry into any stop mode (or Compute Operation), instead of waiting for the bus masters to acknowledge the entry as part of the stop entry sequence. Finally, it can be used to disable selected bus masters or slaves that should remain inactive during a DMA wakeup.

If the flash memory is not being accessed during WAIT and PSTOP modes, then the Flash Doze mode can be used to reduce power consumption, at the expense of a slightly longer wake-up when executing code and vectors from flash. It can also be used to reduce power consumption during Compute Operation when executing code and vectors from SRAM.

7.2.5 Clock gating

To conserve power, the clocks to most modules can be turned off using the CGC bit in the PCC module registers. For more details, see the Clock Distribution and PCC chapters.

7.3 Power Mode Architecture

The power mode architecture consists of a digital power domain and an analog power domain. The device has the following external power connections.

- Digital ground (VSS)
- Digital supply (VDD)
- Analog ground (VSSA)
- Analog supply (VDDA)

The Analog to Digital converters ADC0 and Digital to Analog Converter DAC0 are powered via a separate analog domain: VSSA and VDDA.

GPIO port pads are powered directly from VDD.

7.4 Power modes

The System Mode Controller (SMC) provides multiple power options to allow the user to optimize power consumption for the level of functionality needed .

Depending on the stop requirements of the user application, a variety of stop modes are available that provide state retention, partial power-down or full power-down of certain logic and/or memory. I/O states are held in all modes of operation. The following table compares the various power modes available.

For each run mode, there is a corresponding Wait and Stop mode. Wait modes are similar to ARM Sleep modes. Stop modes (VLPS, STOP) are similar to ARM Sleep Deep mode. The Very Low Power Run (VLPR) operating mode can drastically reduce runtime power when the maximum bus frequency is not required to handle the application needs.

The three primary modes of operation are Run, Wait, and Stop. The WFI instruction invokes both Wait and Stop modes for the chip. The primary modes are augmented in a number of ways to provide lower power based on application needs.

Table 7-1. Chip power modes

Chip mode	Description	Core mode	Normal recovery method
RUN (Normal Run)	• Default mode out of reset • On-chip voltage regulator is on.	Run	—
HRUN (High Speed Run)	Allows maximum performance of chip. • On-chip voltage regulator is on.	High Speed Run	—
WAIT (Normal Wait) - via WFI	Allows peripherals to function while the core is in Sleep mode, reducing power. • NVIC remains sensitive to interrupts • Peripherals continue to be clocked.	Sleep	Interrupt
STOP (Normal Stop) - via WFI	Places chip in static state. Lowest power mode that retains all registers while maintaining LVD protection. • NVIC is disabled. • AWIC is used to wake up from interrupt. • Peripheral clocks are stopped.	Sleep Deep	Interrupt
VLPR (Very Low-Power Run)	On-chip voltage regulator is in a low-power mode that supplies only enough power to run the chip at a reduced frequency. • Reduced frequency Flash access mode (1 MHz) • LVD off	Run	—
VLPW (Very Low-Power Wait) -via WFI	Same as VLPR but with the core in Sleep mode to further reduce power. • NVIC remains sensitive to interrupts (FCLK = ON). • On-chip voltage regulator is in a low-power mode that supplies only enough power to run the chip at a reduced frequency.	Sleep	Interrupt
VLPS (Very Low-Power Stop)-via WFI	Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional. • Peripheral clocks are stopped, but OSC, LPTMR,LPUART, RTC, CMP, TSI can be used. • TPM and UART can optionally be enabled if their clock source is enabled. • NVIC is disabled (FCLK = OFF); AWIC is used to wake up from interrupt.	Sleep Deep	Interrupt

Table continues on the next page...

Table 7-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
	• On-chip voltage regulator is in a low-power mode that supplies only enough power to run the chip at a reduced frequency. • All SRAM is operating (content retained and I/O states held).		
LLS3 (Low-Leakage Stop3)	State retention power mode • Most peripherals are in state retention mode (with clocks stopped), but OSC, LLWU,LPTMR, RTC, CMP, TSI can be used. • NVIC is disabled; LLWU is used to wake up. **NOTE:** The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery • All SRAM is operating (content retained and I/O states held).	Sleep Deep	Wake-up Interrupt[1]
LLS2 (Low-Leakage Stop2)	State retention power mode • Most peripherals are in state retention mode (with clocks stopped), but OSC, LLWU,LPTMR, RTC, CMP, TSI can be used. • NVIC is disabled; LLWU is used to wake up. **NOTE:** The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery • 64 KB SRAM retained, internal logic and I/O states are retained.	Sleep Deep	Wake-up Interrupt[1]
VLLS3 (Very Low-Leakage Stop3)	• Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP, TSI can be used. • NVIC is disabled; LLWU is used to wake up. • SRAM_U and SRAM_L remain powered on (content retained and I/O states held).	Sleep Deep	Wake-up Reset[-1]
VLLS2 (Very Low-Leakage Stop2)	• Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP, TSI can be used. • NVIC is disabled; LLWU is used to wake up. • 64K of SRAM_U remains powered on (content retained and I/O states held).	Sleep Deep	Wake-up Reset[-1]
VLLS1 (Very Low-Leakage Stop1)	• Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP, TSI can be used. • NVIC is disabled; LLWU is used to wake up. • All of SRAM_U and SRAM_L are powered off. • The 32-byte system register file remains powered for customer-critical data • The 16-byte system register file remains powered for customer-critical data	Sleep Deep	Wake-up Reset[-1]
VLLS0 (Very Low-Leakage Stop 0)	• Most peripherals are disabled (with clocks stopped), but LLWU, LPTMR, RTC, TSI can be used. • NVIC is disabled; LLWU is used to wake up. • All of SRAM_U and SRAM_L are powered off. • The 32-byte system register file remains powered for customer-critical data • The 16-byte system register file remains powered for customer-critical data • LPO disabled, optional POR brown-out detection	Sleep Deep	Wake-up Reset[-1]

1. Resumes Normal Run mode operation by executing the LLWU interrupt service routine.

See Chip Power Modes of Operation for more information about power modes.

7.5 Entering and exiting power modes

The WFI instruction invokes wait and stop modes for the chip. The processor exits the low-power mode via an interrupt.

For LLS/VLLS modes, the wake-up sources are limited to LLWU generated wake-ups, LPTMR, CMP, RTC, NMI_b pin, or RESET_b pin assertions. When the NMI_b pin or RESET_b pin have been disabled through associated FTFA_FOPT settings, then these pins are ignored as wakeup sources. The wake-up flow from VLLSx is always through reset.

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

NOTE

The WFE instruction can have the side effect of entering a low-power mode, but that is not its intended usage. See ARM documentation for more on the WFE instruction.

On VLLS recoveries, the I/O pins continue to be held in a static state after code execution begins, allowing software to reconfigure the system before unlocking the I/O.

7.6 Module operation in low-power modes

The table found here illustrates the functionality of each module while the chip is in each of the low power modes.

The standard behavior is shown with some exceptions for Compute Operation (CPO) and Partial Stop2.

Debug modules are discussed separately; see Debug in low-power modes. Number ratings (such as 8 MHz and 2 Mbit/s) represent the maximum frequencies or maximum data rates per mode. A list of terms are used in the following table:

- **FF** = Full functionality. In VLPR and VLPW, the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- **Async** = Fully functional with alternate clock source, provided the selected clock source remains enabled
- **SR** = Module state is retained but not functional.

- **LP** = Module operates in a lower power state
- **OFF** = Modules are powered off or disabled.

Table 7-2. Module operation in low power modes

Modules	VLPR	VLPW	Stop	VLPS	LLSx	VLLSx
Core modules						
NVIC	FF	FF	SR	SR	SR	OFF
System modules						
SMC	FF	FF	FF	FF	FF	FF
LLWU[1]	SR	SR	SR	SR	FF	FF[2]
Regulator	LP	LP	FF	LP	LP	LP in VLLS2/3, OFF in VLLS0/1
HVD/LVD	disabled	disabled	FF	disabled	disabled	disabled
POR (Brown-out Detection)	FF	FF	FF	FF	FF	FF in VLLS1/2/3, optionally disabled in VLLS0[3]
DMA	FF Async in CPO	FF	Async operation	Async operation	Async for modules enabled by the LLWU	OFF
Clocks						
1kHz LPO	FF	FF	FF	FF	Optionally OFF (controlled by STOPCTRL[LPOPO])	OFF in VLLS0, optionally OFF in VLLS1/2/3 (controlled by STOPCTRL[LPOPO])
SCG	SYSOSC, SIRC enabled	SYSOSC, SIRC enabled	SYSOSC, SIRC, FIRC, SPLL enabled	SYSOSC, SIRC enabled	SYSOSC enabled	SYSOSC enabled
DIVCORE Clock	8 MHz max	OFF	OFF	OFF	OFF	OFF
DIVSLOW Clock	1 MHz max	OFF	OFF	OFF	OFF	OFF
Memory and memory interfaces						
Flash	1 MHz max access - no program No register access in CPO	low power	low power	low power	OFF	OFF
SRAM	low power	low power	SR	SR	SR in LLS3, partial SR in LLS2.[4]	SR in VLLS3, partial SR in VLLS2[5], OFF in VLLS0/1.
System Register File	FF	FF	SR	SR	SR	SR
Communication interfaces						

Table continues on the next page...

Table 7-2. Module operation in low power modes (continued)

Modules	VLPR	VLPW	Stop	VLPS	LLSx	VLLSx
USB FS/LS	SR	SR	optional	optional	SR	OFF
USB Voltage Regulator	optional	optional	optional	optional	optional	optional
LPUART0, LPUART1, LPUART2	2 Mbit/s	2 Mbit/s	Async operation FF in PSTOP2	Async operation	SR	OFF
LPSPI0, LPSPI1, LPSPI2	4 Mbit/s	master mode 4 Mbit/s,	Async operation	Async operation	SR	OFF
LPI2C0, LPI2C1, LPI2C2	1 Mbit/s	1 Mbit/s	Async operation	Async operation	SR	OFF
SAI	FF Async operation in CPO	FF	FF with external Async clock FF in PSTOP2	FF with external Async clock	SR	OFF
EMVSIM	FF	FF	FF	FF with external Async clock	OFF	OFF
Timers						
TPM	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation	SR	OFF
LPIT	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation	SR	OFF
LPTMR	FF	FF	Async operation FF in PSTOP2	Async operation	Async operation	Async operation[6]
RTC	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation	Async operation	Async operation[7]
TSTMR	FF	FF	Optional	Optional	OFF	OFF
Analog						
16-bit ADC	FF ADC internal clock only in CPO	FF	ADC internal clock only FF in PSTOP2	ADC internal clock only	SR	OFF
VREF	FF	FF	FF	FF	OFF	OFF
CMP[8]	FF HS or LS compare in CPO	FF	HS or LS compare FF in PSTOP2	HS or LS compare	LS compare	LS compare in VLLS1/3, OFF in VLLS0

Table continues on the next page...

Table 7-2. Module operation in low power modes (continued)

Modules	VLPR	VLPW	Stop	VLPS	LLSx	VLLSx
6-bit DAC	FF SR in CPO	FF	SR FF in PSTOP2	SR	SR	SR, OFF in VLLS0
12-bit DAC	FF SR in CPO	FF	SR FF in PSTOP2	SR	SR	SR
Human-machine interfaces						
GPIO	FF IOPORT write only in CPO	FF	SR output, wakeup input FF in PSTOP2	SR output, wakeup input	SR, pins latched	OFF, pins latched
FlexIO	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation	SR	OFF
TSI	FF Async operation in CPO	Async operation[9]	Async operation[9] FF in PSTOP2	Async operation[9]	Async operation[9]	Async operation[9]
Security						
TRNG	FF SR CPO	FF	SR	SR	SR	OFF
CRC	FF SR Compute	FF	SR	SR	SR	OFF
MMCAU	FF	SR	SR	SR	SR	OFF
Watchdog (WDOG32)	FF	FF	Async operation	Async operation	SR	OFF

1. Using the LLWU module, the external pins available for this chip do not require the associated peripheral function to be enabled. It only requires the function controlling the pin (GPIO or peripheral) to be configured as an input to allow a transition to occur to the LLWU.
2. When LPO clock source is disabled, filters will be bypassed.
3. STOPCTRL[PORPO] in the SMC module controls this option.
4. In LLS2 64kB of SRAM located at 0x2000_0000 - 0x2000_ffff is retained
5. In VLLS2 64kB of SRAM located at 0x2000_0000 - 0x2000_ffff is retained
6. LPO clock source is not available in VLLS0. Also, to use system OSC in VLLS0 it must be configured for bypass (external clock) operation. Pulse counting is available in all modes.
7. RTC is not supported in VLLS0.
8. CMP in stop or VLPS supports high speed or low speed external pin to pin or external pin to DAC compares. CMP in LLS or VLLSx only supports low speed external pin to pin or external pin to DAC compares.
9. TSI wake-up from all low-power modes is limited to a single selectable pin.

Chapter 8
Security

8.1 Introduction

This device implements security based on the mode selected from the flash module.

The following sections provide an overview of flash security and details the effects of security on non-flash modules.

The device security provides:

- Security of the flash and MCU via SEC bit
- Backdoor Key and Freescale Backdoor key access support
- Disable access to the debugger via the SWD interface

8.1.1 Debug security

Debug enables access to the Core for debugging, and also for programming the flash arrays. For the standard Programming mechanism, you make use of the FTFx_SEC register to enable security. Setting the FTFx_SEC [SEC] bits secures the Core from debug access.

8.1.2 Flash security

The device has 2 flash modules:

- Flash0 - 256K bytes
- Flash1 - 256K bytes

Both Flash0 and Flash1 are available to the user for application code. The Flash security follows the structure for other L series devices.

The FTFx module (using FTFx_FSEC register) provides security features:

- Backdoor Key Security Enable (KEYEN)
- Mass Erase Enable (MEEN)
- Freescale Failure Analysis Access Code (FSLACC)
- Flash Secure (SEC) - global secure of all Flash

The flash module provides security information to the MCU based on the state held by FTFA_FSEC[SEC]. The MCU, in turn, confirms the security request and limits access to flash resources. During reset, the flash module initializes FTFA_FSEC using data read from the security byte of the flash configuration field.

NOTE

The security features apply only to external accesses: debug. CPU accesses to the flash are not affected by the status of FTFA_FSEC.

In the unsecured state, all flash commands are available on the programming interfaces either from the debug port (SWD) or from user code execution. When the flash is secured (FTFA_FSEC[SEC] = 00, 01, or 11), the programmer interfaces are only allowed to launch mass erase operations. Additionally, in this mode, the debug port has no access to memory locations.

Chapter 9
Debug

9.1 Introduction

The debug system of this device is based on the ARM CoreSight™ architecture, and is configured to provide the maximum flexibility as allowed by the restrictions of the pinout and other available resources.

The MCU has a single M0+ CPU available for customer application use.

Debug provides register and memory accessibility from the external debugger interface, basic run/halt control, plus 2 breakpoints and 2 watchpoints. Additionally, it supports ARM's Micro Trace Buffer (MTB) capability to provide simple program trace. Only one debug interface is supported: Serial Wire Debug (SWD). The SWD interface provides the capability for debugger tools to interface to the CPU.

9.2 Debug port pin descriptions

The debug port pins default after POR to their SWD functionality.

Table 9-1. Serial wire debug pin description

Pin name	Type	Description
SWD_CLK	Input	Serial Wire Clock This pin is the clock for debug logic when in the Serial Wire Debug mode. This pin is pulled down internally.
SWD_DIO	Input / Output	Serial Wire Debug Data Input/Output The SWD_DIO pin is used by an external debug tool for communication and device control. This pin is pulled up internally.

9.3 Debug and Trace Block diagram

The next diagram shows the Debug and Trace configuration.

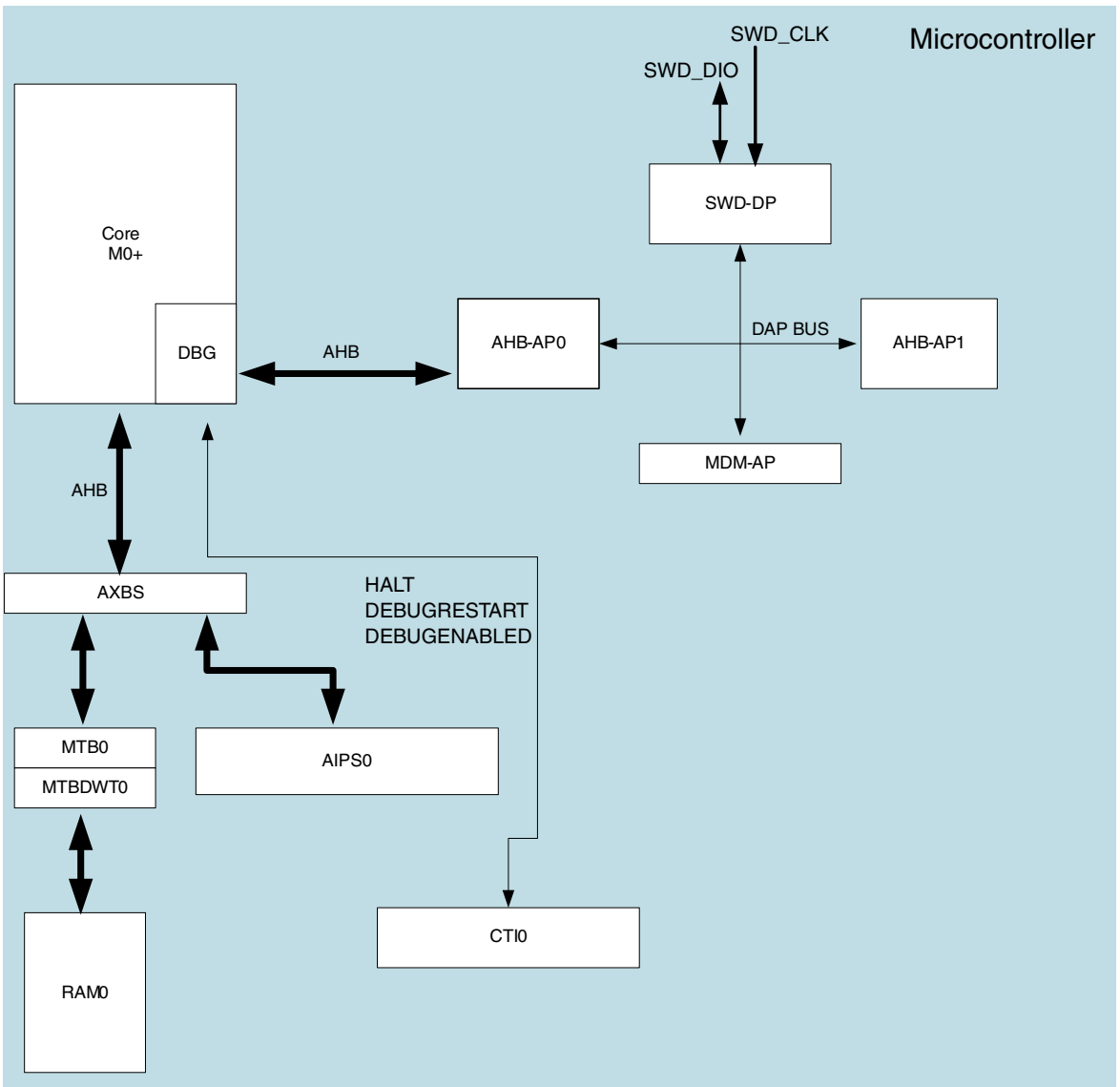


Figure 9-1. Debug and Trace Block Diagram

Details:

- The Debug and Trace uses these ARM-based modules: DP, AHB-AP, DBG, MTB, MTBDWT and CTI.
- The device has 1 ARM based Debug Port (DP) that supports the SWD interface. The DP module connects to a Miscellaneous Debug Module Access Port (MDM-AP) and 2 AHB Access Ports (AHB-AP0 and AHB-AP1).

- The MDM-AP supports flash programming/erase, debug options in the various stop modes, flash security, and resetting the Core. See the MDM-AP registers.
- The AHB-APx module connects to the M0+ core via a private AHB bus, allowing the debugger tool to set up watchpoints, breakpoints, instruction stepping, halting/ starting CPU execution.
- Tracing of code execution deploys ARM's Micro Trace Buffer (MTB) module, which connects to the main AHB-Lite interface and provides the interface to the RAM.

Table 9-2. Debug Access Ports and AP numbers

Debug Access Ports	AP Number
Core0 AHB-AP	AP 0
MDM-AP	AP 1

9.4 SWD status and control registers

Through the ARM Debug Access Port (DAP), the debugger has access to the status and control elements, implemented as registers on the DAP bus as shown in the figure found here.

These registers provide additional control and status for low power mode recovery and typical run-control scenarios. The status register bits also provide a means for the debugger to get updated status of the core, without having to initiate a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

It is important to note that these DAP control and status registers are not memory-mapped within the system memory map, and are only accessible via the Debug Access Port using SWD. The MDM-AP is accessible as Debug Access Port 1 with the available registers shown in this table.

Table 9-3. MDM-AP register summary

Address	Register	Description
0x0100_0000	Status	See MDM-AP Status Register
0x0100_0004	Control	See MDM-AP Control Register
0x0100_00FC	IDR	Read-only identification register that always reads as 0x001C_0020

Core Debug

Via SWD (if the debugger tools can interface to the CPU)

The MDM-AP can support:
-Basic run/halt control
-2 breakpoints
-2 watchpoints

MDM-AP0 has flash erase capability, VLLS debug configuration

MDM-AP0 has:
-Debug disable
-Debug request
-Systems reset request
-Core hold reset

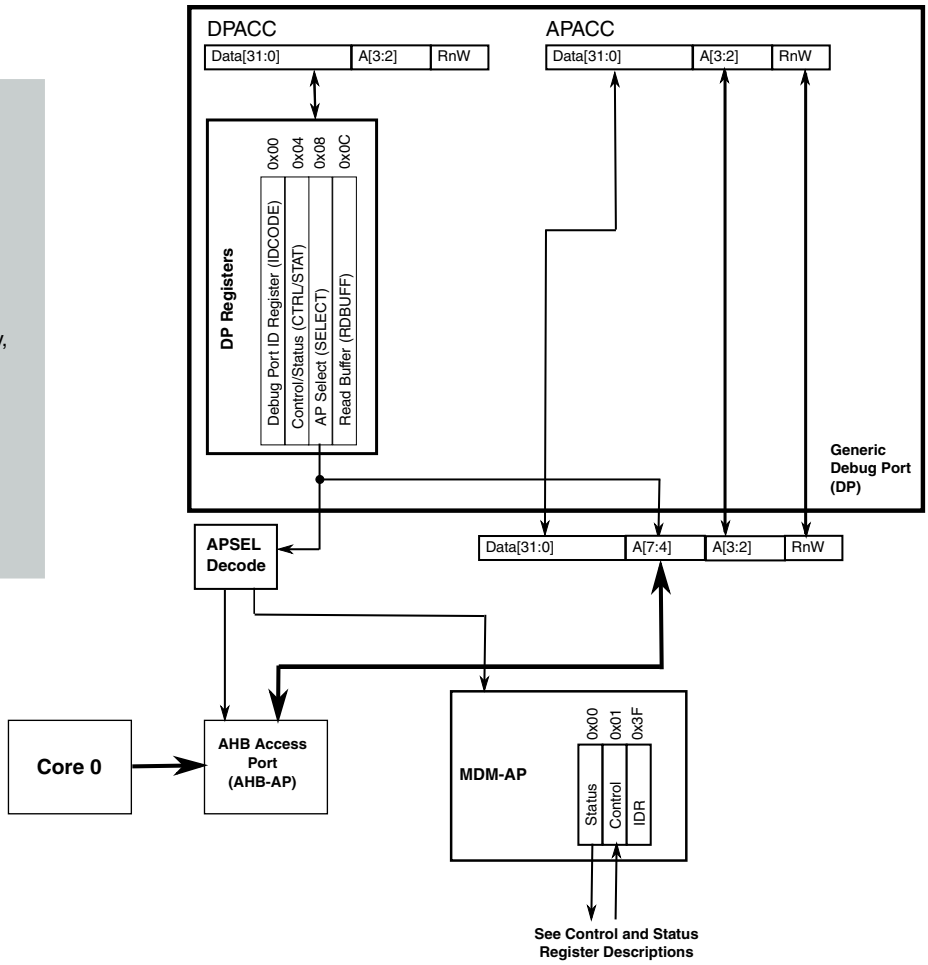


Figure 9-2. MDM AP addressing

9.4.1 MDM-AP Control Register

Table 9-4. MDM-AP Control register assignments

Bit	Name	Secure[1]	Description
0	Flash Mass Erase in Progress	Y	Set to cause mass erase. Cleared by a Power-On Reset (POR). When mass erase is disabled (via MEEN), the erase request does not occur and the Flash Mass Erase in Progress bit continues to assert until the next system reset. **NOTE:** When the reset pin has been disabled and security has been enabled by means of the FSEC register, a mass erase can be performed only by setting both the mass erase and system reset request bits in the MDM-AP register.
1	Debug Disable	N	Set to disable debug. Clear to allow debug operation. When set, it overrides the C_DEBUGEN bit within the DHCSR and force disables Debug logic.
2	Debug Request	N	Set to force the core to halt.

Table continues on the next page...

Table 9-4. MDM-AP Control register assignments (continued)

Bit	Name	Secure[1]	Description
			If the core is in a Stop or Wait mode, then Debug Request bit can be used to wake the core and transition to a halted state.
3	System Reset Request	Y	Set to force a system reset. The system remains held in reset until System Reset Request bit is cleared.
4	Core Hold Reset	N	Configuration bit to control core operation at the end of system reset sequencing. 0 **Normal operation**: At the end of system reset sequencing, release the core from reset (along with the rest of the system). 1 **Suspend operation**: At the end of reset sequencing, hold the core in reset. After the system enters this suspended state, clearing this control bit immediately releases the core from reset and CPU operation begins.
5	VLLSx Debug Request (VLLDBGREQ)	N	Set to configure the system to be held in reset after the next recovery from a VLLSx mode. VLLSx Debug Request bit is ignored on a VLLS wakeup via the Reset pin. During a VLLS wakeup via the Reset pin, the system can be held in reset by holding the reset pin asserted, allowing the debugger to reinitialize the debug modules. VLLSx Debug Request bit holds the system in reset when VLLSx modes are exited, to allow the debugger time to re-initialize debug IP before the debug session continues. The Mode Controller captures VLLSx Debug Request bit logic on entry to VLLSx modes. Upon exit from VLLSx modes, the Mode Controller will hold the system in reset until VLLDBGACK is asserted. VLLDBGREQ clears automatically due to the POR reset generated as part of the VLLSx recovery.
6	VLLSx Debug Acknowledge (VLLDBGACK)	N	Set to release a system being held in reset following a VLLSx recovery VLLSx Debug Acknowledge bit is used by the debugger to release the system reset when it is being held on VLLSx mode exit. The debugger re-initializes all debug IP and then asserts VLLSx Debug Acknowledge bit to allow the Mode Controller to release the system from reset and allow CPU operation to begin. VLLDBGACK is cleared by the debugger or can be left set, because it clears automatically due to the POR reset generated as part of the next VLLSx recovery.
7	LLS, VLLSx Status Acknowledge	N	Set this bit to acknowledge that the DAP LLS and VLLS Status bits have been read. This acknowledge automatically clears the status bits. This bit is used by the debugger to clear the sticky LLS and VLLSx mode entry status bits. This bit is asserted and cleared by the debugger.
8–31	Reserved for future use	N	

1. Command available in secure mode

9.4.2 MDM-AP Status Register

Table 9-5. MDM-AP Status register assignments

Bit	Name	Description
0	Flash Mass Erase Acknowledge	The Flash Mass Erase Acknowledge bit is cleared after a Power-On Reset (POR). The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation. When mass erase is disabled (via MEEN), an erase request due to setting of Flash Mass Erase in Progress bit is not acknowledged.
1	Flash Ready	Indicates Flash has been initialized and the debugger can be configured, even if the system is continuing to be held in reset by the debugger.
2	System Security	Indicates the security state. When secure, the debugger does not have access to the system bus or any memory-mapped peripherals. This bit indicates when the part is locked and no system bus access is possible.
3	System Reset	Indicates the system reset state. 0 System is in reset. 1 System is not in reset.
4	Reserved	
5	Mass Erase Enable	Indicates if the MCU can be mass erased or not 0 Mass erase is disabled. 1 Mass erase is enabled .
6	Backdoor Access Key Enable	Indicates if the MCU has the backdoor access key enabled. 0 Disabled 1 Enabled
7	LP Enabled	Decode of SMC_PMCTRL[STOPM] field to indicate that VLPS, LLS, or VLLSx are the selected power mode for the next time that the ARM Core enters Deep Sleep. 0 Low Power Stop Mode is not enabled. 1 Low Power Stop Mode is enabled. This bit is intended to be used for debug operations in which Run to VLPS is attempted. Per the debug definition, the system actually enters the Stop state. A debugger should interpret a deep sleep indication (with SLEEPDEEP and SLEEPING asserted), in conjunction with LP Enabled bit asserted, as the debugger-VLPS status indication.
8	Very Low Power Mode	Indicates that the current power mode is VLPx. This bit is not ‘sticky’ and should always represent whether VLPx is enabled or not. This bit is used to throttle SWD_CLK frequency up/down.
9	LLS Mode Exit	Indicates an exit from LLS mode has occurred. The debugger will lose communication while the system is in LLS (including access to this register). After communication is re-established, this bit indicates that the system had been in LLS. Because the debug modules held their state during LLS, they do not need to be reconfigured. This bit is set during the LLS recovery sequence. The LLS Mode Exit bit is held until the debugger has had a chance to recognize that LLS was exited, and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register.

Table continues on the next page...

Table 9-5. MDM-AP Status register assignments (continued)

Bit	Name	Description
10	VLLSx Modes Exit	This bit indicates an exit from VLLSx mode has occurred. The debugger will lose communication while the system is in VLLSx (including access to this register). After communication is reestablished, this bit indicates that the system had been in VLLSx. Because the debug modules lose their state during VLLSx modes, they need to be reconfigured. This bit is set during the VLLSx recovery sequence. The VLLSx Mode Exit bit is held until the debugger has had a chance to recognize that a VLLS mode was exited, and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register.
11 – 15	Reserved for future use	Always read 0.
16	CPU0 Core Halted	Indicates that the CPU0 core has entered Debug Halt mode
17	CPU0 Core SLEEPDEEP	Indicates the CPU0 core has entered a low-power mode
18	CPU0 Core SLEEPING	SLEEPING==1 and SLEEPDEEP==0 indicates wait or VLPW mode. SLEEPING==1 and SLEEPDEEP==1 indicates stop or VLPS mode.
20	Reserved	Reserved for future use
21	Reserved	Reserved for future use
22	Reserved	Reserved for future use
23 – 31	Reserved for future use	Always read 0.

9.5 Debug resets

The debug system receives the following sources of reset:

- Debug reset (the CDBGRSTREQ field within the DP CTRL/STAT register) that allows the debugger to reset the debug logic.
- System POR reset

Conversely, the debug system is capable of generating system reset using the following mechanism:

- A system reset in the DAP control register which allows the debugger to hold the system in reset.
- SYSRESETREQ field in the NVIC Application Interrupt and Reset control register
- A system reset in the DAP control register which allows the debugger to hold the core in reset.

9.6 Micro Trace Buffer (MTB)

The Micro Trace Buffer (MTB) provides a simple execution trace capability for the Cortex-M0+ processor.

When enabled, the MTB records changes in program flow reported by the Cortex-M0+ processor, via the execution trace interface, into a configurable region of the SRAM. Subsequently, an off-chip debugger may extract the trace information, which would allow reconstruction of an instruction flow trace. The MTB does not include any form of load/store data trace capability or tracing of any other information.

In addition to providing the trace capability, the MTB also operates as a simple AHB-Lite SRAM controller. The system bus masters, including the processor, have read/write access to all of the SRAM via the AHB-Lite interface, allowing the memory to be also used to store program and data information. The MTB simultaneously stores the trace information into an attached SRAM and allows bus masters to access the memory. The MTB ensures that trace information write accesses to the SRAM take priority over accesses from the AHB-Lite interface.

The MTB includes trace control registers for configuring and triggering the MTB functions. The MTB also supports triggering via TSTART and TSTOP control functions in the MTB DWT module.

9.7 Debug in low-power modes

In low-power modes, in which the debug modules are kept static or powered off, the debugger cannot gather any debug data for the duration of the low-power mode.

- In the case that the debugger is held static, the debug port returns to full functionality as soon as the low-power mode exits and the system returns to a state with active debug.
- In the case that the debugger logic is powered off, the debugger is reset on recovery and must be reconfigured once the low-power mode is exited.

Power mode entry logic monitors Debug Power Up and System Power Up signals from the debug port as indications that a debugger is active. These signals can be changed in RUN, VLPR, WAIT and VLPW. If the debug signal is active and the system attempts to enter Stop or VLPS, FCLK continues to run to support core register access. In these modes in which FCLK is left active the debug modules have access to core registers but not to system memory resources accessed via the crossbar.

With debug enabled, transitions from Run directly to VLPS result in the system entering Stop mode instead. Status bits within the MDM-AP Status register can be evaluated to determine this pseudo-VLPS state.

NOTE

With the debug enabled, transitions from Run --> VLPR --> VLPS are still possible.

In VLLS mode, all debug modules are powered off and reset at wakeup. In LLS mode, the debug modules retain their state but no debug activity is possible.

Going into a VLLSx mode causes all the debug controls and settings to be reset. To give time to the debugger to sync up with the HW, the MDM-AP Control register can be configured to hold the system in reset on recovery so that the debugger can regain control and reconfigure debug logic prior to the system exiting reset and resuming operation.

9.8 Debug and security

When flash security is enabled [FSEC[SEC]!=10], the debug port capabilities are limited in order to prevent exploitation of secure data.

In the secure state, the debugger still has access to the MDM-AP Status Register and can determine the current security state of the device. In the case of a secure device, that has mass erase disabled (FSEC[MEEN] = 10), attempts to mass erase via the debug interface are blocked. When mass erase is disabled (FSEC[MEEN]= 10), the debugger does not have the capability of performing a mass erase operation via writes to MDM-AP Control Register.

Chapter 10
Signal Multiplexing and Signal Descriptions

10.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. Information found here illustrates which of this device's signals are multiplexed on which external pin.

The Port Control block controls which signal is present on the external pin. Refer to that chapter to find which register controls the operation of a specific pin.

For more information about how the Port Control block is integrated into this device, refer to the Signal multiplexing integration section.

10.2 Pinout

10.2.1 Package types

KL28Z device packages include:

- 121-pin XFBGA (Package Your Way)
- 100-pin QFP

10.2.2 KL28Z Signal Multiplexing and Pin Assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT function is available on each pin.

NOTE

The 121-pin XFBGA package for this product is not yet available. However, it is included in a Package Your Way program for Kinetis MCUs. Visit nxp.com/KPYW for more details.

NOTE

- A pull-up resistor (typically 4.7 KΩ) must be connected to the EMVSIM0_IO pin if this pin is configured as EMV SIM function.
- PTB0/1, PTC3/4, PTD4/5/6/7 have both high drive and normal/low drive capability. PTD4, PTD5, PTD6, PTD7, PTE20, PTE21, PTE22, PTE23 are also fast pins. When a high bit rate is required on the communication interface pins, it is recommended to use fast pins. In case of high bus loading, the high drive strength of high drive pins must be enabled by setting the corresponding PORTx_PCRn[DSE] bit.
- RESET_b pin is open drain with internal pullup device and passive analog filter when configured as RESET pin (default state after POR). When this pin is configured to other shared functions, the passive analog filter is disabled.
- NMI0_b pin has pullup device enabled and passive analog filter disabled after POR.
- SWD_DIO pin has pullup device enabled after POR. SWD_CLK has pulldown device enabled after POR.

121 XFBGA	100 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
E4	1	PTE0	ADC0_SE16	ADC0_SE16	PTE0/ RTC_CLKOUT	LPSPI1_SIN	LPUART1_TX		CMP0_OUT	LPI2C1_SDA	
E3	2	PTE1/ LLWU_P0	ADC0_SE17	ADC0_SE17	PTE1/ LLWU_P0	LPSPI1_SOUT	LPUART1_RX			LPI2C1_SCL	
E2	3	PTE2/ LLWU_P1	ADC0_SE18	ADC0_SE18	PTE2/ LLWU_P1	LPSPI1_SCK	LPUART1_ CTS_b			LPI2C1_SDAS	
F4	4	PTE3	ADC0_SE19	ADC0_SE19	PTE3	LPSPI1_SIN	LPUART1_ RTS_b			LPI2C1_SCLS	
H7	5	PTE4/ LLWU_P2	DISABLED		PTE4/ LLWU_P2	LPSPI1_PCS0					
G4	6	PTE5	DISABLED		PTE5	LPSPI1_PCS1					
F3	7	PTE6/ LLWU_P16	DISABLED		PTE6/ LLWU_P16	LPSPI1_PCS2		I2S0_MCLK	USB_SOF_ OUT		
E6	8	VDD	VDD	VDD							
G7	9	VSS	VSS	VSS							

121 XFBGA	100 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
L6	—	VSS	VSS	VSS							
F1	10	USB0_DP	USB0_DP	USB0_DP							
F2	11	USB0_DM	USB0_DM	USB0_DM							
G1	12	VOUT33	VOUT33	VOUT33							
G2	13	VREGIN	VREGIN	VREGIN							
H1	14	PTE16	ADC0_DP1/ ADC0_SE1	ADC0_DP1/ ADC0_SE1	PTE16	LPSPI0_PCS0	LPUART2_TX	TPM0_CLKIN	LPSPI1_PCS3	FXIO0_D0	
H2	15	PTE17/ LLWU_P19	ADC0_DM1/ ADC0_SE5a	ADC0_DM1/ ADC0_SE5a	PTE17/ LLWU_P19	LPSPI0_SCK	LPUART2_RX	TPM1_CLKIN	LPTMR0_ ALT3/ LPTMR1_ALT3	FXIO0_D1	
J1	16	PTE18/ LLWU_P20	ADC0_DP2/ ADC0_SE2	ADC0_DP2/ ADC0_SE2	PTE18/ LLWU_P20	LPSPI0_SOUT	LPUART2_ CTS_b	LPI2C0_SDA		FXIO0_D2	
J2	17	PTE19	ADC0_DM2/ ADC0_SE6a	ADC0_DM2/ ADC0_SE6a	PTE19	LPSPI0_SIN	LPUART2_ RTS_b	LPI2C0_SCL		FXIO0_D3	
K1	18	PTE20	ADC0_DP0/ ADC0_SE0	ADC0_DP0/ ADC0_SE0	PTE20	LPSPI2_SCK	TPM1_CH0	LPUART0_TX		FXIO0_D4	
K2	19	PTE21	ADC0_DM0/ ADC0_SE4a	ADC0_DM0/ ADC0_SE4a	PTE21	LPSPI2_SOUT	TPM1_CH1	LPUART0_RX		FXIO0_D5	
L1	20	PTE22	ADC0_DP3/ ADC0_SE3	ADC0_DP3/ ADC0_SE3	PTE22	LPSPI2_SIN	TPM2_CH0	LPUART2_TX		FXIO0_D6	
L2	21	PTE23	ADC0_DM3/ ADC0_SE7a	ADC0_DM3/ ADC0_SE7a	PTE23	LPSPI2_PCS0	TPM2_CH1	LPUART2_RX		FXIO0_D7	
F5	22	VDDA	VDDA	VDDA							
G5	23	VREFH/ VREF_OUT	VREFH/ VREF_OUT	VREFH/ VREF_OUT							
G6	24	VREFL	VREFL	VREFL							
F6	25	VSSA	VSSA	VSSA							
L3	26	PTE29	CMP1_IN5/ CMP0_IN5/ ADC0_SE4b	CMP1_IN5/ CMP0_IN5/ ADC0_SE4b	PTE29	EMVSIM0_CLK	TPM0_CH2	TPM0_CLKIN			
K5	27	PTE30	DAC0_OUT/ CMP1_IN3/ ADC0_SE23/ CMP0_IN4	DAC0_OUT/ CMP1_IN3/ ADC0_SE23/ CMP0_IN4	PTE30	EMVSIM0_ RST	TPM0_CH3	TPM1_CLKIN			
L4	28	PTE31	DISABLED		PTE31	EMVSIM0_ VCCEN	TPM0_CH4	TPM2_CLKIN	LPI2C0_HREQ		
L5	29	VSS	VSS	VSS							
K6	30	VDD	VDD	VDD							
H5	31	PTE24	ADC0_SE20	ADC0_SE20	PTE24	EMVSIM0_IO	TPM0_CH0		LPI2C0_SCL		
J5	32	PTE25/ LLWU_P21	ADC0_SE21	ADC0_SE21	PTE25/ LLWU_P21	EMVSIM0_PD	TPM0_CH1		LPI2C0_SDA		
H6	33	PTE26	DISABLED		PTE26/ RTC_CLKOUT		TPM0_CH5		LPI2C0_SCLS		USB_CLKIN
J6	34	PTA0	SWD_CLK	TSI0_CH1	PTA0	LPUART0_ CTS_b	TPM0_CH5		LPI2C0_SDAS		SWD_CLK
H8	35	PTA1	DISABLED	TSI0_CH2	PTA1	LPUART0_RX	TPM2_CH0				

121 XFBGA	100 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
J7	36	PTA2	DISABLED	TSI0_CH3	PTA2	LPUART0_TX	TPM2_CH1				
H9	37	PTA3	SWD_DIO	TSI0_CH4	PTA3	LPI2C1_SCL	TPM0_CH0	LPUART0_RTS_b			SWD_DIO
J8	38	PTA4/ LLWU_P3	DISABLED	TSI0_CH5	PTA4/ LLWU_P3	LPI2C1_SDA	TPM0_CH1				NMI0_b
K7	39	PTA5	DISABLED		PTA5	USB_CLKIN	TPM0_CH2		LPI2C2_HREQ	I2S0_TX_BCLK	
E5	—	VDD	VDD	VDD							
G3	—	VSS	VSS	VSS							
K3	40	PTA6	DISABLED		PTA6		TPM0_CH3				
H4	41	PTA7	DISABLED		PTA7	LPSPI0_PCS3	TPM0_CH4		LPI2C2_SDAS		
J9	—	PTA10/ LLWU_P22	DISABLED		PTA10/ LLWU_P22	LPSPI0_PCS2	TPM2_CH0		LPI2C2_SCLS		
J4	—	PTA11/ LLWU_P23	DISABLED		PTA11/ LLWU_P23	LPSPI0_PCS1	TPM2_CH1		LPI2C2_SDA		
K8	42	PTA12	DISABLED		PTA12		TPM1_CH0		LPI2C2_SCL	I2S0_TXD0	
L8	43	PTA13/ LLWU_P4	DISABLED		PTA13/ LLWU_P4		TPM1_CH1		LPI2C2_SDA	I2S0_TX_FS	
K9	44	PTA14	DISABLED		PTA14	LPSPI0_PCS0	LPUART0_TX		LPI2C2_SCL	I2S0_RX_BCLK	I2S0_TXD0
L9	45	PTA15	DISABLED		PTA15	LPSPI0_SCK	LPUART0_RX			I2S0_RXD0	
J10	46	PTA16	DISABLED		PTA16	LPSPI0_SOUT	LPUART0_CTS_b			I2S0_RX_FS	I2S0_RXD0
H10	47	PTA17	ADC0_SE22	ADC0_SE22	PTA17	LPSPI0_SIN	LPUART0_RTS_b			I2S0_MCLK	
L10	48	VDD	VDD	VDD							
K10	49	VSS	VSS	VSS							
L11	50	PTA18	EXTAL0	EXTAL0	PTA18		LPUART1_RX	TPM0_CLKIN			
K11	51	PTA19	XTAL0	XTAL0	PTA19		LPUART1_TX	TPM1_CLKIN		LPTMR0_ALT1/ LPTMR1_ALT1	
J11	52	PTA20	RESET_b		PTA20	LPI2C0_SCLS		TPM2_CLKIN			RESET_b
H11	—	PTA29	DISABLED		PTA29	LPI2C0_SDAS					
G11	53	PTB0/ LLWU_P5	ADC0_SE8/ TSI0_CH0	ADC0_SE8/ TSI0_CH0	PTB0/ LLWU_P5	LPI2C0_SCL	TPM1_CH0			FXIO0_D8	
G10	54	PTB1	ADC0_SE9/ TSI0_CH6	ADC0_SE9/ TSI0_CH6	PTB1	LPI2C0_SDA	TPM1_CH1			FXIO0_D9	
G9	55	PTB2	ADC0_SE12/ TSI0_CH7	ADC0_SE12/ TSI0_CH7	PTB2	LPI2C0_SCL	TPM2_CH0		LPUART0_RTS_b	FXIO0_D10	
G8	56	PTB3	ADC0_SE13/ TSI0_CH8	ADC0_SE13/ TSI0_CH8	PTB3	LPI2C0_SDA	TPM2_CH1	LPSPI1_PCS3	LPUART0_CTS_b	FXIO0_D11	
F11	—	PTB6	DISABLED		PTB6	LPSPI1_PCS2					
E11	57	PTB7	DISABLED		PTB7	LPSPI1_PCS1					
D11	58	PTB8	DISABLED		PTB8	LPSPI1_PCS0				FXIO0_D12	
E10	59	PTB9	DISABLED		PTB9	LPSPI1_SCK				FXIO0_D13	

121 XFBGA	100 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
D10	60	PTB10	DISABLED		PTB10	LPSPI1_PCS0				FXIO0_D14	
C10	61	PTB11	DISABLED		PTB11	LPSPI1_SCK		TPM2_CLKIN		FXIO0_D15	
B10	62	PTB16	TSI0_CH9	TSI0_CH9	PTB16	LPSPI1_SOUT	LPUART0_RX	TPM0_CLKIN	LPSPI2_PCS3	FXIO0_D16	
E9	63	PTB17	TSI0_CH10	TSI0_CH10	PTB17	LPSPI1_SIN	LPUART0_TX	TPM1_CLKIN	LPSPI2_PCS2	FXIO0_D17	
D9	64	PTB18	TSI0_CH11	TSI0_CH11	PTB18		TPM2_CH0	I2S0_TX_BCLK	LPI2C1_HREQ	FXIO0_D18	
C9	65	PTB19	TSI0_CH12	TSI0_CH12	PTB19		TPM2_CH1	I2S0_TX_FS	LPSPI2_PCS1	FXIO0_D19	
F10	66	PTB20	DISABLED		PTB20	LPSPI2_PCS0				CMP0_OUT	
F9	67	PTB21	DISABLED		PTB21	LPSPI2_SCK				CMP1_OUT	
F8	68	PTB22	DISABLED		PTB22	LPSPI2_SOUT					
E8	69	PTB23	DISABLED		PTB23	LPSPI2_SIN					
B9	70	PTC0	ADC0_SE14/ TSI0_CH13	ADC0_SE14/ TSI0_CH13	PTC0	LPSPI2_PCS1		USB_SOF_OUT	CMP0_OUT	I2S0_TXD0	
D8	71	PTC1/ LLWU_P6	ADC0_SE15/ TSI0_CH14	ADC0_SE15/ TSI0_CH14	PTC1/ LLWU_P6	LPI2C1_SCL	LPUART1_RTS_b	TPM0_CH0		I2S0_TXD0	
C8	72	PTC2	ADC0_SE11/ CMP1_IN0/ TSI0_CH15	ADC0_SE11/ CMP1_IN0/ TSI0_CH15	PTC2	LPI2C1_SDA	LPUART1_CTS_b	TPM0_CH1		I2S0_TX_FS	
B8	73	PTC3/ LLWU_P7	CMP1_IN1	CMP1_IN1	PTC3/ LLWU_P7	LPSPI0_PCS1	LPUART1_RX	TPM0_CH2	CLKOUT	I2S0_TX_BCLK	
F7	74	VSS	VSS	VSS							
E7	75	VDD	VDD	VDD							
B11	—	PTC22	DISABLED		PTC22	LPSPI0_PCS3					
C11	—	PTC23	DISABLED		PTC23	LPSPI0_PCS2					
A8	76	PTC4/ LLWU_P8	DISABLED		PTC4/ LLWU_P8	LPSPI0_PCS0	LPUART1_TX	TPM0_CH3	I2S0_MCLK	CMP1_OUT	
D7	77	PTC5/ LLWU_P9	DISABLED		PTC5/ LLWU_P9	LPSPI0_SCK	LPTMR0_ALT2/ LPTMR1_ALT2	I2S0_RXD0		CMP0_OUT	
C7	78	PTC6/ LLWU_P10	CMP0_IN0	CMP0_IN0	PTC6/ LLWU_P10	LPSPI0_SOUT		I2S0_RX_BCLK		I2S0_MCLK	
B7	79	PTC7	CMP0_IN1	CMP0_IN1	PTC7	LPSPI0_SIN	USB_SOF_OUT	I2S0_RX_FS		FXIO0_D20	
A7	80	PTC8	CMP0_IN2	CMP0_IN2	PTC8	LPI2C0_SCL	TPM0_CH4	I2S0_MCLK		FXIO0_D21	
D6	81	PTC9	CMP0_IN3	CMP0_IN3	PTC9	LPI2C0_SDA	TPM0_CH5	I2S0_RX_BCLK		FXIO0_D22	
C6	82	PTC10	DISABLED		PTC10	LPI2C1_SCL		I2S0_RX_FS		FXIO0_D23	
C5	83	PTC11/ LLWU_P11	DISABLED		PTC11/ LLWU_P11	LPI2C1_SDA		I2S0_RXD0			
B6	84	PTC12	DISABLED		PTC12	LPI2C1_SCLS		TPM0_CLKIN			
A6	85	PTC13	DISABLED		PTC13	LPI2C1_SDAS		TPM1_CLKIN			
A5	86	PTC14	DISABLED		PTC14	EMVSIM0_CLK					
B5	87	PTC15	DISABLED		PTC15	EMVSIM0_RST					

121 XFBGA	100 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
A11	88	VSS	VSS	VSS							
—	89	VDD	VDD	VDD							
D5	90	PTC16	DISABLED		PTC16	EMVSIM0_ VCCEN					
C4	91	PTC17	DISABLED		PTC17	EMVSIM0_IO	LPSPI0_PCS3				
B4	92	PTC18	DISABLED		PTC18	EMVSIM0_PD	LPSPI0_PCS2				
A4	—	PTC19	DISABLED		PTC19	LPSPI0_PCS1					
D4	93	PTD0/ LLWU_P12	DISABLED		PTD0/ LLWU_P12	LPSPI0_PCS0	LPUART2_ RTS_b	TPM0_CH0		FXIO0_D0	
D3	94	PTD1	ADC0_SE5b	ADC0_SE5b	PTD1	LPSPI0_SCK	LPUART2_ CTS_b	TPM0_CH1		FXIO0_D1	
C3	95	PTD2/ LLWU_P13	DISABLED		PTD2/ LLWU_P13	LPSPI0_SOUT	LPUART2_RX	TPM0_CH2		FXIO0_D2	
B3	96	PTD3	DISABLED		PTD3	LPSPI0_SIN	LPUART2_TX	TPM0_CH3		FXIO0_D3	
A3	97	PTD4/ LLWU_P14	DISABLED		PTD4/ LLWU_P14	LPSPI1_PCS0	LPUART2_RX	TPM0_CH4	LPUART0_ RTS_b	FXIO0_D4	
A2	98	PTD5	ADC0_SE6b	ADC0_SE6b	PTD5	LPSPI1_SCK	LPUART2_TX	TPM0_CH5	LPUART0_ CTS_b	FXIO0_D5	
B2	99	PTD6/ LLWU_P15	ADC0_SE7b	ADC0_SE7b	PTD6/ LLWU_P15	LPSPI1_SOUT	LPUART0_RX			FXIO0_D6	
A1	100	PTD7	DISABLED		PTD7	LPSPI1_SIN	LPUART0_TX			FXIO0_D7	
A10	—	PTD8/ LLWU_P24	DISABLED		PTD8/ LLWU_P24	LPI2C0_SCL	LPSPI1_PCS1			FXIO0_D24	
A9	—	PTD9	DISABLED		PTD9	LPI2C0_SDA	LPSPI2_PCS3			FXIO0_D25	
B1	—	PTD10	DISABLED		PTD10	LPSPI2_PCS2	LPI2C0_SCLS			FXIO0_D26	
C2	—	PTD11/ LLWU_P25	DISABLED		PTD11/ LLWU_P25	LPSPI2_PCS0	LPI2C0_SDAS			FXIO0_D27	
C1	—	PTD12	DISABLED		PTD12	LPSPI2_SCK				FXIO0_D28	
D2	—	PTD13	DISABLED		PTD13	LPSPI2_SOUT				FXIO0_D29	
D1	—	PTD14	DISABLED		PTD14	LPSPI2_SIN				FXIO0_D30	
E1	—	PTD15	DISABLED		PTD15	LPSPI2_PCS1				FXIO0_D31	
J3	—	NC	NC	NC							
H3	—	NC	NC	NC							
K4	—	NC	NC	NC							
L7	—	NC	NC	NC							

10.2.3 KL28Z Pinouts

The below figure shows the pinout diagram for the devices supported by this document. Many signals may be multiplexed onto a single pin. To determine what signals can be used on which pin, see the previous section.

	1	2	3	4	5	6	7	8	9	10	11	
A	PTD7	PTD5	PTD4/ LLWU_P14	PTC19	PTC14	PTC13	PTC8	PTC4/ LLWU_P8	PTD9	PTD8/ LLWU_P24	VSS	A
B	PTD10	PTD6/ LLWU_P15	PTD3	PTC18	PTC15	PTC12	PTC7	PTC3/ LLWU_P7	PTC0	PTB16	PTC22	B
C	PTD12	PTD11/ LLWU_P25	PTD2/ LLWU_P13	PTC17	PTC11/ LLWU_P11	PTC10	PTC6/ LLWU_P10	PTC2	PTB19	PTB11	PTC23	C
D	PTD14	PTD13	PTD1	PTD0/ LLWU_P12	PTC16	PTC9	PTC5/ LLWU_P9	PTC1/ LLWU_P6	PTB18	PTB10	PTB8	D
E	PTD15	PTE2/ LLWU_P1	PTE1/ LLWU_P0	PTE0	VDD	VDD	VDD	PTB23	PTB17	PTB9	PTB7	E
F	USB0_DP	USB0_DM	PTE6/ LLWU_P16	PTE3	VDDA	VSSA	VSS	PTB22	PTB21	PTB20	PTB6	F
G	VOUT33	VREGIN	VSS	PTE5	VREFH/ VREF_OUT	VREFL	VSS	PTB3	PTB2	PTB1	PTB0/ LLWU_P5	G
H	PTE16	PTE17/ LLWU_P19	NC	PTA7	PTE24	PTE26	PTE4/ LLWU_P2	PTA1	PTA3	PTA17	PTA29	H
J	PTE18/ LLWU_P20	PTE19	NC	PTA11/ LLWU_P23	PTE25/ LLWU_P21	PTA0	PTA2	PTA4/ LLWU_P3	PTA10/ LLWU_P22	PTA16	PTA20	J
K	PTE20	PTE21	PTA6	NC	PTE30	VDD	PTA5	PTA12	PTA14	VSS	PTA19	K
L	PTE22	PTE23	PTE29	PTE31	VSS	VSS	NC	PTA13/ LLWU_P4	PTA15	VDD	PTA18	L
	1	2	3	4	5	6	7	8	9	10	11	

Figure 10-1. 121 XFBGA Pinout Diagram

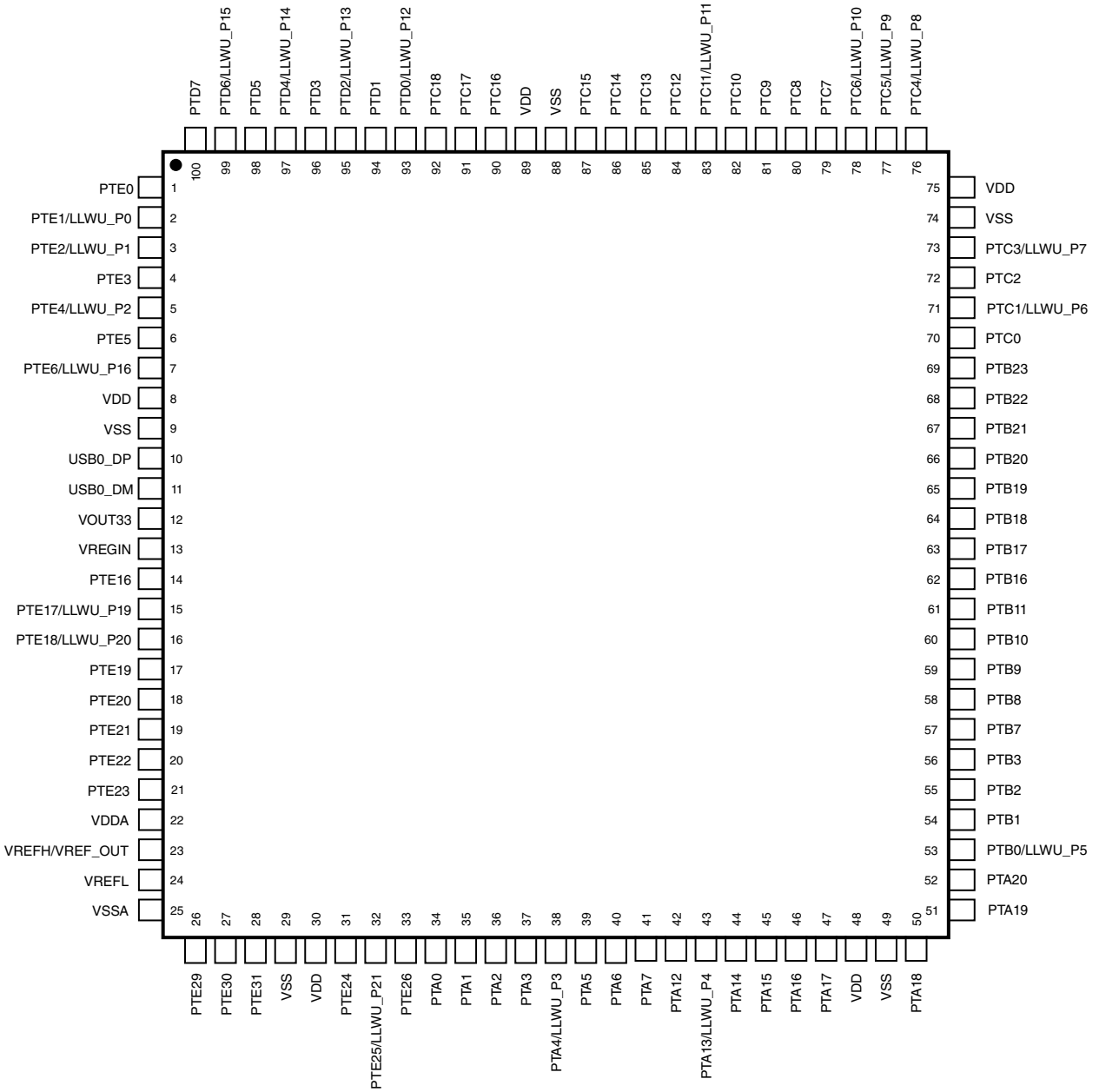


Figure 10-2. 100 LQFP Pinout Diagram

10.3 Module Signal Description Tables

10.3.1 Core modules

Table 10-1. SWD signal descriptions

Chip signal name	Module signal name	Description	I/O
SWD_DIO	SWD_DIO	Serial Wire Debug Data Input/Output The SWD_DIO pin is used by an external debug tool for communication and device control. This pin is pulled up internally.	Input / Output
SWD_CLK	SWD_CLK	Serial Wire Clock This pin is the clock for debug logic when in the Serial Wire Debug mode. This pin is pulled down internally.	Input

10.3.2 System modules

Table 10-2. System signal descriptions

Chip signal name	Module signal name	Description	I/O
NMI0_b	—	Non-maskable interrupt Driving the NMI0_b signal low forces a non-maskable interrupt, if the NMI function is selected on the corresponding pin. **NOTE:** If the NMI function is not required, either for an interrupt or wake-up source, it is recommended that the NMI function be disabled by clearing NMI_DIS.	I
RESET_b	—	Reset bidirectional signal	I/O
VDD	—	MCU power	I
VSS	—	MCU ground	I

10.3.3 Clock modules

Table 10-3. OSC signal descriptions

Chip signal name	Module signal name	Description	I/O
EXTAL0	EXTAL	External clock/Oscillator input	I
XTAL0	XTAL	Oscillator output	O

10.3.4 Memories and memory interfaces

10.3.5 Analog

This table presents the signal descriptions of the ADC0 module.

Table 10-4. ADC0 signal descriptions

Chip signal name	Module signal name	Description	I/O
ADC0_DPn	DADP3–DADP0	Differential Analog Channel Inputs	I
ADC0_DMn	DADM3–DADM0	Differential Analog Channel Inputs	I
ADC0_SEn	AD*n*	Single-Ended Analog Channel Inputs	I
VREFH	V_{REFSH}	Voltage Reference Select High	I
VREFL	V_{REFSL}	Voltage Reference Select Low	I
VDDA	V_{DDA}	Analog Power Supply	I
VSSA	V_{SSA}	Analog Ground	I

This table presents the signal descriptions of the CMP0 module.

Table 10-5. CMP0 signal descriptions

Chip signal name	Module signal name	Description	I/O
CMP0_IN[5:0]	IN[5:0]	Analog voltage inputs	I
CMP0_OUT	CMPO	Comparator output	O

This table presents the signal descriptions of the DAC0 module.

Table 10-6. DAC0 signal descriptions

Chip signal name	Module signal name	Description	I/O
DAC0_OUT	—	DAC output	O

10.3.6 Timer Modules

Table 10-7. TPM0 signal descriptions

Chip signal name	Module signal name	Description	I/O
TPM0_CLKIN	TPM_EXTCLK	External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock.	I

Table continues on the next page...

Table 10-7. TPM0 signal descriptions (continued)

Chip signal name	Module signal name	Description	I/O
TPM0_CH[5:0]	TPM_CHn	TPM channel. A TPM channel pin is configured as output when configured in an output compare, or in PWM mode and the TPM counter is enabled; otherwise the TPM channel pin is an input.	I/O

Table 10-8. TPM1 signal descriptions

Chip signal name	Module signal name	Description	I/O
TPM1_CLKIN	TPM_EXTCLK	External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock.	I
TPM1_CH[1:0]	TPM_CHn	TPM channel. A TPM channel pin is configured as output when configured in an output compare, or in PWM mode and the TPM counter is enabled; otherwise the TPM channel pin is an input.	I/O

Table 10-9. TPM2 signal descriptions

Chip signal name	Module signal name	Description	I/O
TPM2_CLKIN	TPM_EXTCLK	External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock.	I
TPM2_CH[1:0]	TPM_CHn	TPM channel. A TPM channel pin is configured as output when configured in an output compare, or in PWM mode and the TPM counter is enabled; otherwise the TPM channel pin is an input.	I/O

Table 10-10. LPTMR0 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPTMR0_ALT[3:1]	LPTMR0_ALTn	Pulse Counter Input pin	I
LPTMR1_ALT[3:1]	LPTMR1_ALTn	Pulse Counter Input pin	I

Table 10-11. RTC signal descriptions

Chip signal name	Module signal name	Description	I/O
RTC_CLKOUT	RTC_CLKOUT	1 Hz square-wave output or OSC32KCLK	O

10.3.7 Communication interfaces

Table 10-12. USB FS OTG Signal Descriptions

Chip signal name	Module signal name	Description	I/O
USB0_DM	usb_dm	USB D- analog data signal on the USB bus.	I/O
USB0_DP	usb_dp	USB D+ analog data signal on the USB bus.	I/O
USB_CLKIN	—	Alternate USB clock input	I

Table 10-13. USB VREG Signal Descriptions

Chip signal name	Module signal name	Description	I/O
VOUT33	reg33_out	Regulator output voltage	O
VREGIN	reg33_in	Unregulated power supply	I

Table 10-14. LPSPI0 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPSPI0_SOUT	SOUT / DATA[0]	Serial Data Output. Can be configured as serial data input signal. Used as data pin 0 in quad-data and dual-data transfers.	I/O
LPSPI0_SIN	SIN / DATA[1]	Serial Data Input. Can be configured as serial data output signal. Used as data pin 1 in quad-data and dual-data transfers.	I/O
LPSPI0_SCK	SCK	Serial clock. Input in slave mode, output in master mode.	I/O
LPSPI0_PCS[0]	PCS[0]	Peripheral Chip Select. Input in slave mode, output in master mode.	I/O
LPSPI0_PCS[1]	PCS[1] / HREQ	Peripheral Chip Select or Host Request. Host Request pin is selected when HREN=1 and HRSEL=0. Input in either slave mode or when used as Host Request, output in master mode.	I/O
LPSPI0_PCS[2]	PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
LPSPI0_PCS[3]	PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O

Table 10-15. LPSPI1 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPSPI1_SOUT	SOUT / DATA[0]	Serial Data Output. Can be configured as serial data input signal. Used as data pin 0 in quad-data and dual-data transfers.	I/O
LPSPI1_SIN	SIN / DATA[1]	Serial Data Input. Can be configured as serial data output signal. Used as data pin 1 in quad-data and dual-data transfers.	I/O
LPSPI1_SCK	SCK	Serial clock. Input in slave mode, output in master mode.	I/O
LPSPI1_PCS[0]	PCS[0]	Peripheral Chip Select. Input in slave mode, output in master mode.	I/O

Table continues on the next page...

Table 10-15. LPSPI1 signal descriptions (continued)

Chip signal name	Module signal name	Description	I/O
LPSPI1_PCS[1]	PCS[1] / HREQ	Peripheral Chip Select or Host Request. Host Request pin is selected when HREN=1 and HRSEL=0. Input in either slave mode or when used as Host Request, output in master mode.	I/O
LPSPI1_PCS[2]	PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
LPSPI1_PCS[3]	PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O

Table 10-16. LPSPI2 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPSPI2_SOUT	SOUT / DATA[0]	Serial Data Output. Can be configured as serial data input signal. Used as data pin 0 in quad-data and dual-data transfers.	I/O
LPSPI2_SIN	SIN / DATA[1]	Serial Data Input. Can be configured as serial data output signal. Used as data pin 1 in quad-data and dual-data transfers.	I/O
LPSPI2_SCK	SCK	Serial clock. Input in slave mode, output in master mode.	I/O
LPSPI2_PCS[0]	PCS[0]	Peripheral Chip Select. Input in slave mode, output in master mode.	I/O
LPSPI2_PCS[1]	PCS[1] / HREQ	Peripheral Chip Select or Host Request. Host Request pin is selected when HREN=1 and HRSEL=0. Input in either slave mode or when used as Host Request, output in master mode.	I/O
LPSPI2_PCS[2]	PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
LPSPI2_PCS[3]	PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O

Table 10-17. LPI^{2}C0 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPI2C0_SCL	SCL	LPI2C clock line. In 4-wire mode, this is the SCL input pin.	I/O
LPI2C0_SDA	SDA	LPI2C data line. In 4-wire mode, this is the SDA input pin.	I/O
LPI2C0_	HREQ	Host request; can initiate an LPI2C master transfer if asserted and the I2C bus is idle.	I
LPI2C0_SCLS	SCLS	Secondary I2C clock line. In 4-wire mode, this is the SCL output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SCL pin.	I/O
LPI2C0_SDAS	SDAS	Secondary I2C data line. In 4-wire mode, this is the SDA output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SDA pin.	I/O

Table 10-18. LPI2C1 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPI2C1_SCL	SCL	LPI2C clock line. In 4-wire mode, this is the SCL input pin.	I/O
LPI2C1_SDA	SDA	LPI2C data line. In 4-wire mode, this is the SDA input pin.	I/O
LPI2C1_HREQ	HREQ	Host request; can initiate an LPI2C master transfer if asserted and the I2C bus is idle.	I
LPI2C1_SCLS	SCLS	Secondary I2C clock line. In 4-wire mode, this is the SCL output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SCL pin.	I/O
LPI2C1_SDAS	SDAS	Secondary I2C data line. In 4-wire mode, this is the SDA output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SDA pin.	I/O

Table 10-19. LPI2C2 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPI2C2_SCL	SCL	LPI2C clock line. In 4-wire mode, this is the SCL input pin.	I/O
LPI2C2_SDA	SDA	LPI2C data line. In 4-wire mode, this is the SDA input pin.	I/O
LPI2C2_HREQ	HREQ	Host request; can initiate an LPI2C master transfer if asserted and the I2C bus is idle.	I
LPI2C2_SCLS	SCLS	Secondary I2C clock line. In 4-wire mode, this is the SCL output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SCL pin.	I/O
LPI2C2_SDAS	SDAS	Secondary I2C data line. In 4-wire mode, this is the SDA output pin. If LPI2C master/slave are configured to use separate pins, then this is the LPI2C slave SDA pin.	I/O

Table 10-20. LPUART0 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPUART0_TX	TxD	Transmit data	O
LPUART0_RX	RxD	Receive data	I
LPUART0_ CTS_b	LPUART_CTS	Clear to send.	I
LPUART0_ RTS_b	LPUART_RTS	Request to send.	O

Table 10-21. LPUART1 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPUART1_TX	TxD	Transmit data	O
LPUART1_RX	RxD	Receive data	I
LPUART1_ CTS_b	LPUART_CTS	Clear to send.	I
LPUART1_ RTS_b	LPUART_RTS	Request to send.	O

Table 10-22. LPUART2 signal descriptions

Chip signal name	Module signal name	Description	I/O
LPUART2_TX	TxD	Transmit data	O
LPUART2_RX	RxD	Receive data	I
LPUART2_ CTS_b	LPUART_CTS	Clear to send.	I
LPUART2_ RTS_b	LPUART_RTS	Request to send.	O

10.3.8 Human-machine interfaces (HMI)

Table 10-23. GPIO Signal Descriptions

Chip signal name	Module signal name	Description	I/O
PTA[31:0][1]	PORTA31–PORTA0	General-purpose input/output	I/O
PTB[31:0][1]	PORTB31–PORTB0	General-purpose input/output	I/O
PTC[31:0][1]	PORTC31–PORTC0	General-purpose input/output	I/O
PTD[31:0][1]	PORTD31–PORTD0	General-purpose input/output	I/O
PTE[31:0][1]	PORTE31–PORTE0	General-purpose input/output	I/O

1. The available GPIO pins depend on the specific package. See the signal multiplexing section for which exact GPIO signals are available.

Table 10-24. TSI Signal Descriptions

Chip signal name	Module signal name	Description	I/O
TSI0_CH[15:0]	TSI[15:0]	TSI capacitive pins. Switches driver that connects directly to the electrode pins TSI[15:0] can operate as GPIO pins.	I/O

Chapter 11
Analog-to-Digital Converter (ADC)

11.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The 16-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

For the chip specific modes of operation, see the power management information of the device.

11.1.1 Features

Following are the features of the ADC module.

- Linear successive approximation algorithm with up to 16-bit resolution
- Up to four pairs of differential and 24 single-ended external analog inputs
- Output modes:
  - differential 16-bit, 13-bit, 11-bit, and 9-bit modes
  - single-ended 16-bit, 12-bit, 10-bit, and 8-bit modes
- Output format in 2's complement 16-bit sign extended for differential modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion, that is, automatic return to idle after single conversion

- Configurable sample time and conversion speed/power
- Conversion complete/hardware average complete flag and interrupt
- Input clock selectable from up to four sources
- Operation in low-power modes for lower noise
- Asynchronous clock source for lower noise operation with option to output the clock
- Selectable hardware conversion trigger with hardware channel select
- Automatic compare with interrupt for less-than, greater-than or equal-to, within range, or out-of-range, programmable value
- Temperature sensor
- Hardware average function
- Selectable voltage reference: external or alternate
- Self-Calibration mode

11.1.2 Block diagram

The following figure is the ADC module block diagram.

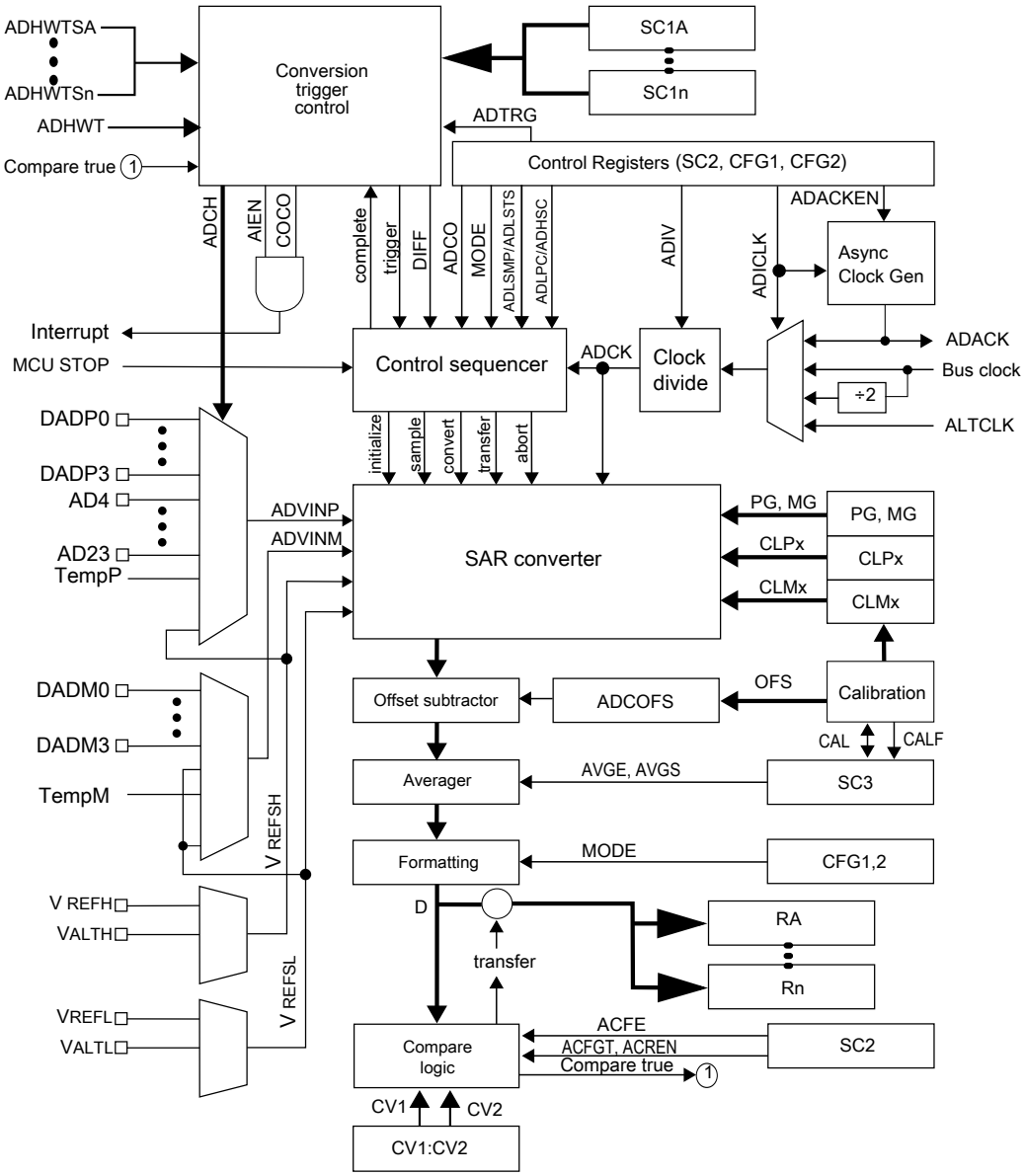


Figure 11-1. ADC block diagram

11.2 ADC signal descriptions

The ADC module supports up to 4 pairs of differential inputs and up to 24 single-ended inputs.

Each differential pair requires two inputs, DADPx and DADMx. The ADC also requires four supply/reference/ground connections.

NOTE

For the number of channels supported on this device as well as information regarding other chip-specific inputs into the ADC block, see the chip-specific ADC configuration information.

Table 11-1. ADC signal descriptions

Signal	Description	I/O
DADP3–DADP0	Differential Analog Channel Inputs	I
DADM3–DADM0	Differential Analog Channel Inputs	I
AD*n*	Single-Ended Analog Channel Inputs	I
V_{REFSH}	Voltage Reference Select High	I
V_{REFSL}	Voltage Reference Select Low	I
V_{DDA}	Analog Power Supply	I
V_{SSA}	Analog Ground	I

11.2.1 Analog Power (V_{DDA})

The ADC analog portion uses V_{DDA} as its power connection. In some packages, V_{DDA} is connected internally to V_{DD}. If externally available, connect the V_{DDA} pin to the same voltage potential as V_{DD}. External filtering may be necessary to ensure clean V_{DDA} for good results.

11.2.2 Analog Ground (V_{SSA})

The ADC analog portion uses V_{SSA} as its ground connection. In some packages, V_{SSA} is connected internally to V_{SS}. If externally available, connect the V_{SSA} pin to the same voltage potential as V_{SS}.

11.2.3 Voltage Reference Select

V_{REFSH} and V_{REFSL} are the high and low reference voltages for the ADC module.

The ADC can be configured to accept one of two voltage reference pairs for V_{REFSH} and V_{REFSL}. Each pair contains a positive reference that must be between the minimum Ref Voltage High and V_{DDA}, and a ground reference that must be at the same potential as V_{SSA}. The two pairs are external (V_{REFH} and V_{REFL}) and alternate (V_{ALTH} and V_{ALTL}). These voltage references are selected using SC2[REFSEL]. The alternate V_{ALTH} and

V_{ALTL} voltage reference pair may select additional external pins or internal sources depending on MCU configuration. See the chip configuration information on the Voltage References specific to this MCU.

In some packages, V_{REFH} is connected in the package to V_{DDA} and V_{REFL} to V_{SSA}. If externally available, the positive reference(s) may be connected to the same potential as V_{DDA} or may be driven by an external source to a level between the minimum Ref Voltage High and the V_{DDA} potential. V_{REFH} must never exceed V_{DDA}. Connect the ground references to the same voltage potential as V_{SSA}.

11.2.4 Analog Channel Inputs (ADx)

The ADC module supports up to 24 single-ended analog inputs. A single-ended input is selected for conversion through the SC1[ADCH] channel select bits when SC1n[DIFF] is low.

11.2.5 Differential Analog Channel Inputs (DADx)

The ADC module supports up to four differential analog channel inputs. Each differential analog input is a pair of external pins, DADPx and DADMx, referenced to each other to provide the most accurate analog to digital readings. A differential input is selected for conversion through SC1[ADCH] when SC1n[DIFF] is high. All DADPx inputs may be used as single-ended inputs if SC1n[DIFF] is low. In certain MCU configurations, some DADMx inputs may also be used as single-ended inputs if SC1n[DIFF] is low. For ADC connections specific to this device, see the chip-specific ADC information.

11.3 Memory map and register definitions

This section describes the ADC registers.

ADC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6000	ADC Status and Control Registers 1 (ADC0_SC1A)	32	R/W	0000_001Fh	11.3.1/184
4006_6004	ADC Status and Control Registers 1 (ADC0_SC1B)	32	R/W	0000_001Fh	11.3.1/184
4006_6008	ADC Configuration Register 1 (ADC0_CFG1)	32	R/W	0000_0000h	11.3.2/188
4006_600C	ADC Configuration Register 2 (ADC0_CFG2)	32	R/W	0000_0000h	11.3.3/189
4006_6010	ADC Data Result Register (ADC0_RA)	32	R	0000_0000h	11.3.4/190

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6014	ADC Data Result Register (ADC0_RB)	32	R	0000_0000h	11.3.4/190
4006_6018	Compare Value Registers (ADC0_CV1)	32	R/W	0000_0000h	11.3.5/192
4006_601C	Compare Value Registers (ADC0_CV2)	32	R/W	0000_0000h	11.3.5/192
4006_6020	Status and Control Register 2 (ADC0_SC2)	32	R/W	0000_0000h	11.3.6/193
4006_6024	Status and Control Register 3 (ADC0_SC3)	32	R/W	0000_0000h	11.3.7/195
4006_6028	ADC Offset Correction Register (ADC0_OFS)	32	R/W	0000_0004h	11.3.8/196
4006_602C	ADC Plus-Side Gain Register (ADC0_PG)	32	R/W	0000_8200h	11.3.9/197
4006_6030	ADC Minus-Side Gain Register (ADC0_MG)	32	R/W	0000_8200h	11.3.10/197
4006_6034	ADC Plus-Side General Calibration Value Register (ADC0_CLPD)	32	R/W	0000_000Ah	11.3.11/198
4006_6038	ADC Plus-Side General Calibration Value Register (ADC0_CLPS)	32	R/W	0000_0020h	11.3.12/199
4006_603C	ADC Plus-Side General Calibration Value Register (ADC0_CLP4)	32	R/W	0000_0200h	11.3.13/199
4006_6040	ADC Plus-Side General Calibration Value Register (ADC0_CLP3)	32	R/W	0000_0100h	11.3.14/200
4006_6044	ADC Plus-Side General Calibration Value Register (ADC0_CLP2)	32	R/W	0000_0080h	11.3.15/200
4006_6048	ADC Plus-Side General Calibration Value Register (ADC0_CLP1)	32	R/W	0000_0040h	11.3.16/201
4006_604C	ADC Plus-Side General Calibration Value Register (ADC0_CLP0)	32	R/W	0000_0020h	11.3.17/201
4006_6054	ADC Minus-Side General Calibration Value Register (ADC0_CLMD)	32	R/W	0000_000Ah	11.3.18/202
4006_6058	ADC Minus-Side General Calibration Value Register (ADC0_CLMS)	32	R/W	0000_0020h	11.3.19/202
4006_605C	ADC Minus-Side General Calibration Value Register (ADC0_CLM4)	32	R/W	0000_0200h	11.3.20/203
4006_6060	ADC Minus-Side General Calibration Value Register (ADC0_CLM3)	32	R/W	0000_0100h	11.3.21/203
4006_6064	ADC Minus-Side General Calibration Value Register (ADC0_CLM2)	32	R/W	0000_0080h	11.3.22/204
4006_6068	ADC Minus-Side General Calibration Value Register (ADC0_CLM1)	32	R/W	0000_0040h	11.3.23/204
4006_606C	ADC Minus-Side General Calibration Value Register (ADC0_CLM0)	32	R/W	0000_0020h	11.3.24/205

11.3.1 ADC Status and Control Registers 1 (ADC*x*_SC1*n*)

SC1A is used for both software and hardware trigger modes of operation.

To allow sequential conversions of the ADC to be triggered by internal peripherals, the ADC can have more than one status and control register: one for each conversion. The SC1B–SC1n registers indicate potentially multiple SC1 registers for use only in hardware trigger mode. See the chip configuration information about the number of SC1n registers specific to this device. The SC1n registers have identical fields, and are used in a "ping-pong" approach to control ADC operation.

At any one point in time, only one of the SC1n registers is actively controlling ADC conversions. Updating SC1A while SC1n is actively controlling a conversion is allowed, and vice-versa for any of the SC1n registers specific to this MCU.

Writing SC1A while SC1A is actively controlling a conversion aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, writes to SC1A subsequently initiate a new conversion, if SC1[ADCH] contains a value other than all 1s (module disabled).

Writing any of the SC1n registers while that specific SC1n register is actively controlling a conversion aborts the current conversion. None of the SC1B-SC1n registers are used for software trigger operation and therefore writes to the SC1B–SC1n registers do not initiate a new conversion.

Address: 4006_6000h base + 0h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								COCO	AIEN	DIFF	ADCH				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

ADCx_SC1n field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 COCO	Conversion Complete Flag This is a read-only field that is set each time a conversion is completed when the compare function is disabled, or SC2[ACFE]=0 and the hardware average function is disabled, or SC3[AVGE]=0. When the compare function is enabled, or SC2[ACFE]=1, COCO is set upon completion of a conversion only if the compare result is true. When the hardware average function is enabled, or SC3[AVGE]=1, COCO is set upon completion of the selected number of conversions (determined by AVGS). COCO in SC1A is also set at the completion of a calibration sequence. COCO is cleared when the respective SC1n register is written or when the respective Rn register is read. 0 Conversion is not completed. 1 Conversion is completed.
6 AIEN	Interrupt Enable Enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt is disabled. 1 Conversion complete interrupt is enabled.
5 DIFF	Differential Mode Enable Configures the ADC to operate in differential mode. When enabled, this mode automatically selects from the differential channels, and changes the conversion algorithm and the number of cycles to complete a conversion. 0 Single-ended conversions and input channels are selected. 1 Differential conversions and input channels are selected.
ADCH	Input channel select Selects one of the input channels. The input channel decode depends on the value of DIFF. DAD0-DAD3 are associated with the input pin pairs DADPx and DADMx. **NOTE:** Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the actual ADC channel assignments for your device, see the Chip Configuration details. The successive approximation converter subsystem is turned off when the channel select bits are all set, that is, ADCH = 11111. This feature allows explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional single conversion from being performed. It is not necessary to set ADCH to all 1s to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes. 00000 When DIFF=0, DADP0 is selected as input; when DIFF=1, DAD0 is selected as input. 00001 When DIFF=0, DADP1 is selected as input; when DIFF=1, DAD1 is selected as input. 00010 When DIFF=0, DADP2 is selected as input; when DIFF=1, DAD2 is selected as input. 00011 When DIFF=0, DADP3 is selected as input; when DIFF=1, DAD3 is selected as input. 00100 When DIFF=0, AD4 is selected as input; when DIFF=1, it is reserved. 00101 When DIFF=0, AD5 is selected as input; when DIFF=1, it is reserved. 00110 When DIFF=0, AD6 is selected as input; when DIFF=1, it is reserved. 00111 When DIFF=0, AD7 is selected as input; when DIFF=1, it is reserved.

Table continues on the next page...

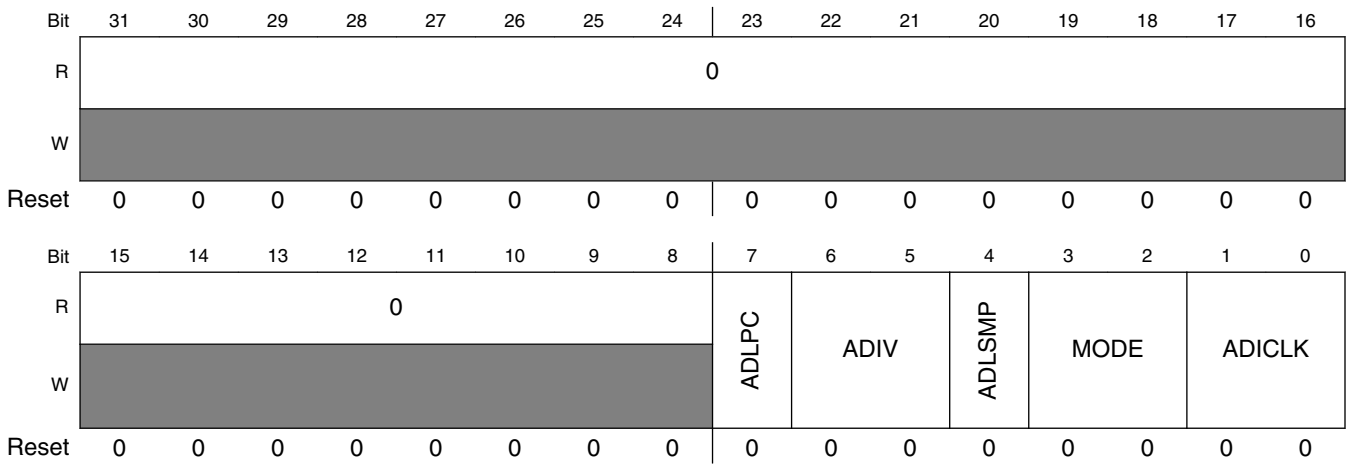
ADC*x*_SC1*n* field descriptions (continued)

Field	Description	
	01000	When DIFF=0, AD8 is selected as input; when DIFF=1, it is reserved.
	01001	When DIFF=0, AD9 is selected as input; when DIFF=1, it is reserved.
	01010	When DIFF=0, AD10 is selected as input; when DIFF=1, it is reserved.
	01011	When DIFF=0, AD11 is selected as input; when DIFF=1, it is reserved.
	01100	When DIFF=0, AD12 is selected as input; when DIFF=1, it is reserved.
	01101	When DIFF=0, AD13 is selected as input; when DIFF=1, it is reserved.
	01110	When DIFF=0, AD14 is selected as input; when DIFF=1, it is reserved.
	01111	When DIFF=0, AD15 is selected as input; when DIFF=1, it is reserved.
	10000	When DIFF=0, AD16 is selected as input; when DIFF=1, it is reserved.
	10001	When DIFF=0, AD17 is selected as input; when DIFF=1, it is reserved.
	10010	When DIFF=0, AD18 is selected as input; when DIFF=1, it is reserved.
	10011	When DIFF=0, AD19 is selected as input; when DIFF=1, it is reserved.
	10100	When DIFF=0, AD20 is selected as input; when DIFF=1, it is reserved.
	10101	When DIFF=0, AD21 is selected as input; when DIFF=1, it is reserved.
	10110	When DIFF=0, AD22 is selected as input; when DIFF=1, it is reserved.
	10111	When DIFF=0, AD23 is selected as input; when DIFF=1, it is reserved.
	11000	Reserved.
	11001	Reserved.
	11010	When DIFF=0, Temp Sensor (single-ended) is selected as input; when DIFF=1, Temp Sensor (differential) is selected as input.
	11011	When DIFF=0,Bandgap (single-ended) is selected as input; when DIFF=1, Bandgap (differential) is selected as input.
	11100	Reserved.
	11101	When DIFF=0,V_{REFSH} is selected as input; when DIFF=1, -V_{REFSH} (differential) is selected as input. Voltage reference selected is determined by SC2[REFSEL].
	11110	When DIFF=0,V_{REFSL} is selected as input; when DIFF=1, it is reserved. Voltage reference selected is determined by SC2[REFSEL].
	11111	Module is disabled.

11.3.2 ADC Configuration Register 1 (ADCx_CFG1)

The configuration Register 1 (CFG1) selects the mode of operation, clock source, clock divide, and configuration for low power or long sample time.

Address: 4006_6000h base + 8h offset = 4006_6008h



ADCx_CFG1 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 ADLPC	Low-Power Configuration Controls the power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 Normal power configuration. 1 Low-power configuration. The power is reduced at the expense of maximum clock speed.
6–5 ADIV	Clock Divide Select Selects the divide ratio used by the ADC to generate the internal clock ADCK. 00 The divide ratio is 1 and the clock rate is input clock. 01 The divide ratio is 2 and the clock rate is (input clock)/2. 10 The divide ratio is 4 and the clock rate is (input clock)/4. 11 The divide ratio is 8 and the clock rate is (input clock)/8.
4 ADLSMP	Sample Time Configuration Selects between different sample times based on the conversion mode selected. This field adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption if continuous conversions are enabled and high conversion rates are not required. When ADLSMP=1, the long sample time select bits, (ADLSTS[1:0]), can select the extent of the long sample time.

Table continues on the next page...

ADCx_CFG1 field descriptions (continued)

Field	Description
	0 Short sample time. 1 Long sample time.
3–2 MODE	Conversion mode selection Selects the ADC resolution mode. 00 When DIFF=0:It is single-ended 8-bit conversion; when DIFF=1, it is differential 9-bit conversion with 2's complement output. 01 When DIFF=0:It is single-ended 12-bit conversion ; when DIFF=1, it is differential 13-bit conversion with 2's complement output. 10 When DIFF=0:It is single-ended 10-bit conversion. ; when DIFF=1, it is differential 11-bit conversion with 2's complement output 11 When DIFF=0:It is single-ended 16-bit conversion..; when DIFF=1, it is differential 16-bit conversion with 2's complement output
ADICLK	Input Clock Select Selects the input clock source to generate the internal clock, ADCK. Note that when the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start, when CFG2[ADACKEN]=0, the asynchronous clock is activated at the start of a conversion and deactivated when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. 00 Bus clock 01 Bus clock divided by 2(BUSCLK/2) 10 Alternate clock (ALTCLK) 11 Asynchronous clock (ADACK)

11.3.3 ADC Configuration Register 2 (ADCx_CFG2)

Configuration Register 2 (CFG2) selects the special high-speed configuration for very high speed conversions and selects the long sample time duration during long sample mode.

Address: 4006_6000h base + Ch offset = 4006_600Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								0			MUXSEL	ADACKEN	ADHSC	ADLSTS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_CFG2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 MUXSEL	ADC Mux Select Changes the ADC mux setting to select between alternate sets of ADC channels. 0 ADxxa channels are selected. 1 ADxxb channels are selected.
3 ADACKEN	Asynchronous Clock Output Enable Enables the asynchronous clock source and the clock source output regardless of the conversion and status of CFG1[ADICLK]. Based on MCU configuration, the asynchronous clock may be used by other modules. See chip configuration information. Setting this field allows the clock to be used even while the ADC is idle or operating from a different clock source. Also, latency of initiating a single or first-continuous conversion with the asynchronous clock selected is reduced because the ADACK clock is already operational. 0 Asynchronous clock output disabled; Asynchronous clock is enabled only if selected by ADICLK and a conversion is active. 1 Asynchronous clock and clock output is enabled regardless of the state of the ADC.
2 ADHSC	High-Speed Configuration Configures the ADC for very high-speed operation. The conversion sequence is altered with 2 ADCK cycles added to the conversion time to allow higher speed conversion clocks. 0 Normal conversion sequence selected. 1 High-speed conversion sequence selected with 2 additional ADCK cycles to total conversion time.
ADLSTS	Long Sample Time Select Selects between the extended sample times when long sample time is selected, that is, when CFG1[ADLSMP]=1. This allows higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. 00 Default longest sample time; 20 extra ADCK cycles; 24 ADCK cycles total. 01 12 extra ADCK cycles; 16 ADCK cycles total sample time. 10 6 extra ADCK cycles; 10 ADCK cycles total sample time. 11 2 extra ADCK cycles; 6 ADCK cycles total sample time.

11.3.4 ADC Data Result Register (ADCx_R*n*)

The data result registers (Rn) contain the result of an ADC conversion of the channel selected by the corresponding status and channel control register (SC1A:SC1n). For every status and channel control register, there is a corresponding data result register.

Unused bits in R n are cleared in unsigned right-aligned modes and carry the sign bit (MSB) in sign-extended 2's complement modes. For example, when configured for 10-bit single-ended mode, D[15:10] are cleared. When configured for 11-bit differential mode, D[15:10] carry the sign bit, that is, bit 10 extended through bit 15.

The following table describes the behavior of the data result registers in the different modes of operation.

Table 11-2. Data result register description

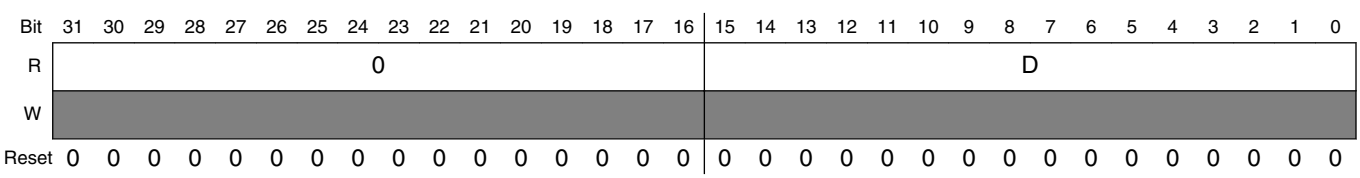
Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Signed 2's complement
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
13-bit differential	S	S	S	S	D	D	D	D	D	D	D	D	D	D	D	D	Sign-extended 2's complement
12-bit single-ended	0	0	0	0	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right-justified
11-bit differential	S	S	S	S	S	S	D	D	D	D	D	D	D	D	D	D	Sign-extended 2's complement
10-bit single-ended	0	0	0	0	0	0	D	D	D	D	D	D	D	D	D	D	Unsigned right-justified
9-bit differential	S	S	S	S	S	S	S	S	D	D	D	D	D	D	D	D	Sign-extended 2's complement
8-bit single-ended	0	0	0	0	0	0	0	0	D	D	D	D	D	D	D	D	Unsigned right-justified

NOTE

S: Sign bit or sign bit extension;

D: Data, which is 2's complement data if indicated

Address: 4006_6000h base + 10h offset + (4d × i), where i=0d to 1d



ADCx_R*n* field descriptions

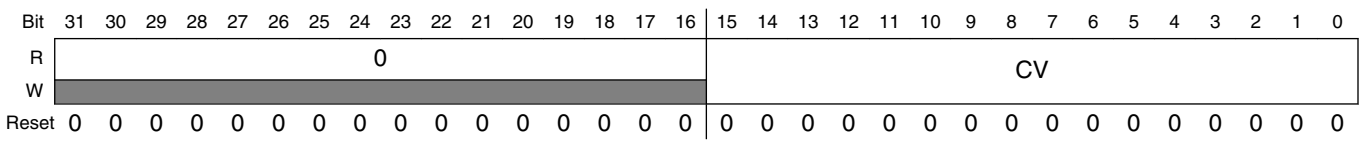
Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
D	Data result

11.3.5 Compare Value Registers (ADC*x*_CV*n*)

The Compare Value Registers (CV1 and CV2) contain a compare value used to compare the conversion result when the compare function is enabled, that is, SC2[ACFE]=1. This register is formatted in the same way as the Rn registers in different modes of operation for both bit position definition and value format using unsigned or sign-extended 2's complement. Therefore, the compare function uses only the CVn fields that are related to the ADC mode of operation.

The compare value 2 register (CV2) is used only when the compare range function is enabled, that is, SC2[ACREN]=1.

Address: 4006_6000h base + 18h offset + (4d × i), where i=0d to 1d



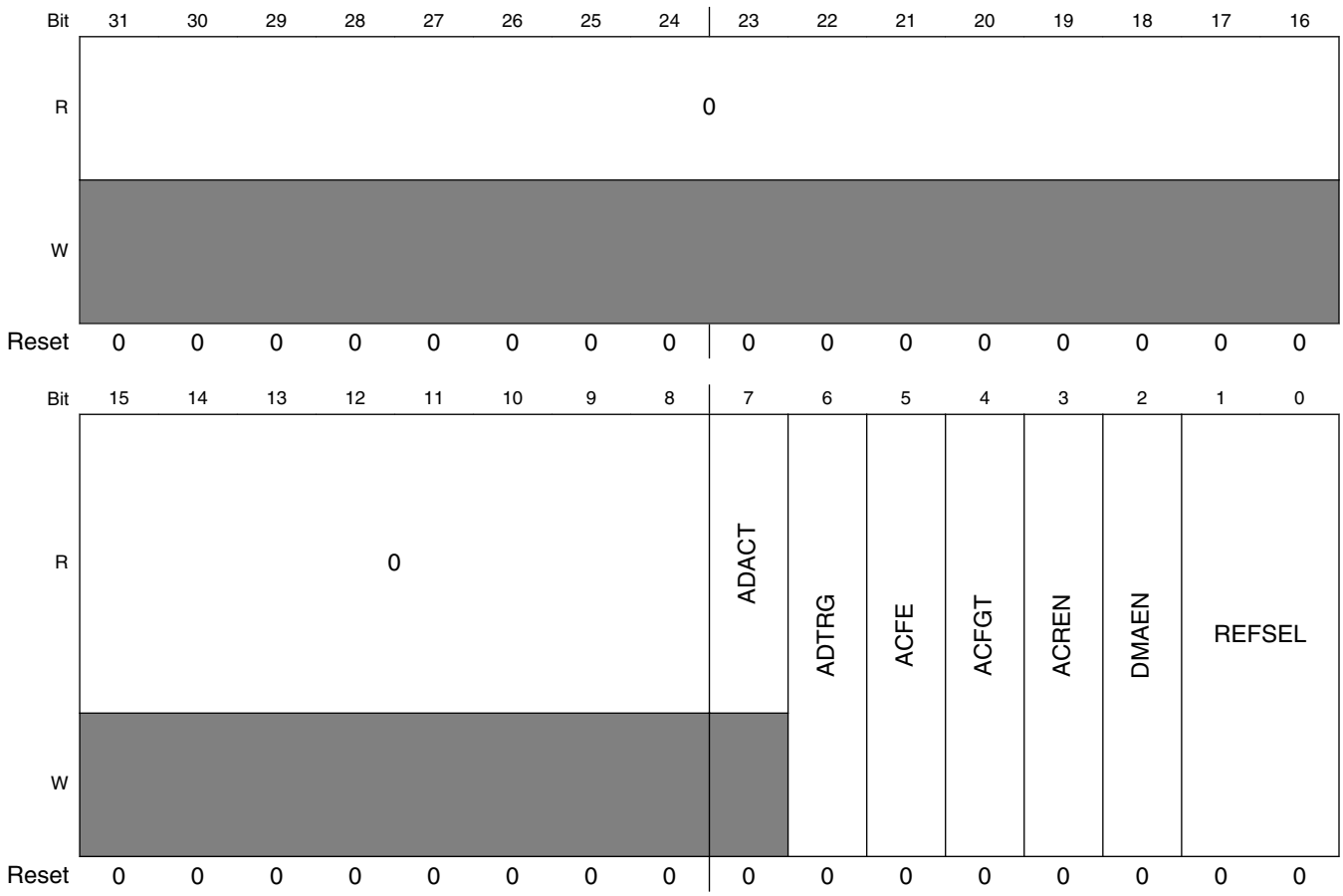
ADC*x*_CV*n* field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CV	Compare Value.

11.3.6 Status and Control Register 2 (ADCx_SC2)

The status and control register 2 (SC2) contains the conversion active, hardware/software trigger select, compare function, and voltage reference select of the ADC module.

Address: 4006_6000h base + 20h offset = 4006_6020h



ADCx_SC2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 ADACT	Conversion Active Indicates that a conversion or hardware averaging is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted. 0 Conversion not in progress. 1 Conversion in progress.
6 ADTRG	Conversion Trigger Select Selects the type of trigger used for initiating a conversion. Two types of trigger are selectable:

Table continues on the next page...

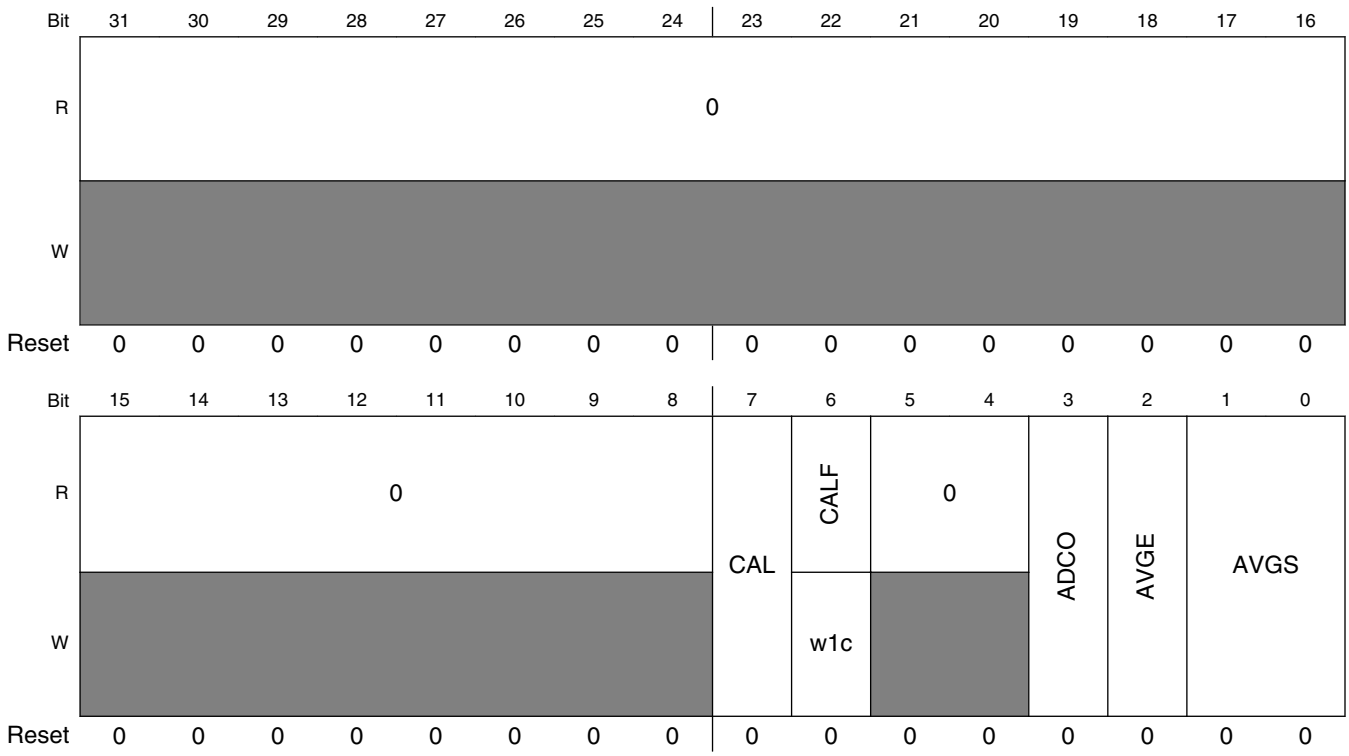
ADCx_SC2 field descriptions (continued)

Field	Description
	• Software trigger: When software trigger is selected, a conversion is initiated following a write to SC1A. • Hardware trigger: When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input after a pulse of the ADHWTSn input. 0 Software trigger selected. 1 Hardware trigger selected.
5 ACFE	Compare Function Enable Enables the compare function. 0 Compare function disabled. 1 Compare function enabled.
4 ACFGT	Compare Function Greater Than Enable Configures the compare function to check the conversion result relative to the CV1 and CV2 based upon the value of ACREN. ACFE must be set for ACFGT to have any effect. 0 Configures less than threshold, outside range not inclusive and inside range not inclusive; functionality based on the values placed in CV1 and CV2. 1 Configures greater than or equal to threshold, outside and inside ranges inclusive; functionality based on the values placed in CV1 and CV2.
3 ACREN	Compare Function Range Enable Configures the compare function to check if the conversion result of the input being monitored is either between or outside the range formed by CV1 and CV2 determined by the value of ACFGT. ACFE must be set for ACFGT to have any effect. 0 Range function disabled. Only CV1 is compared. 1 Range function enabled. Both CV1 and CV2 are compared.
2 DMAEN	DMA Enable 0 DMA is disabled. 1 DMA is enabled and will assert the ADC DMA request during an ADC conversion complete event noted when any of the SC1n[COCO] flags is asserted.
REFSEL	Voltage Reference Selection Selects the voltage reference source used for conversions. 00 Default voltage reference pin pair, that is, external pins V_{REFH} and V_{REFL} 01 Alternate reference pair, that is, V_{ALTH} and V_{ALTL} . This pair may be additional external pins or internal sources depending on the MCU configuration. See the chip configuration information for details specific to this MCU 10 Reserved 11 Reserved

11.3.7 Status and Control Register 3 (ADCx_SC3)

The Status and Control Register 3 (SC3) controls the calibration, continuous convert, and hardware averaging functions of the ADC module.

Address: 4006_6000h base + 24h offset = 4006_6024h



ADCx_SC3 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CAL	Calibration Begins the calibration sequence when set. This field stays set while the calibration is in progress and is cleared when the calibration sequence is completed. CALF must be checked to determine the result of the calibration sequence. Once started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid and CALF will set. Setting CAL will abort any current conversion.
6 CALF	Calibration Failed Flag Displays the result of the calibration sequence. The calibration sequence will fail if SC2[ADTRG] = 1, any ADC register is written, or any stop mode is entered before the calibration sequence completes. Writing 1 to CALF clears it. 0 Calibration completed normally. 1 Calibration failed. ADC accuracy specifications are not guaranteed.

Table continues on the next page...

ADCx_SC3 field descriptions (continued)

Field	Description
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ADCO	Continuous Conversion Enable Enables continuous conversions. 0 One conversion or one set of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion. 1 Continuous conversions or sets of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion.
2 AVGE	Hardware Average Enable Enables the hardware average function of the ADC. 0 Hardware average function disabled. 1 Hardware average function enabled.
AVGS	Hardware Average Select Determines how many ADC conversions will be averaged to create the ADC average result. 00 4 samples averaged. 01 8 samples averaged. 10 16 samples averaged. 11 32 samples averaged.

11.3.8 ADC Offset Correction Register (ADCx_OFS)

The ADC Offset Correction Register (OFS) contains the user-selected or calibration-generated offset error correction value. This register is a 2's complement, left-justified, 16-bit value . The value in OFS is subtracted from the conversion and the result is transferred into the result registers, Rn. If the result is greater than the maximum or less than the minimum result value, it is forced to the appropriate limit for the current mode of operation.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4006_6000h base + 28h offset = 4006_6028h

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

ADCx_OFS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
OFS	Offset Error Correction Value

11.3.9 ADC Plus-Side Gain Register (ADCx_PG)

The Plus-Side Gain Register (PG) contains the gain error correction for the plus-side input in differential mode or the overall conversion in single-ended mode. PG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between PG[15] and PG[14]. This register must be written by the user with the value described in the calibration procedure. Otherwise, the gain error specifications may not be met.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4006_6000h base + 2Ch offset = 4006_602Ch

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

ADCx_PG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PG	Plus-Side Gain

11.3.10 ADC Minus-Side Gain Register (ADCx_MG)

The Minus-Side Gain Register (MG) contains the gain error correction for the minus-side input in differential mode. This register is ignored in single-ended mode. MG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between MG[15] and MG[14]. This register must be written by the user with the value described in the calibration procedure. Otherwise, the gain error specifications may not be met.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4006_6000h base + 30h offset = 4006_6030h

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

ADCx_MG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MG	Minus-Side Gain

11.3.11 ADC Plus-Side General Calibration Value Register (ADCx_CLPD)

The Plus-Side General Calibration Value Registers (CLPx) contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLP0[5:0], CLP1[6:0], CLP2[7:0], CLP3[8:0], CLP4[9:0], CLPS[5:0], and CLPD[5:0]. CLPx are automatically set when the self-calibration sequence is done, that is, CAL is cleared. If these registers are written by the user after calibration, the linearity error specifications may not be met.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4006_6000h base + 34h offset = 4006_6034h

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

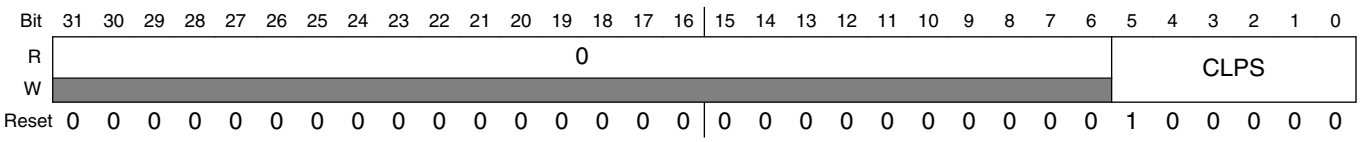
ADCx_CLPD field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPD	Calibration Value Calibration Value

11.3.12 ADC Plus-Side General Calibration Value Register (ADCx_CLPS)

For more information, see CLPD register description.

Address: 4006_6000h base + 38h offset = 4006_6038h



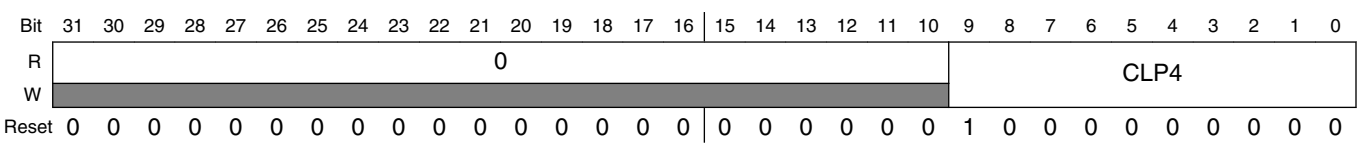
ADCx_CLPS field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLPS	Calibration Value Calibration Value

11.3.13 ADC Plus-Side General Calibration Value Register (ADCx_CLP4)

For more information, see CLPD register description.

Address: 4006_6000h base + 3Ch offset = 4006_603Ch



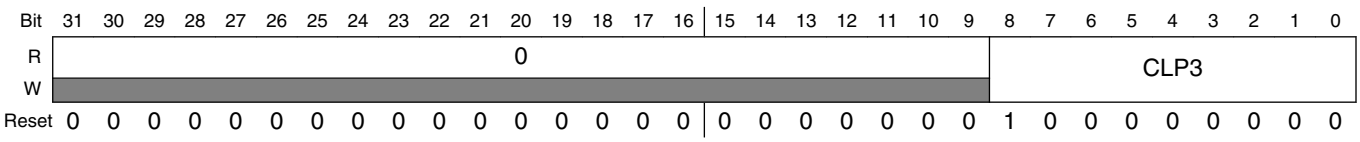
ADCx_CLP4 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP4	Calibration Value Calibration Value

11.3.14 ADC Plus-Side General Calibration Value Register (ADCx_CLP3)

For more information, see CLPD register description.

Address: 4006_6000h base + 40h offset = 4006_6040h



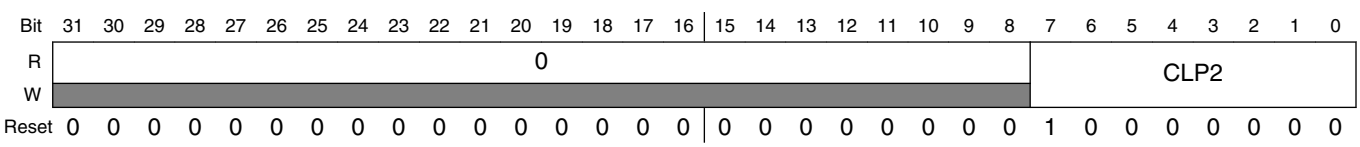
ADCx_CLP3 field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP3	Calibration Value Calibration Value

11.3.15 ADC Plus-Side General Calibration Value Register (ADCx_CLP2)

For more information, see CLPD register description.

Address: 4006_6000h base + 44h offset = 4006_6044h



ADCx_CLP2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP2	Calibration Value Calibration Value

11.3.16 ADC Plus-Side General Calibration Value Register (ADCx_CLP1)

For more information, see CLPD register description.

Address: 4006_6000h base + 48h offset = 4006_6048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																									CLP1						
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

ADCx_CLP1 field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP1	Calibration Value Calibration Value

11.3.17 ADC Plus-Side General Calibration Value Register (ADCx_CLP0)

For more information, see CLPD register description.

Address: 4006_6000h base + 4Ch offset = 4006_604Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																										CLP0					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

ADCx_CLP0 field descriptions

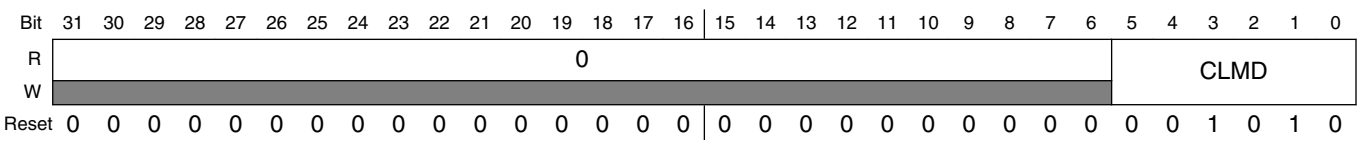
Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLP0	Calibration Value Calibration Value

11.3.18 ADC Minus-Side General Calibration Value Register (ADCx_CLMD)

The Minus-Side General Calibration Value (CLMx) registers contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLM0[5:0], CLM1[6:0], CLM2[7:0], CLM3[8:0], CLM4[9:0], CLMS[5:0], and CLMD[5:0]. CLMx are automatically set when the self-calibration sequence is done, that is, CAL is cleared. If these registers are written by the user after calibration, the linearity error specifications may not be met.

For more information regarding the calibration procedure, please refer to the Calibration function section.

Address: 4006_6000h base + 54h offset = 4006_6054h



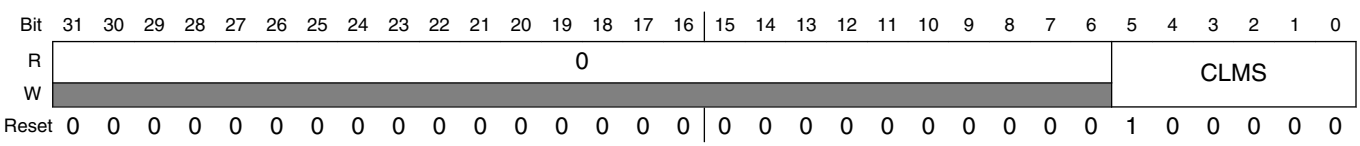
ADCx_CLMD field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLMD	Calibration Value Calibration Value

11.3.19 ADC Minus-Side General Calibration Value Register (ADCx_CLMS)

For more information, see CLMD register description.

Address: 4006_6000h base + 58h offset = 4006_6058h



ADCx_CLMS field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLMS	Calibration Value Calibration Value

11.3.20 ADC Minus-Side General Calibration Value Register (ADCx_CLM4)

For more information, see CLMD register description.

Address: 4006_6000h base + 5Ch offset = 4006_605Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																						CLM4									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

ADCx_CLM4 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLM4	Calibration Value Calibration Value

11.3.21 ADC Minus-Side General Calibration Value Register (ADCx_CLM3)

For more information, see CLMD register description.

Address: 4006_6000h base + 60h offset = 4006_6060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																							CLM3								
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

ADCx_CLM3 field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

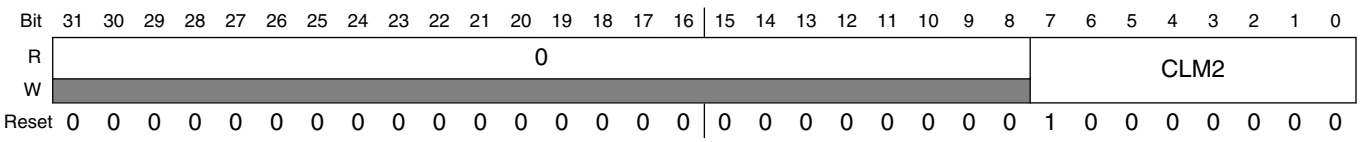
ADCx_CLM3 field descriptions (continued)

Field	Description
CLM3	Calibration Value Calibration Value

11.3.22 ADC Minus-Side General Calibration Value Register (ADCx_CLM2)

For more information, see CLMD register description.

Address: 4006_6000h base + 64h offset = 4006_6064h



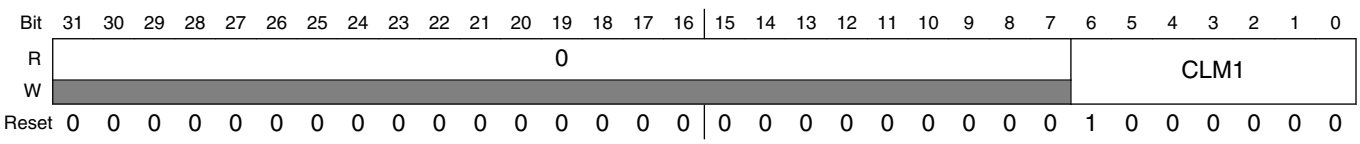
ADCx_CLM2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLM2	Calibration Value Calibration Value

11.3.23 ADC Minus-Side General Calibration Value Register (ADCx_CLM1)

For more information, see CLMD register description.

Address: 4006_6000h base + 68h offset = 4006_6068h



ADCx_CLM1 field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLM1	Calibration Value Calibration Value

11.3.24 ADC Minus-Side General Calibration Value Register (ADCx_CLM0)

For more information, see CLMD register description.

Address: 4006_6000h base + 6Ch offset = 4006_606Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																										CLM0					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

ADCx_CLM0 field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLM0	Calibration Value Calibration Value

11.4 Functional description

The ADC module is disabled during reset, in Low-Power Stop mode, or when SC1n[ADCH] are all high; see the power management information for details. The module is idle when a conversion has completed and another conversion has not been initiated. When it is idle and the asynchronous clock output enable is disabled, or CFG2[ADACKEN]= 0, the module is in its lowest power state. The ADC can perform an analog-to-digital conversion on any of the software selectable channels. All modes perform conversion by a successive approximation algorithm.

To meet accuracy specifications, the ADC module must be calibrated using the on-chip calibration function.

See Calibration function for details on how to perform calibration.

When the conversion is completed, the result is placed in the Rn data registers. The respective SC1n[COCO] is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, or, when SC1n[AIEN]=1.

The ADC module has the capability of automatically comparing the result of a conversion with the contents of the CV1 and CV2 registers. The compare function is enabled by setting SC2[ACFE] and operates in any of the conversion modes and configurations.

The ADC module has the capability of automatically averaging the result of multiple conversions. The hardware average function is enabled by setting SC3[AVGE] and operates in any of the conversion modes and configurations.

NOTE

For the chip specific modes of operation, see the power management information of this MCU.

11.4.1 Clock select and divide control

One of four clock sources can be selected as the clock source for the ADC module.

This clock source is then divided by a configurable value to generate the input clock ADCK, to the module. The clock is selected from one of the following sources by means of CFG1[ADICLK].

- Bus clock. This is the default selection following reset.
- Bus clock divided by two. For higher bus clock rates, this allows a maximum divide-by-16 of the bus clock using CFG1[ADIV].
- ALTCLK: As defined for this MCU. See the chip configuration information. Conversions are possible using ALTCLK as the input clock source while the MCU is in Normal Stop mode.
- Asynchronous clock (ADACK): This clock is generated from a clock source within the ADC module. When the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start CFG2[ADACKEN]=0, ADACK is activated at the start of a conversion and deactivated when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. To avoid the conversion time variability and latency associated with the ADACK clock startup, set CFG2[ADACKEN]=1 and wait the worst-case startup time of 5 μs prior to initiating any conversions using the ADACK clock source. Conversions are possible using ADACK as the input clock source while the MCU is in Normal Stop mode. See Power Control for more information.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC may not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by CFG1[ADIV] and can be divide-by 1, 2, 4, or 8.

11.4.2 Voltage reference selection

The ADC can be configured to accept one of the two voltage reference pairs as the reference voltage (V_{REFSH} and V_{REFSL}) used for conversions.

Each pair contains a positive reference that must be between the minimum Ref Voltage High and V_{DDA}, and a ground reference that must be at the same potential as V_{SSA}. The two pairs are external (V_{REFH} and V_{REFL}) and alternate (V_{ALTH} and V_{ALTL}). These voltage references are selected using SC2[REFSEL]. The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. See the chip configuration information on the voltage references specific to this MCU.

11.4.3 Hardware trigger and channel selects

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when SC2[ADTRG] is set and a hardware trigger select event, ADHWTSn, has occurred.

This source is not available on all MCUs. See the chip-specific ADC information for information on the ADHWT source and the ADHWTSn configurations specific to this MCU.

When an ADHWT source is available and hardware trigger is enabled, that is SC2[ADTRG]=1, a conversion is initiated on the rising-edge of ADHWT after a hardware trigger select event, that is, ADHWTSn, has occurred. If a conversion is in progress when a rising-edge of a trigger occurs, the rising-edge is ignored. In continuous convert configuration, only the initial rising-edge to launch continuous conversions is observed, and until conversion is aborted, the ADC continues to do conversions on the same SCn register that initiated the conversion. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

The hardware trigger select event, ADHWTSn, must be set prior to the receipt of the ADHWT signal. If these conditions are not met, the converter may ignore the trigger or use the incorrect configuration. If a hardware trigger select event is asserted during a conversion, it must stay asserted until the end of current conversion and remain set until the receipt of the ADHWT signal to trigger a new conversion. The channel and status fields selected for the conversion depend on the active trigger select signal:

- ADHWTSA active selects SC1A.
- ADHWTSn active selects SC1n.

Note

Asserting more than one hardware trigger select signal (ADHWTSn) at the same time results in unknown results. To avoid this, select only one hardware trigger select signal (ADHWTSn) prior to the next intended conversion.

When the conversion is completed, the result is placed in the Rn registers associated with the ADHWTSn received. For example:

- ADHWTSA active selects RA register
- ADHWTSn active selects Rn register

The conversion complete flag associated with the ADHWTSn received, that is, SC1n[COCO], is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, that is, SC1[AIEN]=1.

11.4.4 Conversion control

Conversions can be performed as determined by CFG1[MODE] and SC1n[DIFF] as shown in the description of CFG1[MODE].

Conversions can be initiated by a software or hardware trigger.

In addition, the ADC module can be configured for:

- Low-power operation
- Long sample time
- Continuous conversion
- Hardware average
- Automatic compare of the conversion result to a software determined compare value

11.4.4.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A, with SC1n[ADCH] not all 1's, if software triggered operation is selected, that is, when SC2[ADTRG]=0.
- Following a hardware trigger, or ADHWT event, if hardware triggered operation is selected, that is, SC2[ADTRG]=1, and a hardware trigger select event, ADHWTSn, has occurred. The channel and status fields selected depend on the active trigger select signal:
  - ADHWTSA active selects SC1A.

- ADHWTSn active selects SC1n.
- if neither is active, the off condition is selected

Note

Selecting more than one ADHWTSn prior to a conversion completion will result in unknown results. To avoid this, select only one ADHWTSn prior to a conversion completion.

- Following the transfer of the result to the data registers when continuous conversion is enabled, that is, when SC3[ADCO] = 1.

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation, that is, when SC2[ADTRG] = 0, continuous conversions begin after SC1A is written and continue until aborted. In hardware triggered operation, that is, when SC2[ADTRG] = 1 and one ADHWTSn event has occurred, continuous conversions begin after a hardware trigger event and continue until aborted.

If hardware averaging is enabled, a new conversion is automatically initiated after the completion of the current conversion until the correct number of conversions are completed. In software triggered operation, conversions begin after SC1A is written. In hardware triggered operation, conversions begin after a hardware trigger. If continuous conversions are also enabled, a new set of conversions to be averaged are initiated following the last of the selected number of conversions.

11.4.4.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, Rn. If the compare functions are disabled, this is indicated by setting of SC1n[COCO]. If hardware averaging is enabled, the respective SC1n[COCO] sets only if the last of the selected number of conversions is completed. If the compare function is enabled, the respective SC1n[COCO] sets and conversion result data is transferred only if the compare condition is true. If both hardware averaging and compare functions are enabled, then the respective SC1n[COCO] sets only if the last of the selected number of conversions is completed and the compare condition is true. An interrupt is generated if the respective SC1n[AIEN] is high at the time that the respective SC1n[COCO] is set.

11.4.4.3 Aborting conversions

Any conversion in progress is aborted when:

- Writing to SC1A while it is actively controlling a conversion, aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, a write to SC1A initiates a new conversion if SC1A[ADCH] is equal to a value other than all 1s. Writing to any of the SC1B–SC1n registers while that specific SC1B–SC1n register is actively controlling a conversion aborts the current conversion. The SC1(B-n) registers are not used for software trigger operation and therefore writes to the SC1(B-n) registers do not initiate a new conversion.
- A write to any ADC register besides the SC1A-SC1n registers occurs. This indicates that a change in mode of operation has occurred and the current conversion is therefore invalid.
- The MCU is reset or enters Low-Power Stop modes.
- The MCU enters Normal Stop mode with ADACK or Alternate Clock Sources not enabled.

When a conversion is aborted, the contents of the data registers, Rn, are not altered. The data registers continue to be the values transferred after the completion of the last successful conversion. If the conversion was aborted by a reset or Low-Power Stop modes, RA and Rn return to their reset states.

11.4.4.4 Power control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, but the asynchronous clock output is disabled, that is CFG2[ADACKEN]=0, the ADACK clock generator also remains in its idle state (disabled) until a conversion is initiated. If the asynchronous clock output is enabled, that is, CFG2[ADACKEN]=1, it remains active regardless of the state of the ADC or the MCU power mode.

Power consumption when the ADC is active can be reduced by setting CFG1[ADLPC]. This results in a lower maximum value for f_{ADCK}.

11.4.4.5 Sample time and total conversion time

For short sample, that is, when CFG1[ADLSMP]=0, there is a 2-cycle adder for first conversion over the base sample time of four ADCK cycles. For high-speed conversions, that is, when CFG2[ADHSC]=1, there is an additional 2-cycle adder on any conversion. The table below summarizes sample times for the possible ADC configurations.

ADC configuration			Sample time (ADCK cycles)	
CFG1[ADLSMP]	CFG2[ADLSTS]	CFG2[ADHSC]	First or Single	Subsequent
0	X	0	6	4
1	00	0	24	
1	01	0	16	
1	10	0	10	
1	11	0	6	
0	X	1	8	6
1	00	1	26	
1	01	1	18	
1	10	1	12	
1	11	1	8	

The total conversion time depends upon:

- The sample time as determined by CFG1[ADLSMP] and CFG2[ADLSTS]
- The MCU bus frequency
- The conversion mode, as determined by CFG1[MODE] and SC1n[DIFF]
- The high-speed configuration, that is, CFG2[ADHSC]
- The frequency of the conversion clock, that is, f_{ADCK}.

CFG2[ADHSC] is used to configure a higher clock input frequency. This will allow faster overall conversion times. To meet internal ADC timing requirements, CFG2[ADHSC] adds additional ADCK cycles. Conversions with CFG2[ADHSC]=1 take two more ADCK cycles. CFG2[ADHSC] must be used when the ADCLK exceeds the limit for CFG2[ADHSC]=0.

After the module becomes active, sampling of the input begins.

1. CFG1[ADLSMP] and CFG2[ADLSTS] select between sample times based on the conversion mode that is selected.
2. When sampling is completed, the converter is isolated from the input channel and a successive approximation algorithm is applied to determine the digital value of the analog signal.
3. The result of the conversion is transferred to Rn upon completion of the conversion algorithm.

If the bus frequency is less than f_{ADCK}, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled, that is, when CFG1[ADLSMP]=0.

The maximum total conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is selectable by CFG1[ADICLK], and the divide ratio is specified by CFG1[ADIV].

The maximum total conversion time for all configurations is summarized in the equation below. See the following tables for the variables referenced in the equation.

$$\text{ConversionTime} = \text{SFCAdder} + \text{AverageNum} \times (\text{BCT} + \text{LSTAdder} + \text{HSCAdder})$$

Equation 1. Conversion time equation

Table 11-3. Single or first continuous time adder (SFCAdder)

CFG1[ADLSMP]	CFG2[ADACKEN]	CFG1[ADICLK]	Single or first continuous time adder (SFCAdder)
1	x	0x, 10	3 ADCK cycles + 5 bus clock cycles
1	1	11	3 ADCK cycles + 5 bus clock cycles[1]
1	0	11	5 µs + 3 ADCK cycles + 5 bus clock cycles
0	x	0x, 10	5 ADCK cycles + 5 bus clock cycles
0	1	11	5 ADCK cycles + 5 bus clock cycles[1]
0	0	11	5 µs + 5 ADCK cycles + 5 bus clock cycles

1. To achieve this time, CFG2[ADACKEN] must be 1 for at least 5 µs prior to the conversion is initiated.

Table 11-4. Average number factor (AverageNum)

SC3[AVGE]	SC3[AVGS]	Average number factor (AverageNum)
0	xx	1
1	00	4
1	01	8
1	10	16
1	11	32

Table 11-5. Base conversion time (BCT)

Mode	Base conversion time (BCT)
8b single-ended	17 ADCK cycles
9b differential	27 ADCK cycles
10b single-ended	20 ADCK cycles
11b differential	30 ADCK cycles
12b single-ended	20 ADCK cycles
13b differential	30 ADCK cycles

Table continues on the next page...

Table 11-5. Base conversion time (BCT) (continued)

Mode	Base conversion time (BCT)
16b single-ended	25 ADCK cycles
16b differential	34 ADCK cycles

Table 11-6. Long sample time adder (LSTAdder)

CFG1[ADLSMP]	CFG2[ADLSTS]	Long sample time adder (LSTAdder)
0	xx	0 ADCK cycles
1	00	20 ADCK cycles
1	01	12 ADCK cycles
1	10	6 ADCK cycles
1	11	2 ADCK cycles

Table 11-7. High-speed conversion time adder (HSCAdder)

CFG2[ADHSC]	High-speed conversion time adder (HSCAdder)
0	0 ADCK cycles
1	2 ADCK cycles

Note

The ADCK frequency must be between f_{ADCK} minimum and f_{ADCK} maximum to meet ADC specifications.

11.4.4.6 Conversion time examples

The following examples use the Equation 1 on page 212, and the information provided in Table 11-3 through Table 11-7.

11.4.4.6.1 Typical conversion time configuration

A typical configuration for ADC conversion is:

- 10-bit mode, with the bus clock selected as the input clock source
- The input clock divide-by-1 ratio selected
- Bus frequency of 8 MHz
- Long sample time disabled
- High-speed conversion disabled

The conversion time for a single conversion is calculated by using the Equation 1 on page 212, and the information provided in Table 11-3 through Table 11-7. The table below lists the variables of Equation 1 on page 212.

Table 11-8. Typical conversion time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles
AverageNum	1
BCT	20 ADCK cycles
LSTAdder	0
HSCAdder	0

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for a bus clock and an ADCK frequency equal to 8 MHz, the resulting conversion time is 3.75 µs.

11.4.4.6.2 Long conversion time configuration

A configuration for long ADC conversion is:

- 16-bit differential mode with the bus clock selected as the input clock source
- The input clock divide-by-8 ratio selected
- Bus frequency of 8 MHz
- Long sample time enabled
- Configured for longest adder
- High-speed conversion disabled
- Average enabled for 32 conversions

The conversion time for this conversion is calculated by using the Equation 1 on page 212, and the information provided in Table 11-3 through Table 11-7. The following table lists the variables of the Equation 1 on page 212.

Table 11-9. Typical conversion time

Variable	Time
SFCAdder	3 ADCK cycles + 5 bus clock cycles
AverageNum	32
BCT	34 ADCK cycles
LSTAdder	20 ADCK cycles
HSCAdder	0

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for bus clock equal to 8 MHz and ADCK equal to 1 MHz, the resulting conversion time is 57.625 µs, that is, AverageNum. This results in a total conversion time of 1.844 ms.

11.4.4.6.3 Short conversion time configuration

A configuration for short ADC conversion is:

- 8-bit Single-Ended mode with the bus clock selected as the input clock source
- The input clock divide-by-1 ratio selected
- Bus frequency of 20 MHz
- Long sample time disabled
- High-speed conversion enabled

The conversion time for this conversion is calculated by using the Equation 1 on page 212, and the information provided in Table 11-3 through Table 11-7. The table below lists the variables of Equation 1 on page 212.

Table 11-10. Typical conversion time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles
AverageNum	1
BCT	17 ADCK cycles
LSTAdder	0 ADCK cycles
HSCAdder	2

The resulting conversion time is generated using the parameters listed in in the preceding table. Therefore, for bus clock and ADCK frequency equal to 20 MHz, the resulting conversion time is 1.45 µs.

11.4.4.7 Hardware average function

The hardware average function can be enabled by setting SC3[AVGE]=1 to perform a hardware average of multiple conversions. The number of conversions is determined by the AVGS[1:0] bits, which can select 4, 8, 16, or 32 conversions to be averaged. While the hardware average function is in progress, SC2[ADACT] will be set.

After the selected input is sampled and converted, the result is placed in an accumulator from which an average is calculated once the selected number of conversions have been completed. When hardware averaging is selected, the completion of a single conversion will not set SC1n[COCO].

If the compare function is either disabled or evaluates true, after the selected number of conversions are completed, the average conversion result is transferred into the data result registers, Rn, and SC1n[COCO] is set. An ADC interrupt is generated upon the setting of SC1n[COCO] if the respective ADC interrupt is enabled, that is, SC1n[AIEN]=1.

Note

The hardware average function can perform conversions on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the hardware average is completed if SC1n[AIEN] is set.

11.4.5 Automatic compare function

The compare function can be configured to check whether the result is less than or greater-than-or-equal-to a single compare value, or, if the result falls within or outside a range determined by two compare values.

The compare mode is determined by SC2[ACFGT], SC2[ACREN], and the values in the compare value registers, CV1 and CV2. After the input is sampled and converted, the compare values in CV1 and CV2 are used as described in the following table. There are six Compare modes as shown in the following table.

Table 11-11. Compare modes

SC2[ACFGT]	SC2[ACREN]	ADCCV1 relative to ADCCV2	Function	Compare mode description
0	0	—	Less than threshold	Compare true if the result is less than the CV1 registers.
1	0	—	Greater than or equal to threshold	Compare true if the result is greater than or equal to CV1 registers.
0	1	Less than or equal	Outside range, not inclusive	Compare true if the result is less than CV1 **Or** the result is greater than CV2.
0	1	Greater than	Inside range, not inclusive	Compare true if the result is less than CV1 **And** the result is greater than CV2.
1	1	Less than or equal	Inside range, inclusive	Compare true if the result is greater than or equal to CV1 **And** the result is less than or equal to CV2.
1	1	Greater than	Outside range, inclusive	Compare true if the result is greater than or equal to CV1 **Or** the result is less than or equal to CV2.

With SC2[ACREN] =1, and if the value of CV1 is less than or equal to the value of CV2, then setting SC2[ACFGT] will select a trigger-if-inside-compare-range inclusive-of-endpoints function. Clearing SC2[ACFGT] will select a trigger-if-outside-compare-range, not-inclusive-of-endpoints function.

If CV1 is greater than CV2, setting SC2[ACFGT] will select a trigger-if-outside-compare-range, inclusive-of-endpoints function. Clearing SC2[ACFGT] will select a trigger-if-inside-compare-range, not-inclusive-of-endpoints function.

If the condition selected evaluates true, SC1n[COCO] is set.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, SC1n[COCO] is not set and the conversion result data will not be transferred to the result register, Rn. If the hardware averaging function is enabled, the compare function compares the averaged result to the compare values. The same compare function definitions apply. An ADC interrupt is generated when SC1n[COCO] is set and the respective ADC interrupt is enabled, that is, SC1n[AIEN]=1.

Note

The compare function can monitor the voltage on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the compare condition is met.

11.4.6 Calibration function

The ADC contains a self-calibration function that is required to achieve the specified accuracy.

Calibration must be run, or valid calibration values written, after any reset and before a conversion is initiated. The calibration function sets the offset calibration value, the minus-side calibration values, and the plus-side calibration values. The offset calibration value is automatically stored in the ADC offset correction register (OFS), and the plus-side and minus-side calibration values are automatically stored in the ADC plus-side and minus-side calibration registers, CLPx and CLMx. The user must configure the ADC correctly prior to calibration, and must generate the plus-side and minus-side gain calibration results and store them in the ADC plus-side gain register (PG) after the calibration function completes.

Prior to calibration, the user must configure the ADC's clock source and frequency, low power configuration, voltage reference selection, sample time, and high speed configuration according to the application's clock source availability and needs. If the

application uses the ADC in a wide variety of configurations, the configuration for which the highest accuracy is required should be selected, or multiple calibrations can be done for the different configurations. For best calibration results:

- Set hardware averaging to maximum, that is, SC3[AVGE]=1 and SC3[AVGS]=11 for an average of 32
- Set ADC clock frequency f_{ADCK} less than or equal to 4 MHz
- $V_{REFH}=V_{DDA}$
- Calibrate at nominal voltage and temperature

The input channel, conversion mode continuous function, compare function, resolution mode, and differential/single-ended mode are all ignored during the calibration function.

To initiate calibration, the user sets SC3[CAL] and the calibration will automatically begin if the SC2[ADTRG] is 0. If SC2[ADTRG] is 1, SC3[CAL] will not get set and SC3[CALF] will be set. While calibration is active, no ADC register can be written and no stop mode may be entered, or the calibration routine will be aborted causing SC3[CAL] to clear and SC3[CALF] to set. At the end of a calibration sequence, SC1n[COCO] will be set. SC1n[AIEN] can be used to allow an interrupt to occur at the end of a calibration sequence. At the end of the calibration routine, if SC3[CALF] is not set, the automatic calibration routine is completed successfully.

To complete calibration, the user must generate the gain calibration values using the following procedure:

1. Initialize or clear a 16-bit variable in RAM.
2. Add the plus-side calibration results CLP0, CLP1, CLP2, CLP3, CLP4, and CLPS to the variable.
3. Divide the variable by two.
4. Set the MSB of the variable.
5. The previous two steps can be achieved by setting the carry bit, rotating to the right through the carry bit on the high byte and again on the low byte.
6. Store the value in the plus-side gain calibration register PG.
7. Repeat the procedure for the minus-side gain calibration value.

When calibration is complete, the user may reconfigure and use the ADC as desired. A second calibration may also be performed, if desired, by clearing and again setting SC3[CAL].

Overall, the calibration routine may take as many as 14k ADCK cycles and 100 bus cycles, depending on the results and the clock source chosen. For an 8 MHz clock source, this length amounts to about 1.7 ms. To reduce this latency, the calibration values, which are offset, plus-side and minus-side gain, and plus-side and minus-side calibration values, may be stored in flash memory after an initial calibration and recovered prior to the first ADC conversion. This method can reduce the calibration latency to 20 register store operations on all subsequent power, reset, or Low-Power Stop mode recoveries.

Further information on the calibration procedure can be found in the Calibration section of AN3949: ADC16 Calibration Procedure and Programmable Delay Block Synchronization.

11.4.7 User-defined offset function

OFS contains the user-selected or calibration-generated offset error correction value.

This register is a 2’s complement, left-justified. The value in OFS is subtracted from the conversion and the result is transferred into the result registers, Rn. If the result is greater than the maximum or less than the minimum result value, it is forced to the appropriate limit for the current mode of operation.

The formatting of the OFS is different from the data result register, Rn, to preserve the resolution of the calibration value regardless of the conversion mode selected. Lower order bits are ignored in lower resolution modes. For example, in 8-bit single-ended mode, OFS[14:7] are subtracted from D[7:0]; OFS[15] indicates the sign (negative numbers are effectively added to the result) and OFS[6:0] are ignored. The same bits are used in 9-bit differential mode because OFS[15] indicates the sign bit, which maps to D[8]. For 16-bit differential mode, OFS[15:0] are directly subtracted from the conversion result data D[15:0]. In 16-bit single-ended mode, there is no field in the OFS corresponding to the least significant result D[0], so odd values, such as -1 or +1, cannot be subtracted from the result.

OFS is automatically set according to calibration requirements once the self-calibration sequence is done, that is, SC3[CAL] is cleared. The user may write to OFS to override the calibration result if desired. If the OFS is written by the user to a value that is different from the calibration value, the ADC error specifications may not be met. Storing the value generated by the calibration function in memory before overwriting with a user-specified value is recommended.

Note

There is an effective limit to the values of offset that can be set by the user. If the magnitude of the offset is too high, the results of the conversions will cap off at the limits.

The offset calibration function may be employed by the user to remove application offsets or DC bias values. OFS may be written with a number in 2's complement format and this offset will be subtracted from the result, or hardware averaged value. To add an offset, store the negative offset in 2's complement format and the effect will be an addition. An offset correction that results in an out-of-range value will be forced to the minimum or maximum value. The minimum value for single-ended conversions is 0x0000; for a differential conversion it is 0x8000.

To preserve accuracy, the calibrated offset value initially stored in OFS must be added to the user-defined offset. For applications that may change the offset repeatedly during operation, store the initial offset calibration value in flash so it can be recovered and added to any user offset adjustment value and the sum stored in OFS.

11.4.8 Temperature sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs.

The following equation provides an approximate transfer function of the temperature sensor.

$$Temp = 25 - \left(\left(V_{TEMP} - V_{TEMP25} \right) \div m \right)$$

Equation 2. Approximate transfer function of the temperature sensor

where:

- V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature.
- V_{TEMP25} is the voltage of the temperature sensor channel at 25 °C.
- m is referred as temperature sensor slope in the device data sheet. It is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the V_{TEMP25} and temperature sensor slope values from the ADC Electricals table.

In application code, the user reads the temperature sensor channel, calculates V_{TEMP}, and compares to V_{TEMP25}. If V_{TEMP} is greater than V_{TEMP25} the cold slope value is applied in the preceding equation. If V_{TEMP} is less than V_{TEMP25}, the hot slope value is applied in the preceding equation. ADC Electricals table may only specify one temperature sensor slope value. In that case, the user could use the same slope for the calculation across the operational temperature range.

For more information on using the temperature sensor, see the application note titled *Temperature Sensor for the HCS08 Microcontroller Family* (document AN3031).

11.4.9 MCU wait mode operation

Wait mode is a lower-power consumption Standby mode from which recovery is fast because the clock sources remain active.

If a conversion is in progress when the MCU enters Wait mode, it continues until completion. Conversions can be initiated while the MCU is in Wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two; and ADACK are available as conversion clock sources while in Wait mode. The use of ALTCLK as the conversion clock source in Wait is dependent on the definition of ALTCLK for this MCU. See the Chip Configuration information on ALTCLK specific to this MCU.

If the compare and hardware averaging functions are disabled, a conversion complete event sets SC1n[COCO] and generates an ADC interrupt to wake the MCU from Wait mode if the respective ADC interrupt is enabled, that is, when SC1n[AIEN]=1. If the hardware averaging function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, only if the compare conditions are met. If a single conversion is selected and the compare trigger is not met, the ADC will return to its idle state and cannot wake the MCU from Wait mode unless a new conversion is initiated by the hardware trigger.

11.4.10 MCU Normal Stop mode operation

Stop mode is a low-power consumption Standby mode during which most or all clock sources on the MCU are disabled.

11.4.10.1 Normal Stop mode with ADACK disabled

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a stop instruction aborts the current conversion and places the ADC in its Idle state. The contents of the ADC registers, including Rn, are unaffected by Normal Stop mode. After exiting from Normal Stop mode, a software or hardware trigger is required to resume conversions.

11.4.10.2 Normal Stop mode with ADACK enabled

If ADACK is selected as the conversion clock, the ADC continues operation during Normal Stop mode. See the chip-specific ADC information for configuration information for this device.

If a conversion is in progress when the MCU enters Normal Stop mode, it continues until completion. Conversions can be initiated while the MCU is in Normal Stop mode by means of the hardware trigger or if continuous conversions are enabled.

If the compare and hardware averaging functions are disabled, a conversion complete event sets SC1n[COCO] and generates an ADC interrupt to wake the MCU from Normal Stop mode if the respective ADC interrupt is enabled, that is, when SC1n[AIEN]=1. The result register, Rn, will contain the data from the first completed conversion that occurred during Normal Stop mode. If the hardware averaging function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, only if the compare conditions are met. If a single conversion is selected and the compare is not true, the ADC will return to its idle state and cannot wake the MCU from Normal Stop mode unless a new conversion is initiated by another hardware trigger.

11.5 Initialization information

This section gives an example that provides some basic direction on how to initialize and configure the ADC module.

The user can configure the module for 16-bit, 12-bit, 10-bit, or 8-bit single-ended resolution or 16-bit, 13-bit, 11-bit, or 9-bit differential resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. For information used in this example, refer to Table 11-6, Table 11-7, and Table 11-8.

Note

Hexadecimal values are designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

11.5.1 ADC module initialization example

11.5.1.1 Initialization sequence

Before the ADC module can be used to complete conversions, an initialization procedure must be performed. A typical sequence is:

1. Calibrate the ADC by following the calibration instructions in Calibration function.
2. Update CFG to select the input clock source and the divide ratio used to generate ADCK. This register is also used for selecting sample time and low-power configuration.
3. Update SC2 to select the conversion trigger, hardware or software, and compare function options, if enabled.
4. Update SC3 to select whether conversions will be continuous or completed only once (ADCO) and whether to perform hardware averaging.
5. Update SC1:SC1n registers to select whether conversions will be single-ended or differential and to enable or disable conversion complete interrupts. Also, select the input channel which can be used to perform conversions.

11.5.1.2 Pseudo-code example

In this example, the ADC module is set up with interrupts enabled to perform a single 10-bit conversion at low-power with a long sample time on input channel 1, where ADCK is derived from the bus clock divided by 1.

CFG1 = 0x98 (%10011000)

```
        Bit 7   ADLPC   1       Configures for low power, lowers maximum clock speed.
        Bit 6:5 ADIV    00      Sets the ADCK to the input clock ÷ 1.
        Bit 4   ADLSMP  1       Configures for long sample time.
        Bit 3:2   MODE   10     Selects the single-ended 10-bit conversion, differential 11-
bit conversion.
        Bit 1:0   ADICLK   00    Selects the bus clock.
```

SC2 = 0x00 (%00000000)

```
        Bit 7   ADACT   0       Flag indicates if a conversion is in progress.
        Bit 6   ADTRG   0       Software trigger selected.
        Bit 5   ACFE    0       Compare function disabled.
        Bit 4   ACFGT   0       Not used in this example.
        Bit 3   ACREN   0      Compare range disabled.
 Bit 2 DMAEN 0 DMA request disabled.
        Bit 1:0 REFSEL  00     Selects default voltage reference pin pair (External pins
VREFH and VREFL).
```

SC1A = 0x41 (%01000001)

```
        Bit 7   COCO    0       Read-only flag which is set when a conversion completes.
        Bit 6   AIEN    1       Conversion complete interrupt enabled.
 Bit 5 DIFF 0 Single-ended conversion selected.
        Bit 4:0 ADCH    00001   Input channel 1 selected as ADC input channel.
```

RA = 0xxx

```
        Holds results of conversion.
```

CV = 0xxx

```
        Holds compare value when compare function enabled.
```

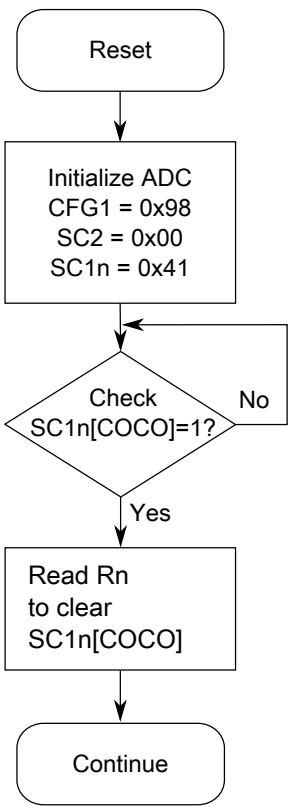


Figure 11-2. Initialization flowchart example

11.6 Application information

The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an ADC.

For guidance on selecting optimum external component values and converter parameters see AN4373: Cookbook for SAR ADC Measurements.

11.6.1 External pins and routing

11.6.1.1 Analog supply pins

Depending on the device, the analog power and ground supplies, V_{DDA} and V_{SSA}, of the ADC module are available as:

- V_{DDA} and V_{SSA} available as separate pins—When available on a separate pin, both V_{DDA} and V_{SSA} must be connected to the same voltage potential as their corresponding MCU digital supply, V_{DD} and V_{SS}, and must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.
- V_{SSA} is shared on the same pin as the MCU digital V_{SS}.
- V_{SSA} and V_{DDA} are shared with the MCU digital supply pins—In these cases, there are separate pads for the analog supplies bonded to the same pin as the corresponding digital supply so that some degree of isolation between the supplies is maintained.

If separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the V_{SSA} pin. This must be the only ground connection between these supplies, if possible. V_{SSA} makes a good single point ground location.

11.6.1.2 Analog voltage reference pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs used by the converter:

- V_{REFSH} is the high reference voltage for the converter.
- V_{REFSL} is the low reference voltage for the converter.

The ADC can be configured to accept one of two voltage reference pairs for V_{REFSH} and V_{REFSL}. Each pair contains a positive reference and a ground reference. The two pairs are external, V_{REFH} and V_{REFL} and alternate, V_{ALTH} and V_{ALTL}. These voltage references are

selected using SC2[REFSEL]. The alternate voltage reference pair, V_{ALTH} and V_{ALTL}, may select additional external pins or internal sources based on MCU configuration. See the chip configuration information on the voltage references specific to this MCU.

In some packages, the external or alternate pairs are connected in the package to V_{DDA} and V_{SSA}, respectively. One of these positive references may be shared on the same pin as V_{DDA} on some devices. One of these ground references may be shared on the same pin as V_{SSA} on some devices.

If externally available, the positive reference may be connected to the same potential as V_{DDA} or may be driven by an external source to a level between the minimum Ref Voltage High and the V_{DDA} potential. The positive reference must never exceed V_{DDA}. If externally available, the ground reference must be connected to the same voltage potential as V_{SSA}. The voltage reference pairs must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the V_{REFH} and V_{REFL} loop. The best external component to meet this current demand is a 0.1 μF capacitor with good high-frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current causes a voltage drop that could result in conversion errors. Inductance in this path must be minimum, that is, parasitic only.

11.6.1.3 Analog input pins

The external analog inputs are typically shared with digital I/O pins on MCU devices.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01 μF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used, they must be placed as near as possible to the package pins and be referenced to V_{SSA}.

For proper conversion, the input voltage must fall between V_{REFH} and V_{REFL}. If the input is equal to or exceeds V_{REFH}, the converter circuit converts the signal to 0xFFF, which is full scale 12-bit representation, 0x3FF, which is full scale 10-bit representation, or 0xFF, which is full scale 8-bit representation. If the input is equal to or less than V_{REFL}, the converter circuit converts it to 0x000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There is a brief current associated with V_{REFL} when the sampling capacitor is charging.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins must not be transitioning during conversions.

11.6.2 Sources of error

11.6.2.1 Sampling error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy.

RAS + RADIN =SC / (FMAX * NUMTAU * CADIN)

Figure 11-3. Sampling equation

Where:

RAS = External analog source resistance

SC = Number of ADCK cycles used during sample window

CADIN = Internal ADC input capacitance

NUMTAU = -ln(LSBERR / 2^N)

LSBERR = value of acceptable sampling error in LSBs

N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode or 16 in 16-bit mode

Higher source resistances or higher-accuracy sampling is possible by setting CFG1[ADLSMP] and changing CFG2[ADLSTS] to increase the sample window, or decreasing ADCK frequency to increase sample time.

11.6.2.2 Pin leakage error

Leakage on the I/O pins can cause conversion error if the external analog source resistance, R_{AS}, is high. If this error cannot be tolerated by the application, keep R_{AS} lower than V_{REFH} / (4 × I_{LEAK} × 2^N) for less than 1/4 LSB leakage error, where N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode, or 16 in 16-bit mode.

11.6.2.3 Noise-induced errors

System noise that occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 μF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 μF low-ESR capacitor from V_{DDA} to V_{SSA}.
- If inductive isolation is used from the primary supply, an additional 1 μF capacitor is placed from V_{DDA} to V_{SSA}.
- V_{SSA}, and V_{REFL}, if connected, is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in Wait or Normal Stop mode before initiating (hardware-triggered conversions) or immediately after initiating (hardware- or software-triggered conversions) the ADC conversion.
  - For software triggered conversions, immediately follow the write to SC1 with a Wait instruction or Stop instruction.
  - For Normal Stop mode operation, select ADACK as the clock source. Operation in Normal Stop reduces V_{DD} noise but increases effective conversion time due to stop recovery.
- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in Wait or Normal Stop mode, or I/O activity cannot be halted, the following actions may reduce the effect of noise on the accuracy:

- Place a 0.01 μF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSA}. This improves noise issues, but affects the sample rate based on the external analog source resistance.
- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1 LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock, that is, ADACK, and averaging. Noise that is synchronous to ADCK cannot be averaged out.

11.6.2.4 Code width and quantization error

The ADC quantizes the ideal straight-line transfer function into 65536 steps in the 16-bit mode.. Each step ideally has the same height, that is, 1 code, and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N-bit converter, where N can be 16, 12, 10, or 8, defined as 1 LSB, is:

$$1LSB=\left(V_{REFH}\right)/2^{N}$$

Equation 3. Ideal code width for an N-bit converter

There is an inherent quantization error due to the digitization of the result. For 8-bit, 10-bit, or 12-bit conversions, the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be ± 1/2 LSB in 8-bit, 10-bit, or 12-bit modes. As a consequence, however, the code width of the first (0x000) conversion is only 1/2 LSB and the code width of the last (0xFF or 0x3FF) is 1.5 LSB.

For 16-bit conversions, the code transitions only after the full code width is present, so the quantization error is -1 LSB to 0 LSB and the code width of each step is 1 LSB.

11.6.2.5 Linearity errors

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors, but the system designers must be aware of these errors because they affect overall accuracy:

- Zero-scale error (E_{ZS}), sometimes called offset: This error is defined as the difference between the actual code width of the first conversion and the ideal code width. This is 1/2 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode. If the first conversion is 0x001, the difference between the actual 0x001 code width and its ideal (1 LSB) is used.
- Full-scale error (E_{FS}): This error is defined as the difference between the actual code width of the last conversion and the ideal code width. This is 1.5 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode. If the last conversion is 0x3FE, the difference between the actual 0x3FE code width and its ideal (1 LSB) is used.
- Differential non-linearity (DNL): This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.

- Integral non-linearity (INL): This error is defined as the highest-value or absolute value that the running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE): This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

11.6.2.6 Code jitter, non-monotonicity, and missing codes

Analog-to-digital converters are susceptible to three special forms of error:

- Code jitter: Code jitter occurs when a given input voltage converts to one of the two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code, and vice-versa. However, even small amounts of system noise can cause the converter to be indeterminate, between two codes, for a range of input voltages around the transition voltage.

  This error may be reduced by repeatedly sampling the input and averaging the result. Additionally, the techniques discussed in Noise-induced errors reduces this error.

- Non-monotonicity: Non-monotonicity occurs when, except for code jitter, the converter converts to a lower code for a higher input voltage.
- Missing codes: Missing codes are those values never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and have no missing codes.

Chapter 12
Crossbar Switch Lite (AXBS-Lite)

12.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The information found here provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows up to four bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave.

12.1.1 Features

The crossbar switch includes these features:

- Symmetric crossbar bus switch implementation
  - Allows concurrent accesses from different masters to different slaves
- Up to single-clock 32-bit transfer
- Programmable configuration for fixed-priority or round-robin slave port arbitration (see the chip-specific information).

12.2 Memory Map / Register Definition

This crossbar switch is designed for minimal gate count. It, therefore, has no memory-mapped configuration registers.

Please see the chip-specific information for information on whether the arbitration method in the crossbar switch is programmable, and by which module.

12.3 Functional Description

12.3.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master simply sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device has no knowledge of whether it actually owns the slave port it is targeting. While the master does not have control of the slave port it is targeting, it simply waits.

After the master has control of the slave port it is targeting, the master remains in control of the slave port until it relinquishes the slave port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it remains parked with the last master to use the slave port. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port.

Chapter 13
Bit Manipulation Engine2 (BME2)

13.1 Introduction

The Bit Manipulation Engine with the BME2 definition (BME2) provides hardware support for atomic read-modify-write memory operations to the first 1 Mbyte of slave peripheral address space in Cortex-M based microcontrollers. The second generation BME2 provides similar atomic read-modify-write capabilities to a larger peripheral address space versus the original implementation. The BME terminology is meant to be equivalent to BME2 throughout this chapter.

This architectural capability is also known as "decorated storage" as it defines a mechanism for providing additional semantics for load and store operations to memory-mapped peripherals beyond just the reading and writing of data values to the addressed memory locations. In the BME definition, the "decoration", that is, the additional semantic information, is encoded into the peripheral address used to reference the memory. The BME2 definition supports up to two AIPS bus controllers, each capable of addressing 512 KB of peripheral address space. BME2 supports an 8-bit or less data field width for bit field inserts and extracts, regardless of reference size. All other BME2 operations are exactly the same as the original definition.

By combining the basic load and store instructions of the ARM Cortex-M instruction set architecture (v6M, v7M) with the concept of decorated storage provided by BME, the resulting implementation provides a robust and efficient read-modify-write capability. The resulting architectural capability defined by this core platform function is targeted at the manipulation of n-bit fields in peripheral registers and is consistent with I/O hardware addressing in the Embedded C standard. For most BME commands, a single core read or write bus cycle is converted into an atomic read-modify-write, that is, an indivisible "read followed by a write" bus sequence.

BME decorated references are only available on system bus transactions generated by the processor core and DMA, and targeted at the 1024 KB peripheral address space based at 0x4000_0000. The decoration semantic is embedded into address bits[28:20], creating a

448 MB space at addresses 0x4400_0000–0x5FFF_FFFF; these bits are stripped out of the actual address sent to the peripheral bus controller and used by the BME to define and control its operation.

13.1.1 Features

The key features of the BME2 include:

- Lightweight implementation of decorated storage for selected address spaces
- Additional access semantics encoded into the reference address
- Resides between crossbar switch slave port(s) and their associated peripheral bridge bus controller(s)
- Two-stage pipeline design matching the AHB system bus protocol
- Combinationally passes non-decorated accesses to peripheral bridge bus controllers
- Conversion of decorated loads and stores from processor core into atomic read-modify-writes
- Decorated loads support unsigned bit field extracts, load-and-{set,clear} 1-bit operations
- Decorated stores support bit field inserts, logical AND, OR, and XOR operations
- Support for byte, halfword and word-sized decorated operations
- Supports minimum signal toggling on AHB output bus to reduce power dissipation

13.1.2 Modes of operation

The BME modules do not support any special modes of operation. As memory-mapped devices located on crossbar slave AHB system bus ports, BME responds strictly on the basis of memory addresses for accesses to the SRAM_U and AIPS controllers.

All functionality associated with the BME modules reside in the core platform's clock domain; this includes their connections with the crossbar slave ports, SRAM_U, and the AIPS controllers.

13.2 Memory map and register definition

The BME module provides a memory-mapped capability and does not include any programming model registers.

The exact set of functions supported by the BME are detailed in the Functional description.

The generic BME2 definition supports a 1024 KB address space based at 0x4000_0000.... . The decorated address space is mapped to the 448 MB region located at 0x4400_0000–0x5FFF_FFFF.

13.3 Functional description

Information found here details the specific functions supported by the BME.

Recall the combination of the basic load and store instructions of the Cortex-M instruction set architecture (v6M, v7M) plus the concept of decorated storage provided by the BME provides a robust and efficient read-modify-write capability. The resulting architectural capability defined by this core platform function is targeted at the manipulation of n-bit fields in peripheral registers and RAM and is consistent with I/O hardware addressing in the Embedded C standard. For most BME commands, a single core read or write bus cycle is converted into an atomic read-modify-write, that is, an indivisible "read followed by a write" bus sequence.

Consider decorated store operations first, then decorated loads.

13.3.1 BME decorated stores

The functions supported by the BME's decorated stores include three logical operators (AND, OR, XOR) plus a bit field insert.

For all these operations, BME converts a single decorated AHB store transaction into a 2-cycle atomic read-modify-write sequence, where the combined read-modify operation is performed in the first AHB data phase, and then the write is performed in the second AHB data phase.

A generic timing diagram of a decorated store showing a peripheral bit field insert operation is shown as follows:

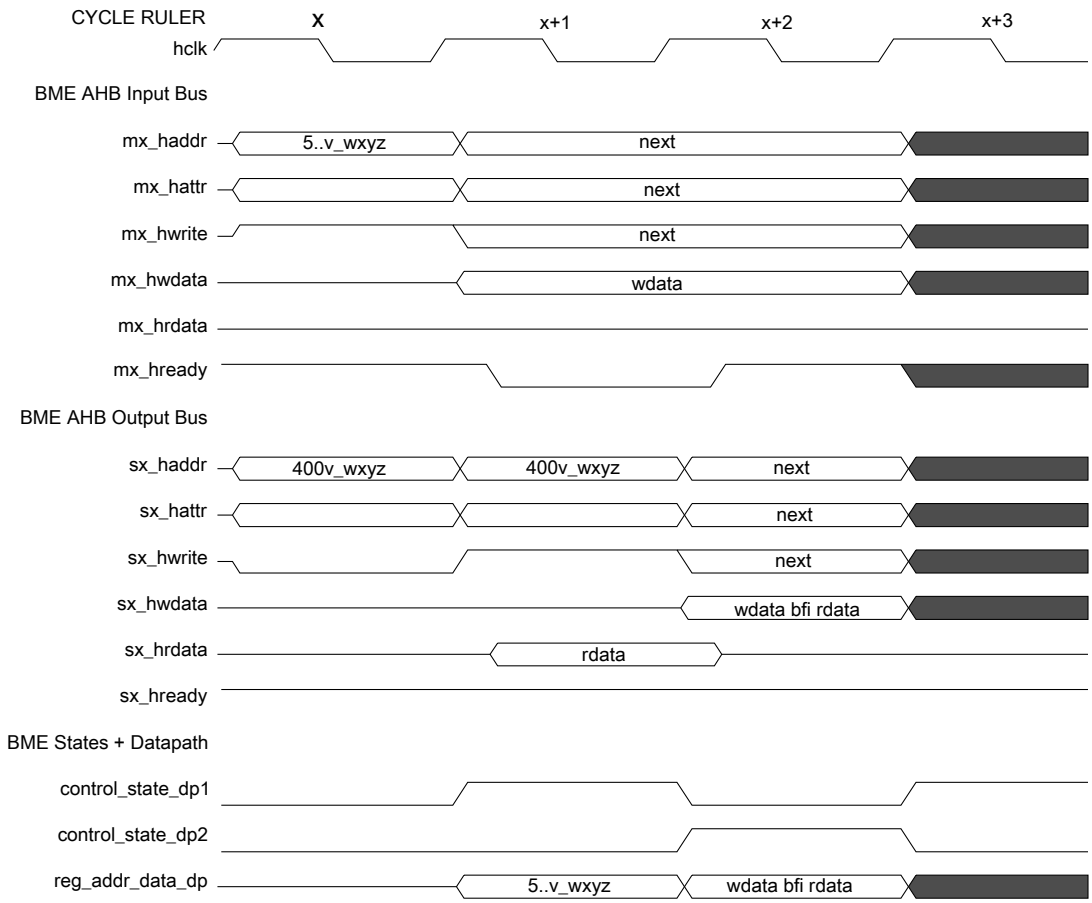


Figure 13-1. Decorated store: bit field insert timing diagram

All the decorated store operations follow the same execution template shown in Figure 13-1, a two-cycle read-modify-write operation:

1. Cycle x, 1st AHB address phase: Write from input bus is translated into a read operation on the output bus using the actual memory address (with the decoration removed) and then captured in a register.
2. Cycle x+1, 2nd AHB address phase: Write access with the registered (but actual) memory address is output
3. Cycle x+1, 1st AHB data phase: Memory read data is modified using the input bus write data and the function defined by the decoration and captured in a data register; the input bus cycle is stalled.
4. Cycle x+2, 2nd AHB data phase: Registered write data is sourced onto the output write data bus.

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

13.3.1.1 Decorated store logical AND (AND)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read;
2. It is then modified by performing a logical AND operation using the write data operand sourced for the system bus cycle
3. Finally, the result of the AND operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

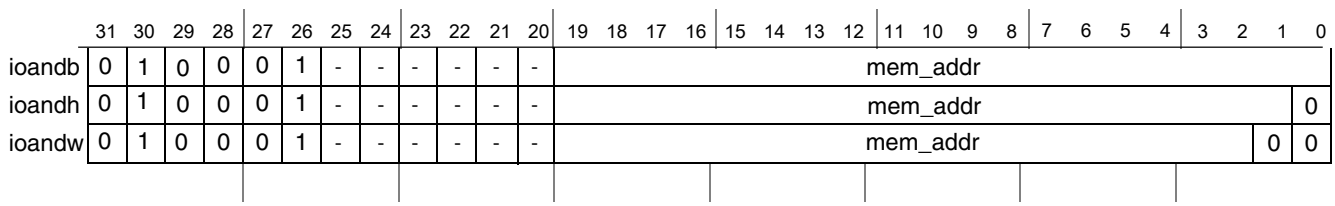


Figure 13-2. Decorated store address: logical AND

See Figure 13-2 where addr[30:29] = 10 for peripheral, addr[28:26] = 001 specifies the AND operation, and mem_addr[19:0] specifies the address offset into the space based at a 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated AND write operation is defined in the following pseudo-code as:

```
ioand<sz>(accessAddress, wdata)                  // decorated store AND
tmp   =  mem[accessAddress & 0xE00FFFFF, size]   // memory read
tmp   =  tmp & wdata                             // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp      // memory write
```

where the operand size <sz> is defined as b(yte, 8-bit), h(alfword, 16-bit) and w(ord, 32-bit). This notation is used throughout the document.

In the cycle definition tables, the notations AHB_ap and AHB_dp refer to the address and data phases of the BME AHB transaction. The cycle-by-cycle BME operations are detailed in the following table.

Table 13-1. Cycle definitions of decorated store: logical AND

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert	Recirculate captured addr + attr to memory as slave_wt	<next>

Table continues on the next page...

Table 13-1. Cycle definitions of decorated store: logical AND (continued)

Pipeline stage	Cycle		
	x	x+1	x+2
	master_wt to slave_rd; Capture address, attributes		
BME AHB_dp	<previous>	Perform memory read; Form (rdata & wdata) and capture destination data in register	Perform write sending registered data to memory

13.3.1.2 Decorated store logical OR (OR)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read.
2. It is then modified by performing a logical OR operation using the write data operand sourced for the system bus cycle.
3. Finally, the result of the OR operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

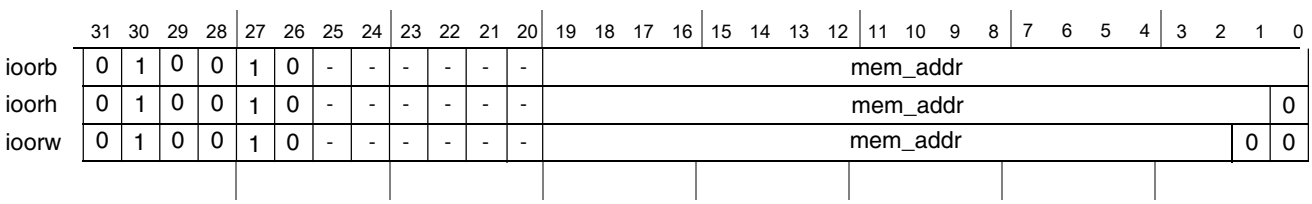


Figure 13-3. Decorated address store: logical OR

See Figure 13-3,where addr[30:29] =10 for peripheral, addr[28:26] = 010 specifies the OR operation, and mem_addr[19:0] specifies the address offset into the space based at a 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated OR write operation is defined in the following pseudo-code as:

```
ioor<sz>(accessAddress, wdata)                  // decorated store OR

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
tmp   =  tmp | wdata                            // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp     // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 13-2. Cycle definitions of decorated store: logical OR

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form (rdata | wdata) and capture destination data in register	Perform write sending registered data to memory

13.3.1.3 Decorated store logical XOR (XOR)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read.
2. It is then modified by performing a logical XOR (exclusive-OR) operation using the write data operand sourced for the system bus cycle.
3. Finally, the result of the XOR operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

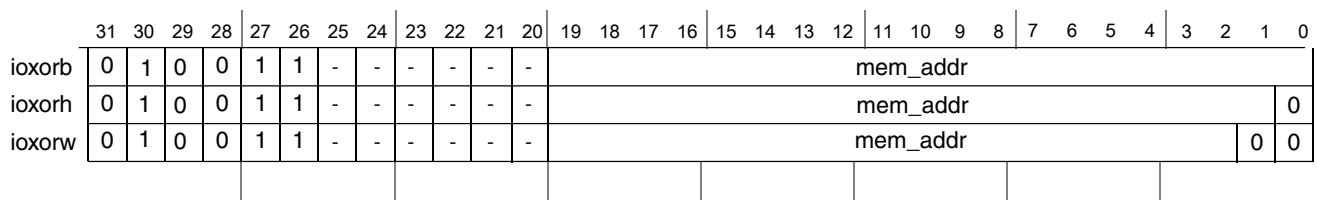


Figure 13-4. Decorated address store: logical XOR

See Figure 13-4, where addr[30:29] =10 for peripheral, addr[28:26] = 011 specifies the XOR operation, and mem_addr[19:0] specifies the address offset into the peripheral space based at a 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated XOR write operation is defined in the following pseudo-code as:

```
ioxor<sz>(accessAddress, wdata)                  // decorated store XOR

tmp   =  mem[accessAddress & 0xE00FFFFF, size]   // memory read
tmp   =  tmp ^ wdata                             // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp      // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 13-3. Cycle definitions of decorated store: logical XOR

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form (rdata ^ wdata) and capture destination data in register	Perform write sending registered data to memory

13.3.1.4 Decorated store bit field insert (BFI)

This command inserts a bit field contained in the write data operand, defined by LSB position (b) and the bit field width (w+1), into the memory "container" defined by the access size associated with the store instruction using an atomic read-modify-write sequence.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). For BME2, the data field width (addr[22:20]) for bit field inserts and extracts is limited to an 8-bit or less value, regardless of reference size.

NOTE

The maximum bit field width is 8 bits. The core performs the required write data lane replication on byte and halfword transfers.

The BFI operation can be used to insert a single bit into a peripheral. For this case, the w field is simply set to 0, indicating a bit field width of 1.

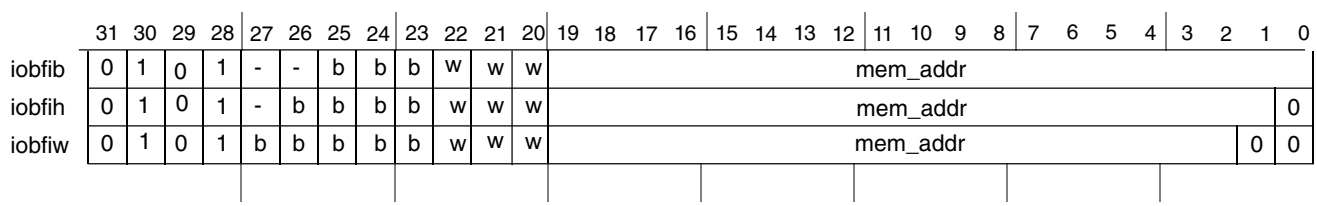


Figure 13-5. Decorated address store: bit field insert

where addr[30:29] =10 for peripheral, addr[28] = 1 signals a BFI operation, addr[27:23] is "b", the LSB identifier, addr[22:20] is "w", the bit field width minus 1 identifier, and addr[19:0] specifies the address offset into the peripheral space based at 0x4000_0000. The "-" indicates an address bit "don't care".

The decorated BFI write operation is defined in the following pseudo-code as:

```
iobfi<sz>(accessAddress, wdata)                  // decorated bit field insert

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
mask  = ((1 << (w+1)) - 1) << b                 // generate bit mask
tmp   =  tmp   & ~mask                          // modify
      |  wdata &  mask
mem[accessAddress & 0xE00FFFFF, size] = tmp     // memory write
```

The write data operand (wdata) associated with the store instruction contains the bit field to be inserted. It must be properly aligned within the right-aligned container, that is, within the lower 8 bits for a byte operation, the lower 16 bits for a halfword, or the entire 32 bits for a word operation.

To illustrate, consider the following example of the insertion of the 3-bit field "xyz" into an 8-bit memory container, initially set to "abcd_efgh". For all cases, w is 2, signaling a bit field width of 3.

```
if b = 0 and the decorated store (strb) Rt register[7:0] = ----_-xyz,
   then destination is "abcd_exyz"
if b = 1 and the decorated store (strb) Rt register[7:0] = ----_xyz-,
   then destination is "abcd_xyzh"
if b = 2 and the decorated store (strb) Rt register[7:0] = ---x_yz--,
   then destination is "abcx_yzgh"
if b = 3 and the decorated store (strb) Rt register[7:0] = --xy_z---,
   then destination is "abxy_zfgh"
if b = 4 and the decorated store (strb) Rt register[7:0] = -xyz_----,
   then destination is "axyz_efgh"
if b = 5 and the decorated store (strb) Rt register[7:0] = xyz-_----,
   then destination is "xyzd_efgh"
if b = 6 and the decorated store (strb) Rt register[7:0] = yz--_----,
   then destination is "yzcd_efgh"
if b = 7 and the decorated store (strb) Rt register[7:0] = z---_----,
   then destination is "zbcd_efgh"
```

Note from the example, when the starting bit position plus the field width exceeds the container size, only part of the source bit field is inserted into the destination memory location. Stated differently, if (b + w+1) > container_width, only the low-order "container_width - b" bits are actually inserted.

The cycle-by-cycle BME operations are detailed in the following table.

Table 13-4. Cycle definitions of decorated store: bit field insert

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Form bitwise ((mask) ? wdata : rdata)) and capture destination data in register	Perform write sending registered data to memory

13.3.2 BME decorated loads

The functions supported by the BME's decorated loads include two single-bit load-and-{set, clear} operators plus unsigned bit field extracts.

For the two load-and-{set, clear} operations, BME converts a single decorated AHB load transaction into a two-cycle atomic read-modify-write sequence, where the combined read-modify operations are performed in the first AHB data phase, and then the write is performed in the second AHB data phase as the original read data is returned to the processor core. For an unsigned bit field extract, the decorated load transaction is stalled for one cycle in the BME as the data field is extracted, then aligned and returned to the processor in the second AHB data phase. This is the only decorated transaction that is not an atomic read-modify-write, as it is a simple data read.

A generic timing diagram of a decorated load showing a peripheral load-and-set 1-bit operation is shown as follows.

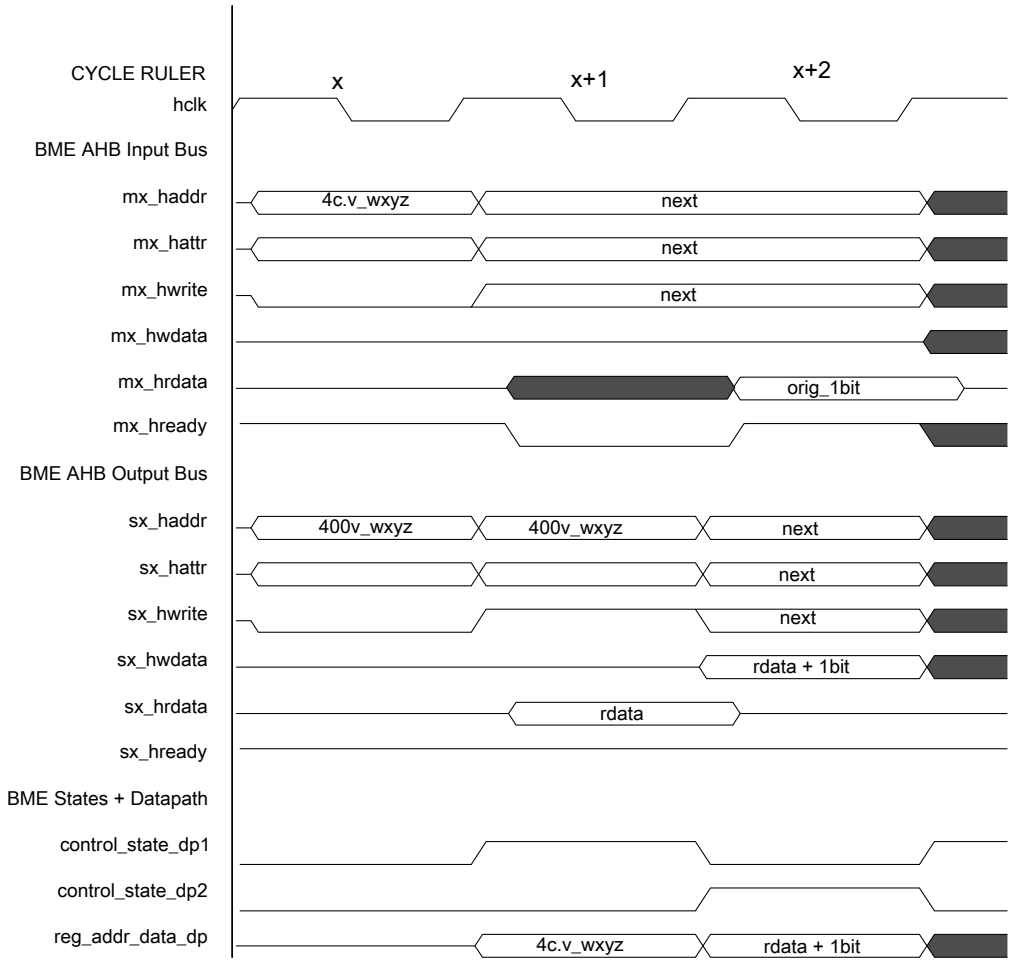


Figure 13-6. Decorated load: load-and-set 1-bit field insert timing diagram

Decorated load-and-{set, clear} 1-bit operations follow the execution template shown in the above figure: a 2-cycle read-modify-write operation:

1. Cycle x, first AHB address phase: Read from input bus is translated into a read operation on the output bus with the actual memory address (with the decoration removed) and then captured in a register
2. Cycle x+1, second AHB address phase: Write access with the registered (but actual) memory address is output
3. Cycle x+1, first AHB data phase: The "original" 1-bit memory read data is captured in a register, while the 1-bit field is set or clear based on the function defined by the decoration with the modified data captured in a register; the input bus cycle is stalled
4. Cycle x+2, second AHB data phase: The selected original 1-bit is right-justified, zero-filled and then driven onto the input read data bus, while the registered write data is sourced onto the output write data bus

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

A generic timing diagram of a decorated load showing an unsigned peripheral bit field operation is shown in the following figure.

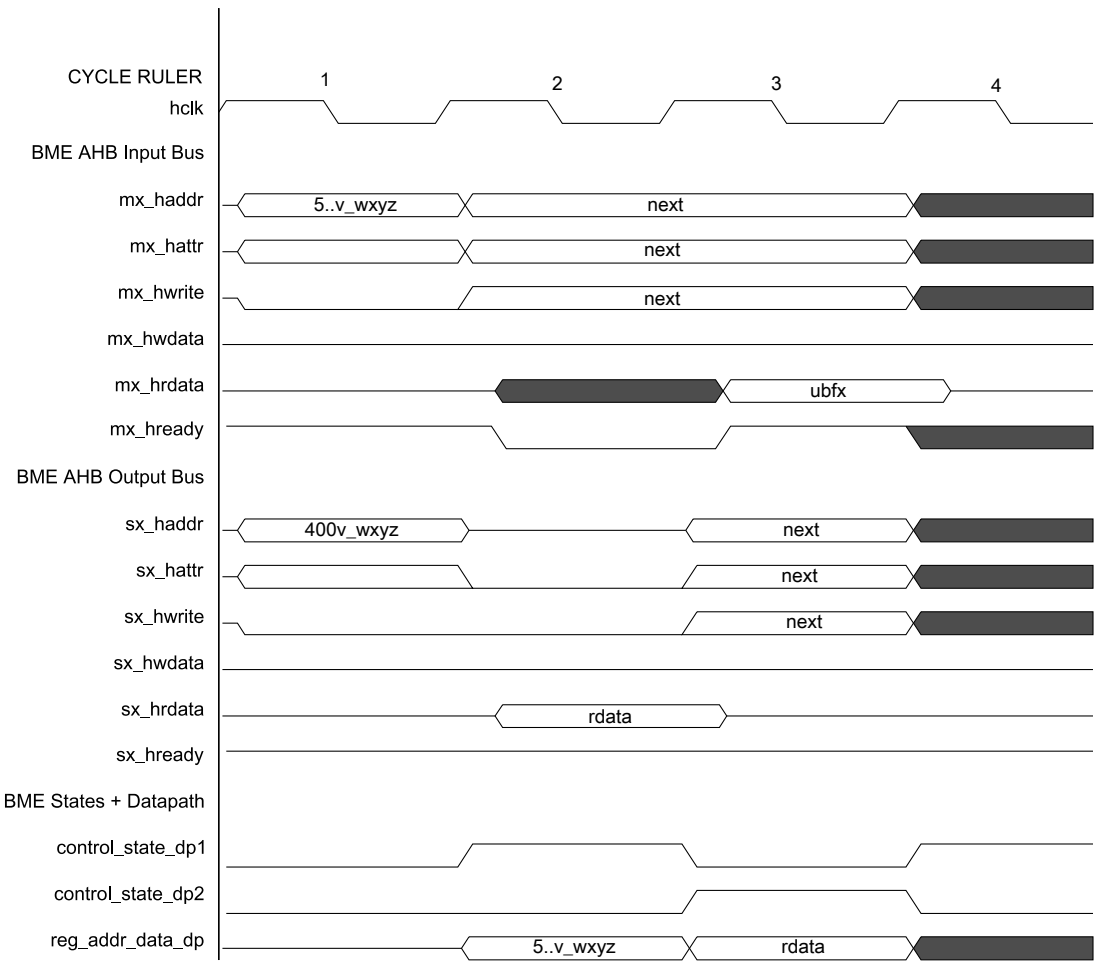


Figure 13-7. Decorated load: unsigned bit field insert timing diagram

The decorated unsigned bit field extract follows the same execution template shown in the above figure, a 2-cycle read operation:

- Cycle x, 1st AHB address phase: Read from input bus is translated into a read operation on the output bus with the actual memory address (with the decoration removed) and then captured in a register
- Cycle x+1, 2nd AHB address phase: Idle cycle

- Cycle x+1, 1st AHB data phase: A bit mask is generated based on the starting bit position and the field width; the mask is AND'ed with the memory read data to isolate the bit field; the resulting data is captured in a data register; the input bus cycle is stalled
- Cycle x+2, 2nd AHB data phase: Registered data is logically right-aligned for proper alignment and driven onto the input read data bus

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

13.3.2.1 Decorated load: load-and-clear 1 bit (LAC1)

This command loads a 1-bit field defined by the LSB position (b) into the core's general purpose destination register (Rt) and zeroes the bit in the memory space after performing an atomic read-modify-write sequence.

The extracted 1-bit data field from the memory address is right-justified and zero-filled in the operand returned to the core.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

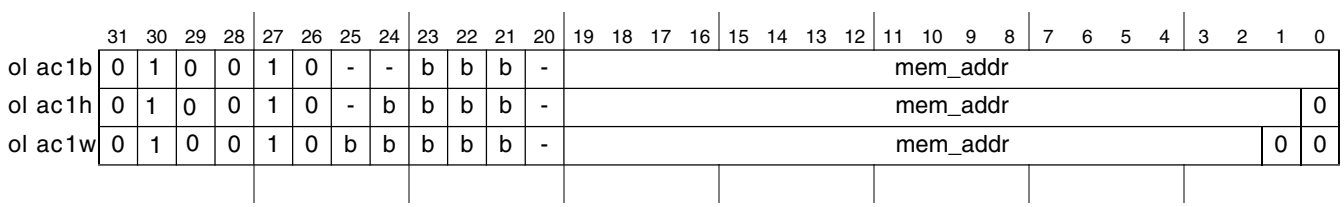


Figure 13-8. Decorated load address: load-and-clear 1 bit

See Figure 13-8 where addr[30:29] = 01 for SRAM_U, addr[30:29] = 10 for peripherals, addr[28:26] = 010 specifies the load-and-clear 1 bit operation, addr[25:21] is "b", the bit identifier, and mem_addr[19:0] specifies the address offset into the space based at 0x2000_0000 for SRAM_U, and 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated load-and-clear 1-bit read operation is defined in the following pseudo-code as:

```
rdata =  iolac1<sz>(accessAddress)              // decorated load-and-clear 1

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
mask  =  1 << b                                 // generate bit mask
rdata = (tmp &  mask) >> b                      // read data returned to core
```

```
tmp    =  tmp & ~mask                              // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp        // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 13-5. Cycle definitions of decorated load: load-and-clear 1 bit

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Extract bit from rdata; Form (rdata & ~mask) and capture destination data in register	Return extracted bit to master; Perform write sending registered data to memory

13.3.2.2 Decorated Load: Load-and-Set 1 Bit (LAS1)

This command loads a 1-bit field defined by the LSB position (b) into the core's general purpose destination register (Rt) and sets the bit in the memory space after performing an atomic read-modify-write sequence.

The extracted one bit data field from the memory address is right justified and zero filled in the operand returned to the core.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

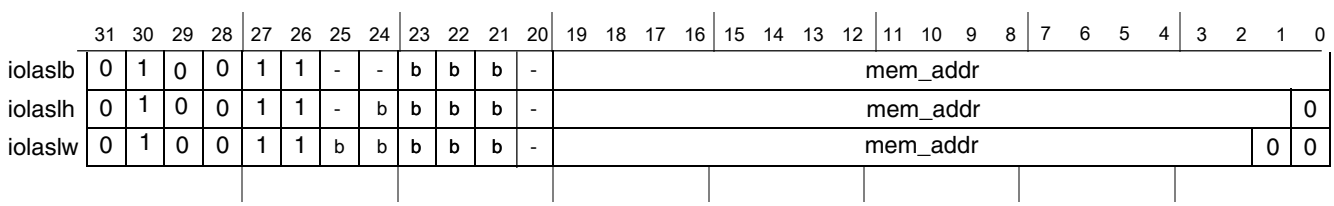


Figure 13-9. Decorated load address: load-and-set 1 bit

where addr[30:29] = 01 for SRAM_U, addr[30:29] = 10 for peripherals, addr[28:26] = 011 specifies the load-and-set 1 bit operation, addr[25:21] is "b", the bit identifier, and mem_addr[19:0] specifies the address offset into the space based at 0x2000_0000 for SRAM_U, and 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated Load-and-Set 1 Bit read operation is defined in the following pseudo-code as:

```
rdata =  iolas1<sz>(accessAddress)              // decorated load-and-set 1

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
mask  =  1 << b                                 // generate bit mask
rdata = (tmp &  mask) >> b                      // read data returned to core
tmp   =  tmp |  mask                            // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp     // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 13-6. Cycle definitions of decorated load: load-and-set 1-bit

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Extract bit from rdata; Form (rdata | mask) and capture destination data in register	Return extracted bit to master; Perform write sending registered data to memory

13.3.2.3 Decorated load unsigned bit field extract (UBFX)

This command extracts a bit field defined by LSB position (b) and the bit field width (w +1) from the memory "container" defined by the access size associated with the load instruction using a two-cycle read sequence.

The extracted bit field from the memory address is right-justified and zero-filled in the operand returned to the core. Recall this is the only decorated operation that does not perform a memory write, that is, UBFX only performs a read.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The maximum bit field width is 8 bits, regardless of reference size.

The use of a UBFX operation is recommended to extract a single bit. For this case, the w field is simply set to 0, indicating a bit field width of 1.

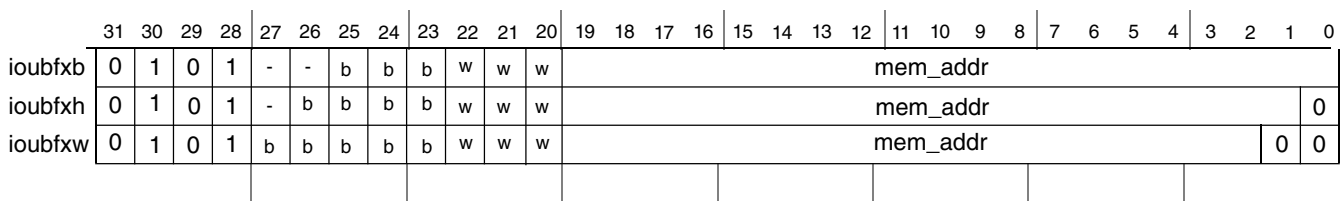


Figure 13-10. Decorated load address: unsigned bit field extract

See Figure 13-10, where addr[30:29] = 10 for peripherals, addr[28] = 1 specifies the unsigned bit field extract operation, addr[27:23] is "b", the LSB identifier, addr[22:20] is "w", the bit field width minus 1 identifier, and mem_addr[19:0] specifies the address offset into the space based at 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated unsigned bit field extract read operation is defined in the following pseudo-code as:

```
rdata =  ioubfx<sz>(accessAddress)              // unsigned bit field extract

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
mask  = ((1 << (w+1)) - 1) << b                 // generate bit mask
rdata = (tmp &  mask) >> b                      // read data returned to core
```

Like the BFI operation, when the starting bit position plus the field width exceeds the container size, only part of the source bit field is extracted from the destination memory location. Stated differently, if (b + w+1) > container_width, only the low-order "container_width - b" bits are actually extracted. The cycle-by-cycle BME operations are detailed in the following table.

Table 13-7. Cycle definitions of decorated load: unsigned bit field extract

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Capture address, attributes	Idle AHB address phase	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Form (rdata & mask) and capture destination data in register	Logically right shift registered data; Return justified rdata to master

13.4 Application information

In this section, GNU assembler macros with C expression operands are presented as examples of the required instructions to perform decorated operations.

This section specifically presents a partial bme.h file defining the assembly language expressions for decorated logical stores: AND, OR, and XOR. Comparable functions for BFI and the decorated loads are more complex and available in the complete BME header file.

These macros use the same function names presented in Functional description.

```
#define IOANDW(ADDR,WDATA)          \
    __asm("ldr    r3, =(1<<26);"    \
          "orr    r3, %[addr];"     \
```

```
          "mov    r2, %[wdata];"    \
          "str    r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOANDH(ADDR,WDATA)          \
    __asm("ldr    r3, =(1<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strh   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOANDB(ADDR,WDATA)          \
    __asm("ldr    r3, =(1<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strb   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORW(ADDR,WDATA)           \
    __asm("ldr    r3, =(1<<27);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "str    r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORH(ADDR,WDATA)           \
    __asm("ldr    r3, =(1<<27);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strh   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORB(ADDR,WDATA)           \
    __asm("ldr    r3, =(1<<27);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strb   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOXORW(ADDR,WDATA)          \
    __asm("ldr    r3, =(3<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "str    r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOXORH(ADDR,WDATA)          \
    __asm("ldr    r3, =(3<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strh   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOXORB(ADDR,WDATA)          \
    __asm("ldr    r3, =(3<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strb   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");
```

Chapter 14
Kinetis ROM Bootloader

14.1 Chip-Specific Information

14.1.1 Kinetis Bootloader Peripheral Pinmux

This device has various peripherals (LPUART, LPI2C, LPSPI , USB) supported by the Kinetis ROM Bootloader. To use an interface for bootloader communications, the peripheral must be enabled in the BCA, as shown in The Kinetis Bootloader Configuration Area (BCA). If the BCA is invalid (such as all 0xFF bytes), then all peripherals will be enabled by default.The next table shows the pads used by the Kinetis ROM Bootloader.

Table 14-1. Kinetis Bootloader Peripheral Pinmux

Peripheral	Instance	Alt Mode	Pins
LPUART	0	2	PTA1, LPUART0_RX
			PTA2, LPUART0_TX
	1	3	PTC3, LPUART1_RX
			PTC4, LPUART1_TX
	2	3	PTD2, LPUART2_RX
			PTD3, LPUART2_TX
I2C	0	5	PTE24, I2C0_SCL
			PTE25, I2C0_SDA
	1	6	PTE1, I2C1_SCL
			PTE0, I2C1_SDA
	2 (I2C instance 2 is only available on 121 XFBGA package[1])	5	PTA12, I2C2_SCL
			PTA11, I2C2_SDA
SPI	0	2	PTA14, SPI0_PCS0
			PTA15, SPI0_SCK

Table continues on the next page...

Table 14-1. Kinetis Bootloader Peripheral Pinmux (continued)

Peripheral	Instance	Alt Mode	Pins
			PTA16, SPI0_MOSI
			PTA17, SPI0_MISO
	1	2	PTD4, SPI1_PCS0
			PTD5, SPI1_SCK
			PTD6, SPI1_MOSI
			PTD7, SPI1_MISO
	2[2]	2	PTB20, SPI2_PCS0
			PTB21, SPI2_SCK
			PTB22, SPI2_MOSI
			PTB23, SPI2_MISO
USB	0	0	USB0_DP
			USB0_DM

1. The 121-pin XFBGA package for this product is not yet available. However, it is included in a Package Your Way program for Kinetis MCUs. Visit nxp.com/KPYW for more details.
2. SPI instances 0 and 2 are only available on 121 XFBGA and 100 LQFP packages.

14.1.2 Bootloader Memory Access

While executing, the Kinetis Bootloader uses the following address ranges in ROM and RAM memory:

- 0x1C00_0000 – 0x1C00_7FFF (ROM)
- 0x1FFF_A000 – 0x1FFF_B5C7 (RAM)

14.2 Introduction

The Kinetis bootloader is the program residing in the on-chip read-only memory (ROM) of a Kinetis microcontroller device. There is hardware logic in place at boot time that either starts execution of an embedded image available on the internal flash memory, or starts the execution of the Kinetis Bootloader from on-chip ROM.

The Kinetis Bootloader's main task is to provision the internal flash memory with an embedded firmware image during manufacturing, or at any time during the life of the Kinetis device. The Kinetis Bootloader does the provisioning by acting as a slave device, and listening to various peripheral ports where a master can start communication.

For the Kinetis device, the Kinetis Bootloader can interface with USB, LPI2C, LPSPI, and LPUART peripherals in slave mode and respond to the commands sent by a master (or host) communicating on one of those ports. The host/master can be a firmware-

download application running on a PC or an embedded host communicating with the Kinetis Bootloader. Regardless of the host/master (PC or embedded host), the Kinetis Bootloader always uses a command protocol to communicate with that host/master. Commands are provided to write to memory (internal flash or RAM), erase flash, and get/set bootloader options and property values. The host application can query the set of available commands.

On start-up, the bootloader reads optional configuration parameters from a fixed area on flash called the bootloader configuration area (BCA). These parameters can be modified by the write memory command or by downloaded flash image. BCA parameters include configuration data such as enabled peripherals, peripheral-specific settings, etc.

This chapter describes Kinetis Bootloader features, functionality, command structure and which peripherals are supported.

Features supported by the Kinetis Bootloader in Kinetis ROM:

- Supports USB, LPI2C, LPSPI, and LPUART peripheral interfaces
- Automatic detection of the active peripheral
- Ability to disable any peripheral
- LPUART peripheral implements autobaud
- Common packet-based protocol for all peripherals
- Packet error detection and retransmission
- Flash-resident configuration options
- Fully supports internal flash security, including ability to mass erase or unlock security via the backdoor key
- Protection of RAM used by the bootloader while it is running
- Provides command to read properties of the device, such as flash and RAM size
- Multiple options for executing the bootloader either at system start-up or under application control at runtime
- Supports internal flash
- Supports encrypted image download

Table 14-2. Commands supported by the Kinetis Bootloader in ROM

Command	Description	When flash security is enabled, then this command is
Call	Runs user application code and returns control to bootloader	Not supported
Execute	Run user application code that never returns control to the bootloader	Not supported
FillMemory	Fill a range of bytes in flash with a word pattern	Not supported
FlashEraseAll	Erase the entire flash array	Not supported
FlashEraseRegion	Erase a range of sectors in flash	Not supported

Table continues on the next page...

Table 14-2. Commands supported by the Kinetis Bootloader in ROM (continued)

Command	Description	When flash security is enabled, then this command is
WriteMemory	Write data to memory	Not supported
ReadMemory	Read data from memory	Not supported
FlashProgramOnce	Writes data provided in a command packet to a specified range of bytes in the program once field	Not supported
FlashReadOnce	Returns the contents of the program once field by given index and byte count	Not supported
FlashReadResource	Returns the contents of the IFR field or Flash firmware ID, by given offset, byte count and option	Not supported
FlashSecurityDisable	Attempt to unlock flash security using the backdoor key	Supported
GetProperty	Get the current value of a property	Supported
ReceiveSbFile	Receive an SB file (which is generated by the `elftosb` tool)	Supported if the SB file is encrypted. See the SB File Decryption Support section for more details.
Reset	Reset the chip	Supported
SetProperty	Attempt to modify a writable property	Supported
FlashEraseAllUnsecure	Erase the entire flash array, including protected sectors	Supported

14.3 Functional Description

The following sub-sections describe the Kinetis Bootloader in ROM functionality.

14.3.1 Memory Maps

While executing, the Kinetis Bootloader uses ROM and RAM memory.

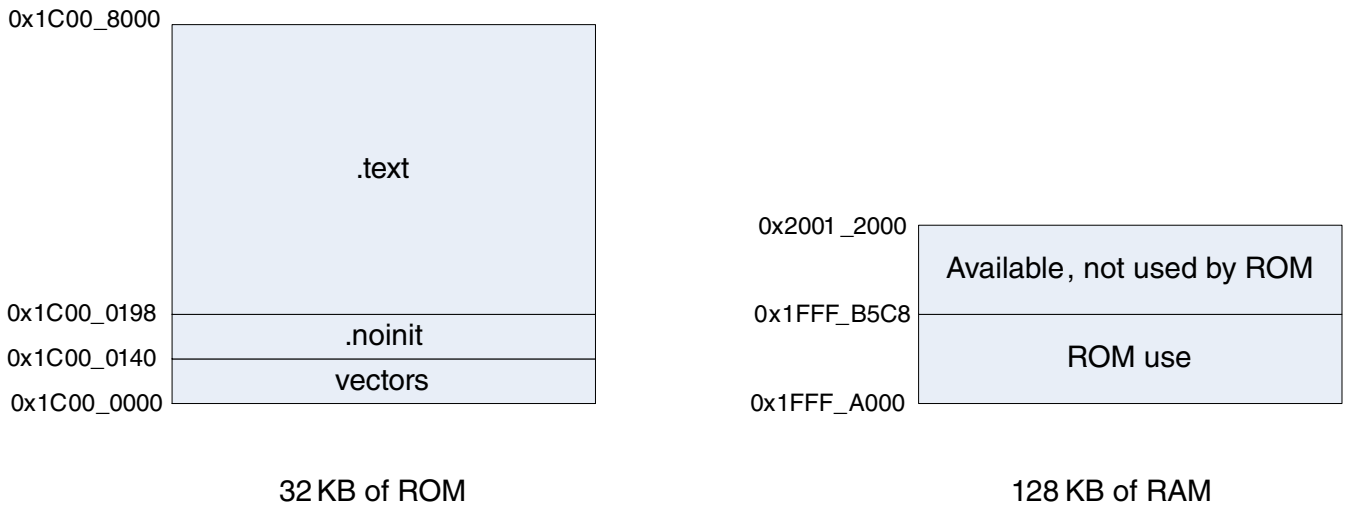


Figure 14-1. Kinetis Bootloader ROM/RAM Memory Maps

14.3.2 The Kinetis Bootloader Configuration Area (BCA)

The Kinetis Bootloader reads data from the Bootloader Configuration Area (BCA) to configure various features of the bootloader. The BCA resides in flash memory at offset 0x3C0, and provides all of the parameters needed to configure the Kinetis Bootloader operation. For uninitialized flash, the Kinetis Bootloader uses a predefined default configuration. A host application can use the Kinetis Bootloader to program the BCA for use during subsequent initializations of the bootloader.

Table 14-3. Configuration Fields for the Kinetis Bootloader

Offset	Size (bytes)	Configuration Field	Description
0x00 - 0x03	4	tag	Magic number to verify bootloader configuration is valid. Must be set to 'kcfg'.
0x04 - 0x07	4	crcStartAddress	Start address for application image CRC check. To generate the CRC, refer to the CRC chapter. If the bits are all set then Kinetis bootloader by default will not perform any CRC check.
0x08 - 0x0B	4	crcByteCount	Byte count for application image CRC check. If the bits are all set then Kinetis bootloader by default will not perform any CRC check.
0x0C - 0x0F	4	crcExpectedValue	Expected CRC value for application CRC check. If the bits are all set then Kinetis bootloader by default will not perform any CRC check.
0x10	1	enabledPeripherals	Bitfield of peripherals to enable.

Table continues on the next page...

Table 14-3. Configuration Fields for the Kinetis Bootloader (continued)

Offset	Size (bytes)	Configuration Field	Description
			bit 0 LPUART bit 1 LPI2C bit 2 LPSPI bit 4 USB Kinetis bootloader will enable the peripheral if corresponding bit is set to 1.
0x11	1	i2cSlaveAddress	If not 0xFF, used as the 7-bit I2C slave address. If 0xFF, defaults to 0x10 for I2C slave address
0x12 - 0x13	2	peripheralDetectionTimeout	Timeout in milliseconds for active peripheral detection. If 0xFFFF, defaults to 5 seconds.
0x14 - 0x15	2	usbVid	Sets the USB Vendor ID reported by the device during enumeration. If 0xFFFF, it defaults to 0x15A2.
0x16 - 0x17	2	usbPid	Sets the USB Product ID reported by the device during enumeration.
0x18 - 0x1B	4	usbStringsPointer	Sets the USB Strings reported by the device during enumeration. Default string values are described in the USB peripheral section.
0x1C	1	clockFlags	See Table 14-5, clockFlags Configuration Field
0x1D	1	clockDivider	Inverted value of the divider to use for core and bus clocks when in high speed mode
0x1F	1	pad byte	N/A
0x20	4	MMCAU configuration pointer	A pointer to the MMCAU configuration structure in memory. See the encryption section for details.
0x24	4	Reserved	-
0x29	7	Reserved	-
0x28	1	qspiPort	Selects the default QSPI port: • If bit 0 is cleared, then the bootloader will choose PORTC as the default QSPI port. • If bit 0 is not cleared, then the default QSPI port is PORTE.
0x30	4	Reserved	-
0x34	12	Reserved	-

NOTE

The flash sector containing the BCA should not be located in the execute-only region, because the Kinetis bootloader cannot read an execute-only region.

The first configuration field 'tag' is a tag value or magic number. The tag value must be set to 'kcfg' for the bootloader configuration data to be recognized as valid. If tag-field verification fails, then the Kinetis Bootloader assumes that the flash is not initialized and uses a predefined default configuration. The tag value is treated as a character string, so bytes 0-3 must be set as shown in the table.

Table 14-4. tag Configuration Field

Offset	tag Byte Value
0	'k' (0x6B)
1	'c' (0x63)
2	'f' (0x66)
3	'g' (0x67)

The flags in the clockFlags configuration field are enabled if the corresponding bit is cleared (0).

Table 14-5. clockFlags Configuration Field

Bit	Flag	Description
0	HighSpeed	Enable high speed mode (i.e., 48 MHz). Read Clock Configuration section for more information on the high speed mode.
1 - 7	Reserved	

14.3.3 Start-up Process

Any of the following conditions will force the hardware to start the Kinetis Bootloader:

- FOPT [7] is set to 1. This forces the ROM to run out of reset.
- The BOOTCFG0 pin is asserted. The pin must be configured as BOOTCFG0 by setting the BOOTPIN_OPT bit of FOPT to 0.
- A user applications running on flash or RAM calls into the Kinetis Bootloader entry point address in ROM, to start Kinetis Bootloader execution.

The FOPT[BOOTSRC_SEL] determines the boot source. The FOPT register is located in the flash configuration field at address 0x40D in the flash memory array. For a complete list of options, see the Boot options section in the Reset and Boot chapter. If FOPT [7] is set to 1, then the device will boot to ROM out of reset. Flash memory defaults to all 1s when erased, so a blank chip will automatically boot to ROM.

The BOOTCFG0 pin is shared with the NMI pin, with NMI being the default usage. Regardless of whether the NMI pin is enabled or not, the NMI functionality is disabled if the ROM is executed out of reset, for as long as the ROM is running.

When the ROM is executed out of reset, vector fetches from the CPU are redirected to the ROM's vector table in ROM memory at offset 0x1C00_0000. This ensures that any exceptions will be handled by the ROM.

After the Kinetis Bootloader has started, the following procedure starts bootloader operations:

1. The RCM_MR [FORCEROM] bits are set, so that the device will reboot back into the ROM if/when the device is reset.
2. Initializes the bootloader's .data and .bss sections.
3. Reads bootloader configuration data from flash at address 0x3C0. The configuration data is only used if the tag field is set to the expected 'kcfg' value. If the tag is incorrect, then the configuration values are set to default, as if the data was all 0xFF bytes.
4. Clocks are configured. See the Clock Configuration section.
5. Enabled peripherals are initialized.
6. The bootloader waits for communication to begin on a peripheral.
   - If detection times out, then the bootloader jumps to the user application in flash. See Bootloader Exit state section.
   - If communication is detected, then all inactive peripherals are shut down, and the command phase is entered.

NOTE

The flash sector containing the vector table should not be located in the execute-only region, because the Kinetis bootloader cannot read the PC and SP addresses in an execute-only region.

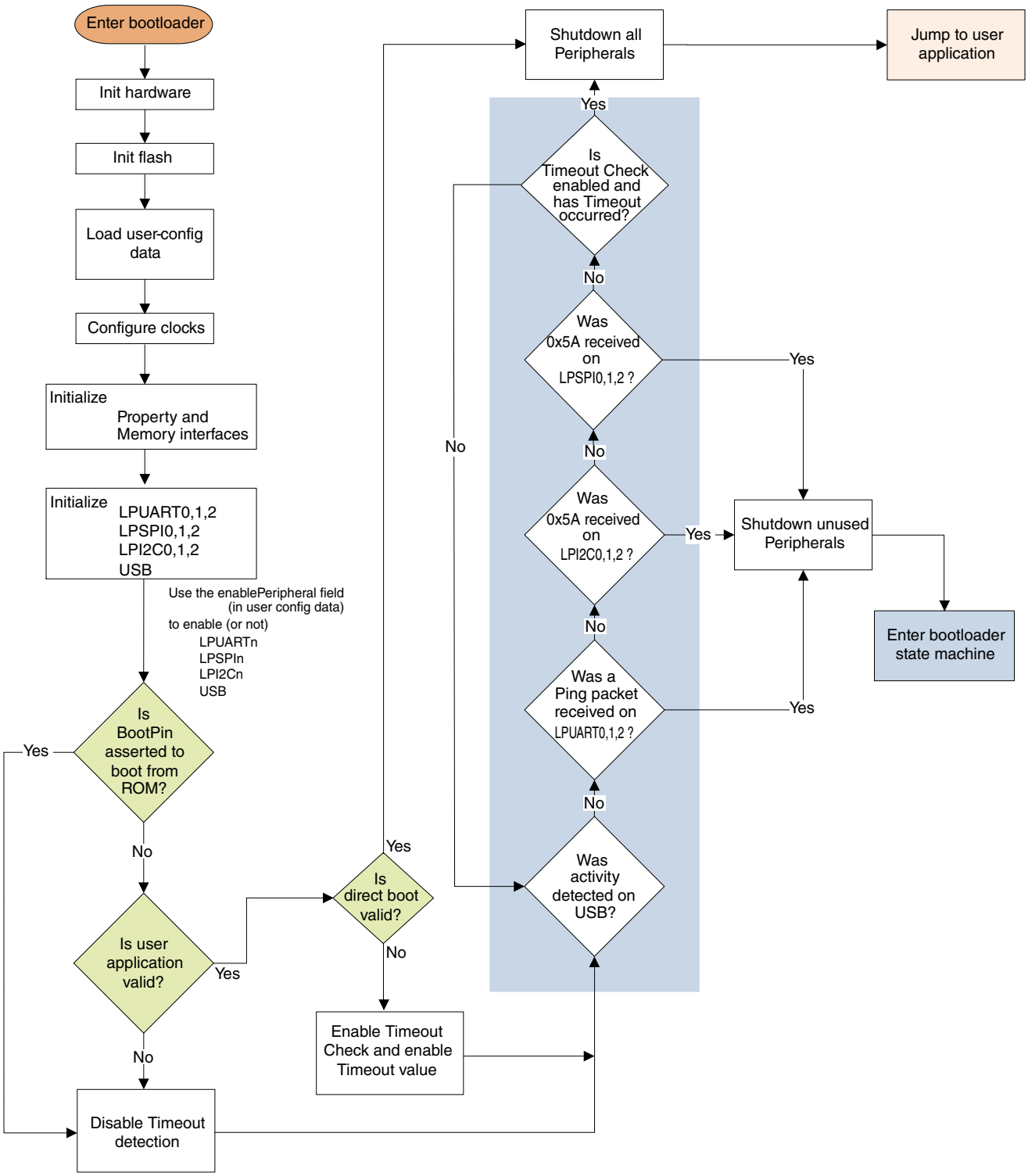


Figure 14-2. Kinetis Bootloader Start-up Flowchart

14.3.4 Clock Configuration

By default, the bootloader does not modify clocks. The Kinetis Bootloader in ROM will use the clock configuration of the chip out of reset unless the clock configuration bits in the FOPT register are cleared, or if a USB peripheral is enabled.

- Alternate clock configurations are supported, by setting fields in the Bootloader Configuration Area (BCA) shown in Table 14-3.
- If the HighSpeed flag of the clockFlags configuration value is cleared, or if a USB peripheral is enabled, the bootloader will enable the internal 48 MHz reference clock.
- In high speed mode, the core and bus clock frequencies are determined by the clockDivider configuration value.
- The core clock divider is set directly from the inverted value of the clockDivider unless a USB peripheral is enabled. If a USB peripheral is enabled and the inverted value of the clockDivider is greater than 2, then the value is reduced to 2, in order to keep the CPU clock above 20 MHz.
- The bus clock divider is set to 1, unless the resulting bus clock frequency would be greater than the maximum supported value. In this case, the bus clock divider is increased until the bus clock frequency is at or below the maximum.
- Note that the maximum baud rate of serial peripherals is related to the core and bus clock frequencies. To achieve the desired baud rates, high speed mode should be enabled in BCA.

14.3.5 Bootloader Entry Point / API Tree

To run the Kinetis Bootloader, a user application simply calls the runBootloader function. To get the address of the entry point, the user application reads the word containing the pointer to the bootloader API tree at offset 0x1C of the bootloader's vector table. The vector table is placed at the base of the bootloader's address range, which for the ROM is 0x1C00_0000; the API tree pointer is at address 0x1C00_001C.

The bootloader API tree is a structure that contains pointers to other structures, which have the function and data addresses for the bootloader. The bootloader entry point is always the 1st word of the API tree.

```
typedef struct BootloaderTree
{
    void (*runBootloader)(void *arg);            //!< Function to start the bootloader
executing.
    standard_version_t version;                  //!< Bootloader version number.
    const char *copyright;                       //!< Copyright string.
    const bootloader_context_t *runtimeContext;  //!< Pointer to the bootloader's runtime
context.
    const flash_driver_interface_t *flashDriver; //!< Flash driver API.
    const aes_driver_interface_t *aesDriver;     //!< AES driver API.
} bootloader_tree_t;
```

The prototype of the entry point is:

```
void run_bootloader(void * arg);
```

The `arg` parameter is currently unused, and is intended for future expansion (for example, passing options to the bootloader). To ensure future compatibility, a value of NULL should be passed for `arg`.

Example: code to get the entry pointer address from the ROM and start the bootloader.

NOTE

This entry must be called in supervisor (privileged) mode.

```
// Variables

uint32_t runBootloaderAddress;

void (*runBootloader)(void * arg);


// Read the function address from the ROM API tree.

runBootloaderAddress = **(uint32_t **)(0x1c00001c);

runBootloader = (void (*)(void * arg))runBootloaderAddress;


// Start the bootloader.

runBootloader(NULL);
```

14.3.6 Bootloader Protocol

This section explains the general protocol for the packet transfers between the host and the Kinetis Bootloader. The description includes the transfer of packets for different transactions, such as commands with no data phase and commands with incoming or outgoing data phase. The next section describes various packet types used in a transaction.

Each command sent from the host is replied to with a response command.

Commands may include an optional data phase:

- If the data phase is **incoming** (from host to bootloader), then the data phase is part of the **original command**.
- If the data phase is **outgoing** (from bootloader to host), then the data phase is part of the **response command**.

NOTE

In all protocols (described in the next subsections), the Ack sent in response to a Command or Data packet can arrive at any time *before, during, or after* the Command/Data packet has processed.

14.3.6.1 Command with no data phase

The protocol for a command with no data phase contains:

- Command packet (from host)
- Generic response command packet (to host)

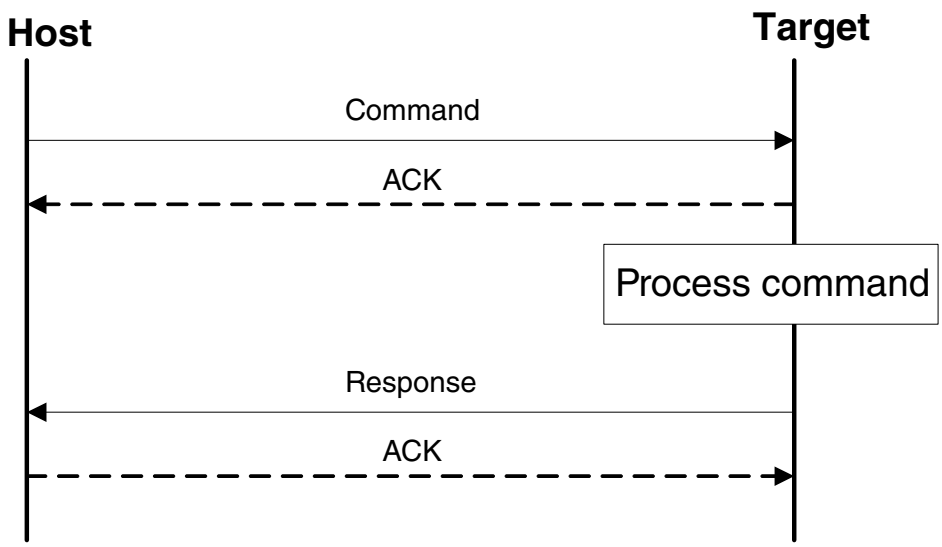


Figure 14-3. Command with No Data Phase

14.3.6.2 Command with incoming data phase

The protocol for a command with an incoming data phase contains:

- Command packet (from host)
- Generic response command packet (to host)
- Incoming data packets (from host)
- Generic response command packet (to host)

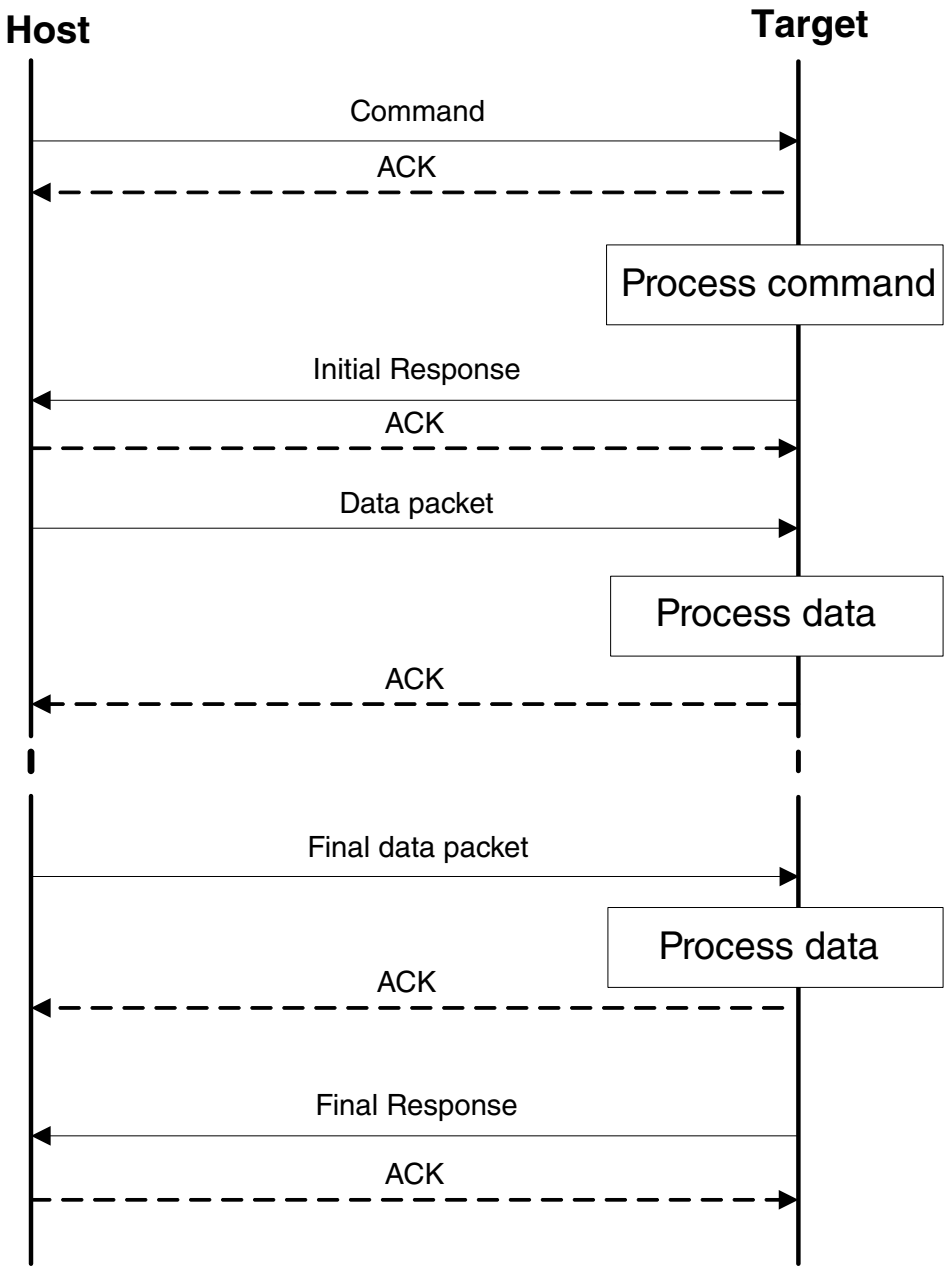


Figure 14-4. Command with incoming data phase

NOTE

- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the Generic Response packet prior to the start of the data phase does not have a status of kStatus_Success, then the data phase is aborted.
- Data phases may be aborted by the receiving side by sending the final Generic Response early with a status of

kStatus_AbortDataPhase. The host may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

14.3.6.3 Command with outgoing data phase

The protocol for a command with an outgoing data phase contains:
- Command packet (from host)
- ReadMemory Response command packet (to host) (kCommandFlag_HasDataPhase set)
- Outgoing data packets (to host)
- Generic response command packet (to host)

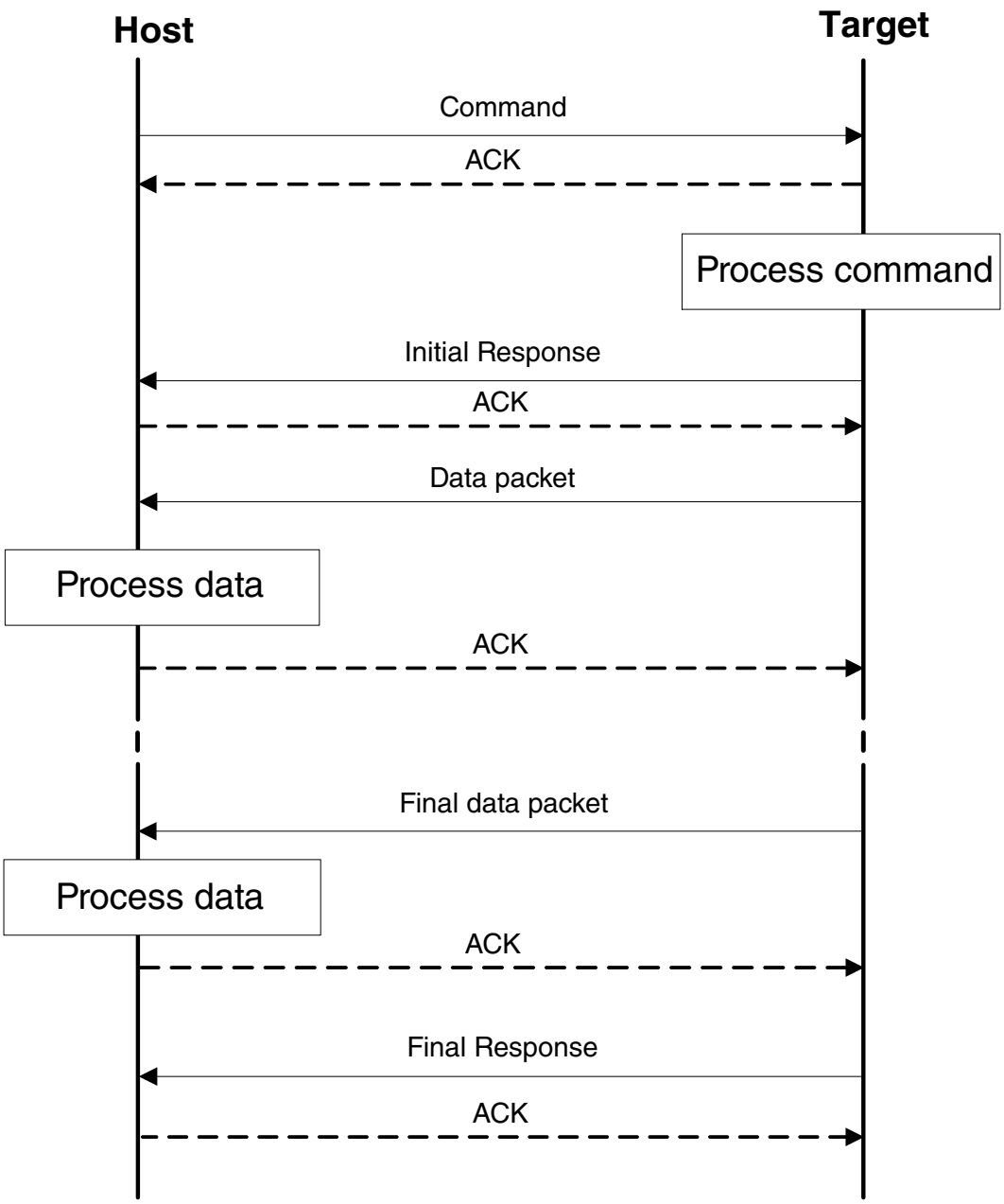


Figure 14-5. Command with outgoing data phase

NOTE

- For the outgoing data phase sequence above, the data phase is really considered part of the response command.
- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the ReadMemory Response command packet prior to the start of the data phase does not contain the kCommandFlag_HasDataPhase flag, then the data phase is aborted.

- Data phases may be aborted by the host sending the final Generic Response early with a status of kStatus_AbortDataPhase. The sending side may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

14.3.7 Bootloader Packet Types

The Kinetis Bootloader device works in slave mode. All data communication is initiated by a host, which is either a PC or an embedded host . The Kinetis Bootloader device is the target, which receives a command or data packet. All data communication between host and target is packetized.

NOTE

The term "target" refers to the "Kinetis Bootloader device."

There are 6 types of packets used in the device:

- Ping packet
- Ping Response packet
- Framing packet
- Command packet
- Data packet
- Response packet

All fields in the packets are in little-endian byte order.

14.3.7.1 Ping packet

The Ping packet is the first packet sent from a host to the target (Kinetis Bootloader), to establish a connection on a selected peripheral. For a LPUART peripheral, the Ping packet is used to determine the baudrate. A Ping packet must be sent before any other communications. In response to a Ping packet, the target sends a Ping Response packet.

Table 14-6. Ping Packet Format

Byte #	Value	Name
0	0x5A	start byte
1	0xA6	ping

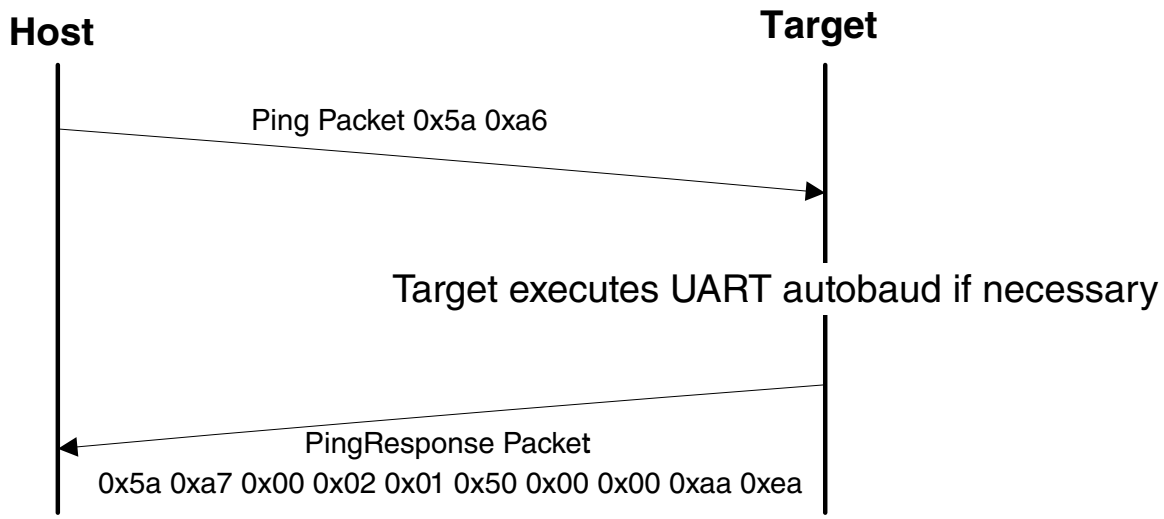


Figure 14-6. Ping Packet Protocol Sequence

14.3.7.2 Ping Response Packet

The target (Kinetis Bootloader) sends a Ping Response packet back to the host after receiving a Ping packet. If communication is over a LPUART peripheral, the target uses the incoming Ping packet to determine the baud rate before replying with the Ping Response packet. Once the Ping Response packet is received by the host, the connection is established, and the host starts sending commands to the target (Kinetis Bootloader).

Table 14-7. Ping Response Packet Format

Byte #	Value	Parameter
0	0x5A	start byte
1	0xA7	Ping response code
2		Protocol bugfix
3		Protocol minor
4		Protocol major
5		Protocol name = 'P' (0x50)
6		Options low
7		Options high
8		CRC16 low
9		CRC16 high

14.3.7.3 Framing Packet

The framing packet is used for flow control and error detection, and it (the framing packet) wraps command and data packets as well.

The framing packet described in this section is used for serial peripherals including LPUART, LPI2C and LPSPI. The USB HID peripheral does not use framing packets. Instead, the packetization inherent in the USB protocol itself is used. Please refer to the USB peripheral section for details.

Table 14-8. Framing Packet Format

Byte #	Value	Parameter	
0	0x5A	start byte	
1		packetType	
2		length_low	Length is a 16-bit field that specifies the entire command or data packet size in bytes.
3		length_high	
4		crc16_low	This is a 16-bit field. The CRC16 value covers entire framing packet, including the start byte and command or data packets, but does not include the CRC bytes. See the CRC16 algorithm after this table.
5		crc16_high	
6 . . .n		Command or Data packet payload	

A special framing packet that contains only a start byte and a packet type is used for synchronization between the host and target.

Table 14-9. Special Framing Packet Format

Byte #	Value	Parameter
0	0x5A	start byte
1	0xA*n*	packetType

The Packet Type field specifies the type of the packet from one of the defined types (below):

Table 14-10. packetType Field

packetType	Name	Description
0xA1	kFramingPacketType_Ack	The previous packet was received successfully; the sending of more packets is allowed.
0xA2	kFramingPacketType_Nak	The previous packet was corrupted and must be re-sent.
0xA3	kFramingPacketType_AckAbort	Data phase is being aborted.
0xA4	kFramingPacketType_Command	The framing packet contains a command packet payload.
0xA5	kFramingPacketType_Data	The framing packet contains a data packet payload.

Table continues on the next page...

Table 14-10. packetType Field (continued)

packetType	Name	Description
0xA6	kFramingPacketType_Ping	Sent to verify the other side is alive. Also used for UART autobaud.
0xA7	kFramingPacketType_PingResponse	A response to Ping; contains the framing protocol version number and options.

CRC16 algorithm:

```
uint16_t crc16_update(const uint8_t * src, uint32_t lengthInBytes)
{
    uint32_t crc = 0;
    uint32_t j;
    for (j=0; j < lengthInBytes; ++j)
    {
        uint32_t i;
        uint32_t byte = src[j];
        crc ^= byte << 8;
        for (i = 0; i < 8; ++i)
        {
            uint32_t temp = crc << 1;
            if (crc & 0x8000)
            {
                temp ^= 0x1021;
            }
            crc = temp;
        }
    }
    return crc;
}
```

14.3.7.4 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 14-11. Command Packet Format

Command Packet Format (32 bytes)										
Command Header (4 bytes)				28 bytes for Parameters (Max 7 parameters)						
Tag	Flags	Rsvd	Param Count	Param1 (32-bit)	Param2 (32-bit)	Param3 (32-bit)	Param4 (32-bit)	Param5 (32-bit)	Param6 (32-bit)	Param7 (32-bit)
byte 0	byte 1	byte 2	byte 3							

Table 14-12. Command Header Format

Byte #	Command Header Field	
0	Command or Response tag	The command header is 4 bytes long, with these fields.
1	Flags	
2	Reserved. Should be 0x00.	
3	ParameterCount	

The header is followed by 32-bit parameters up to the value of the ParameterCount field specified in the header. Because a command packet is 32 bytes long, only 7 parameters can fit into the command packet.

Command packets are also used by the target to send responses back to the host. As mentioned earlier, command packets and data packets are embedded into framing packets for all of the transfers.

Table 14-13. Commands that are supported

Command	Name
0x01	FlashEraseAll
0x02	FlashEraseRegion
0x03	ReadMemory
0x04	WriteMemory
0x05	FillMemory
0x06	FlashSecurityDisable
0x07	GetProperty
0x08	ReceiveSBFile
0x09	Execute
0x0A	Call
0x0B	Reset
0x0C	SetProperty
0x0D	FlashEraseAllUnsecure

Table continues on the next page...

Table 14-13. Commands that are supported (continued)

Command	Name
0x0E	FlashProgramOnce
0x0F	FlashReadOnce
0x10	FlashReadResource
0x11	Reserved

Table 14-14. Responses that are supported

Response	Name
0xA0	GenericResponse
0xA3	ReadMemoryResponse (used for sending responses to ReadMemory command only)
0xA7	GetPropertyResponse (used for sending responses to GetProperty command only)
0xAF	FlashReadOnceResponse (used for sending responses to FlashReadOnce command only)
0xB0	FlashReadResourceResponse (used for sending responses to FlashReadResource command only)

Flags: Each command packet contains a Flag byte. Only bit 0 of the flag byte is used. If bit 0 of the flag byte is set to 1, then data packets will follow in the command sequence. The number of bytes that will be transferred in the data phase is determined by a command-specific parameter in the parameters array.

ParameterCount: The number of parameters included in the command packet.

Parameters: The parameters are word-length (32 bits). With the default maximum packet size of 32 bytes, a command packet can contain up to 7 parameters.

14.3.7.5 Data packet

The data packet carries just the data, either host sending data to target, or target sending data to host. The data transfer direction is determined by the last command sent from the host. The data packet is also wrapped within a framing packet, to ensure the correct packet data is received.

The contents of a data packet are simply the data itself. There are no other fields, so that the most data per packet can be transferred. Framing packets are responsible for ensuring that the correct packet data is received.

14.3.7.6 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses include:

- GenericResponse
- GetPropertyResponse
- ReadMemoryResponse
- FlashReadOnceResponse
- FlashReadResourceResponse

GenericResponse: After the Kinetis Bootloader has processed a command, the bootloader will send a generic response with status and command tag information to the host. The generic response is the last packet in the command protocol sequence. The generic response packet contains the framing packet data and the command packet data (with generic response tag = 0xA0) and a list of parameters (defined in the next section). The parameter count field in the header is always set to 2, for status code and command tag parameters.

Table 14-15. GenericResponse Parameters

Byte #	Parameter	Descripton
0 - 3	Status code	The Status codes are errors encountered during the execution of a command by the target (Kinetis Bootloader). If a command succeeds, then a kStatus_Success code is returned. Table 14-55, Kinetis Bootloader Status Error Codes, lists the status codes returned to the host by the Kinetis Bootloader for ROM.
4 - 7	Command tag	The Command tag parameter identifies the response to the command sent by the host.

GetPropertyResponse: The GetPropertyResponse packet is sent by the target in response to the host query that uses the GetProperty command. The GetPropertyResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a GetPropertyResponse tag value (0xA7).

The parameter count field in the header is set to greater than 1, to always include the status code and one or many property values.

Table 14-16. GetPropertyResponse Parameters

Byte #	Value	Parameter
0 - 3		Status code
4 - 7		Property value

Table continues on the next page...

Table 14-16. GetPropertyResponse Parameters (continued)

Byte #	Value	Parameter
. . .		. . .
		Can be up to maximum 6 property values, limited to the size of the 32-bit command packet and property type.

ReadMemoryResponse: The ReadMemoryResponse packet is sent by the target in response to the host sending a ReadMemory command. The ReadMemoryResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a ReadMemoryResponse tag value (0xA3), the flags field set to kCommandFlag_HasDataPhase (1).

The parameter count set to 2 for the status code and the data byte count parameters shown below.

Table 14-17. ReadMemoryResponse Parameters

Byte #	Parameter	Descripton
0 - 3	Status code	The status of the associated Read Memory command.
4 - 7	Data byte count	The number of bytes sent in the data phase.

FlashReadOnceResponse: The FlashReadOnceResponse packet is sent by the target in response to the host sending a FlashReadOnce command. The FlashReadOnceResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a FlashReadOnceResponse tag value (0xAF), and the flags field set to 0. The parameter count is set to 2 plus *the number of words* requested to be read in the FlashReadOnceCommand.

Table 14-18. FlashReadOnceResponse Parameters

Byte #	Value	Parameter
0 – 3		Status Code
4 – 7		Byte count to read
…		…
		Can be up to 20 bytes of requested read data.

The FlashReadResourceResponse packet is sent by the target in response to the host sending a FlashReadResource command. The FlashReadResourceResponse packet contains the framing packet data and command packet data, with the command/response tag set to a FlashReadResourceResponse tag value (0xB0), and the flags field set to kCommandFlag_HasDataPhase (1).

Table 14-19. FlashReadResourceResponse Parameters

Byte #	Value	Parameter
0 – 3		Status Code
4 – 7		Data byte count

14.3.8 Bootloader Command API

All Kinetis Bootloader command APIs follow the command packet format that is wrapped by the framing packet, as explained in previous sections.

- For a list of commands supported by the Kinetis Bootloader in ROM, see Table 14-2, Commands supported.
- For a list of status codes returned by the Kinetis Bootloader in ROM, see Table 14-55, Kinetis Bootloader Status Error Codes.

NOTE

All the examples in this section depict byte traffic on serial peripherals that use framing packets. USB HID transactions use the USB HID report packets instead of the serial framing packets shown in this section. Please refer to the HID reports section for details of the USB HID packet structure.

14.3.8.1 Call command

The Call command will execute a function that is written in memory at the address sent in the command. The address needs to be a valid memory location residing in accessible flash (internal or external) or in RAM. The command supports the passing of one 32-bit argument. Although the command supports a stack address, at this time the call will still take place using the current stack pointer. After execution of the function, a 32-bit return value will be returned in the generic response message.

Table 14-20. Parameters for Call Command

Byte #	Command
0 - 3	Call address
4 - 7	Argument word
8 - 11	Stack pointer

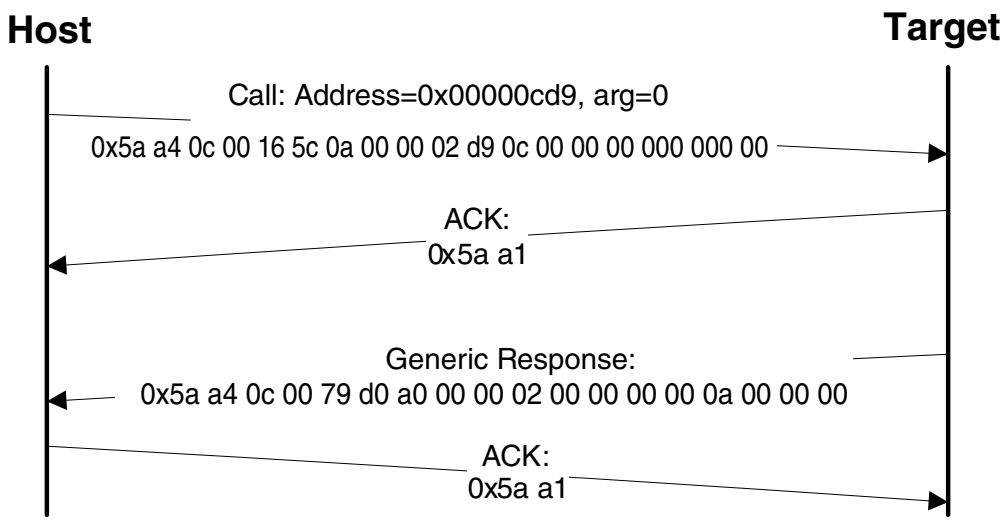


Figure 14-7. Protocol Sequence for Call Command

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code either set to the return value of the function called or set to kStatus_InvalidArgument (105).

14.3.8.2 ReceiveSBFile command

The ReceiveSBFile command (ReceiveSbFile) will start the transfer of an SB file to the target. The command only specifies the size in bytes of the SB file that will be sent in the data phase. The SB file will be processed as it is received by the Bootloader .

Table 14-21. Parameters for ReceiveSBFile Command

Byte #	Command
0 - 3	Byte count

Data Phase: The Receive SB file command has a data phase; the host will send data packets until the number of bytes of data specified in the byteCount parameter of the Receive SB File command are received by the target.

Response: The target (Bootloader) will return a GenericResponse packet with a status code set to the kStatus_Success upon successful execution of the command, or set to an appropriate error code.

14.3.8.3 Execute command

The execute command results in the bootloader setting the program counter to the code at the provided jump address, R0 to the provided argument, and a Stack pointer to the provided stack pointer address. Prior to the jump, the system is returned to the reset state.

The Jump address, function argument pointer, and stack pointer are the parameters required for the Execute command.

Table 14-22. Parameters for Execute Command

Byte #	Command
0 - 3	Jump address
4 - 7	Argument word
8 - 11	Stack pointer address

The Execute command has no data phase.

Response: Before executing the Execute command, the target (Kinetis Bootloader) will validate the parameters and return a GenericResponse packet with a status code either set to kStatus_Success or an appropriate error status code.

14.3.8.4 Reset command

The Reset command will result in bootloader resetting the chip.

The Reset command requires no parameters.

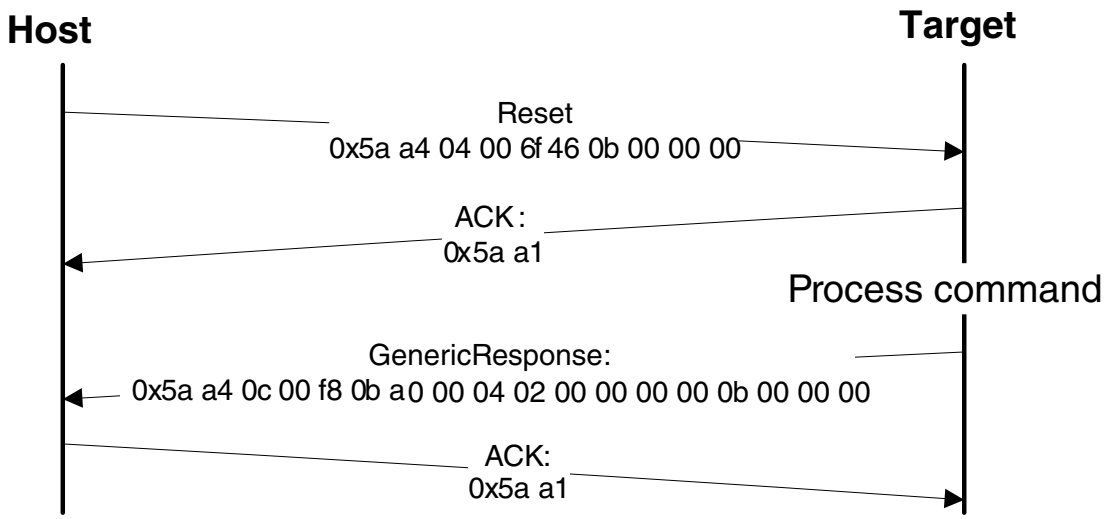


Figure 14-8. Protocol Sequence for Reset Command

Table 14-23. Reset Command Packet Format (Example)

Reset	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0x6F 0x46
Command packet	commandTag	0x0B - reset
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The Reset command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code set to kStatus_Success, before resetting the chip.

14.3.8.5 GetProperty command

The GetProperty command is used to query the bootloader about various properties and settings. Each supported property has a unique 32-bit tag associated with it. The tag occupies the first parameter of the command packet. The target returns a GetPropertyResponse packet with the property values for the property identified with the tag in the GetProperty command.

Properties are the defined units of data that can be accessed with the GetProperty or SetProperty commands. Properties may be read-only or read-write. All read-write properties are 32-bit integers, so they can easily be carried in a command parameter.

For a list of properties and their associated 32-bit property tags supported by the Kinetis Bootloader in ROM, see Table 14-50.

The 32-bit property tag is the only parameter required for GetProperty command.

Table 14-24. Parameters for GetProperty Command

Byte #	Command
0 - 3	Property tag

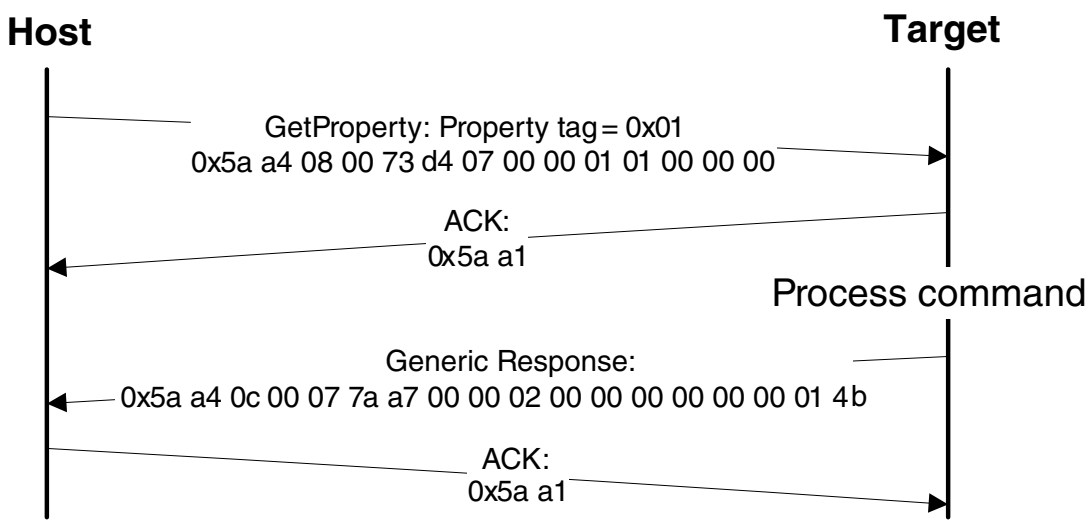


Figure 14-9. Protocol Sequence for GetProperty Command

Table 14-25. GetProperty Command Packet Format (Example)

GetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x08 0x00
	crc16	0x73 0xD4
Command packet	commandTag	0x07 – GetProperty
	flags	0x00
	reserved	0x00
	parameterCount	0x01
	propertyTag	0x00000001 - CurrentVersion

The GetProperty command has no data phase.

Response: In response to a GetProperty command, the target will send a GetPropertyResponse packet with the response tag set to 0xA7. The parameter count indicates the number of parameters sent for the property values, with the first parameter showing status code 0, followed by the property value(s). The next table shows an example of a GetPropertyResponse packet.

Table 14-26. GetProperty Response Packet Format (Example)

GetPropertyResponse	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0c 0x00 (12 bytes)
	crc16	0x07 0x7a
Command packet	responseTag	0xA7
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	status	0x00000000
	propertyValue	0x0000014b - CurrentVersion

14.3.8.6 SetProperty command

The SetProperty command is used to change or alter the values of the properties or options in the Kinetis Bootloader ROM. However, the SetProperty command can only change the value of properties that are writable—see Table 14-50, Properties used by Get/SetProperty Commands. If you try to set a value for a read-only property, then the Kinetis Bootloader will return an error.

The property tag and the new value to set are the 2 parameters required for the SetProperty command.

Table 14-27. Parameters for SetProperty Command

Byte #	Command
0 - 3	Property tag
4 - 7	Property value

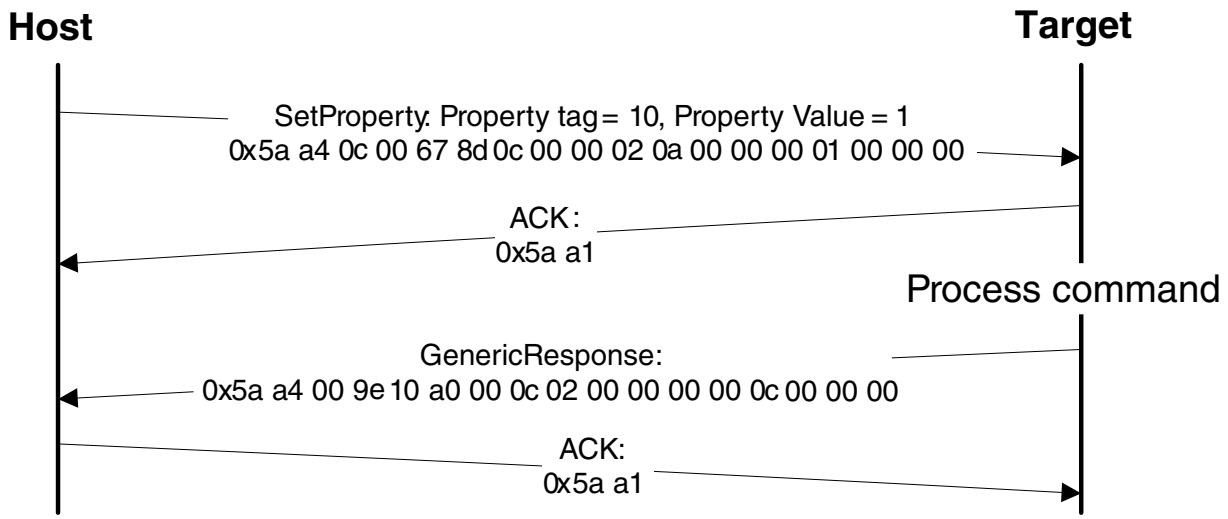


Figure 14-10. Protocol Sequence for SetProperty Command

Table 14-28. SetProperty Command Packet Format (Example)

SetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x67 0x8D
Command packet	commandTag	0x0C – SetProperty with property tag 10
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	propertyTag	0x0000000A - VerifyWrites
	propertyValue	0x00000001

The SetProperty command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with one of following status codes:

Table 14-29. SetProperty Response Status Codes

Status Code
kStatus_Success
kStatus_ReadOnly
kStatus_UnknownProperty
kStatus_InvalidArgument

14.3.8.7 FlashEraseAll command

The FlashEraseAll command performs an erase of the entire flash memory. If any flash regions are protected, then the FlashEraseAll command will fail and return an error status code. Executing the FlashEraseAll command will release flash security if it (flash security) was enabled, by setting the FTFA_FSEC register. However, the FSEC field of the flash configuration field is erased, so unless it is reprogrammed, the flash security will be re-enabled after the next system reset. The Command tag for FlashEraseAll command is 0x01 set in the commandTag field of the command packet.

The FlashEraseAll command requires no parameters.

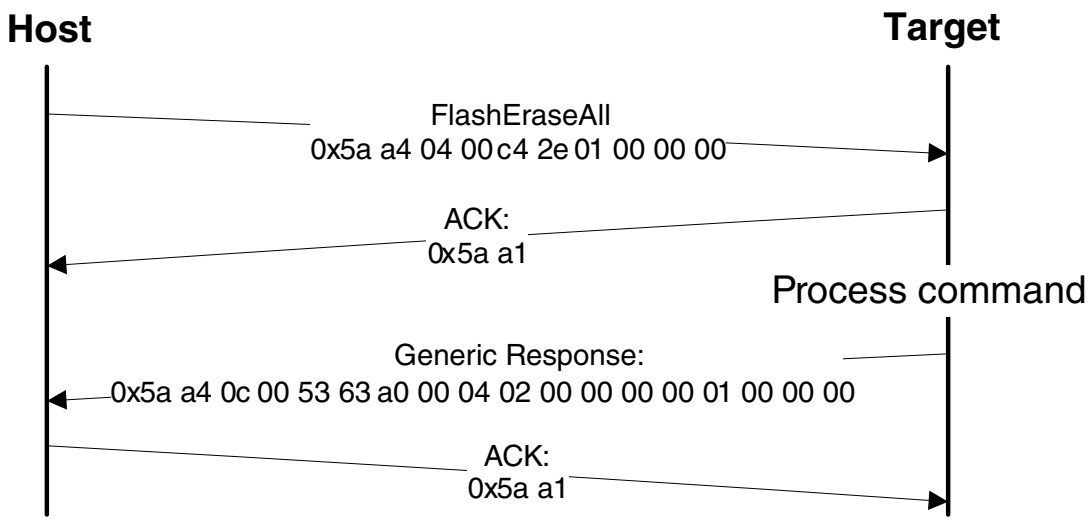


Figure 14-11. Protocol Sequence for FlashEraseAll Command

Table 14-30. FlashEraseAll Command Packet Format (Example)

FlashEraseAll	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0xC4 0x2E
Command packet	commandTag	0x01 - FlashEraseAll
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The FlashEraseAll command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

14.3.8.8 FlashEraseRegion command

The FlashEraseRegion command performs an erase of one or more sectors of the flash memory or a specified range of flash within the connected SPI flash devices.

The start address and number of bytes are the 2 parameters required for the FlashEraseRegion command. The start and byte count parameters must be , or the FlashEraseRegion command will fail and return kStatus_FlashAlignmentError (0x101). If the region specified does not fit in the flash memory space, the FlashEraseRegion command will fail and return kStatus_FlashAddressError (0x102). If any part of the region specified is protected, the FlashEraseRegion command will fail and return kStatus_MemoryRangeInvalid (0x10200).

Table 14-31. Parameters for FlashEraseRegion Command

Byte #	Parameter
0 - 3	Start address
4 - 7	Byte count

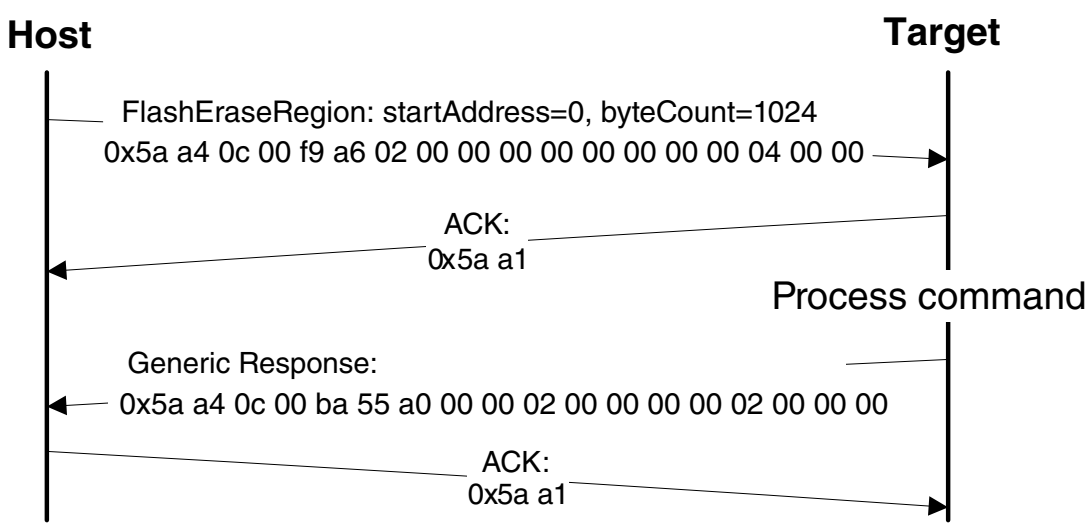


Figure 14-12. Protocol Sequence for FlashEraseRegion Command

Table 14-32. FlashEraseRegion Command Packet Format (Example)

FlashEraseRegion	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0xF9 0x A6
Command packet	commandTag	0x02, kCommandTag_FlashEraseRegion
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	startAddress	0x00 0x00 0x00 0x00 (0x0000_0000)
	byte count	0x00 0x04 0x00 0x00 (0x400)

The FlashEraseRegion command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with one of following error status codes.

Table 14-33. FlashEraseRegion Response Status Codes

Status Code
kStatus_Success (0x0)
kStatus_MemoryRangeInvalid (0x10200)
kStatus_FlashAlignmentError (0x101)
kStatus_FlashAddressError (0x102)
kStatus_FlashAccessError (0x103)
kStatus_FlashProtectionViolation (0x104)
kStatus_FlashCommandFailure (0x105)

14.3.8.9 FlashEraseAllUnsecure command

The FlashEraseAllUnsecure command performs a mass erase of the flash memory, including protected sectors. Flash security is immediately disabled if it (flash security) was enabled, and the FSEC byte in the flash configuration field at address 0x40C is programmed to 0xFE. However, if the mass erase enable option in the FSEC field is disabled, then the FlashEraseAllUnsecure command will fail.

The FlashEraseAllUnsecure command requires no parameters.

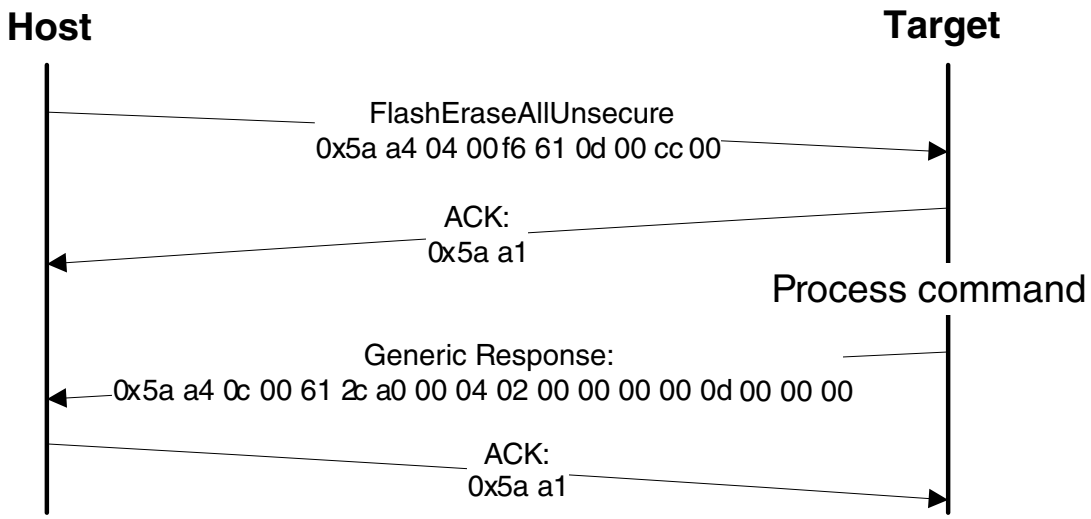


Figure 14-13. Protocol Sequence for FlashEraseAll Command

Table 14-34. FlashEraseAllUnsecure Command Packet Format (Example)

FlashEraseAllUnsecure	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0xF6 0x61
Command packet	commandTag	0x0D - FlashEraseAllUnsecure
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The FlashEraseAllUnsecure command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

14.3.8.10 FlashProgramOnce command

The FlashProgramOnce command writes data (that is provided in a command packet) to a specified range of bytes in the program once field. Special care must be taken when writing to the program once field.

- The program once field only supports programming once, so any attempted to reprogram a program once field will get an error response.
- Writing to the program once field requires the byte count to be 4-byte aligned or 8-byte aligned.

The FlashProgramOnce command uses 3 parameters: index, byteCount, data.

Table 14-35. Parameters for FlashProgramOnce Command

Byte #	Command
0 - 3	Index of program once field
4 - 7	Byte count (must be evenly divisible by 4)
8 - 11	Data
12 - 16	Data

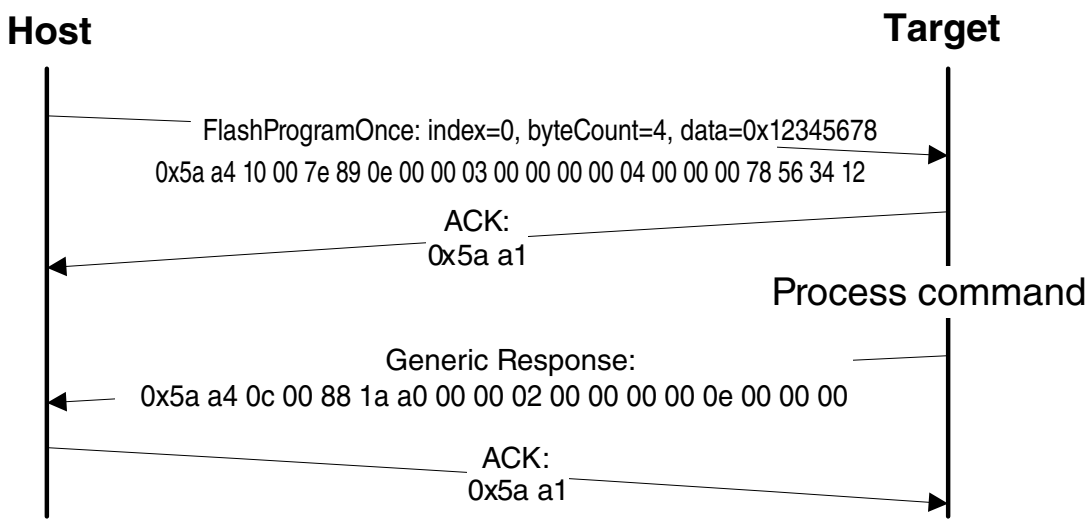


Figure 14-14. Protocol Sequence for FlashProgramOnce Command

Table 14-36. FlashProgramOnce Command Packet Format (Example)

FlashProgramOnce	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0x7E4 0x89
Command packet	commandTag	0x0E – FlashProgramOnce
	flags	0
	reserved	0
	parameterCount	3
	index	0x0000_0000

Table continues on the next page...

Table 14-36. FlashProgramOnce Command Packet Format (Example) (continued)

FlashProgramOnce	Parameter	Value
	byteCount	0x0000_0004
	data	0x1234_5678

Response: upon successful execution of the command, the target (Kinetis Bootloader) will return a GenericResponse packet with a status code set to kStatus_Success, or to an appropriate error status code.

14.3.8.11 FlashReadOnce command

The FlashReadOnce command returns the contents of the program once field by given index and byte count. The FlashReadOnce command uses 2 parameters: index and byteCount.

Table 14-37. Parameters for FlashReadOnce Command

Byte #	Parameter	Description
0 - 3	index	Index of the program once field (to read from)
4 - 7	byteCount	Number of bytes to read and return to the caller

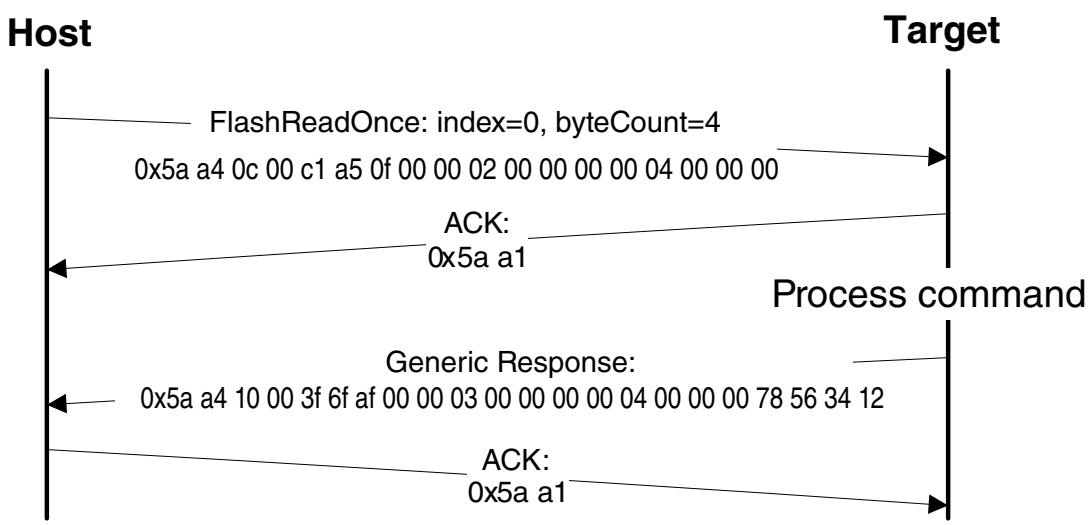


Figure 14-15. Protocol Sequence for FlashReadOnce Command

Table 14-38. FlashReadOnce Command Packet Format (Example)

FlashReadOnce	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4
	length	0x0C 0x00
	crc	0xC1 0xA5
Command packet	commandTag	0x0F – FlashReadOnce
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	index	0x0000_0000
	byteCount	0x0000_0004

Table 14-39. FlashReadOnce Response Format (Example)

FlashReadOnce Response	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4
	length	0x10 0x00
	crc	0x3F 0x6F
Command packet	commandTag	0xAF
	flags	0x00
	reserved	0x00
	parameterCount	0x03
	status	0x0000_0000
	byteCount	0x0000_0004
	data	0x1234_5678

Response: upon successful execution of the command, the target (Kinetis Bootloader) will return a FlashReadOnceResponse packet with a status code set to kStatus_Success, a byte count and corresponding data read from Program Once Field upon successful execution of the command, or will return with a status code set to an appropriate error status code and a byte count set to 0.

14.3.8.12 FlashReadResource command

The FlashReadResource command returns the contents of the IFR field or Flash firmware ID, by given offset, byte count, and option. The FlashReadResource command uses 3 parameters: start address, byteCount, option.

Table 14-40. Parameters for FlashReadResource Command

Byte #	Parameter	Command
0 - 3	start address	Start address of specific non-volatile memory to be read
4 - 7	byteCount	Byte count to be read
8 - 11	option	0: IFR 1: Flash firmware ID

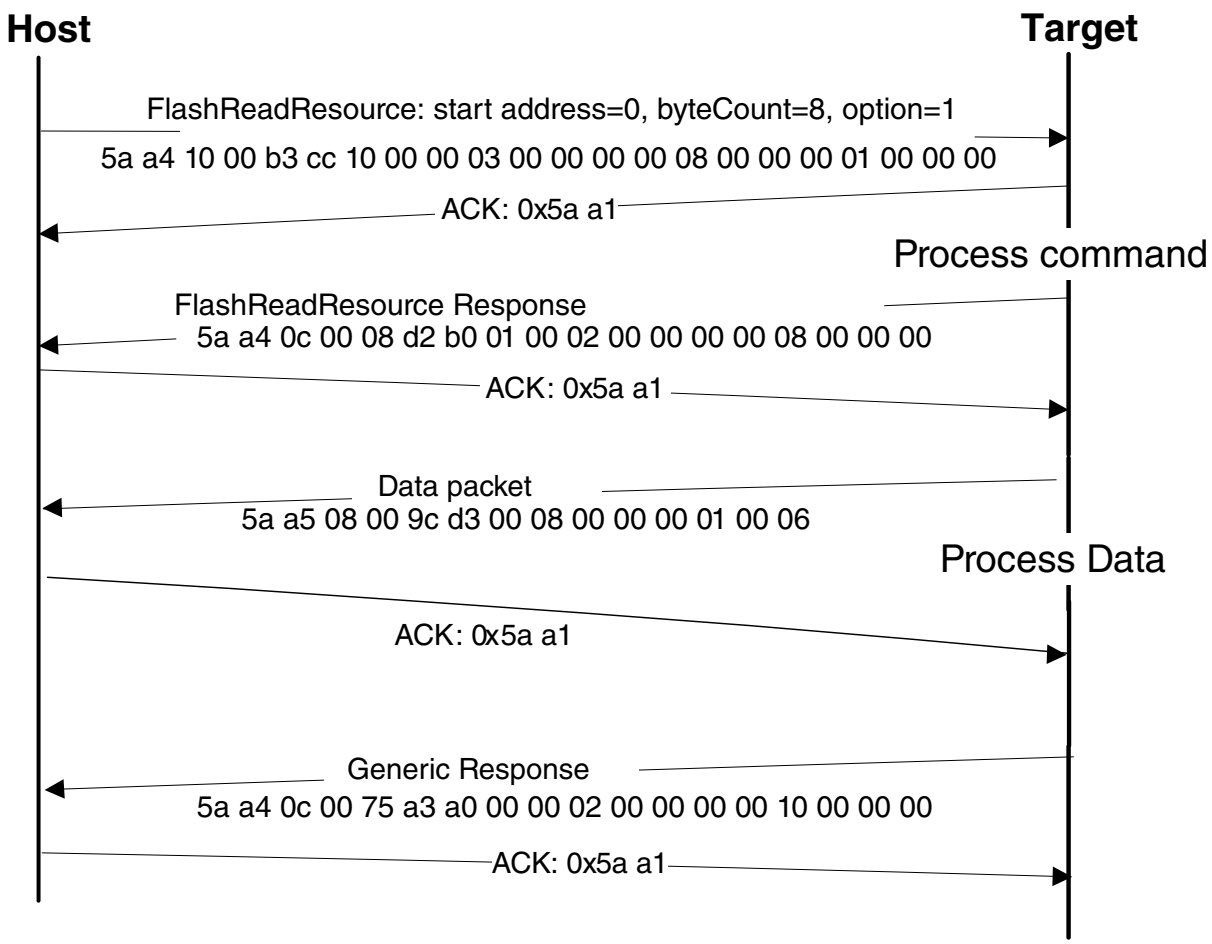


Figure 14-16. Protocol Sequence for FlashReadResource Command

Table 14-41. FlashReadResource Command Packet Format (Example)

FlashReadResource	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4
	length	0x10 0x00
	crc	0xB3 0xCC
Command packet	commandTag	0x10 – FlashReadResource
	flags	0x00

Table continues on the next page...

Table 14-41. FlashReadResource Command Packet Format (Example) (continued)

FlashReadResource	Parameter	Value
	reserved	0x00
	parameterCount	0x03
	startAddress	0x0000_0000
	byteCount	0x0000_0008
	option	0x0000_0001

Table 14-42. FlashReadResource Response Format (Example)

FlashReadResource Response	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4
	length	0x0C 0x00
	crc	0xD2 0xB0
Command packet	commandTag	0xB0
	flags	0x01
	reserved	0x00
	parameterCount	0x02
	status	0x0000_0000
	byteCount	0x0000_0008

Data phase: The FlashReadResource command has a data phase. Because the target (Kinetis Bootloader) works in slave mode, the host must pull data packets until the number of bytes of data *specified in the byteCount parameter of FlashReadResource command* are received by the host.

14.3.8.13 FlashSecurityDisable command

The FlashSecurityDisable command performs the flash security disable operation, by comparing the 8-byte backdoor key (provided in the command) against the backdoor key stored in the flash configuration field (at address 0x400 in the flash).

The backdoor low and high words are the only parameters required for FlashSecurityDisable command.

Table 14-43. Parameters for FlashSecurityDisable Command

Byte #	Command
0 - 3	Backdoor key low word

Table continues on the next page...

Table 14-43. Parameters for FlashSecurityDisable Command (continued)

Byte #	Command
4 - 7	Backdoor key high word

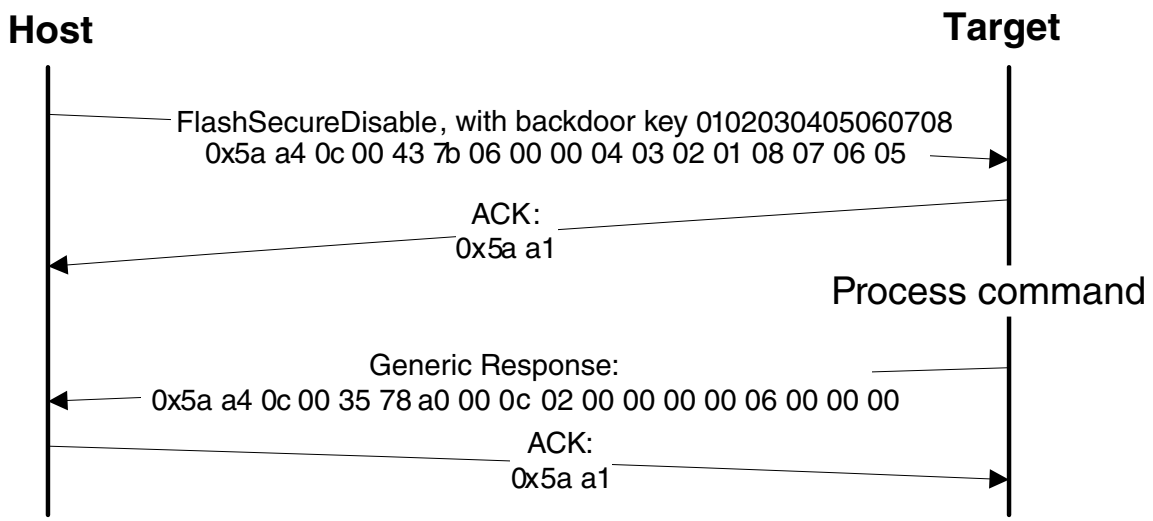


Figure 14-17. Protocol Sequence for FlashSecurityDisable Command

Table 14-44. FlashSecurityDisable Command Packet Format (Example)

FlashSecurityDisable	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x43 0x7B
Command packet	commandTag	0x06 - FlashSecurityDisable
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	Backdoorkey_low	0x04 0x03 0x02 0x01
	Backdoorkey_high	0x08 0x07 0x06 0x05

The FlashSecurityDisable command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

14.3.8.14 FillMemory command

The FillMemory command fills a range of bytes in memory with a data pattern. It follows the same rules as the WriteMemory command. The difference between FillMemory and WriteMemory is that a data pattern is included in FillMemory command parameter, and there is no data phase for the FillMemory command, while WriteMemory does have a data phase.

Table 14-45. Parameters for FillMemory Command

Byte #	Command
0 - 3	Start address of memory to fill
4 - 7	Number of bytes to write with the pattern • The start address should be 32-bit aligned. • The number of bytes must be evenly divisible by 4.
8 - 11	32-bit pattern

- To fill with a byte pattern (8-bit), the byte must be replicated 4 times in the 32-bit pattern.
- To fill with a short pattern (16-bit), the short value must be replicated 2 times in the 32-bit pattern.

For example, to fill a byte value with 0xFE, the word pattern would be 0xFEFEFEFE; to fill a short value 0x5AFE, the word pattern would be 0x5AFE5AFE.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll, FlashEraseRegion, or FlashEraseAllUnsecure command.
- Writing to flash requires the start address to be .
- If the VerifyWrites property is set to true, then writes to flash will also perform a flash verify program operation.

When writing to RAM, the start address need not be aligned, and the data will not be padded.

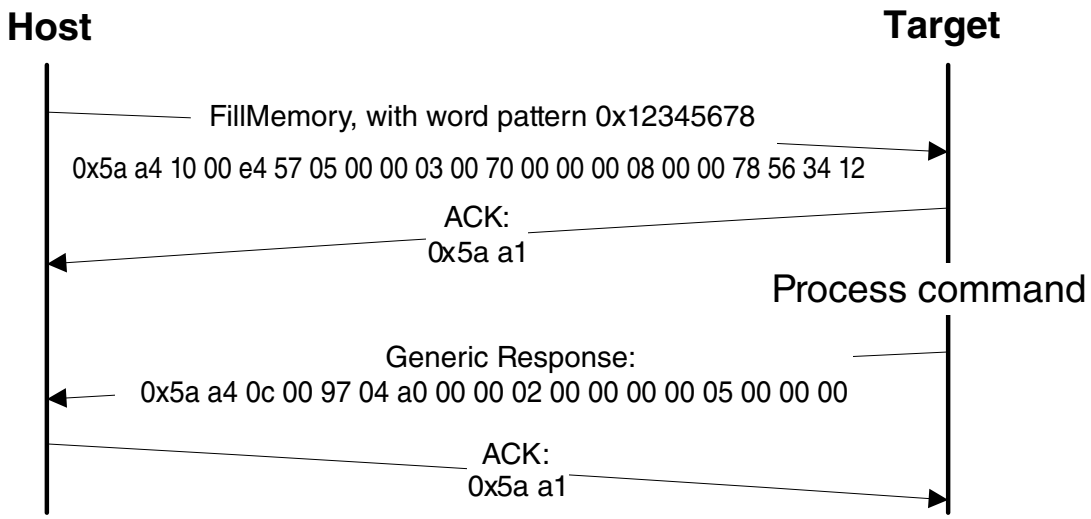


Figure 14-18. Protocol Sequence for FillMemory Command

Table 14-46. FillMemory Command Packet Format (Example)

FillMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0xE4 0x57
Command packet	commandTag	0x05 – FillMemory
	flags	0x00
	Reserved	0x00
	parameterCount	0x03
	startAddress	0x00007000
	byteCount	0x00000800
	patternWord	0x12345678

The FillMemory command has no data phase.

Response: upon successful execution of the command, the target (Kinetis Bootloader) will return a GenericResponse packet with a status code set to kStatus_Success, or to an appropriate error status code.

14.3.8.15 WriteMemory command

The WriteMemory command writes data provided in the data phase to a specified range of bytes in memory (flash or RAM). However, if flash protection is enabled, then writes to protected sectors will fail.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll, FlashEraseRegion, or FlashEraseAllUnsecure command.
- Writing to flash requires the start address to be .
- The byte count will be rounded up to a multiple of , and the trailing bytes will be filled with the flash erase pattern (0xff).
- If the VerifyWrites property is set to true, then writes to flash will also perform a flash verify program operation.

When writing to RAM, the start address need not be aligned, and the data will not be padded.

The start address and number of bytes are the 2 parameters required for WriteMemory command.

Table 14-47. Parameters for WriteMemory Command

Byte #	Command
0 - 3	Start address
4 - 7	Byte count

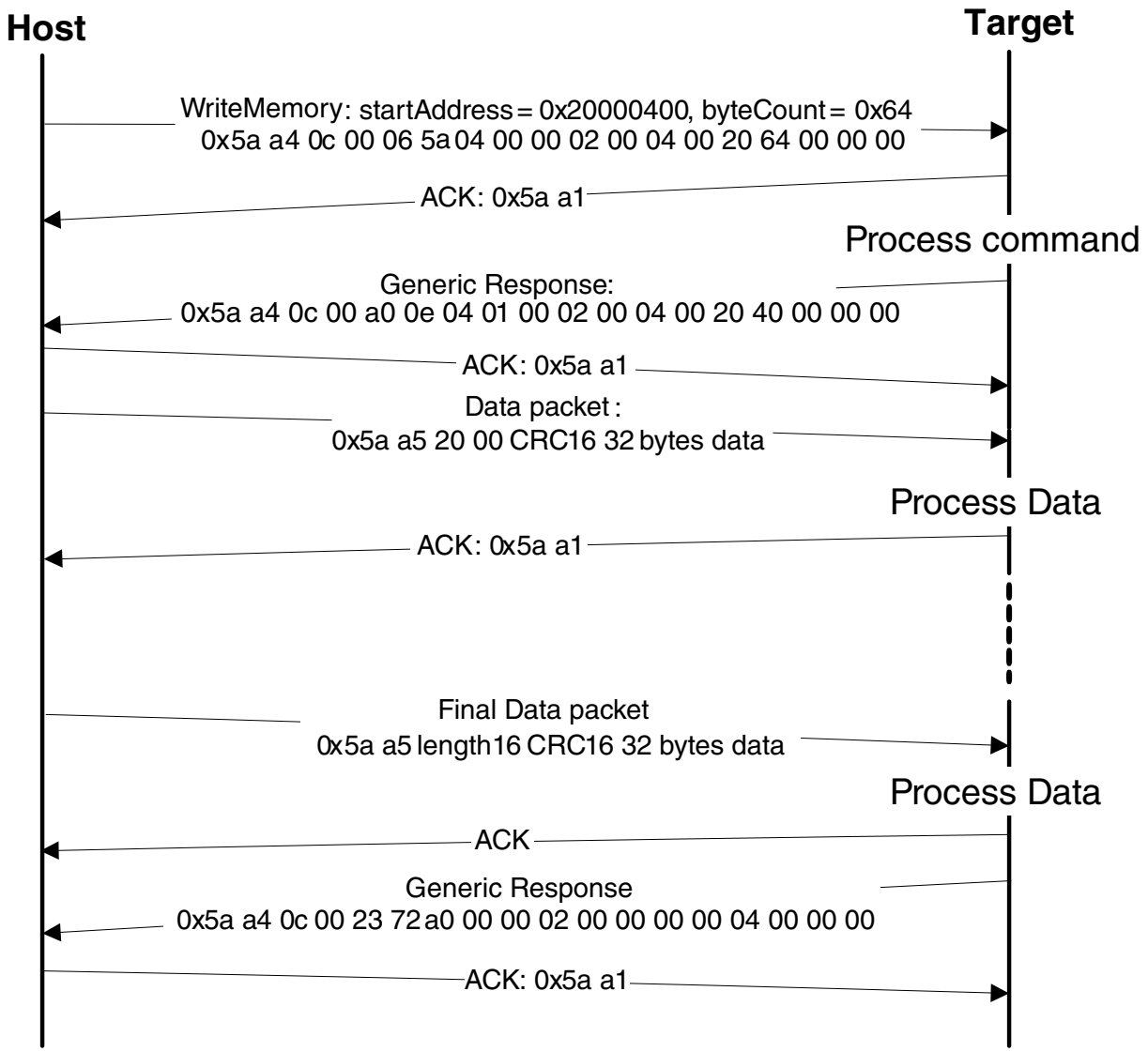


Figure 14-19. Protocol Sequence for WriteMemory Command

Table 14-48. WriteMemory Command Packet Format (Example)

WriteMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x06 0x5A
Command packet	commandTag	0x04 - writeMemory
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	startAddress	0x20000400
	byteCount	0x00000064

Data Phase: The WriteMemory command has a data phase; the host will send data packets until the number of bytes of data specified in the byteCount parameter of the WriteMemory command are received by the target.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code set to kStatus_Success upon successful execution of the command, or to an appropriate error status code.

14.3.8.16 Read memory command

The ReadMemory command returns the contents of memory at the given address, for a specified number of bytes. This command can read any region of memory accessible by the CPU and not protected by security.

The start address and number of bytes are the 2 parameters required for ReadMemory command.

Table 14-49. Parameters for read memory command

Byte	Parameter	Description
0-3	Start address	Start address of memory to read from
4-7	Byte count	Number of bytes to read and return to caller

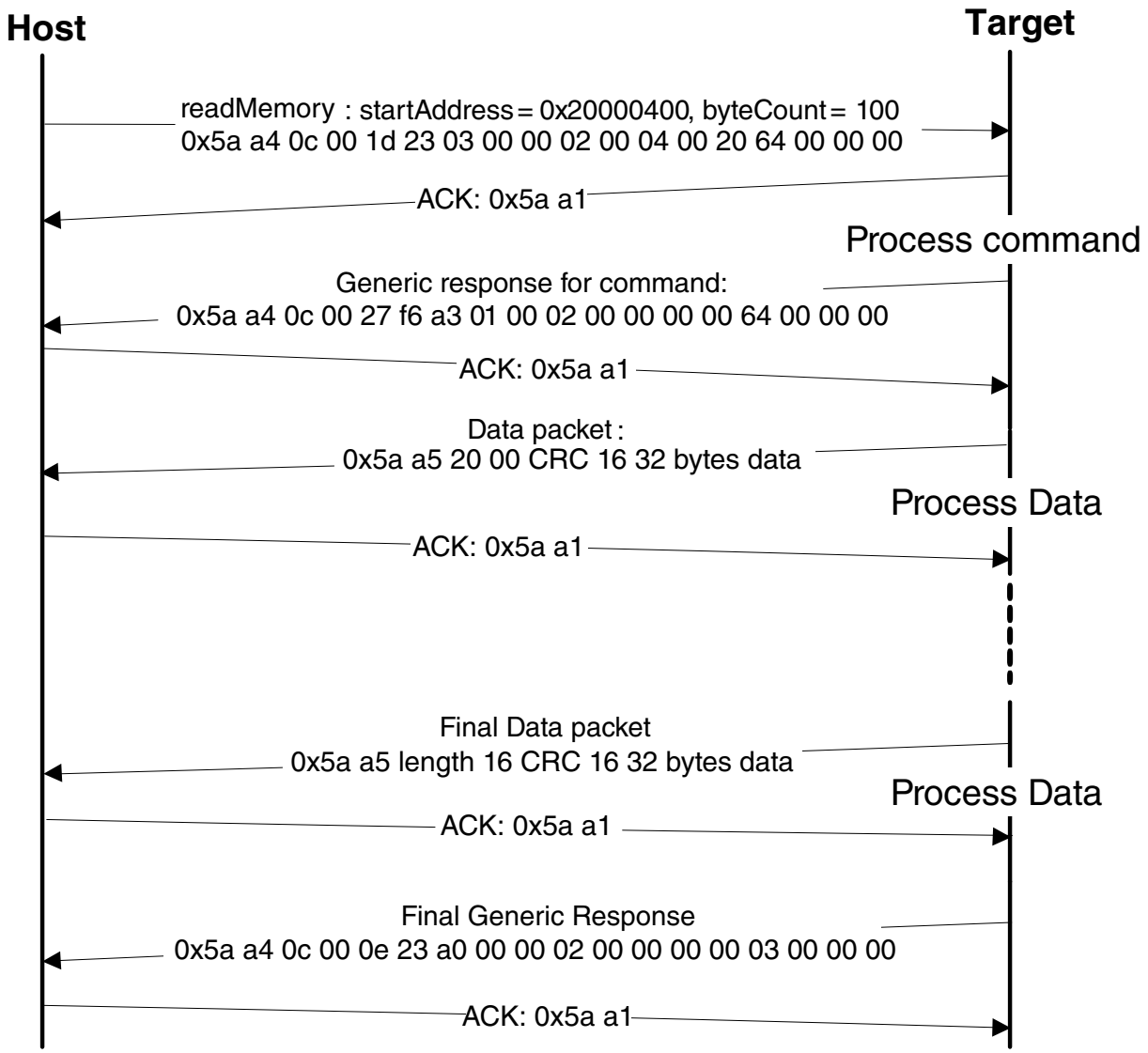


Figure 14-20. Command sequence for read memory

ReadMemory	Parameter	Value
Framing packet	Start byte	0x5A0xA4,
	packetType	kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x1D 0x23
Command packet	commandTag	0x03 - readMemory
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	startAddress	0x20000400
	byteCount	0x00000064

Data Phase: The ReadMemory command has a data phase. Since the target (Kinetis Bootloader) works in slave mode, the host need pull data packets until the number of bytes of data specified in the byteCount parameter of ReadMemory command are received by host.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

14.3.9 Bootloader Exit state

The Kinetis Bootloader tries to reconfigure the system back to the reset state in the following situations:

- After completion of an Execute command, but before jumping to the specified entry point.
- After a peripheral detection timeout, but before jumping to the application entry point.

In general, all peripherals are reset. However, the application must consider the following situations:

- If VLPR mode is active during a system boot, then the bootloader will exit VLPR mode on entry (to bootloader operation), and VLPR mode will not be restored after exiting from the bootloader.
- If high speed clocking is selected in the Bootloader Configuration Area, then high speed clocks will be retained after exiting from the bootloader.
- Upon exit from the bootloader, the bootloader restores the VTOR register to its default value (0x0).

14.4 Peripherals Supported

This section describes the peripherals supported by the Kinetis ROM Bootloader. To use an interface for bootloader communications, the peripheral must be enabled in the BCA, as shown in Table 14-3. If the BCA is invalid (such as all 0xFF bytes), then all peripherals will be enabled by default.

14.4.1 LPI2C Peripheral

The Kinetis Bootloader in ROM supports loading data into flash via the LPI2C peripheral, where the LPI2C peripheral serves as the LPI2C slave. A 7-bit slave address is used during the transfer.

Customizing an LPI2C slave address is also supported. This feature is enabled if the Bootloader Configuration Area (BCA) (shown in Table 14-3) is enabled (tag field is filled with 'kcfg') and the i2cSlaveAddress field is filled with a value other than 0xFF. 0x10 is used as the default LPI2C slave address.

The maximum supported LPI2C baud rate depends on corresponding clock configuration field in the BCA. Typical supported baud rate is 400 kbps with factory settings. Actual supported baud rate may be lower or higher than 400 kbps, depending on the actual value of the clockFlags and the clockDivider fields.

Because the LPI2C peripheral serves as an LPI2C slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

- An incoming packet is sent by the host with a selected LPI2C slave address and the direction bit is set as write.
- An outgoing packet is read by the host with a selected LPI2C slave address and the direction bit is set as read.
- 0x00 will be sent as the response to host if the target is busy with processing or preparing data.

The following flow charts demonstrate the communication flow of how the host reads ping packet, ACK and response from the target.

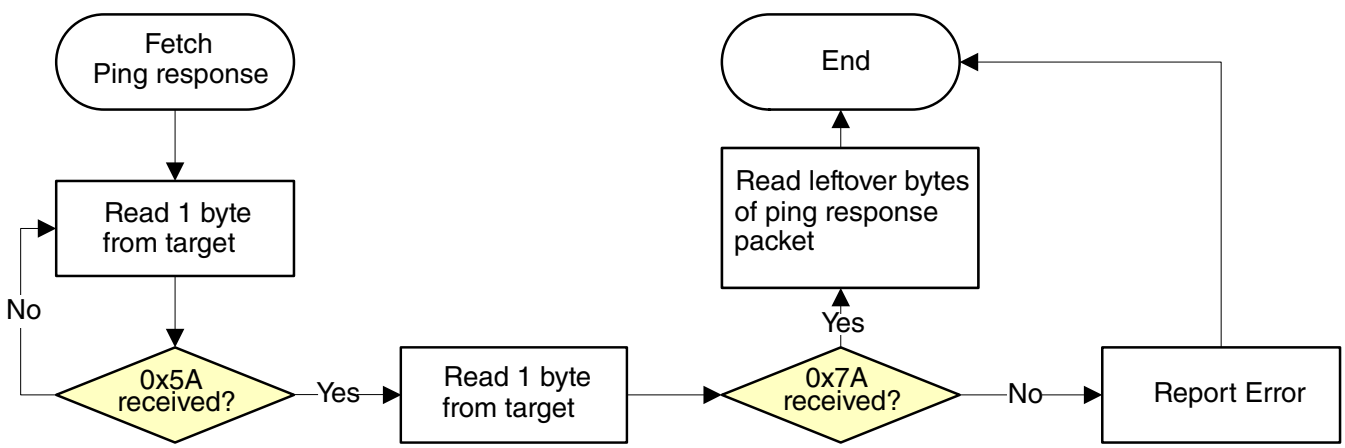


Figure 14-21. Host reads ping response from target via LPI2C

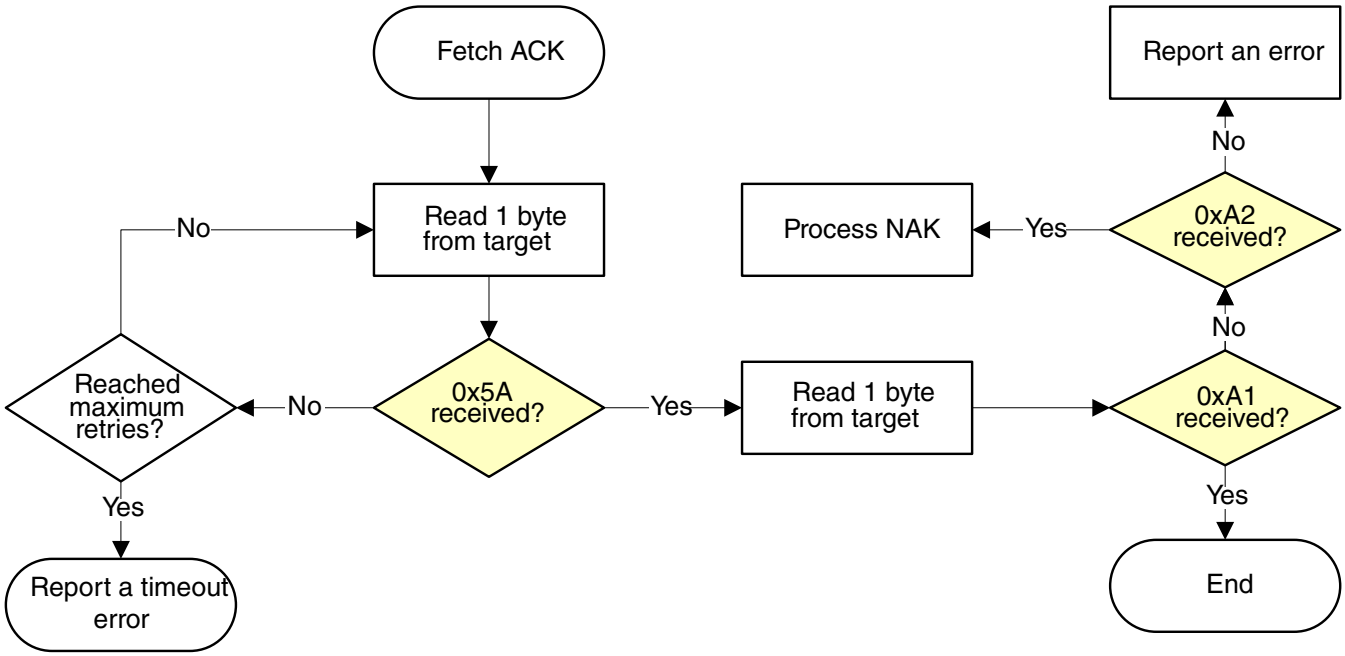


Figure 14-22. Host reads ACK packet from target via LPI2C

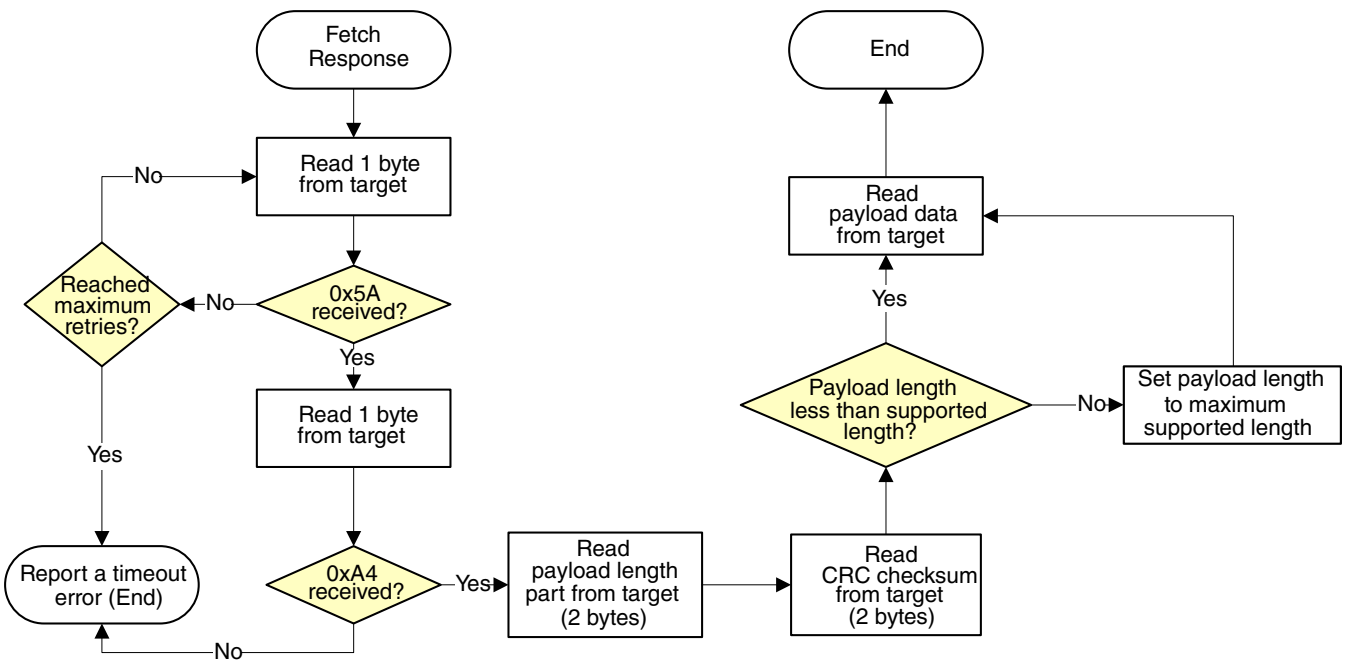


Figure 14-23. Host reads response from target via LPI2C

14.4.2 LPSPI Peripheral

The Kinetis Bootloader in ROM supports loading data into flash via the LPSPI peripheral, where the LPSPI peripheral serves as a LPSPI slave.

Maximum supported baud rate of LPSPI depends on the clock configuration fields in the Bootloader Configuration Area (BCA) shown in Table 14-3. The typical supported baud rate is 400 kbps with the factory settings. The actual baud rate is lower or higher than 400 kbps, depending on the actual value of the clockFlags and clockDivider fields in the BCA.

The LPSPI peripheral uses the following bus attributes:

- Clock Phase = 1 (Second Edge)
- Clock Polarity = 1 (Active Low)

Because the LPSPI peripheral in ROM serves as a LPSPI slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

The transfer on LPSPI is slightly different from I2C:

- Host will receive 1 byte after it sends out any byte.
- Received bytes should be ignored when host is sending out bytes to target
- Host starts reading bytes by sending 0x00s to target
- The byte 0x00 will be sent as response to host if target is under the following conditions:
  - Processing incoming packet
  - Preparing outgoing data
  - Received invalid data

The LPSPI bus configuration is:

- Phase = 1; data is sampled on rising edges
- Polarity = 1; idle is high
- MSB is transmitted first

For any transfer where the target does not have actual data to send, the target (slave) is responsible for ensuring that 0x00 bytes will be returned to the host (master). The host uses framing packets to identify real data and not "dummy" 0x00 bytes (which do not have framing packets).

The following flowcharts demonstrate how the host reads a ping response, an ACK and a command response from target via LPSPI.

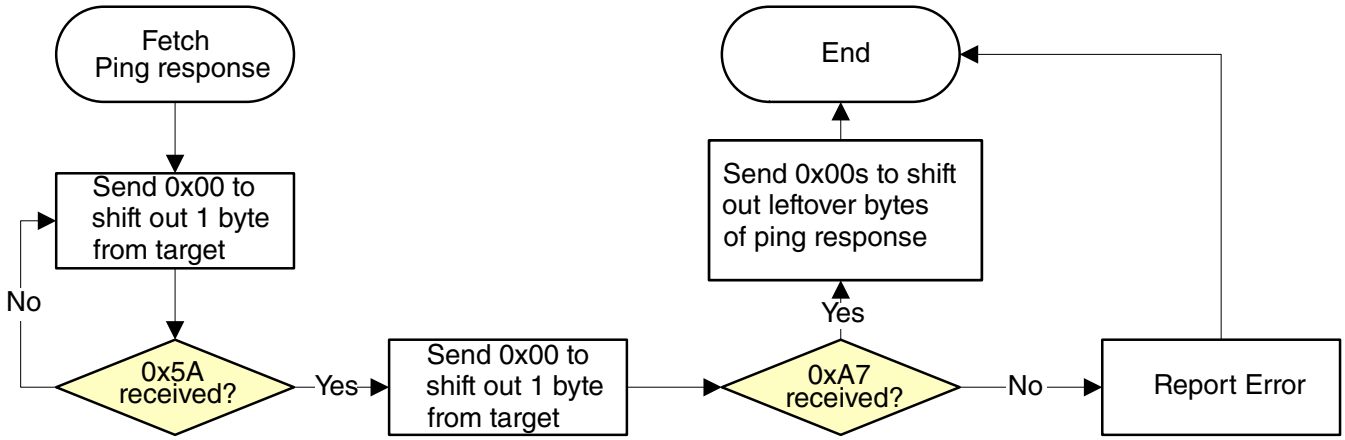


Figure 14-24. Host reads ping packet from target via LPSPI

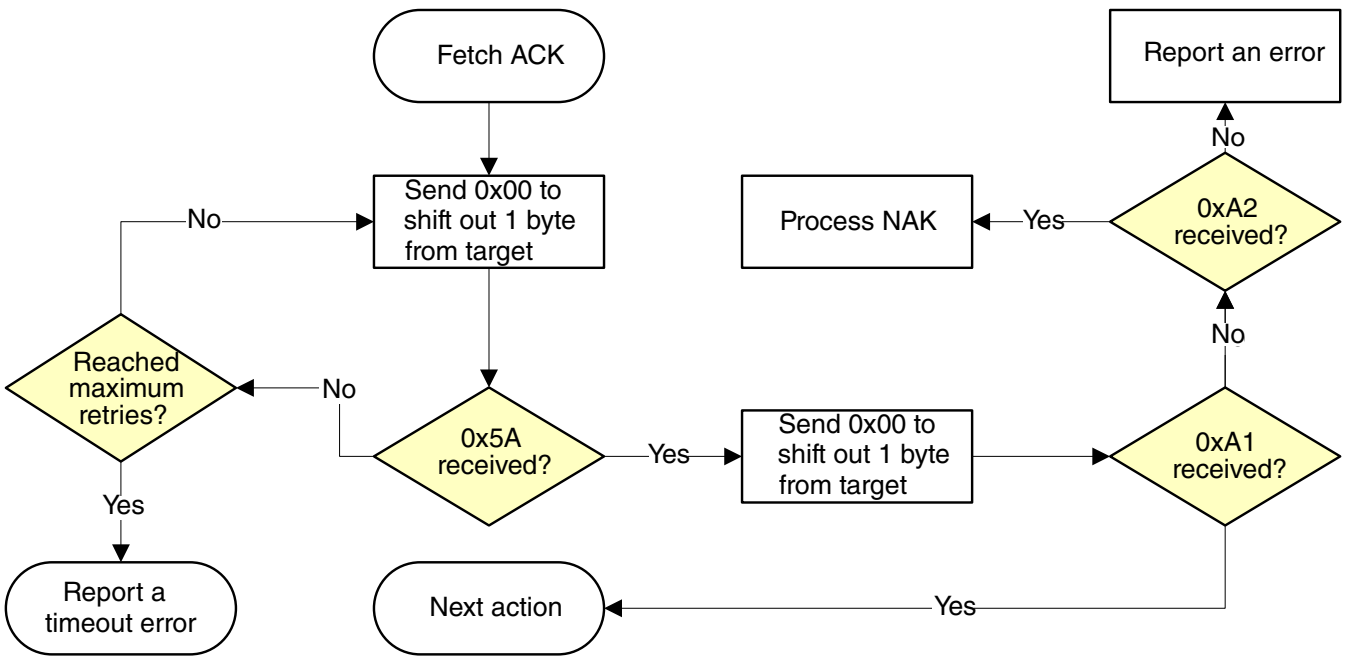


Figure 14-25. Host reads ACK from target via LPSPI

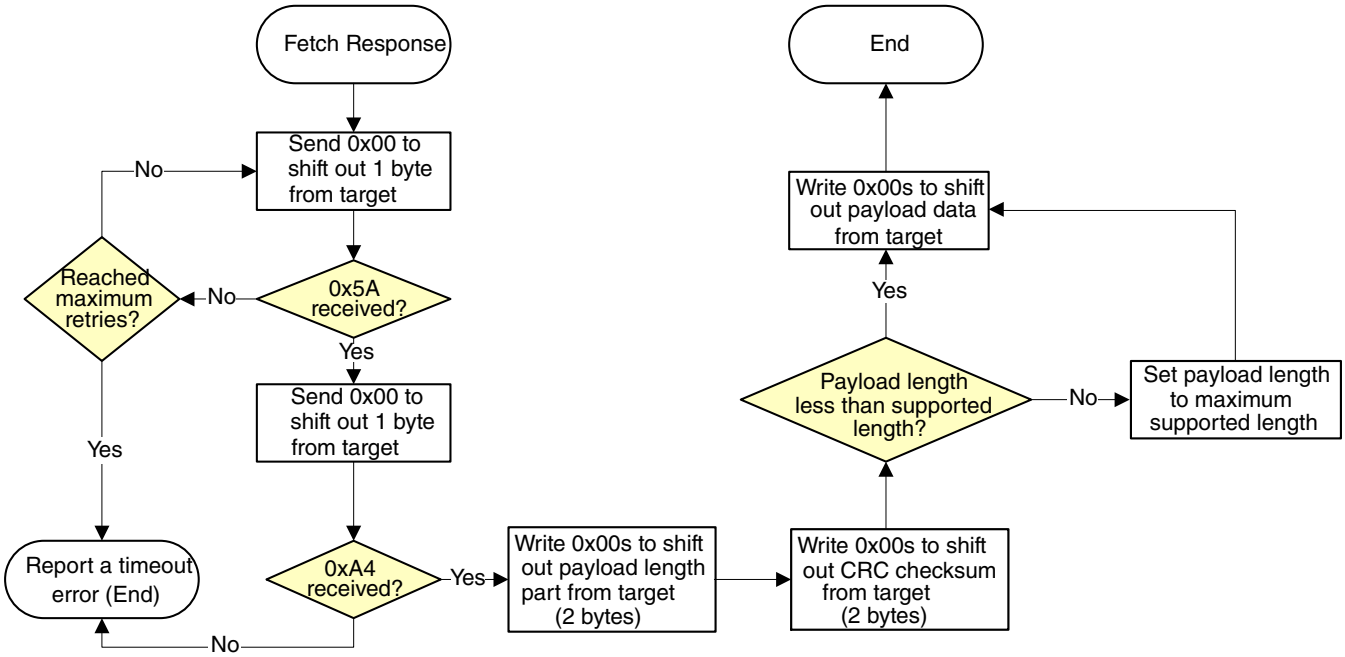


Figure 14-26. Host reads response from target via LPSPI

14.4.3 LPUART Peripheral

The Kinetis Bootloader integrates an autobaud detection algorithm for the LPUART peripheral, thereby providing flexible baud rate choices.

Autobaud feature: If LPUART*n* is used to connect to the bootloader, then the LPUART*n*_RX pin must be kept high and not left floating during the detection phase in order to comply with the autobaud detection algorithm. After the bootloader detects the ping packet (0x5A 0xA6) on LPUART*n*_RX, the bootloader firmware executes the autobaud sequence. If the baudrate is successfully detected, then the bootloader will send a ping packet response [(0x5A 0xA7), protocol version (4 bytes), protocol version options (2 bytes) and crc16 (2 bytes)] at the detected baudrate. The Kinetis Bootloader then enters a loop, waiting for bootloader commands via the LPUART peripheral.

NOTE

- The autobaud feature requires a ping packet with a higher accuracy (+/-3%), or the ping packet will be ignored as noise.
- The data bytes of the ping packet must be sent continuously (with no more than 80 ms between bytes) in a fixed LPUART transmission mode (8-bit data, no parity bit and 1 stop bit). If the bytes of the ping packet are sent one-by-one with more than 80 ms delay between them, then the

autobaud detection algorithm may calculate an incorrect baud rate. In this case, the autobaud detection state machine should be reset.

Supported baud rates: The baud rate is closely related to the MCU core and system clock frequencies. Typical baud rates supported are 9600, 19200, 38400, 57600, and 115200. Of course, to influence the performance of autobaud detection, the clock configuration in BCA can be changed.

Packet transfer: After autobaud detection succeeds, bootloader communications can take place over the LPUART peripheral. The following flow charts show:

- How the host detects an ACK from the target
- How the host detects a ping response from the target
- How the host detects a command response from the target

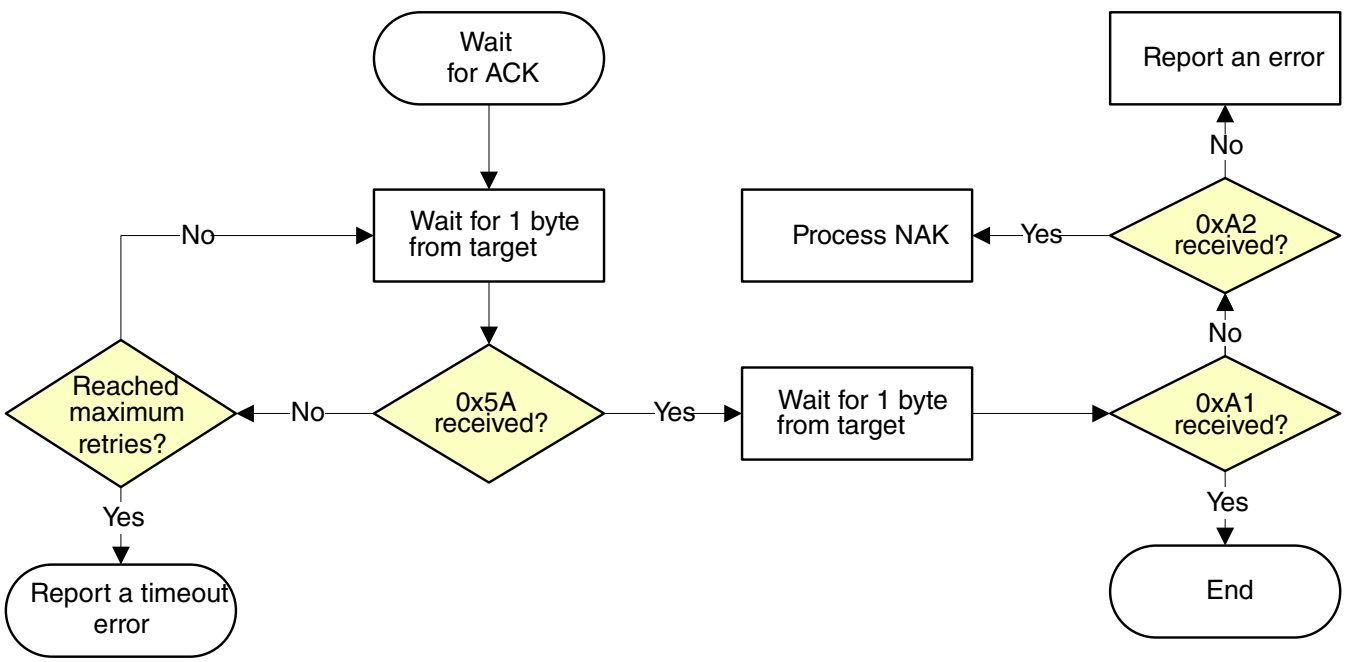


Figure 14-27. Host reads an ACK from target via LPUART

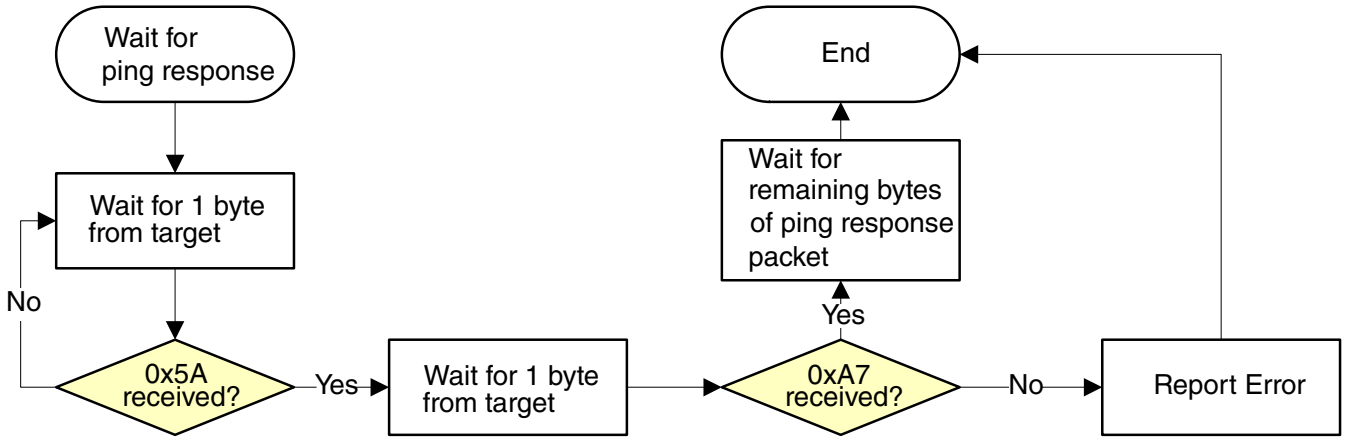


Figure 14-28. Host reads a ping response from target via LPUART

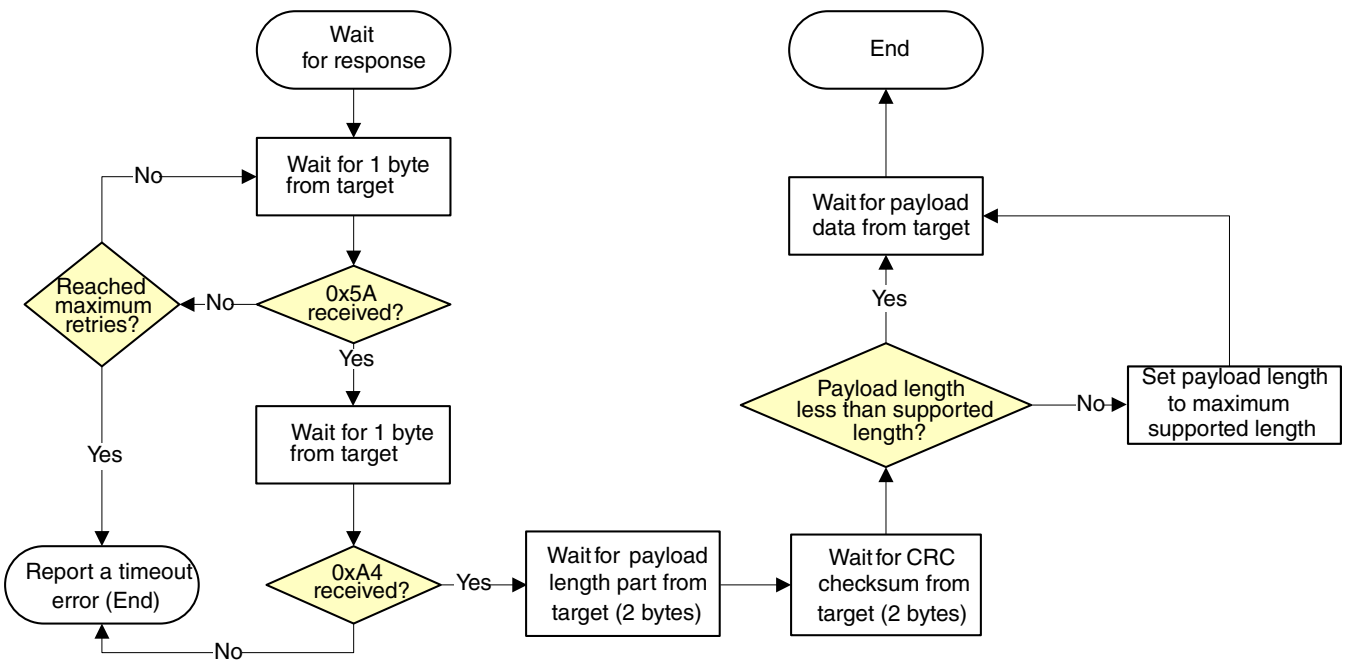


Figure 14-29. Host reads a command response from target via LPUART

14.4.4 USB peripheral

The Kinetis Bootloader in the ROM supports loading data into flash via the USB peripheral. The target is implemented as a USB HID class.

USB HID does not use framing packets; instead the packetization inherent in the USB protocol itself is used. The ability for the device to NAK Out transfers (until they can be received) provides the required flow control; the built-in CRC of each USB packet provides the required error detection.

14.4.4.1 Clock configuration

The ROM supports the crystal-less USB feature. If the USB peripheral is enabled, then the ROM enables the 48-MHz . The ROM also enables the USB clock recovery feature (by setting USBx_CLK_RECOVER_CTRL[CLOCK_RECOVER_EN] to 1).

14.4.4.2 Device descriptor

The Kinetis ROM configures the default USB VID/PID/Strings as below:

Default VID/PID:

- VID = 0x15A2
- PID = 0x0073

Default Strings:

- Manufacturer [1] = "Freescale Semiconductor Inc." (Note that Freescale Semiconductor is now NXP Semiconductors.)
- Product [2] = "Kinetis Bootloader"

You can customize the USB VID/PID/Strings with the Bootloader Configuration Area (BCA) of the flash. See Table 14-3. For example, the USB VID and PID can be customized by writing the new VID to the usbVid(BCA + 0x14) field and the new PID to the usbPid(BCA + 0x16) field of the BCA in flash. To change the USB strings, you need to prepare a structure (like the one shown below) in the flash, and then write the address of the g_languages structure to the usbStringsPointer(BCA + 0x18) field of the BCA.

```
g_languages = { USB_STR_0,
sizeof(USB_STR_0),
(uint_16)0x0409,
(const uint_8 **)g_string_descriptors,
g_string_desc_size};
the USB_STR_0, g_string_descriptors and g_string_desc_size are defined as below.
USB_STR_0[4] =    {0x02,
0x03,
0x09,
0x04
};
g_string_descriptors[4] =
{ USB_STR_0,
USB_STR_1,
USB_STR_2,
USB_STR_3};
g_string_desc_size[4] =
{ sizeof(USB_STR_0),
sizeof(USB_STR_1),
sizeof(USB_STR_2),
sizeof(USB_STR_3)};
```

You can make your own structure of USB_STR_1, USB_STR_2, USB_STR_3:

- USB_STR_1 is used for the manufacturer string.
- USB_STR_2 is used for the product string.
- USB_STR_3 is used for the serial number string.

By default, the 3 strings are defined as below:

```
USB_STR_1[] =
{  sizeof(USB_STR_1),
USB_STRING_DESCRIPTOR,
'F',0,
'r',0,
'e',0,
'e',0,
's',0,
'c',0,
'a',0,
'l',0,
'e',0,
' ',0,
'S',0,
'e',0,
'm',0,
'i',0,
'c',0,
'o',0,
'n',0,
'd',0,
'u',0,
'c',0,
't',0,
'o',0,
'r',0,
' ',0,
'I',0,
'n',0,
'c',0,
'.',0
};

USB_STR_2[] =
{  sizeof(USB_STR_2),
USB_STRING_DESCRIPTOR,
'U',0,
'S',0,
'B',0,
' ',0,
'C',0,
'O',0,
'M',0,
'P',0,
'O',0,
'S',0,
'I',0,
'T',0,
'E',0,
' ',0,
'D',0,
'E',0,
'V',0,
'I',0,
'C',0,
'E',0
};
```

```
USB_STR_3[] =
{  sizeof(USB_STR_3),
USB_STRING_DESCRIPTOR,
'M',0,
'C',0,
'U',0,
' ',0,
'H',0,
'I',0,
'D',0,
' ',0,
'G',0,
'E',0,
'N',0,
'E',0,
'R',0,
'I',0,
'C',0,
' ',0,
'D',0,
'E',0,
'V',0,
'I',0,
'C',0,
'E',0
};
```

14.4.4.3 Endpoints

The HID peripheral uses 3 endpoints:

- Control (0)
- Interrupt IN (1)
- Interrupt OUT (2)

The Interrupt OUT endpoint is optional for HID class devices, but the Kinetis Bootloader uses it as a pipe, where the firmware can NAK send requests from the USB host.

14.4.4.4 HID reports

There are 4 HID reports defined and used by the bootloader USB HID peripheral. The report ID determines the direction and type of packet sent in the report; otherwise, the contents of all reports are the same.

Report ID	Packet Type	Direction
1	Command	OUT
2	Data	OUT
3	Command	IN
4	Data	IN

For all reports, these properties apply:

Usage Min	1
Usage Max	1
Logical Min	0
Logical Max	255
Report Size	8
Report Count	34

Each report has a maximum size of 34 bytes. This is derived from the minimum bootloader packet size of 32 bytes, plus a 2-byte report header that indicates the length (in bytes) of the packet sent in the report.

NOTE

In the future, the maximum report size may be increased, to support transfers of larger packets. Alternatively, additional reports may be added with larger maximum sizes.

The actual data sent in all of the reports looks like:

0	Report ID
1	Packet Length LSB
2	Packet Length MSB
3	Packet[0]
4	Packet[1]
5	Packet[2]
	...
N+3-1	Packet[N-1]

This data includes the Report ID, which is required if more than one report is defined in the HID report descriptor. The actual data sent and received has a maximum length of 35 bytes. The Packet Length header is written in little-endian format, and it is set to the size (in bytes) of the packet sent in the report. This size does not include the Report ID or the Packet Length header itself. During a data phase, a packet size of 0 indicates a data phase abort request from the receiver.

14.5 Get/SetProperty Command Properties

This section lists the properties of the GetProperty and SetProperty commands.

Table 14-50. Properties used by Get/SetProperty Commands, sorted by Value

Property	Writable	Tag Value	Size	Description
CurrentVersion	No	01h	4	Current bootloader version.
AvailablePeripherals	No	02h	4	The set of peripherals supported on this chip.
FlashStartAddress	No	03h	4	Start address of program flash.
FlashSizeInBytes	No	04h	4	Size in bytes of program flash.
FlashSectorSize	No	05h	4	The size in bytes of one sector of program flash. This is the minimum erase size.
FlashBlockCount	No	06h	4	Number of blocks in the flash array.
AvailableCommands	No	07h	4	The set of commands supported by the bootloader.
VerifyWrites	Yes	0Ah	4	Controls whether the bootloader will verify writes to flash. VerifyWrites feature is enabled by default. 0 - No verification is done. 1 - Enable verification.
MaxPacketSize	No	0Bh	4	Maximum supported packet size for the currently active peripheral interface.
ReservedRegions	No	0Ch	16	List of memory regions reserved by the bootloader. Returned as value pairs (<start-address-of-region>, <end-address-of-region>). • If HasDataPhase flag is not set, then the Response packet parameter count indicates the number of pairs. • If HasDataPhase flag is set, then the second parameter is the number of bytes in the data phase.
RAMStartAddress	No	0Eh	4	Start address of RAM
RAMSizeInBytes	No	0Fh	4	Size in bytes of RAM
SystemDeviceId	No	10h	4	Value of the Kinetis System Device Identification register.
FlashSecurityState	No	11h	4	Indicates whether Flash security is enabled 0 - Flash security is disabled 1 - Flash security is enabled
UniqueDeviceId	No	12h	16	Unique device identification, value of Kinetis Unique Identification registers (16 for K series devices, 12 for KL series devices)
FacSupport	No	13h	4	FAC (Flash Access Control) support flag 0 - FAC not supported 1 - FAC supported
FlashAcessSegmentSize	No	14h	4	The size in bytes of 1 segment of flash
FlashAcessSegmentCount	No	15h	4	FAC segment count (The count of flash access segments within the flash model.)
FlashReadMargin	Yes	16h	4	The margin level setting for flash erase and program verify commands. 0 = Normal 1 = User (default)

Table continues on the next page...

Table 14-50. Properties used by Get/SetProperty Commands, sorted by Value (continued)

Property	Writable	Tag Value	Size	Description
				2 = Factory
TargetVersion	No	18h	4	SoC target build version number

14.5.1 Property Definitions

Get/Set property definitions are provided in this section.

14.5.1.1 CurrentVersion Property

The value of this property is a 4-byte structure containing the current version of the bootloader.

Table 14-51. Fields of CurrentVersion property:

Bits	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'K' (0x4B)	Major version	Minor version	Bugfix version

14.5.1.2 AvailablePeripherals Property

The value of this property is a bitfield that lists the peripherals supported by the bootloader and the hardware on which it is running.

Table 14-52. Peripheral bits:

Bit	[31:7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Peripheral	Reserved	USB DFU	USB CDC	USB HID	Reserved	LPSPI Slave	LPI2C Slave	LPUART

If the peripheral is available, then the corresponding bit will be set in the property value. All reserved bits must be set to 0.

14.5.1.3 AvailableCommands Property

This property value is a bitfield with set bits indicating the commands enabled in the bootloader. Only commands that can be sent from the host to the target are listed in the bitfield. Response commands such as GenericResponse are excluded.

The bit number that identifies whether a command is present is the command's tag value minus 1. 1 is subtracted from the command tag because the lowest command tag value is 0x01. To get the bit mask for a given command, use this expression:

```
mask = 1 << (tag - 1)
```

Table 14-53. Command bits:

Bit	[31:18]	[17]	[16]	[15]	[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Command	Reserved	Reserved	Reserved	FlashReadResource	FlashReadOnce	FlashProgramOnce	FlashEraseAllUnsecure	SetProperty	Reset	Call	Execute	ReceiveSBFile	GetProperty	FlashSecurityDisable	FillMemory	WriteMemory	ReadMemory	FlashEraseRegion	FlashEraseAll

14.5.1.4 TargetVersion Property

The value of this property is a 4-byte structure containing the target version of the bootloader. The TargetVersion number returned returned for this device is 1.0.0.

Table 14-54. Fields of TargetVersion property:

Bits	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'T' (0x4B)	Major version	Minor version	Bugfix version

14.6 SB File Decryption Support

The bootloader supports the loading and flashing of applications using an encrypted SB file format. Using the ReceiveSbFile command, the host can send an SB file that contains various commands and data that will be executed on the Kinetis bootloader (the target). An unencrypted SB file can be flashed to the target only when flash security is disabled. When flash security is enabled, all memory writes and reads are disabled, with the exception being an encrypted SB file that can be sent to the target. The SB file is

decrypted using an AES-128 style key (16 bytes). Before you send the encrypted file, the key must be written to the target using the FlashProgramOnce command. The SB key begins at offset 0x30 in the program once registers.

For example, to program a key of 000102030405060708090A0B0C0D0E0F:

- FlashProgramOnce to index 0x30 with 4 bytes of 0x03020100
- FlashProgramOnce to index 0x31 with 4 bytes of 0x07060504
- FlashProgramOnce to index 0x32 with 4 bytes of 0x0B0A0908
- FlashProgramOnce to index 0x33 with 4 bytes of 0x0F0E0D0C

If no key is programmed in these registers, then an attempt is made to decrypt the file using an all-zero key.

14.6.1 Decryption using MMCAU

The MMCAU code must flashed to the device memory in a separate step. The location of MMCAU initialization, encryption, and decryption functions does not matter, as long as it is on accessible memory. The provided functions are expected to have the following prototypes:

```
    void mmcau_aes_init(unsigned int* key, unsigned int* keySchedule, unsigned int* rcon);
    void mmcau_aes_encrypt(unsigned int* in, unsigned int* key, unsigned int* keySchedule,
unsigned int* out);
    void mmcau_aes_decrypt(unsigned int* in, unsigned int* key, unsigned int* keySchedule,
unsigned int* out);
```

NOTE

The mmCAU software library source code can be downloaded from www.nxp.com.

In order to decrypt an encrypted SB file using the MMCAU peripheral, the following information must be provided:

- The Bootloader Configuration Area (BCA) contains a pointer to an MMCAU set-up structure. The location in the configuration area (relative to the BCA) is at offset 0x20, as shown in the BCA layout in this document. The pointer is a little endian memory location that contains the following structure:

```
        typedef struct mmcau_function_info
{
    uint32_t tag;
    uint32_t length;
    uint32_t aes_init_start;
    uint32_t aes_encrypt_start;
    uint32_t aes_decrypt_start;
} mmcau_function_info_t;
```

For decryption to work properly, the mmcau_function_info structure must contain valid values for all the fields in this structure:
  a. The tag field must equal 'kcau'.
  b. The length field must equal the size of the total length of the aes_init, aes_encrypt, and aes_decrypt functions. All of these function must be in a contiguous region of memory that fits within 1280 bytes. All code in the functions must be relocatable.
  c. aes_init_start contains the memory location of the aes_init function.
  d. aes_encrypt_start contains the memory location of the aes_encrypt function.
  e. aes_decrypt_start contains the memory location of the aes_decrypt function.
- The MMCAU information structure and the actual MMCAU functions do not need to be in contiguous memory.
- A ready-to-use library contains these functions, but this library needs to be rebuilt to be compatible with the prototypes above. Specifically, the rcon variable needs to be passed into the functions, to make the code relocatable.

After an encrypted SB file starts to be received by the bootloader, the information for setting up the MMCAU is retrieved and the decryption process begins.

- If the information is incorrect or if the functions do not meet the above requirements, then the bootloader can become unresponsive, so be sure that everything is correctly implemented as above.
- If there is a recoverable error, then the bootloader will return a relevant error code.

If everything is operating properly, then the exact same response will be received as though the bootloader was receiving an unencrypted file via the ReceiveSbFile command.

14.7 CRC-32 Check on Application Data

Using CRC-32 and a given address range, the ROM bootloader supports performing an application integrity check. To properly configure this functionality, the following fields in the bootloader configuration area must be set:

- Set crcStartAddress to the start address that should be used for the CRC check.
- Set crcByteCount to the number of bytes to run the CRC check on, from the start address.
- Set crcExpectedValue to the value that the CRC calculation should result in.

Considerations:

- If all of the above fields are unset (all 0xFF bytes for crcStartAddress, crcByteCount, and crcExpectedValue), then the ROM bootloader returns kStatus_AppCrcCheckInvalid.

- If any one of the above fields are set (crcStartAddress, crcByteCount, and crcExpectedValue), then the ROM bootloader checks if the given address range of the application is valid and if the application just resides in internal flash:
  - If false, then the bootloader returns kStatus_AppCrcCheckOutOfRange.
  - If true, then the CRC check occurs. If the CRC check fails, then the bootloader returns kStatus_AppCrcCheckFailed; if the CRC check succeeds, then it returns kStatus_AppCrcCheckPassed.
  - If the bootloader returns kStatus_AppCrcCheckOutOfRange or kStatus_AppCrcCheckFailed, then an external pin (PTA6) will also be asserted, to indicate CRC check failure.

NOTE

PTA6 is only available on the 121 MAP BGA and 100 LQFP packages.

  - Only if kStatus_AppCrcCheckPassed is returned, will the application be jumped to; otherwise the bootloader will stay active, and wait for further commands.
- The CRC32 algorithm in ROM uses a polynomial (0x04C1_1DB7), and the initial seed is 0xFFFF_FFFF.

14.8 Kinetis Bootloader Status Error Codes

This section describes the status error codes that the Kinetis Bootloader returns to the host.

Table 14-55. Kinetis Bootloader Status Error Codes, sorted by Value

Error Code	Value	Description
kStatus_Success	0	Operation succeeded without error.
kStatus_Fail	1	Operation failed with a generic error.
kStatus_ReadOnly	2	Requested value cannot be changed because it is read-only.
kStatus_OutOfRange	3	Requested value is out of range.
kStatus_InvalidArgument	4	The requested command's argument is undefined.
kStatus_Timeout	5	A timeout occurred.
kStatus_FlashSizeError	100	Not used.
kStatus_FlashAlignmentError	101	Address or length does not meet required alignment.
kStatus_FlashAddressError	102	Address or length is outside addressable memory.
kStatus_FlashAccessError	103	The FTFA_FSTAT[ACCERR] bit is set.
kStatus_FlashProtectionViolation	104	The FTFA_FSTAT[FPVIOL] bit is set.
kStatus_FlashCommandFailure	105	The FTFA_FSTAT[MGSTAT0] bit is set.

Table continues on the next page...

Table 14-55. Kinetis Bootloader Status Error Codes, sorted by Value (continued)

Error Code	Value	Description
kStatus_FlashUnknownProperty	106	Unknown Flash property.
kStatus_FlashEraseKeyError	107	The key provided does not match the programmed flash key.
kStatus_FlashRegionExecuteOnly	108	The area of flash is protected as execute only.
kStatus_I2C_SlaveTxUnderrun	200	I2C Slave TX Underrun error.
kStatus_I2C_SlaveRxOverrun	201	I2C Slave RX Overrun error.
kStatus_I2C_AribtrationLost	202	I2C Arbitration Lost error.
kStatus_SPI_SlaveTxUnderrun	300	SPI Slave TX Underrun error.
kStatus_SPI_SlaveRxOverrun	301	SPI Slave RX Overrun error.
kStatus_SPI_Timeout	302	SPI tranfser timed out.
kStatus_SPI_Busy	303	SPI instance is already busy performing a transfer.
kStatus_SPI_NoTransferInProgress	304	Attempt to abort a transfer when no transfer was in progress.
kStatus_UnknownCommand	10000	The requested command value is undefined.
kStatus_SecurityViolation	10001	Command is disallowed because flash security is enabled.
kStatus_AbortDataPhase	10002	Abort the data phase early.
kStatus_Ping	10003	Internal: received ping during command phase.
kStatusRomLdrSectionOverrun	10100	The loader has finished processing the SB file.
kStatusRomLdrSignature	10101	The signature of the SB file is incorrect.
kStatusRomLdrSectionLength	10102	The section length in chunks is invalid.
kStatusRomLdrUnencryptedOnly	10103	An encrypted SB file has been sent and decryption support is not available.
kStatusRomLdrEOFReached	10104	The end of the SB file has been reached.
kStatusRomLdrChecksum	10105	The checksum of a command tag block is invalid.
kStatusRomLdrCrc32Error	10106	The CRC-32 of the data for a load command is incorrect.
kStatusRomLdrUnknownCommand	10107	An unknown command was found in the SB file.
kStatusRomLdrIdNotFound	10108	There was no bootable section found in the SB file.
kStatusRomLdrDataUnderrun	10109	The SB state machine is waiting for more data.
kStatusRomLdrJumpReturned	10110	The function that was jumped to by the SB file has returned.
kStatusRomLdrCallFailed	10111	The call command in the SB file failed.
kStatusRomLdrKeyNotFound	10112	A matching key was not found in the SB file's key dictionary to unencrypt the section.
kStatusRomLdrSecureOnly	10113	The SB file sent is unencrypted and security on the target is enabled.
kStatusRomLdrResetReturned	10114	The SB file reset operation has unexpectedly returned.
kStatusMemoryRangeInvalid	10200	Memory range conflicts with a protected region.
kStatus_UnknownProperty	10300	The requested property value is undefined.
kStatus_ReadOnlyProperty	10301	The requested property value cannot be written.

Table continues on the next page...

Table 14-55. Kinetis Bootloader Status Error Codes, sorted by Value (continued)

Error Code	Value	Description
kStatus_InvalidPropertyValue	10302	The specified property value is invalid.
kStatus_AppCrcCheckPassed	10400	CRC check is valid and passed.
kStatus_AppCrcCheckFailed	10401	CRC check is valid but failed.
kStatus_AppCrcCheckInactive	10402	CRC check is inactive.
kStatus_AppCrcCheckInvalid	10403	CRC check is invalid, because the BCA is invalid or the CRC parameters are unset (all 0xFF bytes).
kStatus_AppCrcCheckOutOfRange	10404	CRC check is valid but addresses are out of range.

Chapter 15
Cryptographic Acceleration Unit (CAU)

15.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The memory-mapped Cryptographic Acceleration Unit (CAU) is a coprocessor that is connected to the processor's Private Peripheral Bus (PPB). It supports the hardware implementation of a set of specialized operations to improve the throughput of software-based security encryption/decryption operations and message digest functions.

The CAU supports acceleration of the DES, 3DES, AES, MD5, SHA-1, and SHA-256 algorithms. NXP provides an optimized C-function library that provides the appropriate software building blocks to implement higher-level security functions.

15.2 CAU Block Diagram

A simplified block diagram is given below that illustrates the CAU and a table to show its parts.

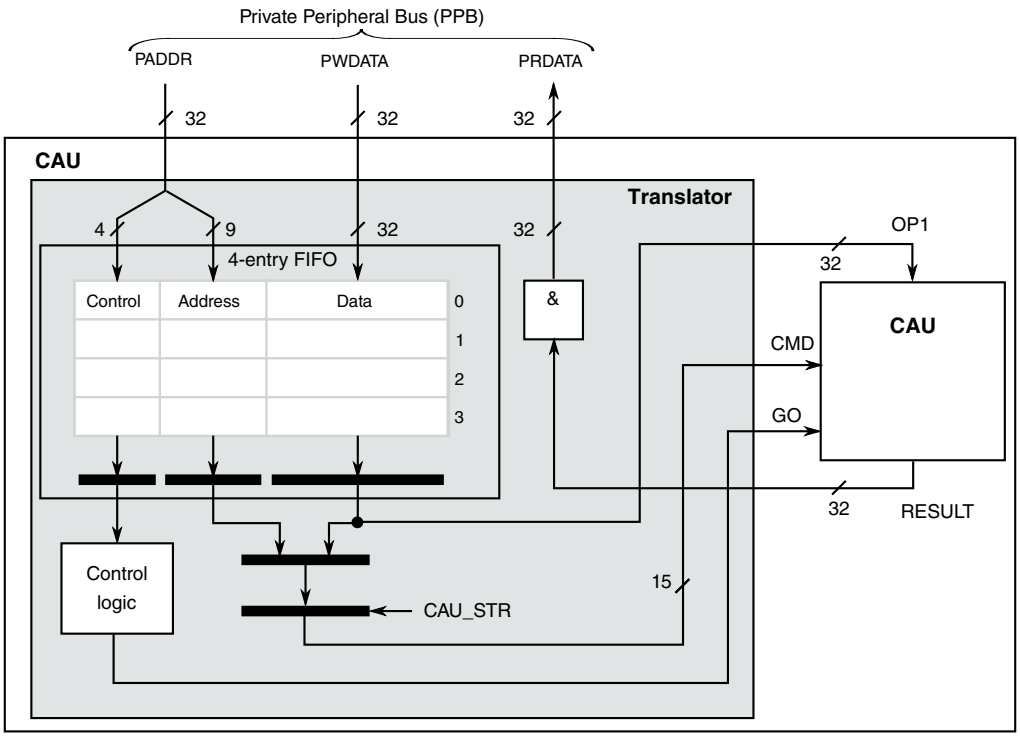


Figure 15-1. CAU block diagram

Table 15-1. CAU parts table

Item	Description
Translator submodule	Provides the bridge between the PPB interface and the CAU module. Passes memory-mapped commands and data on the PPB to/from the CAU
4-entry FIFO	Contains commands and input operands and the associated control captured from the PPB and sent to the CAU
CAU	3-terminal block with a command and optional input operand and a result bus. More details in following figure.

The following figure shows the CAU block in more detail.

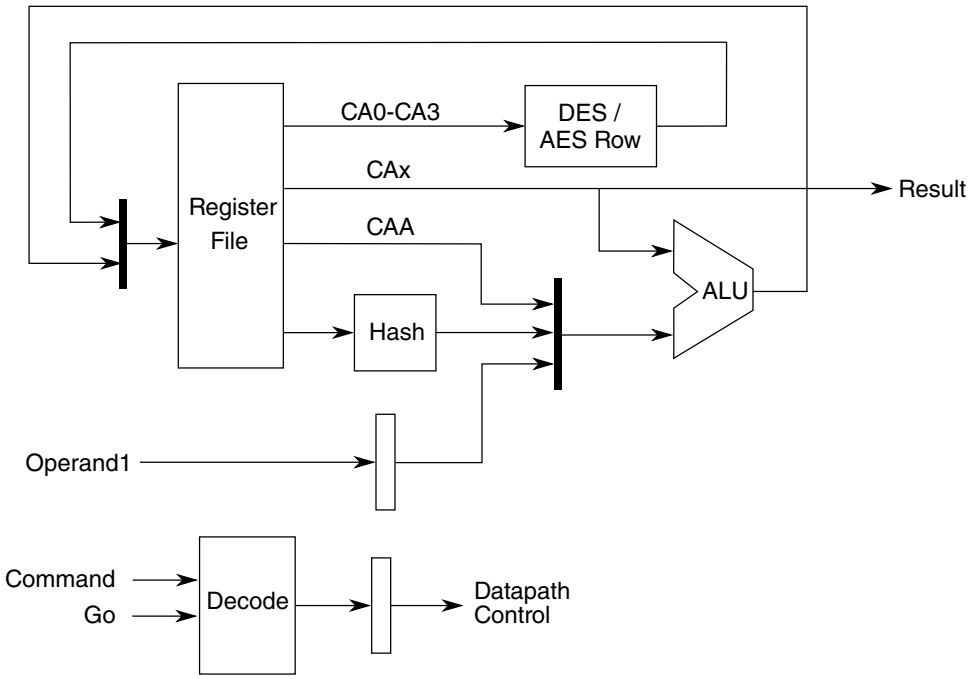


Figure 15-2. Top-level CAU block diagram

15.3 Overview

As the name suggests, the CAU provides a mechanism for memory-mapped register reads and writes to be transformed into specific commands and operands sent to the CAU coprocessor.

The CAU translator module performs the following functions:

- All the required functions affecting the transmission of commands to the CAU module.
- If needed, stalling the PPB transactions based on the state of the 4-entry command/ data FIFO.
- Some basic integrity checks on PPB operations.

The set of implemented algorithms provides excellent support for network security standards, such as SSL and IPsec. Additionally, using the CAU efficiently permits the implementation of any higher level functions or modes of operation, such as HMAC, CBC, and so on based on the supported algorithms.

The cryptographic algorithms are implemented partially in software with only functions critical to increasing performance implemented in hardware. The CAU allows for efficient, fine-grained partitioning of functions between hardware and software:

- Implement the innermost security kernel functions using the coprocessor instructions.
- Implement higher level functions in software by using the standard processor instructions.

This partitioning of functions is key to minimizing size of the CAU while maintaining a high level of throughput. Using software for some functions also simplifies the CAU design. The CAU implements a set of coprocessor commands that operate on a register file of 32-bit registers.

15.4 Features

The CAU includes the following distinctive features:

- Supports DES, 3DES, AES, MD5, SHA-1, and SHA-256 algorithms
- Simple, flexible programming model
- Ability to send up to three commands in one data write operation

15.5 Memory map/Register definition

The CAU contains multiple registers used by each of the supported algorithms. The following table shows registers that are applicable to each supported algorithm and indicates the corresponding letter designations for each algorithm.

For more information on these letter designations, see the supported algorithm specifications.

Code	Register	DES	AES	MD5	SHA-1	SHA-256
0	CAU Status Register (CASR)	—	—	—	—	—
1	CAU Accumulator (CAA)	—	—	a	T	T
2	General-Purpose Register 0 (CA0)	C	W0	—	A	A
3	General-Purpose Register 1 (CA1)	D	W1	b	B	B

Table continues on the next page...

Code	Register	DES	AES	MD5	SHA-1	SHA-256
4	General-Purpose Register 2 (CA2)	L	W2	c	C	C
5	General-Purpose Register 3 (CA3)	R	W3	d	D	D
6	General-Purpose Register 4 (CA4)	—	—	—	E	E
7	General-Purpose Register 5 (CA5)	—	—	—	W	F
8	General-Purpose Register 6 (CA6)	—	—	—	—	G
9	General-Purpose Register 7 (CA7)	—	—	—	—	H
10	General-Purpose Register 8 (CA8)	—	—	—	—	W/T_1

The CAU supports only 32-bit operations and register accesses. All registers support read, write, and ALU operations. However, only bits 1–0 of the CASR are writeable. Bits 31–2 of the CASR must be written as 0 for compatibility with future versions of the CAU.

The codes listed in this section are used in the memory-mapped commands. For more details on this, see CAU programming model.

NOTE

In the following table, the "address" or "offset" refers to the command code value for the CAU registers.

CAU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_5000	Status Register (CAU0_CASR)	32	R/W	2000_0000h	15.5.1/322
F000_5004	Accumulator (CAU0_CAA)	32	R/W	0000_0000h	15.5.2/323

Table continues on the next page...

CAU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_5008	General Purpose Register (CAU0_CA0)	32	R/W	0000_0000h	15.5.3/323
F000_500C	General Purpose Register (CAU0_CA1)	32	R/W	0000_0000h	15.5.3/323
F000_5010	General Purpose Register (CAU0_CA2)	32	R/W	0000_0000h	15.5.3/323
F000_5014	General Purpose Register (CAU0_CA3)	32	R/W	0000_0000h	15.5.3/323
F000_5018	General Purpose Register (CAU0_CA4)	32	R/W	0000_0000h	15.5.3/323
F000_501C	General Purpose Register (CAU0_CA5)	32	R/W	0000_0000h	15.5.3/323
F000_5020	General Purpose Register (CAU0_CA6)	32	R/W	0000_0000h	15.5.3/323
F000_5024	General Purpose Register (CAU0_CA7)	32	R/W	0000_0000h	15.5.3/323
F000_5028	General Purpose Register (CAU0_CA8)	32	R/W	0000_0000h	15.5.3/323

15.5.1 Status Register (CAUx_CASR)

CASR contains the status and configuration for the CAU.

Address: F000_5000h base + 0h offset = F000_5000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VER				0											
W																
Reset	0	0	1	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
R	0														DPE	IC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CAUx_CASR field descriptions

Field	Description
31–28 VER	CAU Version Indicates CAU version. 0x1 Initial CAU version. 0x2 Second version, added support for SHA-256 algorithm (This is the value on this device).
27–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 DPE	DES Parity Error Indicates whether the DES parity error is detected. 0 No error detected. 1 DES key parity error detected.

Table continues on the next page...

CAUx_CASR field descriptions (continued)

Field	Description
0 IC	Illegal Command Indicates an illegal instruction has been executed. 0 No illegal commands issued. 1 Illegal command issued.

15.5.2 Accumulator (CAUx_CAA)

Commands use the CAU accumulator for storage of results and as an operand for the cryptographic algorithms.

Address: F000_5000h base + 4h offset = F000_5004h

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

CAUx_CAA field descriptions

Field	Description
ACC	Accumulator Stores results of various CAU commands.

15.5.3 General Purpose Register (CAUx_CA*n*)

The General Purpose Register is used in the CAU commands for storage of results and as operands for various cryptographic algorithms.

Address: F000_5000h base + 8h offset + (4d × i), where i=0d to 8d

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

CAUx_CA*n* field descriptions

Field	Description
CAn	General Purpose Registers Used by the CAU commands. Some cryptographic operations work with specific registers.

15.6 Functional description

This section discusses the programming model and operation of the CAU.

15.6.1 CAU programming model

The 4-entry FIFO is indirectly mapped into a 4-KB address space associated with the CAU located on this device. This address space is effectively split into two equal regions:

- one used to directly write commands for CAU load operations
- the other used to send commands and input operands for CAU loads

Data writes on the PPB are loaded into this FIFO and automatically converted into CAU load operands by the CAU translator. Data reads on the PPB are converted into CAU store register operations where the result is returned to the processor as the read data value.

The CAU requires a 15-bit command, and optionally, a 32-bit input operand, for each CAU load, PPB write. The 15-bit command includes the 9-bit opcode and other bits statically formed by the CAU translator logic controlling the CAU.

The following figure shows the 4-KB address space and the mapping of the CAU commands in this space.

NOTE

- Although the indirect store/load portion of the address space in the figure below shows only the indirect load/store commands, direct load commands can also be used in this space. However, it is more efficient to use the direct load portion of the address space.
- Accesses to the reserved space in the direct load space are terminated with an error, while accesses to the reserved space in the indirect load/store space are detected as an illegal CAU command. See CAU integrity checks for details.

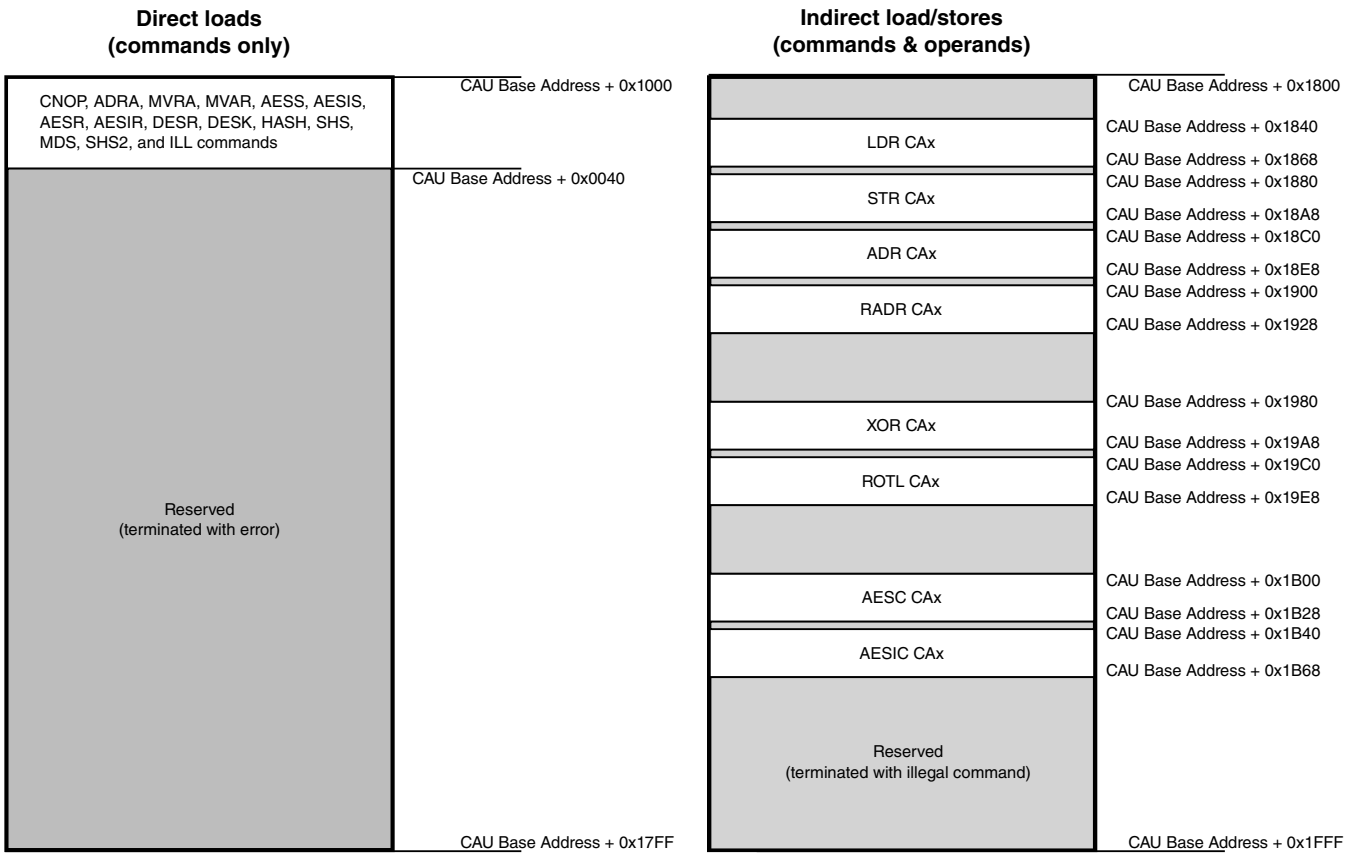


Figure 15-3. CAU memory map

15.6.1.1 Direct loads

The CAU supports writing multiple commands in each 32-bit direct write operation. Each 9-bit opcode also includes a valid bit. Therefore, one, two, or three commands can be transmitted in a single 32-bit PPB write. The following figure illustrates the accepted formats for the 32-bit CAU write data value:

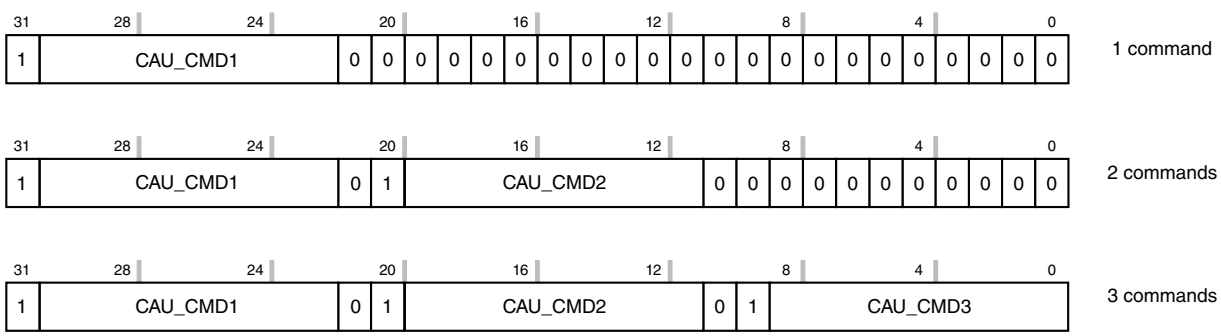


Figure 15-4. Direct loads

15.6.1.2 Indirect loads

For CAU load operations requiring a 32-bit input operand, the address contains the 9-bit opcode to be passed to the CAU while the data is the 32-bit operand. Specifically, the CAU address and data for these indirect writes is shown in the figure below.

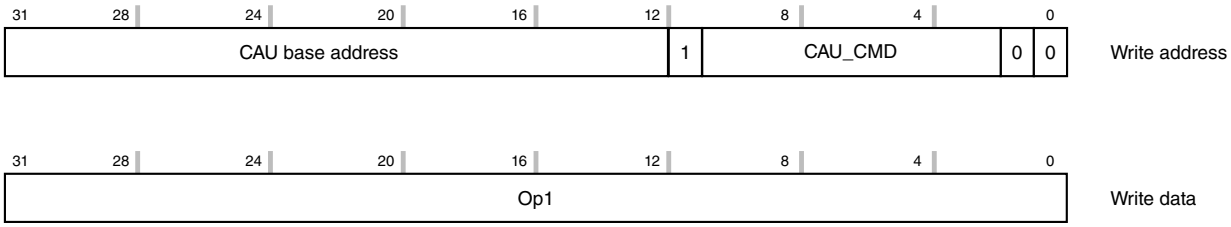


Figure 15-5. Indirect loads

15.6.1.3 Indirect stores

For CAU store operations, a PPB read is performed with the appropriate CAU store register opcode embedded in the address. This appears as another indirect command. The detail of Indirect stores is shown in the figure below.

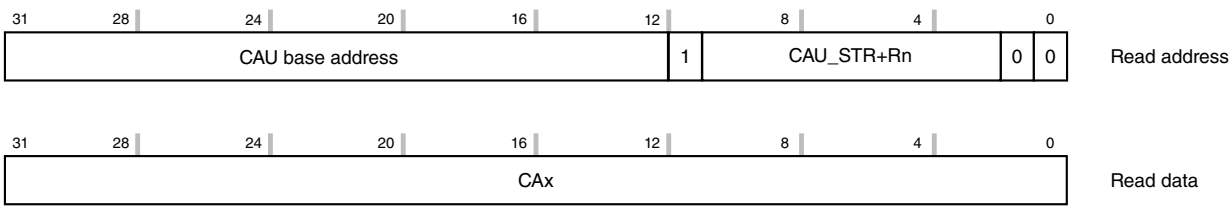


Figure 15-6. Indirect store

15.6.2 CAU integrity checks

If an illegal operation or access is attempted, the PPB bus cycle is terminated with an error response and the operation is aborted and not sent to the CAU.

The CAU performs a series of address and data integrity checks as described in the following sections. The results of these checks are logically summed together and, if appropriate, a PPB error termination is generated.

15.6.2.1 Address integrity checks

The CAU address checking includes the following. See Figure 15-3 for the CAU memory map details.

- Any CAU reference using a non-0-modulo-4 byte address (addr[1:0] ≠ 00) generates an error termination.
- For CAU writes:
  - Only the first 64 bytes of the 2-KB direct write address space can be referenced. Attempting to access regions beyond the first 64 bytes terminates with an error.
  - The second 2-KB space defines the indirect address-as-command region and any reference in this space is allowed by the CAU.

NOTE

The CAU contains error logic to detect any illegal command sent to it. Accordingly, there are address values in this upper 2-KB region of the address space that are passed to the CAU, and then detected as illegal commands. If the CAU detects an illegal command, it sets the CASR[IC] flag and performs no operation.

- For CAU reads:
  - Any attempted read from the first 2-KB region of the address space (an attempted direct read) is illegal and produces an error termination.
  - Within the second 2-KB region of the address space, i.e., addr[11] = 1, only a 64-byte space is treated as a legal CAU store operation. The allowable addresses are defined as:

    addr[11:0] = 1000_10xx_xx_00

    where the 4-bit xxxx value specifies the CAU register number. The CAU supports a subset of the allowable register numbers, 0x0 - 0xA. Attempting a store of a reserved register produces an undefined result.

15.6.2.2 Data integrity checks

Direct writes can send 1, 2, or 3 commands to the CAU in a single 32-bit transfer. As shown in Figure 15-4, the commands include a valid bit located at bits 31, 20, and 9 of the write data where:

- Bit 31 is the valid bit for the first command
- Bit 20 is the valid bit for the second command
- Bit 9 is the valid bit for the third command

The direct write data check validates the combination of these three valid bits. The following table presents the three legal states associated with these bits:

Value of bits 31, 20, and 9	Number of commands included
100	1
110	2
111	3

All other combinations of bits 31, 20, and 9 are illegal and generate an error termination.

15.6.3 CAU commands

The CAU supports the commands shown in the following table. All other encodings are reserved. The CASR[IC] bit is set if an undefined command is issued. A specific illegal command (ILL) is defined to allow software self-checking. Reserved commands must not be issued so as to ensure compatibility with future implementations.

The CMD field specifies the 9-bit CAU opcode for the operation command.

See Assembler equate values for a set of assembly constants used in the command descriptions here. If supported by the assembler, macros can also be created for each instruction. The value CA*x* should be interpreted as any CAU register (CASR, CAA, and CA*n*).

Table 15-2. CAU commands

Type	Command name	Description	CMD									Operation
			8	7	6	5	4	3	2	1	0	
Direct load	CNOP	No Operation	0x000									—
Indirect load	LDR	Load Reg	0x01					CAx				Op1 → CAx
Indirect store	STR	Store Reg	0x02					CAx				CAx → Result
Indirect load	ADR	Add	0x03					CAx				CAx + Op1 → CAx
Indirect load	RADR	Reverse and Add	0x04					CAx				CAx + ByteRev(Op1) → CAx
Direct load	ADRA	Add Reg to Acc	0x05					CAx				CAx + CAA → CAA
Indirect load	XOR	Exclusive Or	0x06					CAx				CAx ^ Op1 → CAx
Indirect load	ROTL	Rotate Left	0x07					CAx				(CAx <<< (Op1 % 32)) | (CAx >>> (32 - (Op1 % 32))) → CAx
Direct load	MVRA	Move Reg to Acc	0x08					CAx				CAx → CAA
Direct load	MVAR	Move Acc to Reg	0x09					CAx				CAA → CAx
Direct load	AESS	AES Sub Bytes	0x0A					CAx				SubBytes(CAx) → CAx
Direct load	AESIS	AES Inv Sub Bytes	0x0B					CAx				InvSubBytes(CAx) → CAx
Indirect load	AESC	AES Column Op	0x0C					CAx				MixColumns(CAx)^Op1→ CAx
Indirect load	AESIC	AES Inv Column Op	0x0D					CAx				InvMixColumns(CAx^Op1) → CAx

Table continues on the next page...

Table 15-2. CAU commands (continued)

Type	Command name	Description	CMD									Operation
			8	7	6	5	4	3	2	1	0	
Direct load	AESR	AES Shift Rows	0x0E0									ShiftRows(CA0-CA3) → CA0-CA3
Direct load	AESIR	AES Inv Shift Rows	0x0F0									InvShiftRows(CA0-CA3)→ CA0-CA3
Direct load	DESR	DES Round	0x10					IP	FP	KS[1:0]		DES Round(CA0-CA3) →CA0-CA3
Direct load	DESK	DES Key Setup	0x11					0	0	CP	DC	DES Key Op(CA0-CA1)→ CA0-CA1 Key Parity Error & CP → CASR[1]
Direct load	HASH	Hash Function	0x12					0	HF[2:0]			Hash Func(CA1-CA3)+CAA→ CAA
Direct load	SHS	Secure Hash Shift	0x130									CAA <<< 5→ CAA, CAA→CA0, CA0→CA1, CA1 <<< 30 → CA2, CA2→CA3, CA3→CA4
Direct load	MDS	Message Digest Shift	0x140									CA3→CAA, CAA→CA1, CA1→CA2, CA2→CA3,
Direct load	SHS2	Secure Hash Shift 2	0x150									CAA→CA0, CA0→CA1, CA1 → CA2, CA2→CA3, CA3 + CA8 →CA4, CA4 → CA5, CA5 → CA6, CA6 → CA7
Direct load	ILL	Illegal Command	0x1F0									0x1→CASR[IC]

15.6.3.1 Coprocessor No Operation (CNOP)

The CNOP command is the coprocessor no-op. It is issued by the CAU and consumes a location in the CAU FIFO, but has no effect on any CAU register.

15.6.3.2 Load Register (LDR)

The LDR command loads CAx with the source data specified by the write data.

15.6.3.3 Store Register (STR)

The STR command returns the value of CAx specified in the read address to the destination specified as read data.

15.6.3.4 Add to Register (ADR)

The ADR command adds the source operand specified by the write data to CAx and stores the result in CAx.

15.6.3.5 Reverse and Add to Register (RADR)

The RADR command performs a byte reverse on the source operand specified by the write data, adds that value to CAx, and stores the result in CAx. The table below shows an example.

Table 15-3. RADR command example

Operand	CAx before	CAx after
0x0102_0304	0xA0B0_C0D0	0xA4B3_C2D1

15.6.3.6 Add Register to Accumulator (ADRA)

The ADRA command adds CAx to CAA and stores the result in CAA.

15.6.3.7 Exclusive Or (XOR)

The XOR command performs an exclusive-or of the source operand specified by the write data with CAx and stores the result in CAx.

15.6.3.8 Rotate Left (ROTL)

ROTL rotates the CAx bits to the left with the result stored back to CAx. The number of bits to rotate is the value specified by the write data modulo 32.

15.6.3.9 Move Register to Accumulator (MVRA)

The MVRA command moves the value from the source register CAx to the destination register CAA.

15.6.3.10 Move Accumulator to Register (MVAR)

The MVAR command moves the value from source register CAA to the destination register CAx.

15.6.3.11 AES Substitution (AESS)

The AESS command performs the AES byte substitution operation on CAx and stores the result back to CAx.

15.6.3.12 AES Inverse Substitution (AESIS)

The AESIS command performs the AES inverse byte substitution operation on CAx and stores the result back to CAx.

15.6.3.13 AES Column Operation (AESC)

The AESC command performs the AES column operation on the contents of CAx. It then performs an exclusive-or of that result with the source operand specified by the write data and stores the result in CAx.

15.6.3.14 AES Inverse Column Operation (AESIC)

The AESIC command performs an exclusive-or operation of the source operand specified by the write data on the contents of CAx followed by the AES inverse mix column operation on that result and stores the result back in CAx.

15.6.3.15 AES Shift Rows (AESR)

The AESR command performs the AES shift rows operation on registers CA0, CA1, CA2, and CA3. The table below shows an example.

Table 15-4. AESR command example

Register	Before	After
CA0	0x0102_0304	0x0106_0B00
CA1	0x0506_0708	0x050A_0F04
CA2	0x090A_0B0C	0x090E_0308
CA3	0x0D0E_0F00	0x0D02_070C

Where:

- row 1 = CA0[31:24], CA1[31:24], CA2[31:24], CA3[31:24]
- row 2 = CA0[23:16], CA1[23:16], CA2[23:16], CA3[23:16]
- row 3 = CA0[15:8], CA1[15:8], CA2[15:8], CA3[15:8]
- row 4 = CA0[7:0], CA1[7:0], CA2[7:0], CA3[7:0]

15.6.3.16 AES Inverse Shift Rows (AESIR)

The AESIR command performs the AES inverse shift rows operation on registers CA0, CA1, CA2, and CA3. The table below shows an example.

Table 15-5. AESIR command example

Register	Before	After
CA0	0x0106_0B00	0x0102_0304
CA1	0x050A_0F04	0x0506_0708
CA2	0x090E_0308	0x090A_0B0C
CA3	0x0D02_070C	0x0D0E_0F00

Where:

- row 1 = CA0[31:24], CA1[31:24], CA2[31:24], CA3[31:24]
- row 2 = CA0[23:16], CA1[23:16], CA2[23:16], CA3[23:16]
- row 3 = CA0[15:8], CA1[15:8], CA2[15:8], CA3[15:8]
- row 4 = CA0[7:0], CA1[7:0], CA2[7:0], CA3[7:0]

15.6.3.17 DES Round (DESR)

The DESR command performs a round of the DES algorithm and a key schedule update with the following source and destination designations: CA0=C, CA1=D, CA2=L, CA3=R. If the IP bit is set, DES initial permutation performs on CA2 and CA3 before the round operation. If the FP bit is set, DES final permutation, that is, inverse initial permutation, performs on CA2 and CA3 after the round operation. The round operation

uses the source values from registers CA0 and CA1 for the key addition operation. The KSx field specifies the shift for the key schedule operation to update the values in CA0 and CA1. The following table defines the specific shift function performed based on the KSx field.

Table 15-6. Key shift function codes

KSx code	KSx define	Shift function
0	KSL1	Left 1
1	KSL2	Left 2
2	KSR1	Right 1
3	KSR2	Right 2

15.6.3.18 DES Key setup (DESK)

The DESK command performs the initial key transformation, permuted choice 1, defined by the DES algorithm on CA0 and CA1 with CA0 containing bits 1–32 of the key and CA1 containing bits 33–64 of the key[1] . If the DC bit is set, no shift operation performs and the values C_0 and D_0 store back to CA0 and CA1, respectively. The DC bit must be set for decrypt operations. If the DC bit is not set, a left shift by one also occurs and the values C_1 and D_1 store back to CA0 and CA1, respectively. The DC bit should be cleared for encrypt operations. If the CP bit is set and a key parity error is detected, CASR[DPE] bit is set; otherwise, it is cleared.

15.6.3.19 Hash Function (HASH)

The HASH command performs a hashing operation on a set of registers and adds that result to the value in CAA and stores the result in CAA. The specific hash function performed is based on the HFx field as defined in the table below.

This table uses the following terms:

- ROTR^n(CAx): rotate CAx register right *n* times
- SHR^n(CAx): shift CAx register right *n* times

Table 15-7. Hash Function codes

HFx code	HFx define	Hash Function	Hash logic
0	HFF	MD5 F()	(CA1 & CA2) | ($\overline{\text{CA1}}$ & CA3)
1	HFG	MD5 G()	(CA1 & CA3) | (CA2 & $\overline{\text{CA3}}$)

Table continues on the next page...

1. The DES algorithm numbers the most significant bit of a block as bit 1 and the least significant as bit 64.

Table 15-7. Hash Function codes (continued)

HFx code	HFx define	Hash Function	Hash logic
2	HFH	MD5 H(), SHA Parity()	CA1 ^ CA2 ^ CA3
3	HFI	MD5 I()	CA2 ^ (CA1 | $\overline{CA3}$)
4	HFC	SHA Ch()	(CA1 & CA2) ^ ($\overline{CA1}$ & CA3)
5	HFM	SHA Maj()	(CA1 & CA2) ^ (CA1 & CA3) ^ (CA2 & CA3)
6	HF2C	SHA-256 Ch()	(CA4 & CA5) ^ ($\overline{CA1}$ & CA6)
7	HF2M	SHA-256 Maj()	(CA0 & CA1) ^ (CA0 & CA2) ^ (CA1 & CA2)
8	HF2S	SHA-256 Sigma 0	$ROTR^{2}$(CA0) ^ $ROTR^{13}$(CA0) ^ $ROTR^{22}$(CA0)
9	HF2T	SHA-256 Sigma 1	$ROTR^{6}$(CA4) ^ $ROTR^{11}$(CA4) ^ $ROTR^{25}$(CA4)
A	HF2U	SHA-256 Sigma 0	$ROTR^{7}$(CA8) ^ $ROTR^{18}$(CA8) ^ SHR^{3}(CA8)
B	HF2V	SHA-256 Sigma 1	$ROTR^{17}$(CA8) ^ $ROTR^{19}$(CA8) ^ SHR^{10}(CA8)

15.6.3.20 Secure Hash Shift (SHS)

The SHS command does a set of parallel register-to-register move and shift operations for implementing SHA-1. The following source and destination assignments are made:

Register	Value prior to command	Value after command executes
CA4	CA4	CA3
CA3	CA3	CA2
CA2	CA2	CA1<<<30
CA1	CA1	CA0
CA0	CA0	CAA
CAA	CAA	CAA<<<5

15.6.3.21 Message Digest Shift (MDS)

The MDS command does a set of parallel register-to-register move operations for implementing MD5. The following source and destination assignments are made:

Register	Value prior to command	Value after command executes
CA3	CA3	CA2
CA2	CA2	CA1
CA1	CA1	CAA
CAA	CAA	CA3

15.6.3.22 Secure Hash Shift 2 (SHS2)

The SHS2 command does an addition and a set of register to register moves in parallel for implementing SHA-256. The following source and destination assignments are made:

Register	Value prior to command	Value after command executes
CA7	CA7	CA6
CA6	CA6	CA5
CA5	CA5	CA4
CA4	CA4	CA3+CA8
CA3	CA3	CA2
CA2	CA2	CA1
CA1	CA1	CA0
CA0	CA0	CAA

15.6.3.23 Illegal command (ILL)

The ILL command is a specific illegal command that sets CASR[IC]. All other illegal commands are reserved for use in future implementations.

15.7 Application/initialization information

This section discusses how to initialize and use the CAU.

15.7.1 Code example

A code fragment is shown below as an example of how the CAU is used. This example shows the round function of the AES algorithm. Core registers are defined as follows:

- R1 points to the key schedule
- R3 contains 3 direct CAU commands
- R8 contains 2 direct CAU commands
- R9 contains an indirect CAU command
- FP points to the CAU indirect command address space
- IP points to the CAU direct command space

```
movw    fp, #:lower16:MMCAU_PPB_INDIRECT            @ fp -> MMCAU_PPB_INDIRECT
movt    fp, #:upper16:MMCAU_PPB_INDIRECT
movw    ip, #:lower16:MMCAU_PPB_DIRECT              @ ip -> MMCAU_PPB_DIRECT
movt    ip, #:upper16:MMCAU_PPB_DIRECT
```

```
# r3 = mmcau_3_cmds(AESS+CA0,AESS+CA1,AESS+CA2)
    movw    r3, #:lower16:(0x80100200+(AESS+CA0)<<22+(AESS+CA1)<<11+AESS+CA2)
    movt    r3, #:upper16:(0x80100200+(AESS+CA0)<<22+(AESS+CA1)<<11+AESS+CA2)

# r8 = mmcau_2_cmds(AESS+CA3,AESR)
    movw    r8, #:lower16:(0x80100000+(AESS+CA3)<<22+(AESR)<<11)
    movt    r8, #:upper16:(0x80100000+(AESS+CA3)<<22+(AESR)<<11)

    add     r9, fp, $((AESC+CA0)<<2)                        @ r9 = mmcau_cmd(AESC+CA0)

    str     r3, [ip]                                      @ sub bytes w0, w1, w2
    str     r8, [ip]                                      @ sub bytes w3, shift rows
    ldmia   r1!, {r4-r7}                                  @ get next 4 keys; r1++
    stmia   r9, {r4-r7}                                   @ mix columns, add keys
```

15.7.2 Assembler equate values

The following equates ease programming of the CAU.

```
; CAU Registers (CAx)
     .set CASR,0x0
     .set CAA,0x1
     .set CA0,0x2
     .set CA1,0x3
     .set CA2,0x4
     .set CA3,0x5
     .set CA4,0x6
     .set CA5,0x7
     .set CA6,0x8
     .set CA7,0x9
     .set CA8,0xA
; CAU Commands
     .set CNOP,0x000
     .set LDR,0x010
     .set STR,0x020
     .set ADR,0x030
     .set RADR,0x040
     .set ADRA,0x050
     .set XOR,0x060
     .set ROTL,0x070
     .set MVRA,0x080
     .set MVAR,0x090
     .set AESS,0x0A0
     .set AESIS,0x0B0
     .set AESC,0x0C0
     .set AESIC,0x0D0
     .set AESR,0x0E0
     .set AESIR,0x0F0
     .set DESR,0x100
     .set DESK,0x110
     .set HASH,0x120
     .set SHS,0x130
     .set MDS,0x140
     .set SHS2,0x150
     .set ILL,0x1F0
; DESR  Fields
     .set IP,0x08         ; initial permutation
     .set FP,0x04         ; final permutation
     .set KSL1,0x00       ; key schedule left 1 bit
     .set KSL2,0x01       ; key schedule left 2 bits
     .set KSR1,0x02       ; key schedule right 1 bit
     .set KSR2,0x03       ; key schedule right 2 bits
; DESK Field
     .set DC,0x01         ; decrypt key schedule
```

```
     .set CP,0x02          ; check parity
; HASH Functions Codes
     .set HFF,0x0          ; MD5 F() CA1&CA2 | ~CA1&CA3
     .set HFG,0x1          ; MD5 G() CA1&CA3 | CA2&~CA3
     .set HFH,0x2          ; MD5 H(), SHA Parity() CA1^CA2^CA3
     .set HFI,0x3          ; MD5 I()  CA2^(CA1|~CA3)
     .set HFC,0x4          ; SHA Ch() CA1&CA2 ^ ~CA1&CA3
     .set HFM,0x5          ; SHA Maj() CA1&CA2 ^ CA1&CA3 ^ CA2&CA3
     .set HF2C,0x6         ; SHA-256 Ch() CA4&CA5 ^ ~CA4&CA6
     .set HF2M,0x7         ; SHA-256 Maj() CA0&CA1 ^ CA0&CA2 ^ CA1&CA2
     .set HF2S,0x8         ; SHA-256 Sigma 0 ROTR2(CA0)^ROTR13(CA0)^ROTR22(CA0)
     .set HF2T,0x9         ; SHA-256 Sigma 1 ROTR6(CA4)^ROTR11(CA4)^ROTR25(CA4)
     .set HF2U,0xA         ; SHA-256 sigma 0 ROTR7(CA8)^ROTR18(CA8)^SHR3(CA8)
     .set HF2V,0xB         ; SHA-256 sigma 1 ROTR17(CA8)^ROTR19(CA8)^SHR10(CA8)
```

Chapter 16
Comparator (CMP)

16.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal is provided by the 6-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed. The 64-tap resistor ladder network divides the supply reference V_{in} into 64 voltage levels. A 6-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/64$. V_{in} can be selected from two voltage sources, V_{in1} and V_{in2}. The 6-bit DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

16.1.1 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control

- Selectable interrupt on rising-edge, falling-edge, or both rising or falling edges of the comparator output
- Selectable inversion on comparator output
- Capability to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered:
    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Two software selectable performance levels:
  - Shorter propagation delay at the expense of higher power
  - Low power, with longer propagation delay
- DMA transfer support
  - A comparison event can be selected to trigger a DMA transfer
- Functional in all modes of operation
- The window and filter functions are not available in the following modes:
  - Stop
  - VLPS
  - LLS
  - VLLSx

16.1.2 6-bit DAC key features

The 6-bit DAC has the following features:

- 6-bit resolution
- Selectable supply reference source
- Power Down mode to conserve power when not in use
- Option to route the output to internal comparator input

16.1.3 ANMUX key features

The ANMUX has the following features:

- Two 8-to-1 channel mux
- Operational over the entire supply range

16.1.4 CMP, DAC and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

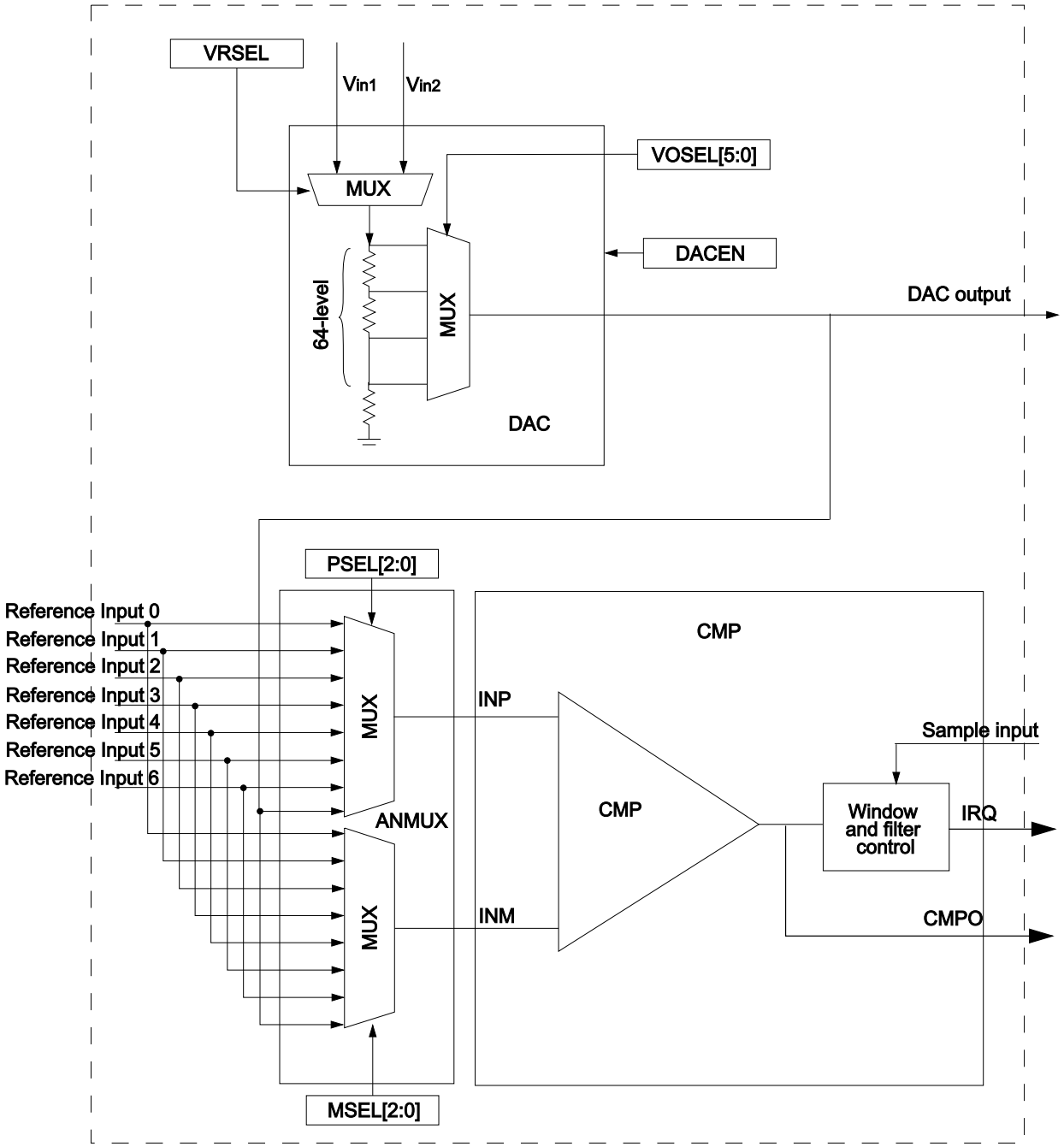


Figure 16-1. CMP, DAC and ANMUX block diagram

16.1.5 CMP block diagram

The following figure shows the block diagram for the CMP module.

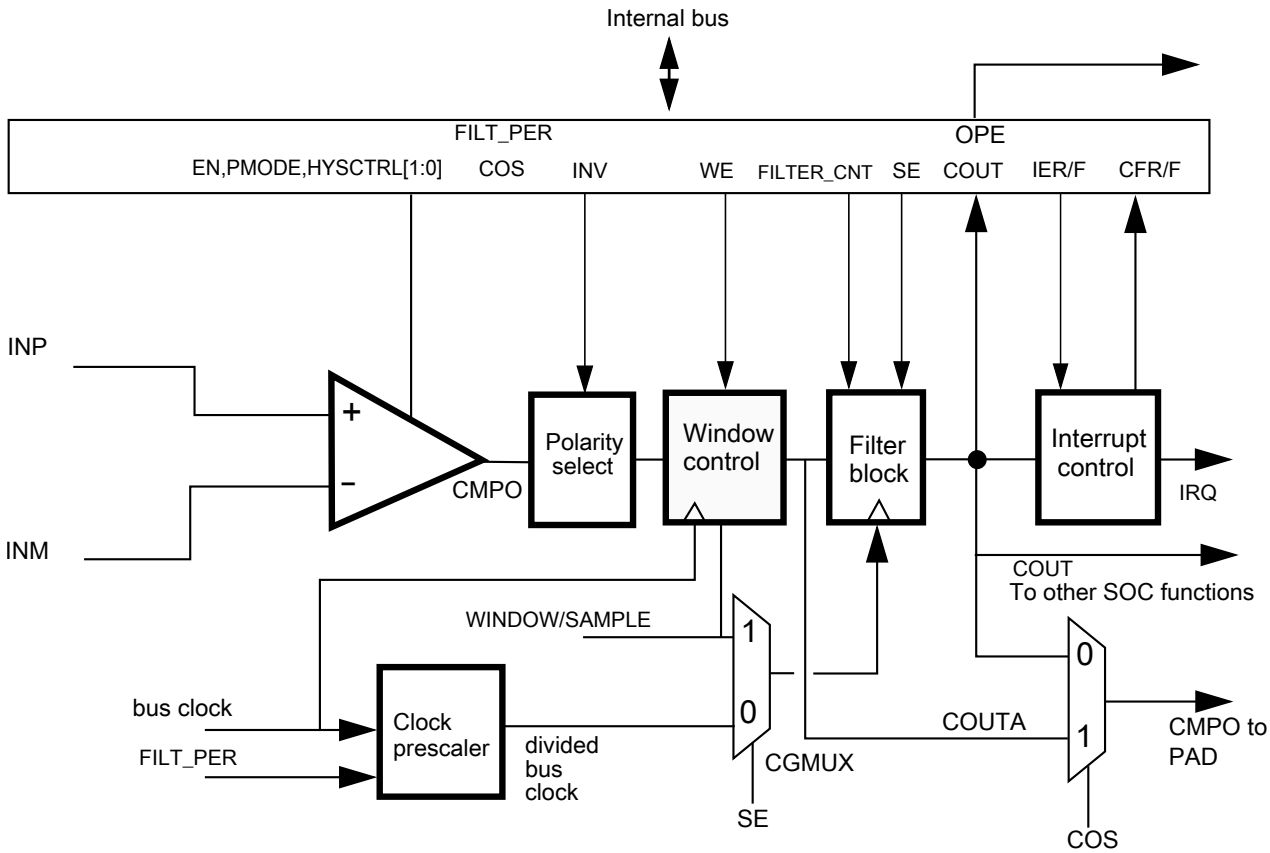


Figure 16-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter block is bypassed when not in use.
- The Filter block acts as a simple sampler if the filter is bypassed and CR0[FILTER_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and CR0[FILTER_CNT] is set greater than 0x01.
  - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
  - If CR1[SE] = 0, the divided bus clock is used as sampling clock

- If enabled, the Filter block will incur up to one bus clock additional latency penalty on COUT due to the fact that COUT, which is crossing clock domain boundaries, must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

16.2 Memory map/register definitions

CMP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_E000	CMP Control Register 0 (CMP0_CR0)	8	R/W	00h	16.2.1/344
4006_E001	CMP Control Register 1 (CMP0_CR1)	8	R/W	00h	16.2.2/345
4006_E002	CMP Filter Period Register (CMP0_FPR)	8	R/W	00h	16.2.3/347
4006_E003	CMP Status and Control Register (CMP0_SCR)	8	R/W	00h	16.2.4/347
4006_E004	DAC Control Register (CMP0_DACCR)	8	R/W	00h	16.2.5/348
4006_E005	MUX Control Register (CMP0_MUXCR)	8	R/W	00h	16.2.6/349
400E_F000	CMP Control Register 0 (CMP1_CR0)	8	R/W	00h	16.2.1/344
400E_F001	CMP Control Register 1 (CMP1_CR1)	8	R/W	00h	16.2.2/345
400E_F002	CMP Filter Period Register (CMP1_FPR)	8	R/W	00h	16.2.3/347
400E_F003	CMP Status and Control Register (CMP1_SCR)	8	R/W	00h	16.2.4/347
400E_F004	DAC Control Register (CMP1_DACCR)	8	R/W	00h	16.2.5/348
400E_F005	MUX Control Register (CMP1_MUXCR)	8	R/W	00h	16.2.6/349

16.2.1 CMP Control Register 0 (CMPx_CR0)

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0	0	HYSTCTR	
Write								
Reset	0	0	0	0	0	0	0	0

CMPx_CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count

Table continues on the next page...

CMPx_CR0 field descriptions (continued)

Field	Description
	Represents the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency, see the Functional description. 000 Filter is disabled. If SE = 1, then COUT is a logic 0. This is not a legal state, and is not recommended. If SE = 0, COUT = COUTA. 001 One sample must agree. The comparator output is simply sampled. 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level are device-specific. See the Data Sheet of the device for the exact values. 00 Level 0 01 Level 1 10 Level 2 11 Level 3

16.2.2 CMP Control Register 1 (CMPx_CR1)

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SE	WE	TRIGM	PMODE	INV	COS	OPE	EN
Reset	0	0	0	0	0	0	0	0

CMPx_CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing 1s to both field locations because this "11" case is reserved and may change in future implementations. 0 Sampling mode is not selected. 1 Sampling mode is selected.
6 WE	Windowing Enable

Table continues on the next page...

CMPx_CR1 field descriptions (continued)

Field	Description
	At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing 1s to both field locations because this "11" case is reserved and may change in future implementations. 0 Windowing mode is not selected. 1 Windowing mode is selected.
5 TRIGM	Trigger Mode Enable CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1. In addition, the CMP should be enabled. If the DAC is to be used as a reference to the CMP, it should also be enabled. CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare. Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received. See the chip configuration for details about the external timer resource. 0 Trigger mode is disabled. 1 Trigger mode is enabled.
4 PMODE	Power Mode Select See the electrical specifications table in the device Data Sheet for details. 0 Low-Speed (LS) Comparison mode selected. In this mode, CMP has slower output propagation delay and lower current consumption. 1 High-Speed (HS) Comparison mode selected. In this mode, CMP has faster output propagation delay and higher current consumption.
3 INV	Comparator INVERT Allows selection of the polarity of the analog comparator function. It is also driven to the COUT output, on both the device pin and as SCR[COUT], when OPE=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select 0 Set the filtered comparator output (CMPO) to equal COUT. 1 Set the unfiltered comparator output (CMPO) to equal COUTA.
1 OPE	Comparator Output Pin Enable 0 CMPO is not available on the associated CMPO output pin. If the comparator does not own the pin, this field has no effect. 1 CMPO is available on the associated CMPO output pin. The comparator output (CMPO) is driven out on the associated CMPO output pin if the comparator owns the pin. If the comparator does not own the field, this bit has no effect.
0 EN	Comparator Module Enable Enables the Analog Comparator module. When the module is not enabled, it remains in the off state, and consumes no power. When the user selects the same input from analog mux to the positive and negative port, the comparator is disabled automatically.

Table continues on the next page...

CMPx_CR1 field descriptions (continued)

Field	Description
	0 Analog Comparator is disabled. 1 Analog Comparator is enabled.

16.2.3 CMP Filter Period Register (CMPx_FPR)

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	FILT_PER							
Reset	0	0	0	0	0	0	0	0

CMPx_FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when CR1[SE]=0. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional description. This field has no effect when CR1[SE]=1. In that case, the external SAMPLE signal is used to determine the sampling period.

16.2.4 CMP Status and Control Register (CMPx_SCR)

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read	0	DMAEN	0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

CMPx_SCR field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DMAEN	DMA Enable Control Enables the DMA transfer triggered from the CMP module. When this field is set, a DMA request is asserted when CFR or CFF is set. 0 DMA is disabled. 1 DMA is enabled.

Table continues on the next page...

CMPx_SCR field descriptions (continued)

Field	Description
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
3 IEF	Comparator Interrupt Enable Falling Enables the CFF interrupt from the CMP. When this field is set, an interrupt will be asserted when CFF is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 CFR	Analog Comparator Flag Rising Detects a rising-edge on COUT, when set, during normal operation. CFR is cleared by writing 1 to it. During Stop modes, CFR is level sensitive . 0 Rising-edge on COUT has not been detected. 1 Rising-edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling Detects a falling-edge on COUT, when set, during normal operation. CFF is cleared by writing 1 to it. During Stop modes, CFF is level sensitive . 0 Falling-edge on COUT has not been detected. 1 Falling-edge on COUT has occurred.
0 COUT	Analog Comparator Output Returns the current value of the Analog Comparator output, when read. The field is reset to 0 and will read as CR1[INV] when the Analog Comparator module is disabled, that is, when CR1[EN] = 0. Writes to this field are ignored.

16.2.5 DAC Control Register (CMPx_DACCR)

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read Write	DACEN	VRSEL	VOSEL					
Reset	0	0	0	0	0	0	0	0

CMPx_DACCR field descriptions

Field	Description
7 DACEN	DAC Enable Enables the DAC. When the DAC is disabled, it is powered down to conserve power.

Table continues on the next page...

CMPx_DACCR field descriptions (continued)

Field	Description
	0 DAC is disabled. 1 DAC is enabled.
6 VRSEL	Supply Voltage Reference Source Select 0 V_{in1} is selected as resistor ladder network supply reference. 1 V_{in2} is selected as resistor ladder network supply reference.
VOSEL	DAC Output Voltage Select Selects an output voltage from one of 64 distinct levels. `DACO = (V in /64) * (VOSEL[5:0] + 1)` , so the DACO range is from V_{in} /64 to V_{in} .

16.2.6 MUX Control Register (CMPx_MUXCR)

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read Write	PSTM	0	PSEL			MSEL		
Reset	0	0	0	0	0	0	0	0

CMPx_MUXCR field descriptions

Field	Description
7 PSTM	Pass Through Mode Enable This bit is used to enable to MUX pass through mode. Pass through mode is always available but for some devices this feature must be always disabled due to the lack of package pins. 0 Pass Through Mode is disabled. 1 Pass Through Mode is enabled.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–3 PSEL	Plus Input Mux Control Determines which input is selected for the plus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams. **NOTE:** When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7

Table continues on the next page...

CMPx_MUXCR field descriptions (continued)

Field	Description
MSEL	Minus Input Mux Control Determines which input is selected for the minus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams. **NOTE:** When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7

16.3 Functional description

The CMP module can be used to compare two analog input voltages applied to INP and INM.

CMPO is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CR1[INV] = 1.

SCR[IER] and SCR[IEF] are used to select the condition which will cause the CMP module to assert an interrupt to the processor. SCR[CFF] is set on a falling-edge and SCR[CFR] is set on rising-edge of the comparator output. The optionally filtered CMPO can be read directly through SCR[COUT].

16.3.1 CMP functional modes

There are the following main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, CR0[FILTER_CNT], can be clocked from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CR1[WE]. When set, the comparator output is sampled only when WINDOW=1. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 16-1. Comparator sample/filter controls

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
1	0	X	X	X	X	**Disabled** See the Disabled mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode** See the Continuous mode (#s 2A & 2B).
2B	1	0	0	X	0x00	
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode** See the Sampled, Non-Filtered mode (#s 3A & 3B).
3B	1	0	0	0x01	> 0x00	
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode** See the Sampled, Filtered mode (#s 4A & 4B).
4B	1	0	0	> 0x01	> 0x00	
5A	1	1	0	0x00	X	**Windowed mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA. See the Windowed mode (#s 5A & 5B).
5B	1	1	0	X	0x00	
6	1	1	0	0x01	0x01–0xFF	**Windowed/Resampled mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled on an interval determined by FILT_PER to generate COUT. See the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01–0xFF	**Windowed/Filtered mode**

Table continues on the next page...

Table 16-1. Comparator sample/filter controls (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
						Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. See the Windowed/Filtered mode (#7).
All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] are illegal.						

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it must be configured to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings must be changed only after setting CR1[SE]=0 and CR0[FILTER_CNT]=0x00. This resets the filter to a known state.

16.3.1.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

16.3.1.2 Continuous mode (#s 2A & 2B)

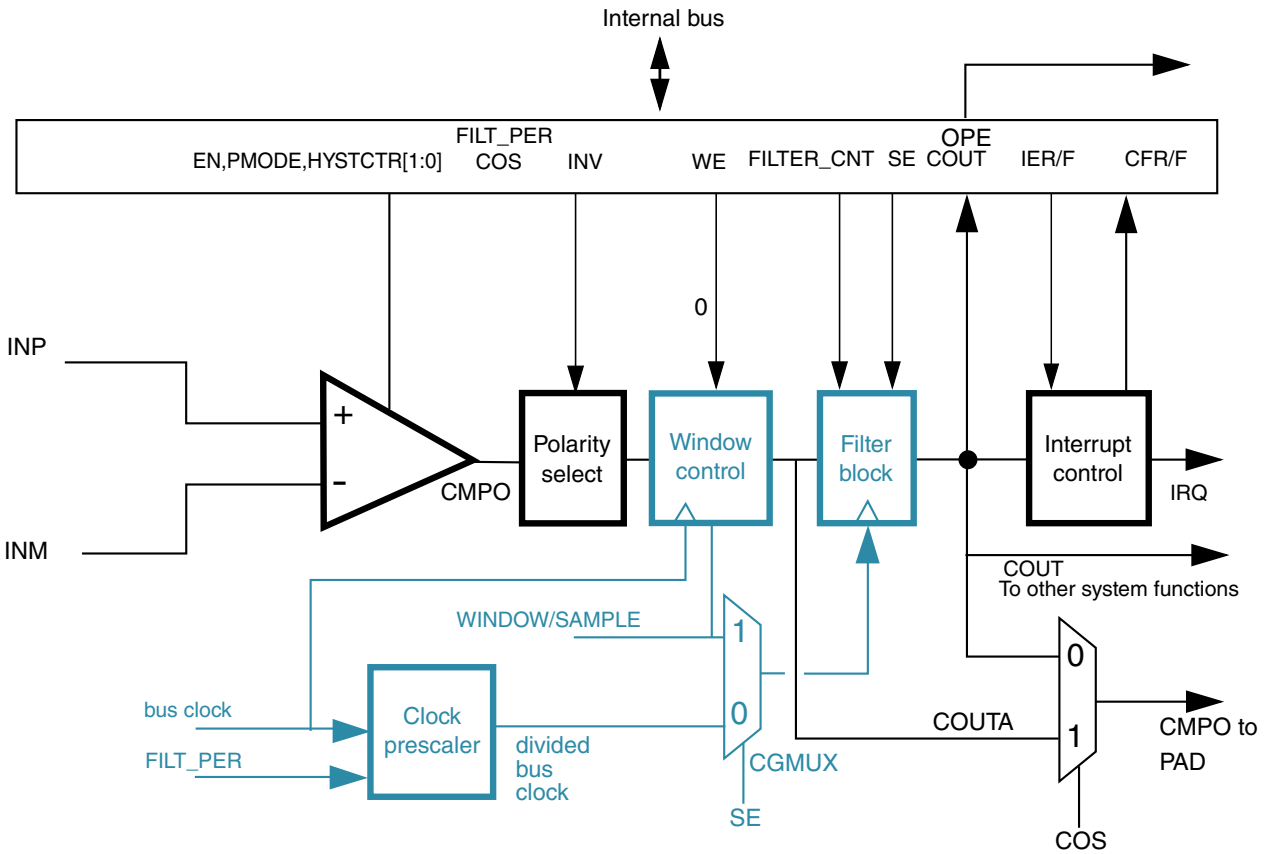


Figure 16-3. Comparator operation in Continuous mode

NOTE

See the chip configuration section for the source of sample/ window input.

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed. SCR[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational unclocked mode. COUT and COUTA are identical.

For control configurations which result in disabling the filter block, see the Filter Block Bypass Logic diagram.

16.3.1.3 Sampled, Non-Filtered mode (#s 3A & 3B)

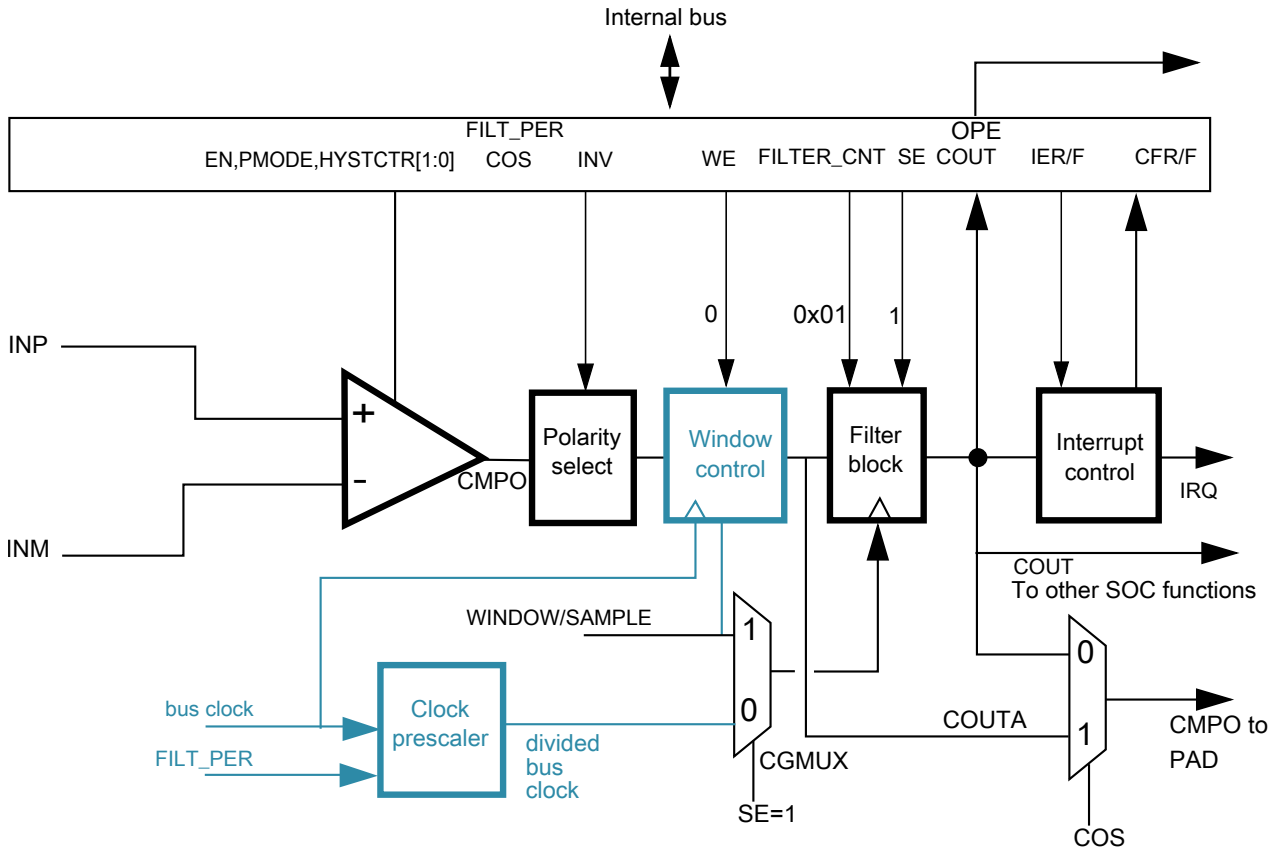


Figure 16-4. Sampled, Non-Filtered (# 3A): sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising-edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

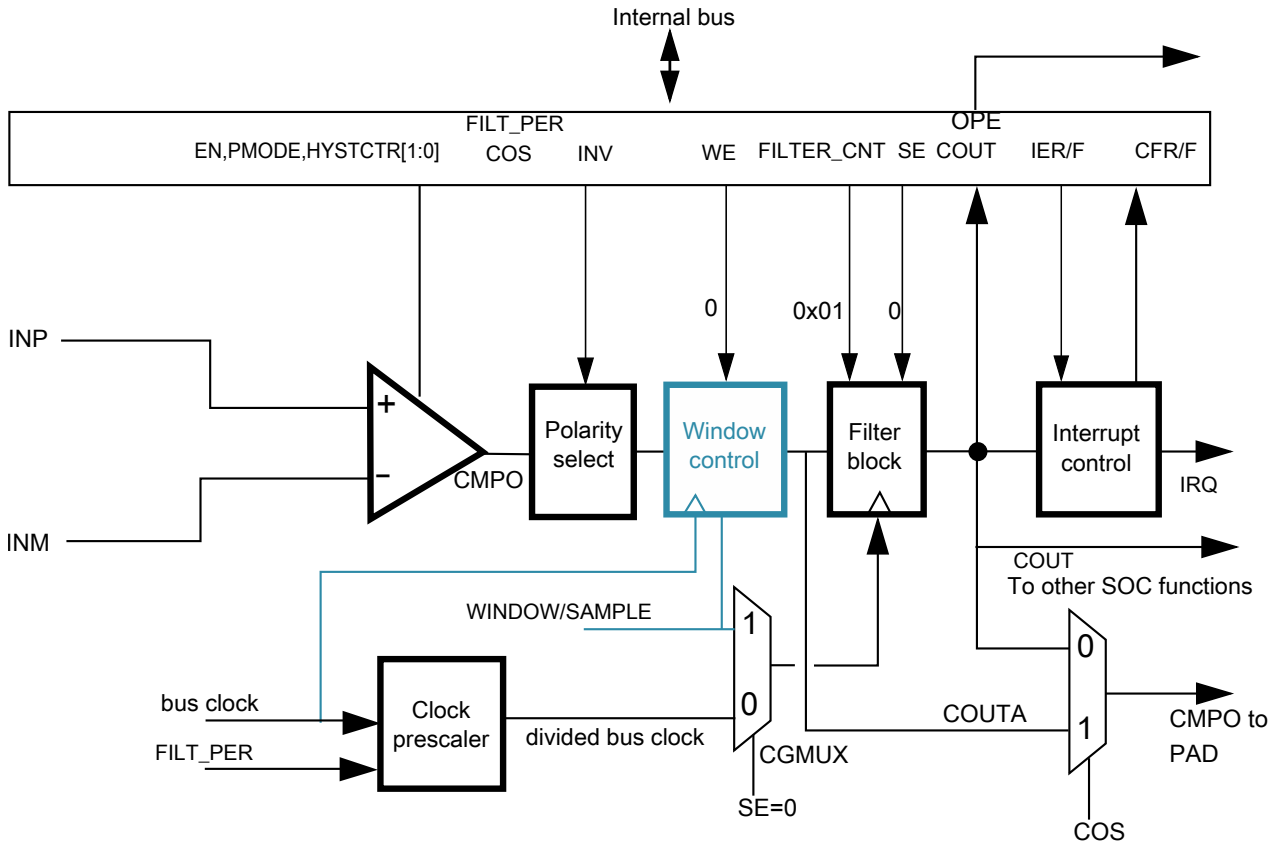


Figure 16-5. Sampled, Non-Filtered (# 3B): sampling interval internally derived

16.3.1.4 Sampled, Filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that, now, CR0[FILTER_CNT]>1, which activates filter operation.

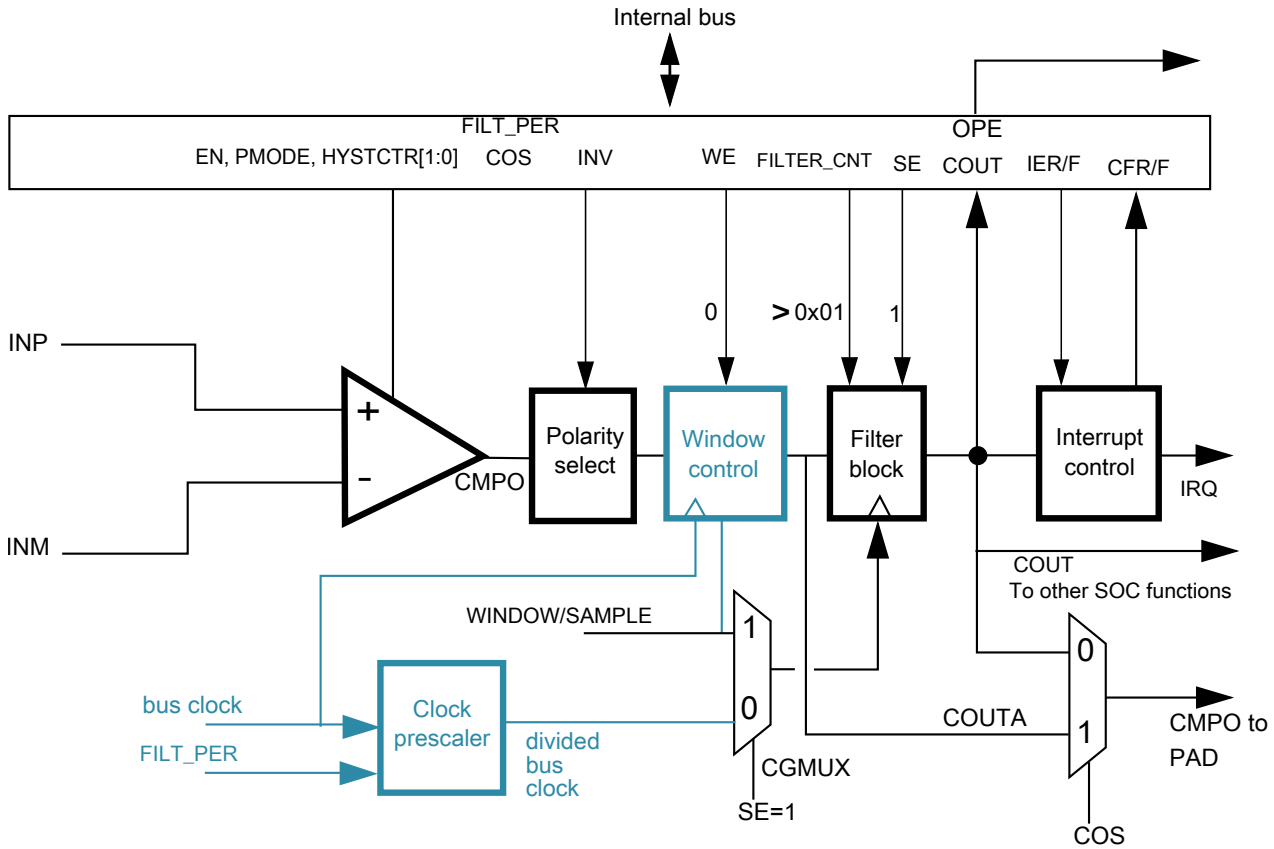


Figure 16-6. Sampled, Filtered (# 4A): sampling point externally driven

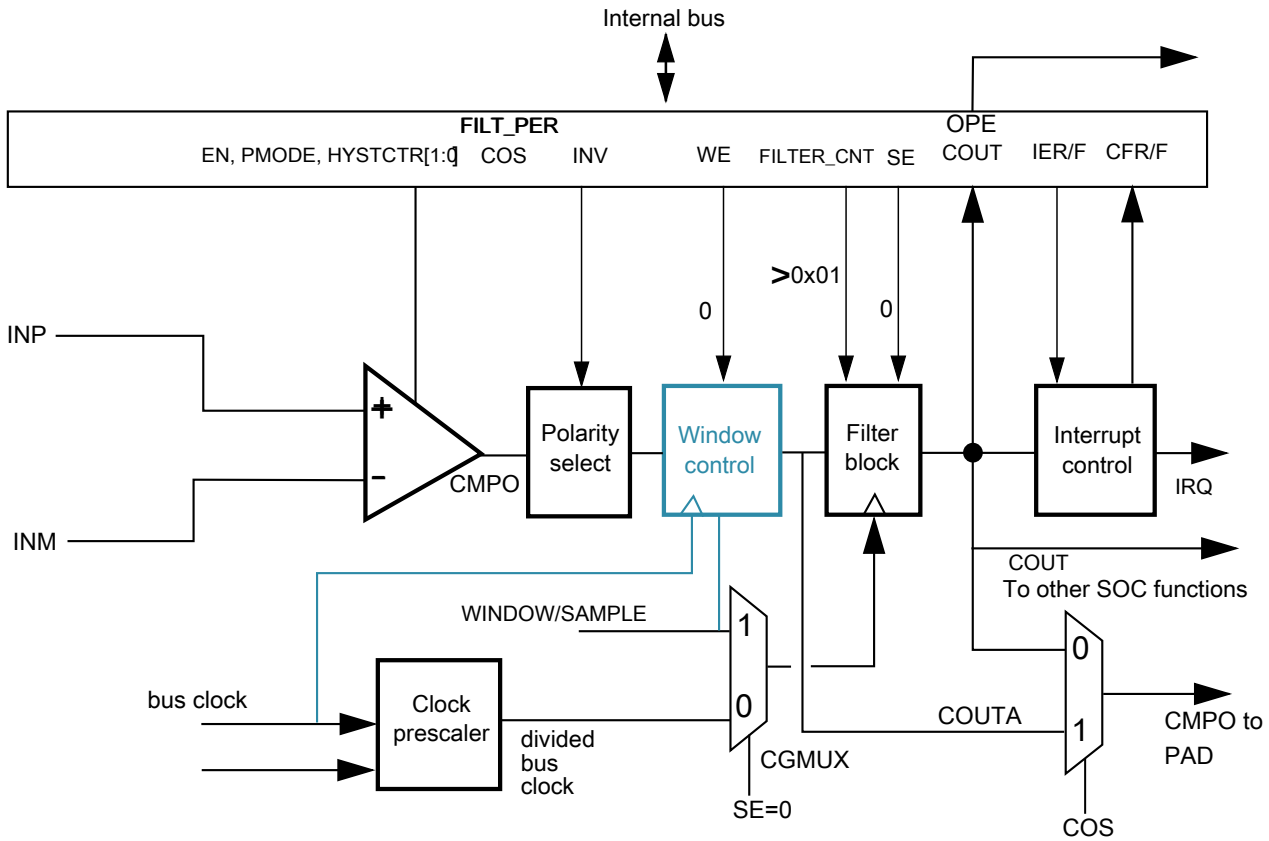


Figure 16-7. Sampled, Filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, CR0[FILTER_CNT]>1, which activates filter operation.

16.3.1.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the Windowed mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

NOTE

The analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

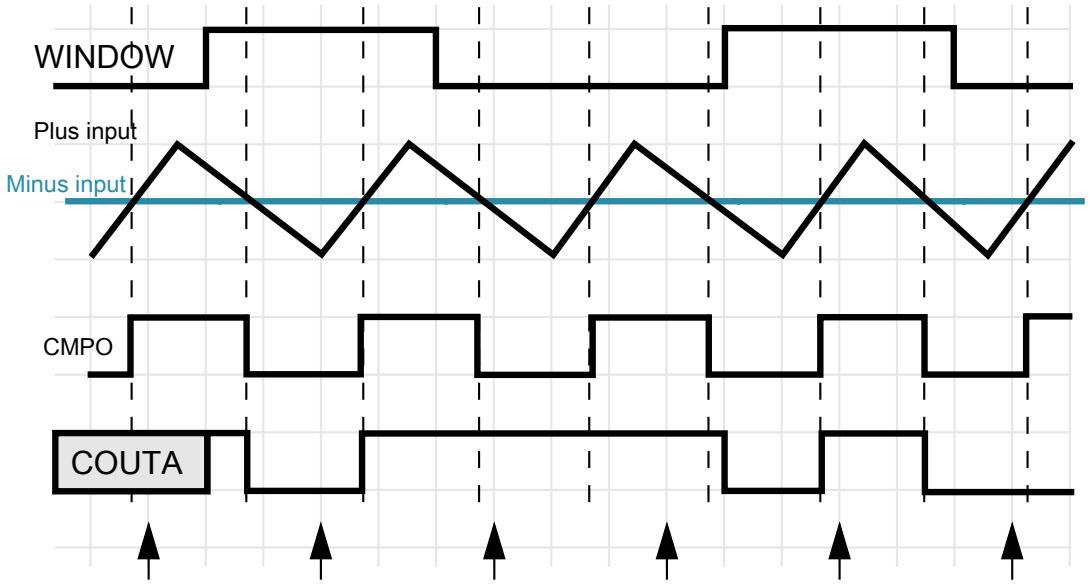


Figure 16-8. Windowed mode operation

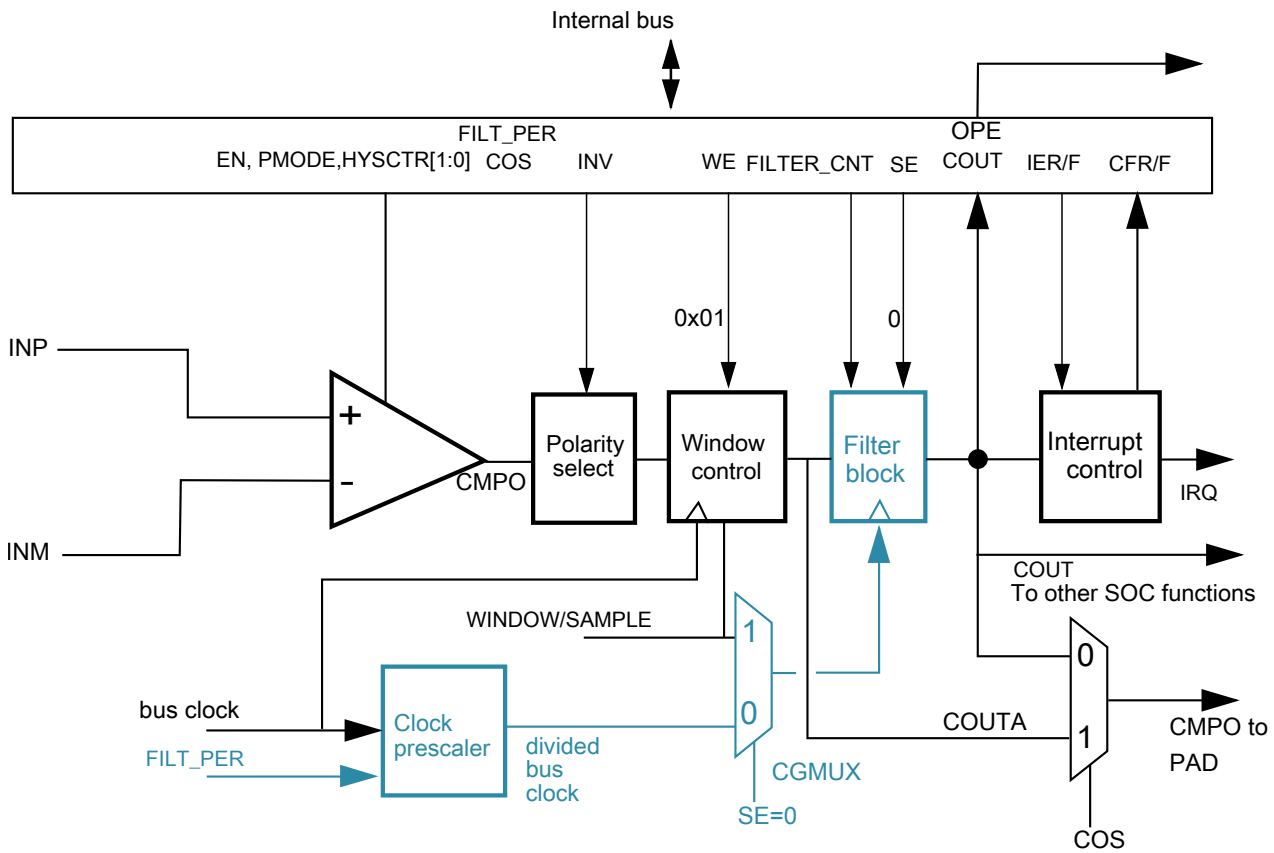


Figure 16-9. Windowed mode

For control configurations which result in disabling the filter block, see Filter Block Bypass Logic diagram.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

16.3.1.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 16-8, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows in the figure. Again, prop delays and latency are ignored for the sake of clarity.

This example was generated solely to demonstrate operation of the comparator in windowed/resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

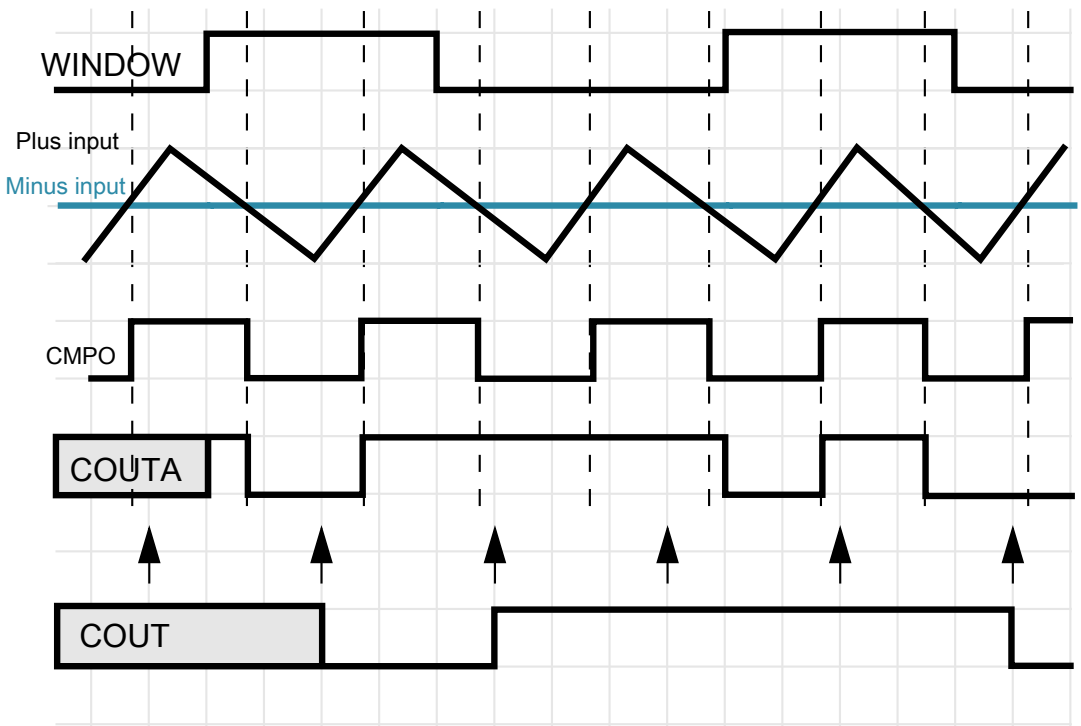


Figure 16-10. Windowed/resampled mode operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of CR0[FILTER_CNT] must be 1.

16.3.1.7 Windowed/Filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it uses both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + ((CR0[FILTER_CNT] * FPR[FILT_PER]) + 1) * bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

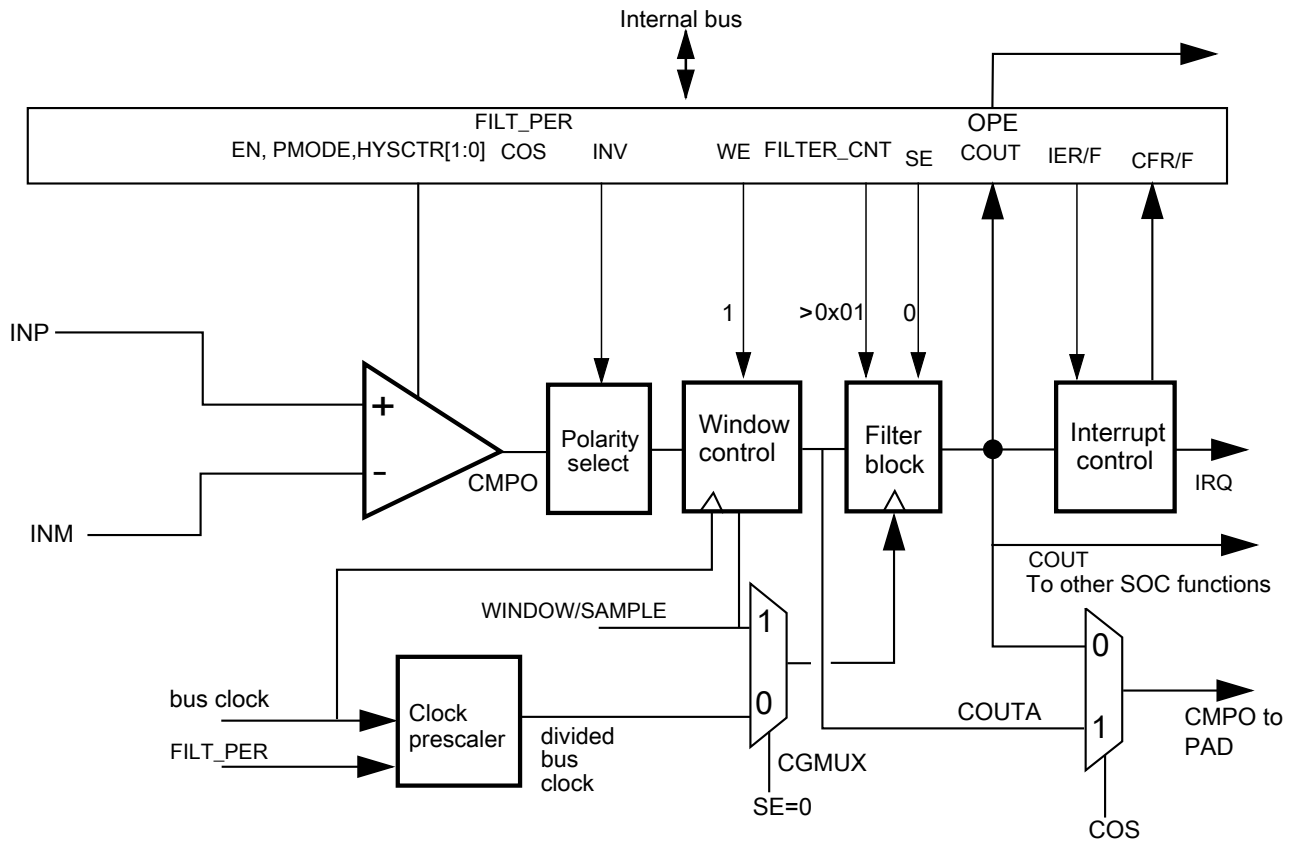


Figure 16-11. Windowed/Filtered mode

16.3.2 Power modes

16.3.2.1 Wait mode operation

During Wait and VLPW modes, the CMP, if enabled, continues to operate normally and a CMP interrupt can wake the MCU.

16.3.2.2 Stop mode operation

Depending on clock restrictions related to the MCU core or core peripherals, the MCU is brought out of stop when a compare event occurs and the corresponding interrupt is enabled. Similarly, if CR1[OPE] is enabled, the comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. In Stop modes, the comparator can be operational in both:

- High-Speed (HS) Comparison mode when CR1[PMODE] = 1
- Low-Speed (LS) Comparison mode when CR1[PMODE] = 0

It is recommended to use the LS mode to minimize power consumption.

If stop is exited with a reset, all comparator registers are put into their reset state.

16.3.2.3 Low-Leakage mode operation

When the chip is in Low-Leakage modes:

- The CMP module is partially functional and is limited to Low-Speed mode, regardless of CR1[PMODE] setting
- Windowed, Sampled, and Filtered modes are not supported
- The CMP output pin is latched and does not reflect the compare output state.

The positive- and negative-input voltage can be supplied from external pins or the DAC output. The MCU can be brought out of the Low-Leakage mode if a compare event occurs and the CMP interrupt is enabled. After wakeup from low-leakage modes, the CMP module is in the reset state except for SCR[CFF] and SCR[CFR].

16.3.3 Startup and operation

A typical startup sequence is listed here.

- The time required to stabilize COUT will be the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function and filter. See the Data Sheets for power-on delays of the comparators. The windowing function has a maximum of one bus clock period delay. The filter delay is specified in the Low-pass filter.
- During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and SCR[CFR]/SCR[CFF]

to reflect an input change or a configuration change to one of the components involved in the data path.
- When programmed for filtering modes, COUT will initially be equal to 0, until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

16.3.4 Low-pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT.

Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by FPR[FILT_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high-frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

16.3.4.1 Enabling filter modes

Filter modes can be enabled by:

- Setting CR0[FILTER_CNT] > 0x01 and
- Setting FPR[FILT_PER] to a nonzero value or setting CR1[SE]=1

If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

The filter output will be at logic 0 when first initalized, and will subsequently change when all the consecutive CR0[FILTER_CNT] samples agree that the output value has changed. In other words, SCR[COUT] will be 0 for some initial period, even when COUTA is at logic 1.

Setting both CR1[SE] and FPR[FILT_PER] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CR0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CR0[FILTER_CNT] samples agree that the output value has changed.

16.3.4.2 Latency issues

The value of FPR[FILT_PER] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of CR0[FILTER_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CR0[FILTER_CNT].

The values of FPR[FILT_PER] or SAMPLE period and CR0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CR0[FILTER_CNT].

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

Table 16-2. Comparator sample/filter maximum latencies

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		T_{PD} + (FPR[FILT_PER] * T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CR0[FILTER_CNT] * T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] x T_{per}) + T_{per}

Table continues on the next page...

Table 16-2. Comparator sample/filter maximum latencies (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum latency[1]
5A	1	1	0	0x00	X	Windowed mode	$T_{PD} + T_{per}$
5B	1	1	0	X	0x00		$T_{PD} + T_{per}$
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (FPR[FILT_PER] * T_{per}) + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] x T_{per}) + $2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

16.4 CMP interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both.

The following table gives the conditions in which the interrupt request is asserted and deasserted.

When	Then
SCR[IER] and SCR[CFR] are set	The interrupt request is asserted
SCR[IEF] and SCR[CFF] are set	The interrupt request is asserted
SCR[IER] and SCR[CFR] are cleared for a rising-edge interrupt	The interrupt request is deasserted
SCR[IEF] and SCR[CFF] are cleared for a falling-edge interrupt	The interrupt request is deasserted

16.5 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it sends a transfer completing indicator that deasserts the DMA transfer request and clears the flag to allow a subsequent change on comparator output to occur and force another DMA request.

The comparator can remain functional in STOP modes.

When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

16.6 CMP Asynchronous DMA support

The comparator can remain functional in STOP modes.

When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

16.7 Digital-to-analog converter

The figure found here shows the block diagram of the DAC module.

It contains a 64-tap resistor ladder network and a 64-to-1 multiplexer, which selects an output voltage from one of 64 distinct levels that outputs from DACO. It is controlled through the DAC Control Register (DACCR). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down or disabled when not in use. When in Disabled mode, DACO is connected to the analog ground.

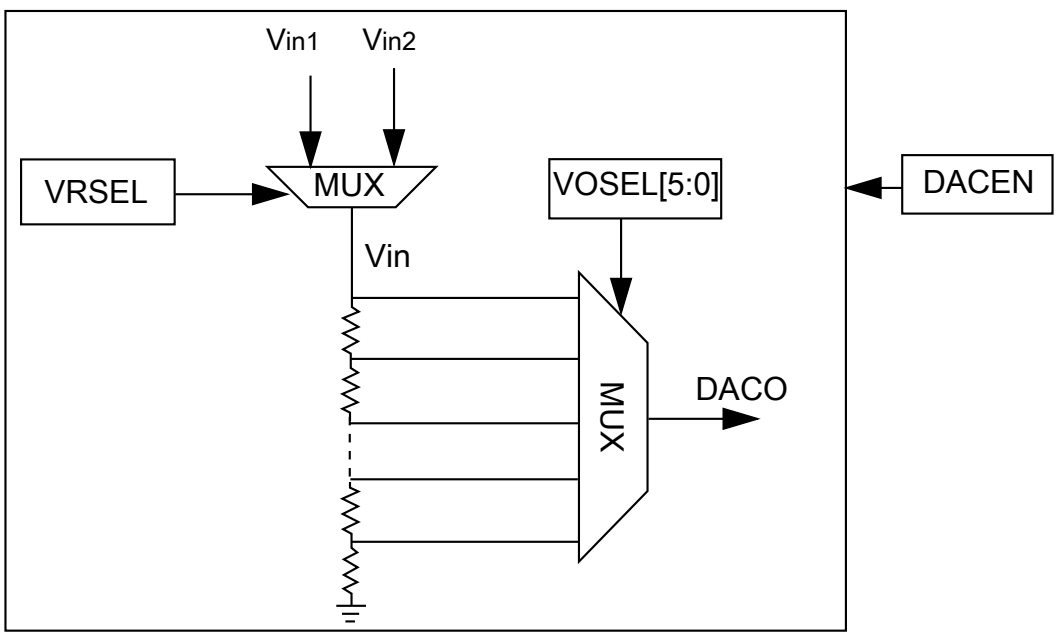


Figure 16-12. 6-bit DAC block diagram

16.8 DAC functional description

This section provides DAC functional description information.

16.8.1 Voltage reference source select

- V_{in1} connects to the primary voltage source as supply reference of 64 tap resistor ladder
- V_{in2} connects to an alternate voltage source

16.9 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

16.10 DAC clocks

This module has a single clock input, the bus clock.

16.11 DAC interrupts

This module has no interrupts.

16.12 CMP Trigger Mode

CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1.

In addition, the CMP must be enabled. If the DAC is to be used as a reference to the CMP, it must also be enabled.

CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare.

Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received.

See the chip configuration chapter for details about the external timer resource.

Chapter 17
Cyclic Redundancy Check (CRC)

17.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial and other parameters required to implement a 16-bit or 32-bit CRC standard.

The 16/32-bit code is calculated for 32 bits of data at a time.

17.1.1 Features

Features of the CRC module include:

- Hardware CRC generator circuit using a 16-bit or 32-bit programmable shift register
- Programmable initial seed value and polynomial
- Option to transpose input data or output data (the CRC result) bitwise or bytewise. This option is required for certain CRC standards. A bytewise transpose operation is not possible when accessing the CRC data register via 8-bit accesses. In this case, the user's software must perform the bytewise transpose function.
- Option for inversion of final CRC result
- 32-bit CPU register programming interface

17.1.2 Block diagram

The following is a block diagram of the CRC.

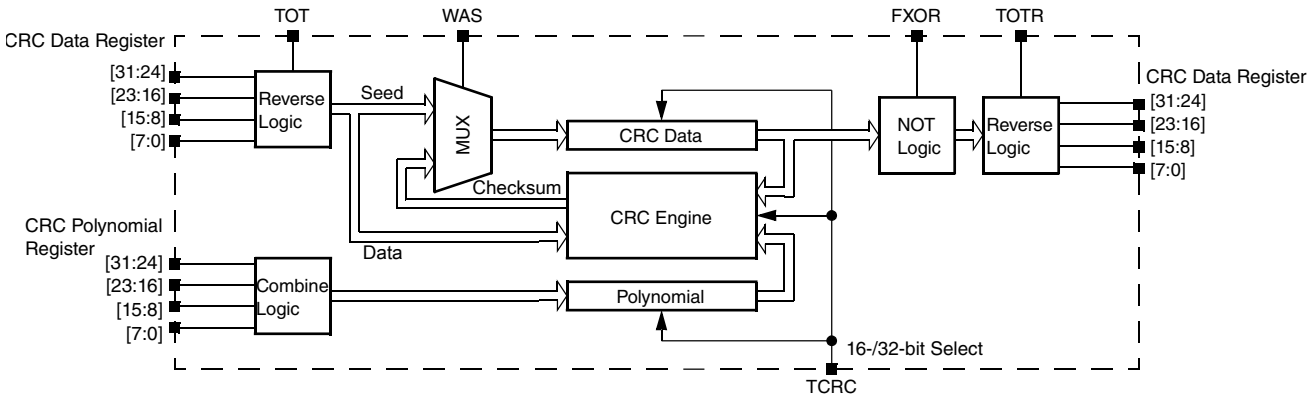


Figure 17-1. Programmable cyclic redundancy check (CRC) block diagram

17.1.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

17.1.3.1 Run mode

This is the basic mode of operation.

17.1.3.2 Low-power modes (Wait or Stop)

Any CRC calculation in progress stops when the MCU enters a low-power mode that disables the module clock. It resumes after the clock is enabled or via the system reset for exiting the low-power mode. Clock gating for this module is dependent on the MCU.

17.2 Memory map and register descriptions

CRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_8000	CRC Data register (CRC_DATA)	32	R/W	FFFF_FFFFh	17.2.1/371
4007_8004	CRC Polynomial register (CRC_GPOLY)	32	R/W	0000_1021h	17.2.2/372
4007_8008	CRC Control register (CRC_CTRL)	32	R/W	0000_0000h	17.2.3/372

17.2.1 CRC Data register (CRC_DATA)

The CRC Data register contains the value of the seed, data, and checksum. When CTRL[WAS] is set, any write to the data register is regarded as the seed value. When CTRL[WAS] is cleared, any write to the data register is regarded as data for general CRC computation.

In 16-bit CRC mode, the HU and HL fields are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous; with MSB of data value written first.

After all data values are written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is available in the LU and LL fields. In 32-bit CRC mode, all fields contain the result. Reads of this register at any time return the intermediate CRC value, provided the CRC module is configured.

Address: 4007_8000h base + 0h offset = 4007_8000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HU								HL								LU								LL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

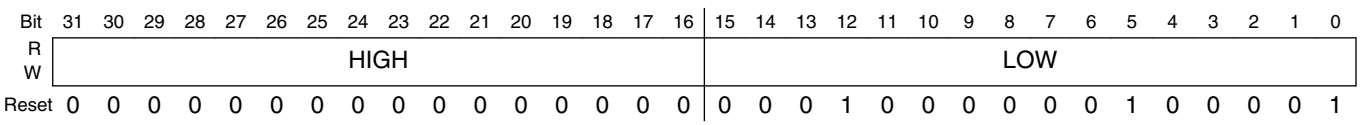
CRC_DATA field descriptions

Field	Description
31–24 HU	CRC High Upper Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
23–16 HL	CRC High Lower Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
15–8 LU	CRC Low Upper Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.
LL	CRC Low Lower Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.

17.2.2 CRC Polynomial register (CRC_GPOLY)

This register contains the value of the polynomial for the CRC calculation. The HIGH field contains the upper 16 bits of the CRC polynomial, which are used only in 32-bit CRC mode. Writes to the HIGH field are ignored in 16-bit CRC mode. The LOW field contains the lower 16 bits of the CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 4007_8000h base + 4h offset = 4007_8004h



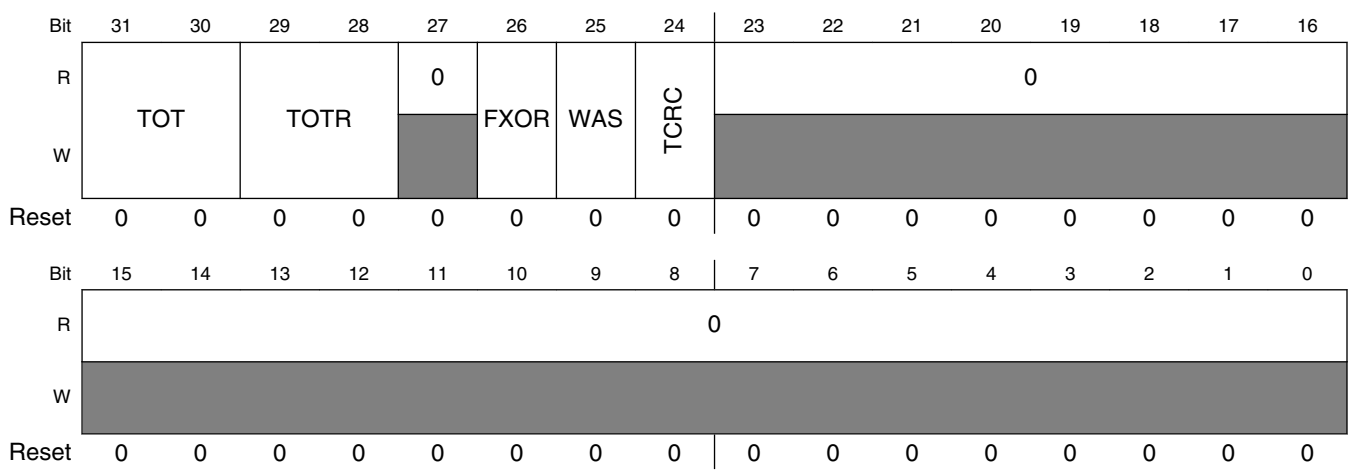
CRC_GPOLY field descriptions

Field	Description
31–16 HIGH	High Polynominal Half-word Writable and readable in 32-bit CRC mode (CTRL[TCRC] is 1). This field is not writable in 16-bit CRC mode (CTRL[TCRC] is 0).
LOW	Low Polynominal Half-word Writable and readable in both 32-bit and 16-bit CRC modes.

17.2.3 CRC Control register (CRC_CTRL)

This register controls the configuration and working of the CRC module. Appropriate bits must be set before starting a new CRC calculation. A new CRC calculation is initialized by asserting CTRL[WAS] and then writing the seed into the CRC data register.

Address: 4007_8000h base + 8h offset = 4007_8008h



CRC_CTRL field descriptions

Field	Description
31–30 TOT	Type Of Transpose For Writes Defines the transpose configuration of the data written to the CRC data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
29–28 TOTR	Type Of Transpose For Read Identifies the transpose configuration of the value read from the CRC Data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 FXOR	Complement Read Of CRC Data Register Some CRC protocols require the final checksum to be XORed with 0xFFFFFFFF or 0xFFFF. Asserting this bit enables on the fly complementing of read data. 0 No XOR on reading. 1 Invert or complement the read value of the CRC Data register.
25 WAS	Write CRC Data Register As Seed When asserted, a value written to the CRC data register is considered a seed value. When deasserted, a value written to the CRC data register is taken as data for CRC computation. 0 Writes to the CRC data register are data values. 1 Writes to the CRC data register are seed values.
24 TCRC	Width of CRC protocol. 0 16-bit CRC protocol. 1 32-bit CRC protocol.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

17.3 Functional description

17.3.1 CRC initialization/reinitialization

To enable the CRC calculation, the user must program CRC_CTRL[WAS], CRC_GPOLY,necessary parameters for transposition and CRC result inversion in the applicable registers. Asserting CRC_CTRL[WAS] enables the programming of the seed value into the CRC_DATA register.

After a completed CRC calculation, the module can be reinitialized for a new CRC computation by reasserting CRC_CTRL[WAS] and programming a new, or previously used, seed value. All other parameters must be set before programming the seed value and subsequent data values.

17.3.2 CRC calculations

In 16-bit and 32-bit CRC modes, data values can be programmed 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes can lead to an incorrect CRC computation.

17.3.2.1 16-bit CRC

To compute a 16-bit CRC:

1. Clear CRC_CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 16-bit polynomial to the CRC_GPOLY[LOW] field. The CRC_GPOLY[HIGH] field is not usable in 16-bit CRC mode.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to CRC_DATA[LU:LL]. CRC_DATA[HU:HL] are not used.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[LU:LL].

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

17.3.2.2 32-bit CRC

To compute a 32-bit CRC:

1. Set CRC_CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 32-bit polynomial to CRC_GPOLY[HIGH:LOW].
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to CRC_DATA[HU:HL:LU:LL].
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[HU:HL:LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[HU:HL:LU:LL]. The CRC is calculated bytewise, and two clocks are required to complete one CRC calculation.

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

17.3.3 Transpose feature

By default, the transpose feature is not enabled. However, some CRC standards require the input data and/or the final checksum to be transposed. The user software has the option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on the fly while being read or written.

Some protocols use little endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

17.3.3.1 Types of transpose

The CRC module provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] or CTRL[TOTR] fields, according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from the CRC data register:

1. CTRL[TOT] or CTRL[TOTR] is 00.

No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 01

   Bits in a byte are transposed, while bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}

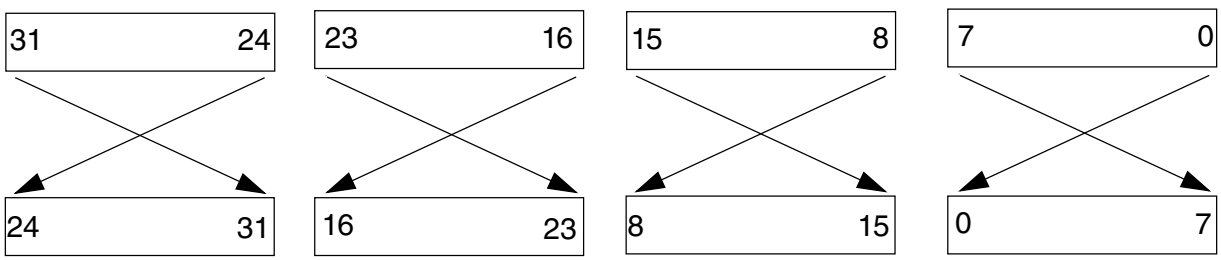


Figure 17-2. Transpose type 01

3. CTRL[TOT] or CTRL[TOTR] is 10.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes = {reg[0:7], reg[8:15],reg[16:23], reg[24:31]}

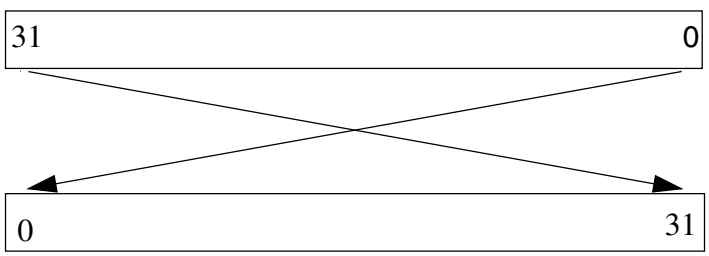


Figure 17-3. Transpose type 10

4. CTRL[TOT] or CTRL[TOTR] is 11.

   Bytes are transposed, but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}

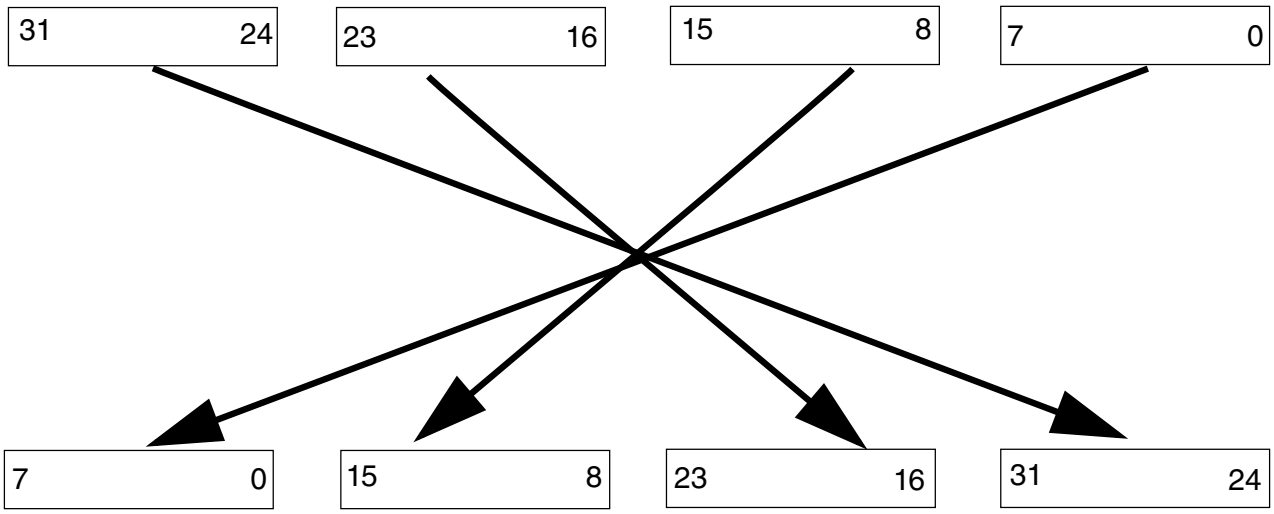


Figure 17-4. Transpose type 11

NOTE

- For 8-bit and 16-bit write accesses to the CRC data register, the data is transposed with zeros on the unused byte or bytes (taking 32 bits as a whole), but the CRC is calculated on the valid byte(s) only.
- When reading the CRC data register for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in the CRC[HU:HL] fields. The user software must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

17.3.4 CRC result complement

When CTRL[FXOR] is set, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in the CRC data register every time the CRC data register is read. When CTRL[FXOR] is cleared, reading the CRC data register accesses the raw checksum value.

Chapter 18
12-bit Digital-to-Analog Converter (DAC)

18.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The 12-bit digital-to-analog converter (DAC) is a low-power, general-purpose DAC. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator, op-amps, or ADC.

18.2 Features

The features of the DAC module include:

- On-chip programmable reference generator output. The voltage output range is from 1⁄4096 V_{in} to V_{in}, and the step is 1⁄4096 V_{in}, where V_{in} is the input voltage.
- V_{in} can be selected from two reference sources
- Static operation in Normal Stop mode
- 16-word data buffer supported with configurable watermark and multiple operation modes
- DMA support

18.3 Block diagram

The block diagram of the DAC module is as follows:

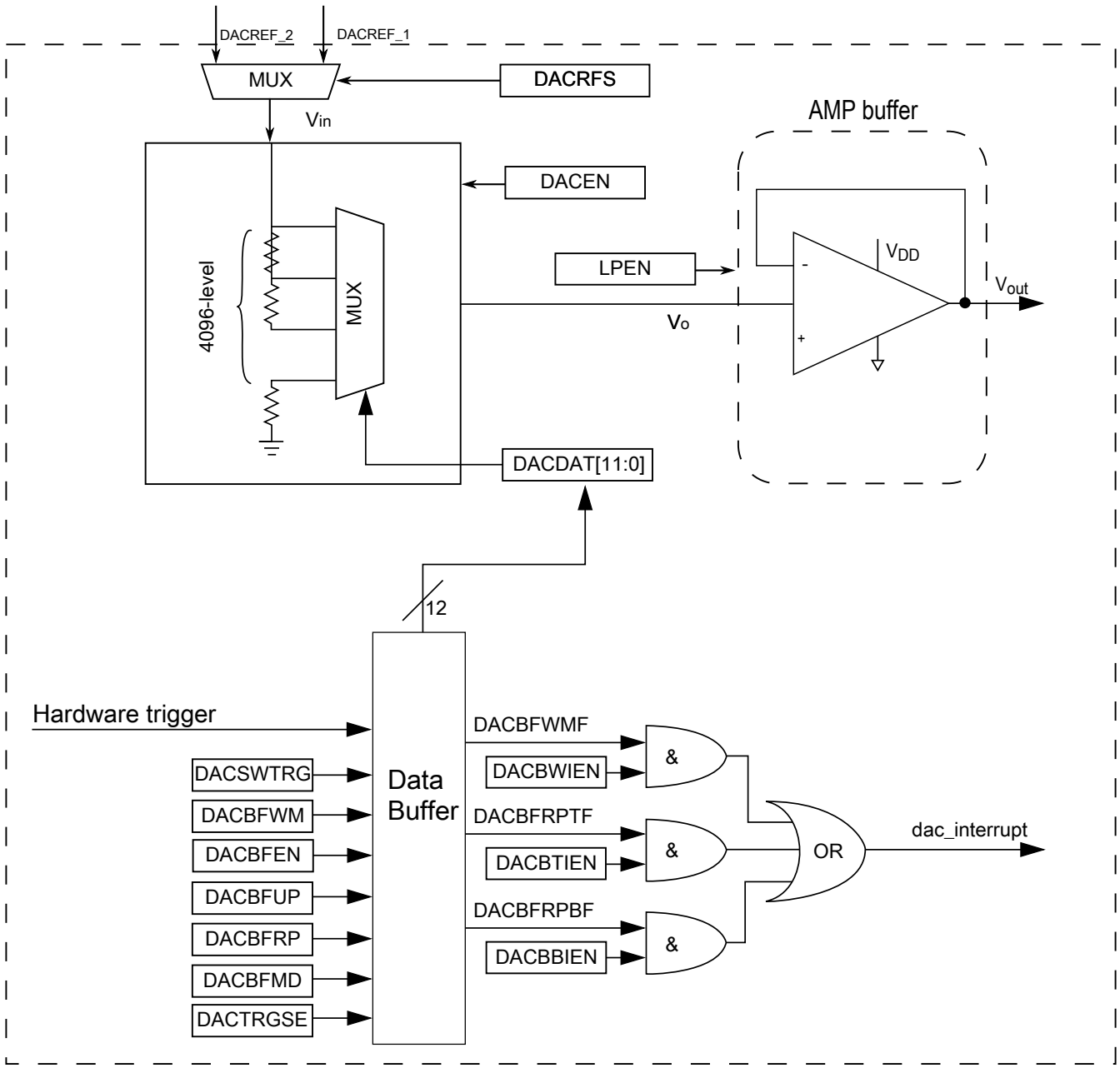


Figure 18-1. DAC block diagram

18.4 Memory map/register definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

DAC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A000	DAC Data Low Register (DAC0_DAT0L)	8	R/W	00h	18.4.1/382
4006_A001	DAC Data High Register (DAC0_DAT0H)	8	R/W	00h	18.4.2/382
4006_A002	DAC Data Low Register (DAC0_DAT1L)	8	R/W	00h	18.4.1/382
4006_A003	DAC Data High Register (DAC0_DAT1H)	8	R/W	00h	18.4.2/382
4006_A004	DAC Data Low Register (DAC0_DAT2L)	8	R/W	00h	18.4.1/382
4006_A005	DAC Data High Register (DAC0_DAT2H)	8	R/W	00h	18.4.2/382
4006_A006	DAC Data Low Register (DAC0_DAT3L)	8	R/W	00h	18.4.1/382
4006_A007	DAC Data High Register (DAC0_DAT3H)	8	R/W	00h	18.4.2/382
4006_A008	DAC Data Low Register (DAC0_DAT4L)	8	R/W	00h	18.4.1/382
4006_A009	DAC Data High Register (DAC0_DAT4H)	8	R/W	00h	18.4.2/382
4006_A00A	DAC Data Low Register (DAC0_DAT5L)	8	R/W	00h	18.4.1/382
4006_A00B	DAC Data High Register (DAC0_DAT5H)	8	R/W	00h	18.4.2/382
4006_A00C	DAC Data Low Register (DAC0_DAT6L)	8	R/W	00h	18.4.1/382
4006_A00D	DAC Data High Register (DAC0_DAT6H)	8	R/W	00h	18.4.2/382
4006_A00E	DAC Data Low Register (DAC0_DAT7L)	8	R/W	00h	18.4.1/382
4006_A00F	DAC Data High Register (DAC0_DAT7H)	8	R/W	00h	18.4.2/382
4006_A010	DAC Data Low Register (DAC0_DAT8L)	8	R/W	00h	18.4.1/382
4006_A011	DAC Data High Register (DAC0_DAT8H)	8	R/W	00h	18.4.2/382
4006_A012	DAC Data Low Register (DAC0_DAT9L)	8	R/W	00h	18.4.1/382
4006_A013	DAC Data High Register (DAC0_DAT9H)	8	R/W	00h	18.4.2/382
4006_A014	DAC Data Low Register (DAC0_DAT10L)	8	R/W	00h	18.4.1/382
4006_A015	DAC Data High Register (DAC0_DAT10H)	8	R/W	00h	18.4.2/382
4006_A016	DAC Data Low Register (DAC0_DAT11L)	8	R/W	00h	18.4.1/382
4006_A017	DAC Data High Register (DAC0_DAT11H)	8	R/W	00h	18.4.2/382
4006_A018	DAC Data Low Register (DAC0_DAT12L)	8	R/W	00h	18.4.1/382
4006_A019	DAC Data High Register (DAC0_DAT12H)	8	R/W	00h	18.4.2/382
4006_A01A	DAC Data Low Register (DAC0_DAT13L)	8	R/W	00h	18.4.1/382
4006_A01B	DAC Data High Register (DAC0_DAT13H)	8	R/W	00h	18.4.2/382
4006_A01C	DAC Data Low Register (DAC0_DAT14L)	8	R/W	00h	18.4.1/382
4006_A01D	DAC Data High Register (DAC0_DAT14H)	8	R/W	00h	18.4.2/382
4006_A01E	DAC Data Low Register (DAC0_DAT15L)	8	R/W	00h	18.4.1/382
4006_A01F	DAC Data High Register (DAC0_DAT15H)	8	R/W	00h	18.4.2/382
4006_A020	DAC Status Register (DAC0_SR)	8	R/W	02h	18.4.3/382
4006_A021	DAC Control Register (DAC0_C0)	8	R/W	00h	18.4.4/383
4006_A022	DAC Control Register 1 (DAC0_C1)	8	R/W	00h	18.4.5/385
4006_A023	DAC Control Register 2 (DAC0_C2)	8	R/W	0Fh	18.4.6/386

18.4.1 DAC Data Low Register (DAC*x*_DAT*n*L)

Address: 4006_A000h base + 0h offset + (2d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read Write	DATA0							
Reset	0	0	0	0	0	0	0	0

DAC*x*_DAT*n*L field descriptions

Field	Description
DATA0	DATA0 When the DAC buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula: $V_{out} = V_{in} * (1 + DACDAT0[11:0])/4096$ When the DAC buffer is enabled, DATA is mapped to the 16-word buffer.

18.4.2 DAC Data High Register (DAC*x*_DAT*n*H)

Address: 4006_A000h base + 1h offset + (2d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	0				DATA1			
Write								
Reset	0	0	0	0	0	0	0	0

DAC*x*_DAT*n*H field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA1	DATA1 When the DAC Buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula. $V_{out} = V_{in} * (1 + DACDAT0[11:0])/4096$ When the DAC buffer is enabled, DATA[11:0] is mapped to the 16-word buffer.

18.4.3 DAC Status Register (DAC*x*_SR)

If DMA is enabled, the flags can be cleared automatically by DMA when the DMA request is done. Writing 0 to a field clears it whereas writing 1 has no effect. After reset, DACBFRPTF is set and can be cleared by software, if needed. The flags are set only when the data buffer status is changed.

NOTE

Do not use 32/16-bit accesses to this register.

Address: 4006_A000h base + 20h offset = 4006_A020h

Bit	7	6	5	4	3	2	1	0
Read	0					DACBFWMF	DACBFRPTF	DACBFRPBF
Write								
Reset	0	0	0	0	0	0	1	0

DACx_SR field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 DACBFWMF	DAC Buffer Watermark Flag 0 The DAC buffer read pointer has not reached the watermark level. 1 The DAC buffer read pointer has reached the watermark level.
1 DACBFRPTF	DAC Buffer Read Pointer Top Position Flag 0 The DAC buffer read pointer is not zero. 1 The DAC buffer read pointer is zero.
0 DACBFRPBF	DAC Buffer Read Pointer Bottom Position Flag 0 The DAC buffer read pointer is not equal to C2[DACBFUP]. 1 The DAC buffer read pointer is equal to C2[DACBFUP].

18.4.4 DAC Control Register (DACx_C0)

NOTE

Do not use 32- or 16-bit accesses to this register.

Address: 4006_A000h base + 21h offset = 4006_A021h

Bit	7	6	5	4	3	2	1	0
Read	DACEN	DACRFS	DACTRGSEL	0	LPEN	DACBWIEN	DACBTIEN	DACBBIEN
Write				DACSWTRG				
Reset	0	0	0	0	0	0	0	0

DACx_C0 field descriptions

Field	Description
7 DACEN	DAC Enable Starts the Programmable Reference Generator operation.

Table continues on the next page...

DACx_C0 field descriptions (continued)

Field	Description
	0 The DAC system is disabled. 1 The DAC system is enabled.
6 DACRFS	DAC Reference Select 0 The DAC selects DACREF_1 as the reference voltage. 1 The DAC selects DACREF_2 as the reference voltage.
5 DACTRGSEL	DAC Trigger Select 0 The DAC hardware trigger is selected. 1 The DAC software trigger is selected.
4 DACSWTRG	DAC Software Trigger Active high. This is a write-only field, which always reads 0. If DAC software trigger is selected and buffer is enabled, writing 1 to this field will advance the buffer read pointer once. 0 The DAC soft trigger is not valid. 1 The DAC soft trigger is valid.
3 LPEN	DAC Low Power Control **NOTE:** See the 12-bit DAC electrical characteristics of the device data sheet for details on the impact of the modes below. 0 High-Power mode 1 Low-Power mode
2 DACBWIEN	DAC Buffer Watermark Interrupt Enable 0 The DAC buffer watermark interrupt is disabled. 1 The DAC buffer watermark interrupt is enabled.
1 DACBTIEN	DAC Buffer Read Pointer Top Flag Interrupt Enable 0 The DAC buffer read pointer top flag interrupt is disabled. 1 The DAC buffer read pointer top flag interrupt is enabled.
0 DACBBIEN	DAC Buffer Read Pointer Bottom Flag Interrupt Enable 0 The DAC buffer read pointer bottom flag interrupt is disabled. 1 The DAC buffer read pointer bottom flag interrupt is enabled.

18.4.5 DAC Control Register 1 (DAC*x*_C1)

NOTE

Do not use 32- or 16-bit accesses to this register.

Address: 4006_A000h base + 22h offset = 4006_A022h

Bit	7	6	5	4	3	2	1	0
Read	DMAEN	0		DACBFWM		DACBFMD		DACBFEN
Write								
Reset	0	0	0	0	0	0	0	0

DAC*x*_C1 field descriptions

Field	Description
7 DMAEN	DMA Enable Select 0 DMA is disabled. 1 DMA is enabled. When DMA is enabled, the DMA request will be generated by original interrupts. The interrupts will not be presented on this module at the same time.
6–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4–3 DACBFWM	DAC Buffer Watermark Select Controls when SR[DACBFWMF] is set. When the DAC buffer read pointer reaches the word defined by this field, which is 1–4 words away from the upper limit (DACBUP), SR[DACBFWMF] will be set. This allows user configuration of the watermark interrupt. 00 1 word 01 2 words 10 3 words 11 4 words
2–1 DACBFMD	DAC Buffer Work Mode Select 00 Normal mode 01 Swing mode 10 One-Time Scan mode 11 Reserved
0 DACBFEN	DAC Buffer Enable 0 Buffer read pointer is disabled. The converted data is always the first word of the buffer. 1 Buffer read pointer is enabled. The converted data is the word that the read pointer points to. It means converted data can be from any word of the buffer.

18.4.6 DAC Control Register 2 (DAC*x*_C2)

Address: 4006_A000h base + 23h offset = 4006_A023h

Bit	7	6	5	4	3	2	1	0
Read Write	DACBFRP				DACBFUP			
Reset	0	0	0	0	1	1	1	1

DACx_C2 field descriptions

Field	Description
7–4 DACBFRP	DAC Buffer Read Pointer Keeps the current value of the buffer read pointer.
DACBFUP	DAC Buffer Upper Limit Selects the upper limit of the DAC buffer. The buffer read pointer cannot exceed it.

18.5 Functional description

The 12-bit DAC module can select one of the two reference inputs—DACREF_1 and DACREF_2 as the DAC reference voltage, V_{in} by C0 [DACRFS]. See the chip-specific DAC information to determine the source options for DACREF_1 and DACREF_2.

When the DAC is enabled, it converts the data in DACDAT0[11:0] or the data from the DAC data buffer to a stepped analog output voltage. The output voltage range is from V_{in} to $V_{in}/4096$, and the step is $V_{in}/4096$.

18.5.1 DAC data buffer operation

When the DAC is enabled and the buffer is not enabled, the DAC module always converts the data in DAT0 to the analog output voltage.

When both the DAC and the buffer are enabled, the DAC converts the data in the data buffer to analog output voltage. The data buffer read pointer advances to the next word whenever a hardware or software trigger event occurs.

The data buffer can be configured to operate in Normal mode, Swing mode, One-Time Scan mode. When the buffer operation is switched from one mode to another, the read pointer does not change. The read pointer can be set to any value between 0 and C2[DACBFUP] by writing C2[DACBFRP].

18.5.1.1 DAC data buffer interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer. SR[DACBFRPBF] is set when the DAC buffer read pointer reaches the DAC buffer upper limit, that is, C2[DACBFRP] = C2[DACBFUP]. SR[DACBFRPTF] is set when the DAC read pointer is equal to the start position, 0. Finally, SR[DACBFWMF] is set when the DAC buffer read pointer has reached the position defined by C1[DACBFWM]. C1[DACBFWM] can be used to generate an interrupt when the DAC buffer read pointer is between 1 to 4 words from C2[DACBFUP].

18.5.1.2 Modes of DAC data buffer operation

The following table describes the different modes of data buffer operation for the DAC module.

Table 18-1. Modes of DAC data buffer operation

Modes	Description
Buffer Normal mode	This is the default mode. The buffer works as a circular buffer. The read pointer increases by one, every time the trigger occurs. When the read pointer reaches the upper limit, it goes to 0 directly in the next trigger event.
Buffer Swing mode	This mode is similar to the normal mode. However, when the read pointer reaches the upper limit, it does not go to 0. It will descend by 1 in the next trigger events until 0 is reached.
Buffer One-time Scan mode	The read pointer increases by 1 every time the trigger occurs. When it reaches the upper limit, it stops there. If read pointer is reset to the address other than the upper limit, it will increase to the upper address and stop there again. **NOTE:** If the software set the read pointer to the upper limit, the read pointer will not advance in this mode.

18.5.2 DMA operation

When DMA is enabled, DMA requests are generated instead of interrupt requests. The DMA Done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

18.5.3 Resets

During reset, the DAC is configured in the default mode and is disabled.

18.5.4 Low-Power mode operation

The following table shows the wait mode and the stop mode operation of the DAC module.

Table 18-2. Modes of operation

Modes of operation	Description
Wait mode	The DAC will operate normally, if enabled.
Stop mode	If enabled, the DAC module continues to operate normally, with the hardware trigger initiating the conversion. In low-power stop modes, the DAC is fully shut down.

NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

Chapter 19
Direct Memory Access Multiplexer (DMAMUX)

19.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

19.1.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the 8 DMA channels. This process is illustrated in the following figure.

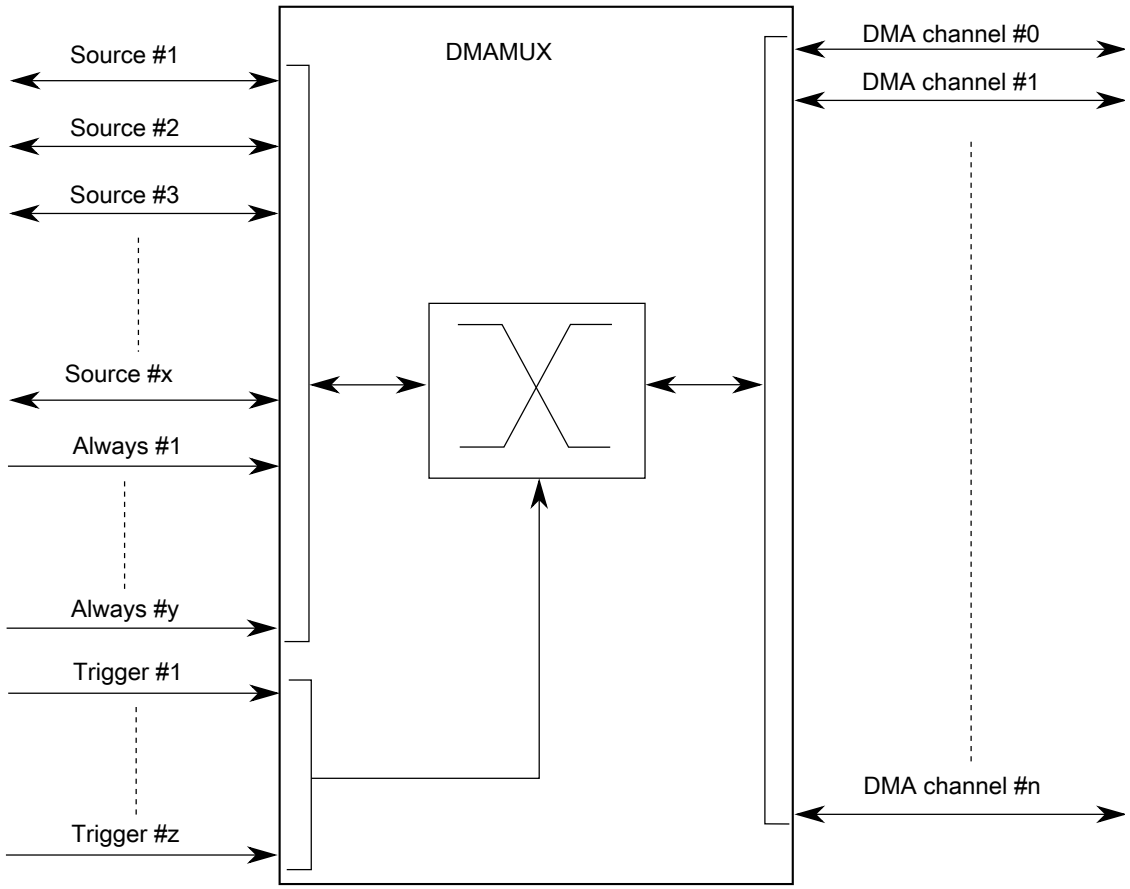


Figure 19-1. DMAMUX block diagram

19.1.2 Features

The DMAMUX module provides these features:

- Up to 59 peripheral slots and up to four always-on slots can be routed to 8 channels.
- 8 independently selectable DMA channel routers.
  - The first four channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the possible peripheral DMA slots or to one of the always-on slots.

19.1.3 Modes of operation

The following operating modes are available:

- Disabled mode

In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.

- Normal mode

  In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.

- Periodic Trigger mode

  In this mode, a DMA source may only request a DMA transfer, such as when a transmit buffer becomes empty or a receive buffer becomes full, periodically.

  Configuration of the period is done in the registers of the periodic interrupt timer (LPIT). This mode is available only for channels 0–3.

19.2 External signal description

The DMAMUX has no external pins.

19.3 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

DMAMUX memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_1000	Channel Configuration register (DMAMUX0_CHCFG0)	8	R/W	00h	19.3.1/392
4002_1001	Channel Configuration register (DMAMUX0_CHCFG1)	8	R/W	00h	19.3.1/392
4002_1002	Channel Configuration register (DMAMUX0_CHCFG2)	8	R/W	00h	19.3.1/392
4002_1003	Channel Configuration register (DMAMUX0_CHCFG3)	8	R/W	00h	19.3.1/392
4002_1004	Channel Configuration register (DMAMUX0_CHCFG4)	8	R/W	00h	19.3.1/392
4002_1005	Channel Configuration register (DMAMUX0_CHCFG5)	8	R/W	00h	19.3.1/392
4002_1006	Channel Configuration register (DMAMUX0_CHCFG6)	8	R/W	00h	19.3.1/392
4002_1007	Channel Configuration register (DMAMUX0_CHCFG7)	8	R/W	00h	19.3.1/392

19.3.1 Channel Configuration register (DMAMUX*x*_CHCFG*n*)

Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Setting multiple CHCFG registers with the same source value will result in unpredictable behavior. This is true, even if a channel is disabled (ENBL==0).

Before changing the trigger or source settings, a DMA channel must be disabled via CHCFGn[ENBL].

Address: 4002_1000h base + 0h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read Write	ENBL	TRIG	SOURCE					
Reset	0	0	0	0	0	0	0	0

DMAMUX*x*_CHCFG*n* field descriptions

Field	Description
7 ENBL	DMA Channel Enable Enables the DMA channel. 0 DMA channel is disabled. This mode is primarily used during configuration of the DMAMux. The DMA has separate channel enables/disables, which should be used to disable or reconfigure a DMA channel. 1 DMA channel is enabled
6 TRIG	DMA Channel Trigger Enable Enables the periodic trigger capability for the triggered DMA channel. 0 Triggering is disabled. If triggering is disabled and ENBL is set, the DMA Channel will simply route the specified source to the DMA channel. (Normal mode) 1 Triggering is enabled. If triggering is enabled and ENBL is set, the DMAMUX is in Periodic Trigger mode.
SOURCE	DMA Channel Source (Slot) Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about the peripherals and their slot numbers.

19.4 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels.

As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in Enabling and configuring sources is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

Functionally, the DMAMUX channels may be divided into two classes:

- Channels that implement the normal routing functionality plus periodic triggering capability
- Channels that implement only the normal routing functionality

19.4.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first 4 channels of the DMAMUX provide a special periodic triggering capability that can be used to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention.

The trigger is generated by the periodic interrupt timer (LPIT); as such, the configuration of the periodic triggering interval is done via configuration registers in the LPIT. See the section on periodic interrupt timer for more information on this topic.

Note

Because of the dynamic nature of the system (due to DMA channel priorities, bus arbitration, interrupt service routine lengths, etc.), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

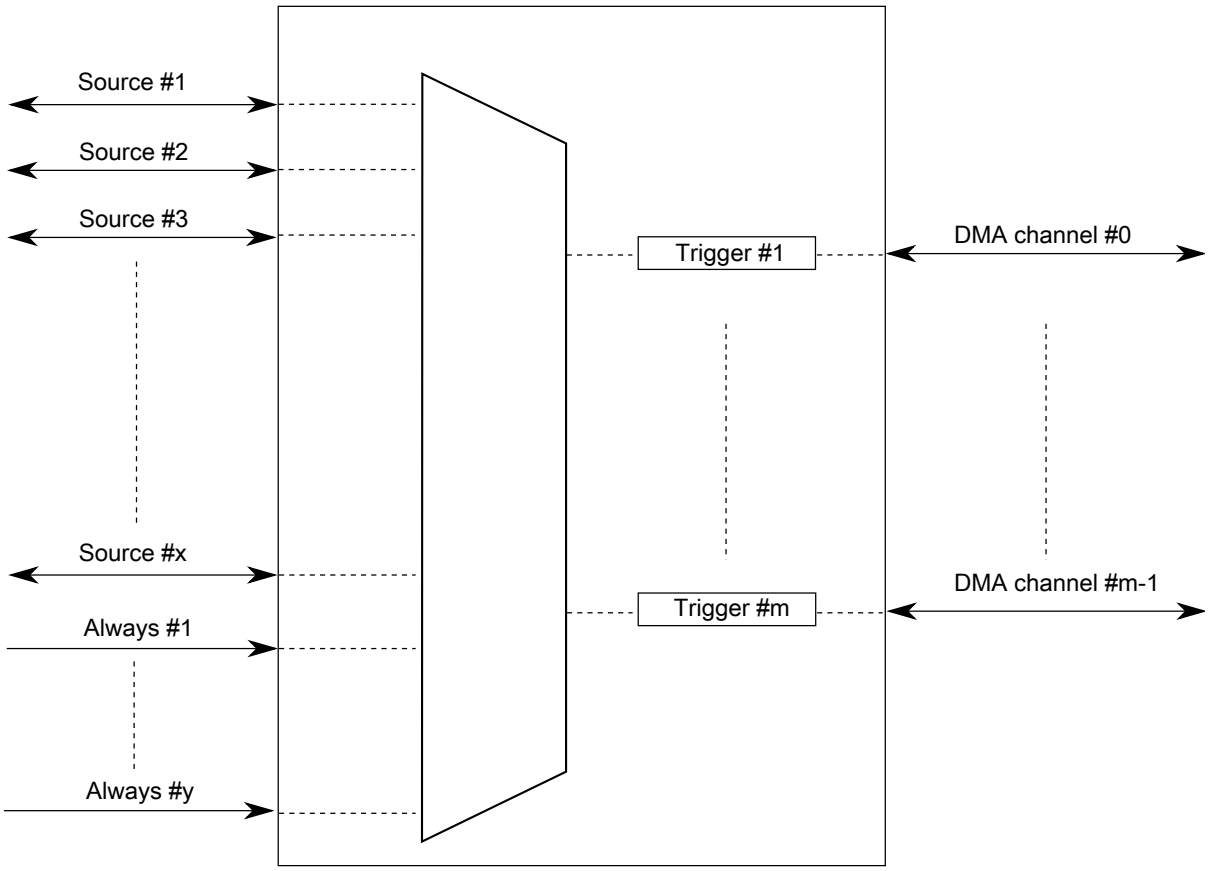


Figure 19-2. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.

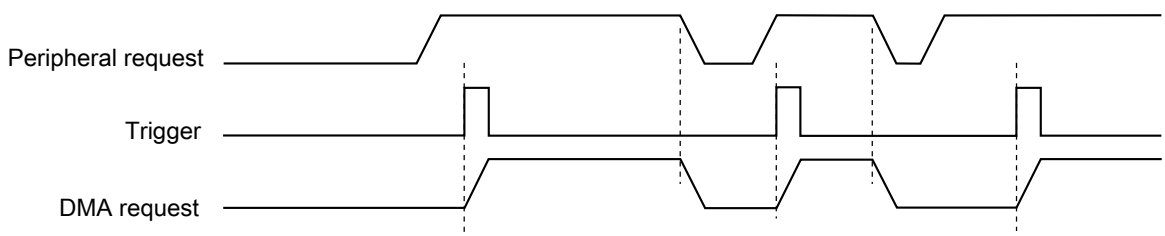


Figure 19-3. DMAMUX channel triggering: normal operation

After the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.

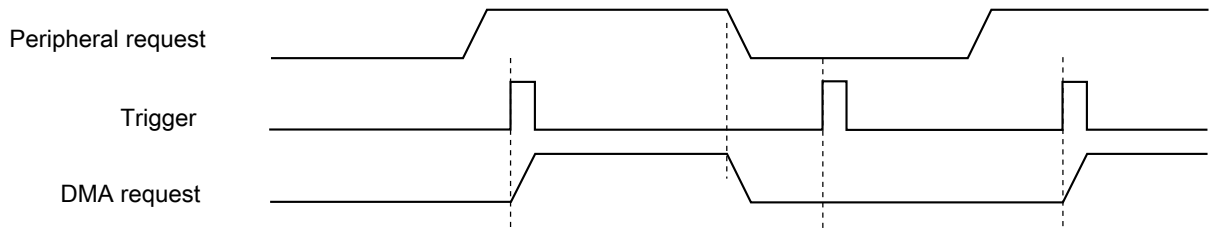


Figure 19-4. DMAMUX channel triggering: ignored trigger

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful for two types of situations:

- Periodically polling external devices on a particular bus

  As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI will request DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.

- Using the GPIO ports to drive or sample waveforms

  By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

A more detailed description of the capability of each trigger, including resolution, range of values, and so on, may be found in the periodic interrupt timer section.

19.4.2 DMA channels with no triggering capability

The other channels of the DMAMUX provide the normal routing functionality as described in Modes of operation.

19.4.3 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are four additional DMA sources that are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability).
- Performing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Performing DMA transfers from memory to the external bus, or vice-versa—Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation—Any DMA transfer that should be explicitly started by software.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are:

- Transfer all data in a single minor loop.

  By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use explicit software reactivation.

  In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers *after every minor loop*. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use an always-enabled DMA source.

In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and pointing to an "always enabled" source. Note that the reactivation of the channel can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of packets of data from one source to another, without processor intervention.

19.5 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

19.5.1 Reset

The reset state of each individual bit is shown in Memory map/register definition. In summary, after reset, all channels are disabled and must be explicitly enabled before use.

19.5.2 Enabling and configuring sources

To enable a source with periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 4 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0x00 to CHCFG1.

2. Configure channel 1 in the DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write 0xC5 to CHCFG1.

The following code example illustrates steps 1 and 4 above:

```
void DMAMUX_Init(uint8_t DMA_CH, uint8_t DMAMUX_SOURCE)
{
    DMAMUX_0.CHCFG[DMA_CH].B.SOURCE = DMAMUX_SOURCE;
    DMAMUX_0.CHCFG[DMA_CH].B.ENBL   = 1;
    DMAMUX_0.CHCFG[DMA_CH].B.TRIG   = 1;
}
```

To enable a source, without periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 4 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is set while CHCFG[TRIG] is cleared.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0x00 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
```

```
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;
*CHCFG1 = 0x85;
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the appropriate section for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00 to CHCFG8.
3. Write 0x87 to CHCFG8. (In this example, setting CHCFG[TRIG] would have no effect due to the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
```

```
In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;
*CHCFG8 = 0x87;
```

Chapter 20
Enhanced Direct Memory Access (eDMA)

20.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The enhanced direct memory access (eDMA) controller is a second-generation module capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 8 channels

20.1.1 eDMA system block diagram

Figure 20-1 illustrates the components of the eDMA system, including the eDMA module ("engine").

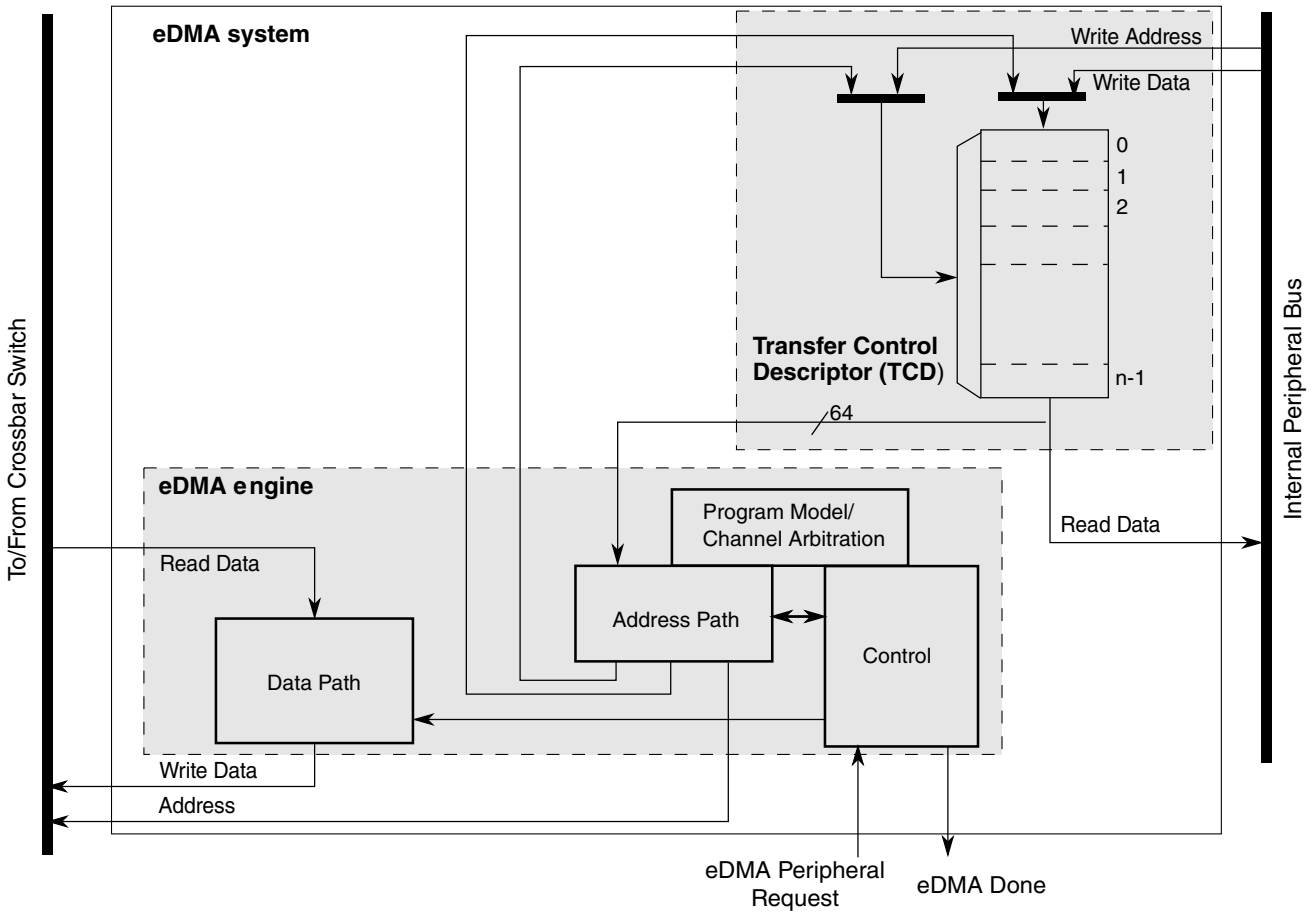


Figure 20-1. eDMA system block diagram

20.1.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer-control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 20-1. eDMA engine submodules

Submodule	Function
Address path	This block implements registered versions of two channel transfer control descriptors, channel x and channel y, and manages all master bus-address calculations. All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the first channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by DCHPRI*n*[ECP]) where a large data move operation can be preempted to minimize the time another channel is blocked from execution. When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes

Table continues on the next page...

Table 20-1. eDMA engine submodules (continued)

Submodule	Function
	the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of the next TCD*n* from memory as part of a scatter/gather operation.
Data path	This block implements the bus master read/write datapath. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the 1st stage of the bus pipeline (address phase), while the data path module implements the 2nd stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/ destination write operations until the number of bytes specified in the minor loop byte count has moved. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, two reads are performed, then one 32-bit write.

The transfer-control descriptor local memory is further partitioned into:

Table 20-2. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in the event of simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

20.1.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for enhanced addressing modes

- 8-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Connections to the crossbar switch for bus mastering the data movement
- Transfer control descriptor (TCD) organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

20.2 Modes of operation

The eDMA operates in the following modes:

Table 20-3. Modes of operation

Mode	Description
Normal	In Normal mode, the eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with the eDMA.

Table continues on the next page...

Table 20-3. Modes of operation (continued)

Mode	Description
	A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the transfer control descriptor (TCD). The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	DMA operation is configurable in Debug mode via the control register: • If CR[EDBG] is cleared, the DMA continues to operate. • If CR[EDBG] is set, the eDMA stops transferring data. If Debug mode is entered while a channel is active, the eDMA continues operation until the channel retires.
Wait	Before entering Wait mode, the DMA attempts to complete its current transfer. After the transfer completes, the device enters Wait mode.

20.3 Memory map/register definition

The eDMA's programming model is partitioned into two regions:

- The first region defines a number of registers providing control functions
- The second region corresponds to the local transfer control descriptor (TCD) memory

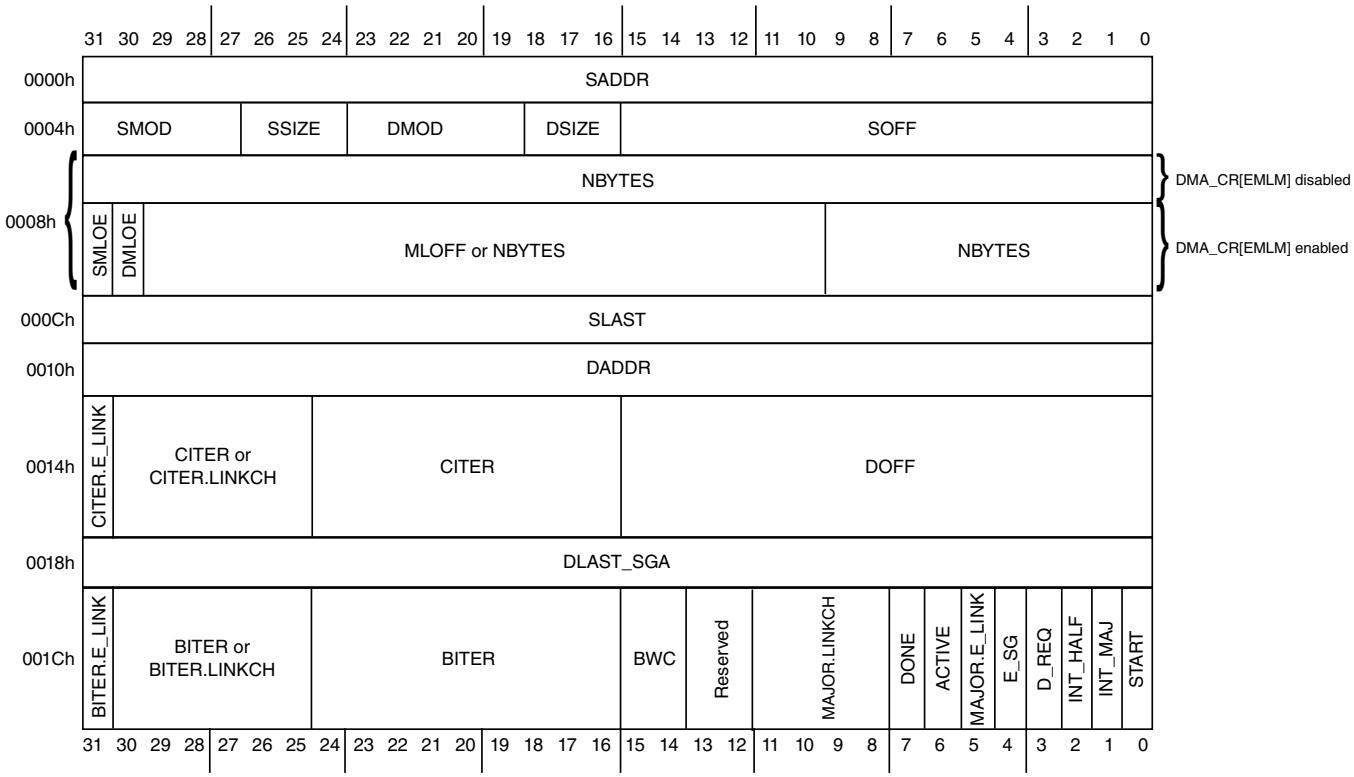
20.3.1 TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the desired data movement operation. The channel descriptors are stored in the local memory in sequential order: channel 0, channel 1, ... channel 7. Each TCD*n* definition is presented as 11 registers of 16 or 32 bits.

20.3.2 TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

20.3.3 TCD structure



20.3.4 Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

DMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_8000	Control Register (DMA0_CR)	32	R/W	See section	20.3.1/412
4000_8004	Error Status Register (DMA0_ES)	32	R	0000_0000h	20.3.2/415
4000_800C	Enable Request Register (DMA0_ERQ)	32	R/W	0000_0000h	20.3.3/417
4000_8014	Enable Error Interrupt Register (DMA0_EEI)	32	R/W	0000_0000h	20.3.4/419
4000_8018	Clear Enable Error Interrupt Register (DMA0_CEEI)	8	W (always reads 0)	00h	20.3.5/420
4000_8019	Set Enable Error Interrupt Register (DMA0_SEEI)	8	W (always reads 0)	00h	20.3.6/421

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_801A	Clear Enable Request Register (DMA0_CERQ)	8	W (always reads 0)	00h	20.3.7/422
4000_801B	Set Enable Request Register (DMA0_SERQ)	8	W (always reads 0)	00h	20.3.8/423
4000_801C	Clear DONE Status Bit Register (DMA0_CDNE)	8	W (always reads 0)	00h	20.3.9/424
4000_801D	Set START Bit Register (DMA0_SSRT)	8	W (always reads 0)	00h	20.3.10/425
4000_801E	Clear Error Register (DMA0_CERR)	8	W (always reads 0)	00h	20.3.11/426
4000_801F	Clear Interrupt Request Register (DMA0_CINT)	8	W (always reads 0)	00h	20.3.12/427
4000_8024	Interrupt Request Register (DMA0_INT)	32	R/W	0000_0000h	20.3.13/428
4000_802C	Error Register (DMA0_ERR)	32	R/W	0000_0000h	20.3.14/429
4000_8034	Hardware Request Status Register (DMA0_HRS)	32	R	0000_0000h	20.3.15/431
4000_8044	Enable Asynchronous Request in Stop Register (DMA0_EARS)	32	R/W	0000_0000h	20.3.16/433
4000_8100	Channel n Priority Register (DMA0_DCHPRI3)	8	R/W	See section	20.3.17/434
4000_8101	Channel n Priority Register (DMA0_DCHPRI2)	8	R/W	See section	20.3.17/434
4000_8102	Channel n Priority Register (DMA0_DCHPRI1)	8	R/W	See section	20.3.17/434
4000_8103	Channel n Priority Register (DMA0_DCHPRI0)	8	R/W	See section	20.3.17/434
4000_8104	Channel n Priority Register (DMA0_DCHPRI7)	8	R/W	See section	20.3.17/434
4000_8105	Channel n Priority Register (DMA0_DCHPRI6)	8	R/W	See section	20.3.17/434
4000_8106	Channel n Priority Register (DMA0_DCHPRI5)	8	R/W	See section	20.3.17/434
4000_8107	Channel n Priority Register (DMA0_DCHPRI4)	8	R/W	See section	20.3.17/434
4000_9000	TCD Source Address (DMA0_TCD0_SADDR)	32	R/W	Undefined	20.3.18/435
4000_9004	TCD Signed Source Address Offset (DMA0_TCD0_SOFF)	16	R/W	Undefined	20.3.19/435
4000_9006	TCD Transfer Attributes (DMA0_TCD0_ATTR)	16	R/W	Undefined	20.3.20/436
4000_9008	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD0_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD0_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD0_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_900C	TCD Last Source Address Adjustment (DMA0_TCD0_SLAST)	32	R/W	Undefined	20.3.24/440

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9010	TCD Destination Address (DMA0_TCD0_DADDR)	32	R/W	Undefined	20.3.25/440
4000_9014	TCD Signed Destination Address Offset (DMA0_TCD0_DOFF)	16	R/W	Undefined	20.3.26/441
4000_9016	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD0_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_9016	DMA0_TCD0_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_9018	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD0_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_901C	TCD Control and Status (DMA0_TCD0_CSR)	16	R/W	Undefined	20.3.30/444
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD0_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD0_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_9020	TCD Source Address (DMA0_TCD1_SADDR)	32	R/W	Undefined	20.3.18/435
4000_9024	TCD Signed Source Address Offset (DMA0_TCD1_SOFF)	16	R/W	Undefined	20.3.19/435
4000_9026	TCD Transfer Attributes (DMA0_TCD1_ATTR)	16	R/W	Undefined	20.3.20/436
4000_9028	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD1_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD1_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD1_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_902C	TCD Last Source Address Adjustment (DMA0_TCD1_SLAST)	32	R/W	Undefined	20.3.24/440
4000_9030	TCD Destination Address (DMA0_TCD1_DADDR)	32	R/W	Undefined	20.3.25/440
4000_9034	TCD Signed Destination Address Offset (DMA0_TCD1_DOFF)	16	R/W	Undefined	20.3.26/441
4000_9036	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD1_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_9036	DMA0_TCD1_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_9038	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD1_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_903C	TCD Control and Status (DMA0_TCD1_CSR)	16	R/W	Undefined	20.3.30/444
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD1_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD1_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_9040	TCD Source Address (DMA0_TCD2_SADDR)	32	R/W	Undefined	20.3.18/435

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9044	TCD Signed Source Address Offset (DMA0_TCD2_SOFF)	16	R/W	Undefined	20.3.19/435
4000_9046	TCD Transfer Attributes (DMA0_TCD2_ATTR)	16	R/W	Undefined	20.3.20/436
4000_9048	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD2_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD2_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD2_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_904C	TCD Last Source Address Adjustment (DMA0_TCD2_SLAST)	32	R/W	Undefined	20.3.24/440
4000_9050	TCD Destination Address (DMA0_TCD2_DADDR)	32	R/W	Undefined	20.3.25/440
4000_9054	TCD Signed Destination Address Offset (DMA0_TCD2_DOFF)	16	R/W	Undefined	20.3.26/441
4000_9056	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD2_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_9056	DMA0_TCD2_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_9058	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD2_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_905C	TCD Control and Status (DMA0_TCD2_CSR)	16	R/W	Undefined	20.3.30/444
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD2_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD2_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_9060	TCD Source Address (DMA0_TCD3_SADDR)	32	R/W	Undefined	20.3.18/435
4000_9064	TCD Signed Source Address Offset (DMA0_TCD3_SOFF)	16	R/W	Undefined	20.3.19/435
4000_9066	TCD Transfer Attributes (DMA0_TCD3_ATTR)	16	R/W	Undefined	20.3.20/436
4000_9068	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD3_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD3_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD3_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_906C	TCD Last Source Address Adjustment (DMA0_TCD3_SLAST)	32	R/W	Undefined	20.3.24/440
4000_9070	TCD Destination Address (DMA0_TCD3_DADDR)	32	R/W	Undefined	20.3.25/440
4000_9074	TCD Signed Destination Address Offset (DMA0_TCD3_DOFF)	16	R/W	Undefined	20.3.26/441
4000_9076	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD3_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9076	DMA0_TCD3_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_9078	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD3_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_907C	TCD Control and Status (DMA0_TCD3_CSR)	16	R/W	Undefined	20.3.30/444
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD3_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD3_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_9080	TCD Source Address (DMA0_TCD4_SADDR)	32	R/W	Undefined	20.3.18/435
4000_9084	TCD Signed Source Address Offset (DMA0_TCD4_SOFF)	16	R/W	Undefined	20.3.19/435
4000_9086	TCD Transfer Attributes (DMA0_TCD4_ATTR)	16	R/W	Undefined	20.3.20/436
4000_9088	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD4_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD4_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD4_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_908C	TCD Last Source Address Adjustment (DMA0_TCD4_SLAST)	32	R/W	Undefined	20.3.24/440
4000_9090	TCD Destination Address (DMA0_TCD4_DADDR)	32	R/W	Undefined	20.3.25/440
4000_9094	TCD Signed Destination Address Offset (DMA0_TCD4_DOFF)	16	R/W	Undefined	20.3.26/441
4000_9096	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD4_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_9096	DMA0_TCD4_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_9098	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD4_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_909C	TCD Control and Status (DMA0_TCD4_CSR)	16	R/W	Undefined	20.3.30/444
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD4_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD4_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_90A0	TCD Source Address (DMA0_TCD5_SADDR)	32	R/W	Undefined	20.3.18/435
4000_90A4	TCD Signed Source Address Offset (DMA0_TCD5_SOFF)	16	R/W	Undefined	20.3.19/435
4000_90A6	TCD Transfer Attributes (DMA0_TCD5_ATTR)	16	R/W	Undefined	20.3.20/436
4000_90A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD5_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD5_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD5_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_90AC	TCD Last Source Address Adjustment (DMA0_TCD5_SLAST)	32	R/W	Undefined	20.3.24/440
4000_90B0	TCD Destination Address (DMA0_TCD5_DADDR)	32	R/W	Undefined	20.3.25/440
4000_90B4	TCD Signed Destination Address Offset (DMA0_TCD5_DOFF)	16	R/W	Undefined	20.3.26/441
4000_90B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD5_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_90B6	DMA0_TCD5_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_90B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD5_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_90BC	TCD Control and Status (DMA0_TCD5_CSR)	16	R/W	Undefined	20.3.30/444
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD5_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD5_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_90C0	TCD Source Address (DMA0_TCD6_SADDR)	32	R/W	Undefined	20.3.18/435
4000_90C4	TCD Signed Source Address Offset (DMA0_TCD6_SOFF)	16	R/W	Undefined	20.3.19/435
4000_90C6	TCD Transfer Attributes (DMA0_TCD6_ATTR)	16	R/W	Undefined	20.3.20/436
4000_90C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD6_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD6_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD6_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_90CC	TCD Last Source Address Adjustment (DMA0_TCD6_SLAST)	32	R/W	Undefined	20.3.24/440
4000_90D0	TCD Destination Address (DMA0_TCD6_DADDR)	32	R/W	Undefined	20.3.25/440
4000_90D4	TCD Signed Destination Address Offset (DMA0_TCD6_DOFF)	16	R/W	Undefined	20.3.26/441
4000_90D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD6_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_90D6	DMA0_TCD6_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_90D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD6_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_90DC	TCD Control and Status (DMA0_TCD6_CSR)	16	R/W	Undefined	20.3.30/444

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD6_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD6_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448
4000_90E0	TCD Source Address (DMA0_TCD7_SADDR)	32	R/W	Undefined	20.3.18/435
4000_90E4	TCD Signed Source Address Offset (DMA0_TCD7_SOFF)	16	R/W	Undefined	20.3.19/435
4000_90E6	TCD Transfer Attributes (DMA0_TCD7_ATTR)	16	R/W	Undefined	20.3.20/436
4000_90E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA0_TCD7_NBYTES_MLNO)	32	R/W	Undefined	20.3.21/437
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA0_TCD7_NBYTES_MLOFFNO)	32	R/W	Undefined	20.3.22/437
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA0_TCD7_NBYTES_MLOFFYES)	32	R/W	Undefined	20.3.23/439
4000_90EC	TCD Last Source Address Adjustment (DMA0_TCD7_SLAST)	32	R/W	Undefined	20.3.24/440
4000_90F0	TCD Destination Address (DMA0_TCD7_DADDR)	32	R/W	Undefined	20.3.25/440
4000_90F4	TCD Signed Destination Address Offset (DMA0_TCD7_DOFF)	16	R/W	Undefined	20.3.26/441
4000_90F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD7_CITER_ELINKYES)	16	R/W	Undefined	20.3.27/441
4000_90F6	DMA0_TCD7_CITER_ELINKNO	16	R/W	Undefined	20.3.28/443
4000_90F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA0_TCD7_DLASTSGA)	32	R/W	Undefined	20.3.29/444
4000_90FC	TCD Control and Status (DMA0_TCD7_CSR)	16	R/W	Undefined	20.3.30/444
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA0_TCD7_BITER_ELINKYES)	16	R/W	Undefined	20.3.31/447
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA0_TCD7_BITER_ELINKNO)	16	R/W	Undefined	20.3.32/448

20.3.1 Control Register (DMA*x*_CR)

The CR defines the basic operating configuration of the DMA.

Arbitration can be configured to use either a fixed-priority or a round-robin scheme. For fixed-priority arbitration, the highest priority channel requesting service is selected to execute. The channel priority registers assign the priorities; see the DCHPRIn registers. For round-robin arbitration, the channel priorities are ignored and channels are cycled through (from high to low channel number) without regard to priority.

NOTE

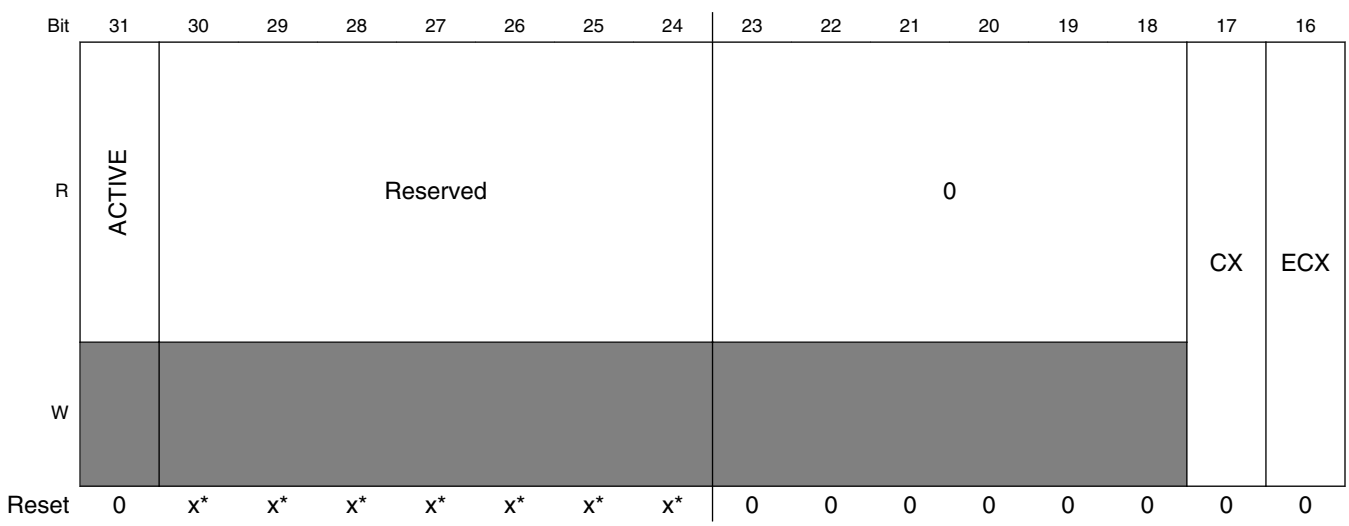
For correct operation, writes to the CR register must be performed only when the DMA channels are inactive; that is, when TCDn_CSR[ACTIVE] bits are cleared.

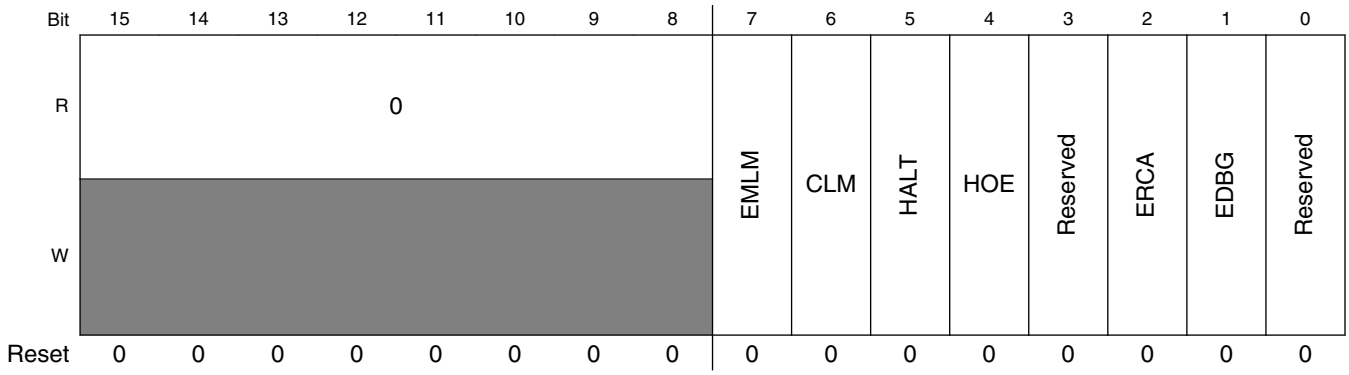
Minor loop offsets are address offset values added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. When minor loop offsets are enabled, the minor loop offset (MLOFF) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both prior to the addresses being written back into the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (EMLM is 1), TCDn word2 is redefined. A portion of TCDn word2 is used to specify multiple fields: a source enable bit (SMLOE) to specify the minor loop offset should be applied to the source address (TCDn_SADDR) upon minor loop completion, a destination enable bit (DMLOE) to specify the minor loop offset should be applied to the destination address (TCDn_DADDR) upon minor loop completion, and the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

When minor loop mapping is disabled (EMLM is 0), all 32 bits of TCDn word2 are assigned to the NBYTES field.

Address: 4000_8000h base + 0h offset = 4000_8000h





* Notes:
- x = Undefined at reset.

DMAx_CR field descriptions

Field	Description
31 ACTIVE	DMA Active Status 0 eDMA is idle. 1 eDMA is executing a channel.
30–24 Reserved	This field is reserved. Reserved
23–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 CX	Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The CX bit clears itself after the cancel has been honored. This cancel retires the channel normally as if the minor loop was completed.
16 ECX	Error Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer in the same fashion as the CX bit. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The ECX bit clears itself after the cancel is honored. In addition to cancelling the transfer, ECX treats the cancel as an error condition, thus updating the Error Status register (DMAx_ES) and generating an optional error interrupt.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 EMLM	Enable Minor Loop Mapping 0 Disabled. TCDn.word2 is defined as a 32-bit NBYTES field. 1 Enabled. TCDn.word2 is redefined to include individual enable fields, an offset field, and the NBYTES field. The individual enable fields allow the minor loop offset to be applied to the source address, the destination address, or both. The NBYTES field is reduced when either offset is enabled.
6 CLM	Continuous Link Mode

Table continues on the next page...

DMAx_CR field descriptions (continued)

Field	Description
	NOTE: Do not use continuous link mode with a channel linking to itself if there is only one minor loop iteration per service request, e.g., if the channel's NBYTES value is the same as either the source or destination size. The same data transfer profile can be achieved by simply increasing the NBYTES value, which provides more efficient, faster processing. 0 A minor loop channel link made to itself goes through channel arbitration before being activated again. 1 A minor loop channel link made to itself does not go through channel arbitration before being activated again. Upon minor loop completion, the channel activates again if that channel has a minor loop channel link enabled and the link channel is itself. This effectively applies the minor loop offsets and restarts the next minor loop.
5 HALT	Halt DMA Operations 0 Normal operation 1 Stall the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this bit is cleared.
4 HOE	Halt On Error 0 Normal operation 1 Any error causes the HALT bit to set. Subsequently, all service requests are ignored until the HALT bit is cleared.
3 Reserved	This field is reserved. Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0 Fixed priority arbitration is used for channel selection . 1 Round robin arbitration is used for channel selection .
1 EDBG	Enable Debug 0 When in debug mode, the DMA continues to operate. 1 When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the system exits debug mode or the EDBG bit is cleared.
0 Reserved	This field is reserved. Reserved

20.3.2 Error Status Register (DMAx_ES)

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- A configuration error, that is:
  - An illegal setting in the transfer-control descriptor, or
  - An illegal priority register setting in fixed-arbitration
- An error termination to a bus master read or write cycle
- A cancel transfer with error bit that will be set when a transfer is canceled via the corresponding cancel transfer control bit

See the Error Reporting and Handling section for more details.

Address: 4000_8000h base + 4h offset = 4000_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	0														ECX
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
R	0	CPE	0			ERRCHN			SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMAx_ES field descriptions

Field	Description
31 VLD	Logical OR of all ERR status bits 0 No ERR bits are set. 1 At least one ERR bit is set indicating a valid error exists that has not been cleared.
30–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ECX	Transfer Canceled 0 No canceled transfers 1 The last recorded entry was a canceled transfer by the error cancel transfer input
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 CPE	Channel Priority Error 0 No channel priority error 1 The last recorded error was a configuration error in the channel priorities . Channel priorities are not unique.
13–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error, excluding CPE errors, or last recorded error canceled transfer.
7 SAE	Source Address Error 0 No source address configuration error. 1 The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].
6 SOE	Source Offset Error 0 No source offset configuration error 1 The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].
5 DAE	Destination Address Error

Table continues on the next page...

DMAx_ES field descriptions (continued)

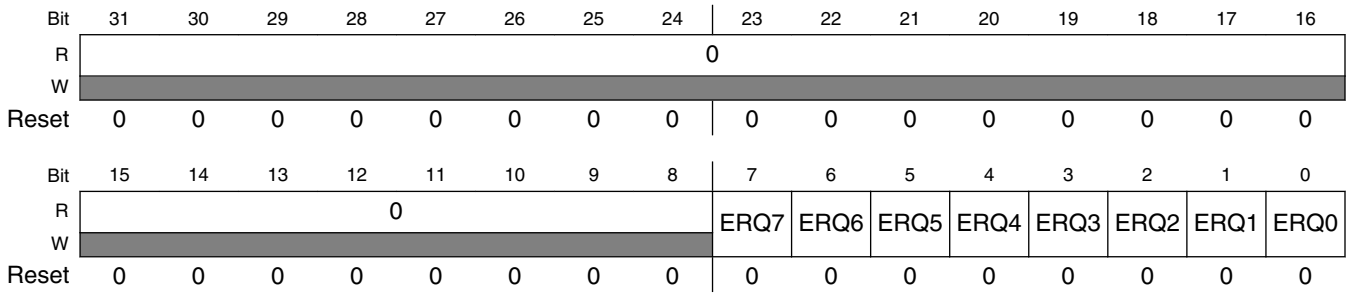
Field	Description
	0 No destination address configuration error 1 The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].
4 DOE	Destination Offset Error 0 No destination offset configuration error 1 The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].
3 NCE	NBYTES/CITER Configuration Error 0 No NBYTES/CITER configuration error 1 The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields. • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], or • TCDn_CITER[CITER] is equal to zero, or • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0 No scatter/gather configuration error 1 The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.
1 SBE	Source Bus Error 0 No source bus error 1 The last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0 No destination bus error 1 The last recorded error was a bus error on a destination write

20.3.3 Enable Request Register (DMAx_ERQ)

The ERQ register provides a bit map for the 8 channels to enable the request signal for each channel. The state of any given channel enable is directly affected by writes to this register; it is also affected by writes to the SERQ and CERQ registers. These registers are provided so the request enable for a single channel can easily be modified without needing to perform a read-modify-write sequence to the ERQ.

DMA request input signals and this enable request flag must be asserted before a channel's hardware service request is accepted. The state of the DMA enable request flag does not affect a channel service request made explicitly through software or a linked channel request.

Address: 4000_8000h base + Ch offset = 4000_800Ch



DMAx_ERQ field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 ERQ7	Enable DMA Request 7 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
6 ERQ6	Enable DMA Request 6 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
5 ERQ5	Enable DMA Request 5 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
4 ERQ4	Enable DMA Request 4 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
3 ERQ3	Enable DMA Request 3 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
2 ERQ2	Enable DMA Request 2 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
1 ERQ1	Enable DMA Request 1 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
0 ERQ0	Enable DMA Request 0 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

20.3.4 Enable Error Interrupt Register (DMAx_EEI)

The EEI register provides a bit map for the 8 channels to enable the error interrupt signal for each channel. The state of any given channel's error interrupt enable is directly affected by writes to this register; it is also affected by writes to the SEEI and CEEI. These registers are provided so that the error interrupt enable for a single channel can easily be modified without the need to perform a read-modify-write sequence to the EEI register.

The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller.

Address: 4000_8000h base + 14h offset = 4000_8014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								EEI7	EEI6	EEI5	EEI4	EEI3	EEI2	EEI1	EEI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMAx_EEI field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 EEI7	Enable Error Interrupt 7 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
6 EEI6	Enable Error Interrupt 6 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
5 EEI5	Enable Error Interrupt 5 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
4 EEI4	Enable Error Interrupt 4 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
3 EEI3	Enable Error Interrupt 3 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

Table continues on the next page...

DMAx_EEI field descriptions (continued)

Field	Description
2 EEI2	Enable Error Interrupt 2 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
1 EEI1	Enable Error Interrupt 1 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
0 EEI0	Enable Error Interrupt 0 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

20.3.5 Clear Enable Error Interrupt Register (DMAx_CEEI)

The CEEI provides a simple memory-mapped mechanism to clear a given bit in the EEI to disable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be cleared. Setting the CAEE bit provides a global clear function, forcing the EEI contents to be cleared, disabling all DMA request inputs. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 18h offset = 4000_8018h

Bit	7	6	5	4	3	2	1	0
Read	0	0					0	
Write	NOP	CAEE		0			CEEI	
Reset	0	0	0	0	0	0	0	0

DMAx_CEEI field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEE	Clear All Enable Error Interrupts 0 Clear only the EEI bit specified in the CEEI field 1 Clear all bits in EEI
5–3 Reserved	This field is reserved.

Table continues on the next page...

DMAx_CEEI field descriptions (continued)

Field	Description
CEEI	Clear Enable Error Interrupt Clears the corresponding bit in EEI

20.3.6 Set Enable Error Interrupt Register (DMAx_SEEI)

The SEEI provides a simple memory-mapped mechanism to set a given bit in the EEI to enable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be set. Setting the SAEE bit provides a global set function, forcing the entire EEI contents to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 19h offset = 4000_8019h

Bit	7	6	5	4	3	2	1	0
Read	0	0					0	
Write	NOP	SAEE			0		SEEI	
Reset	0	0	0	0	0	0	0	0

DMAx_SEEI field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAEE	Sets All Enable Error Interrupts 0 Set only the EEI bit specified in the SEEI field. 1 Sets all bits in EEI
5–3 Reserved	This field is reserved.
SEEI	Set Enable Error Interrupt Sets the corresponding bit in EEI

20.3.7 Clear Enable Request Register (DMA*x*_CERQ)

The CERQ provides a simple memory-mapped mechanism to clear a given bit in the ERQ to disable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be cleared. Setting the CAER bit provides a global clear function, forcing the entire contents of the ERQ to be cleared, disabling all DMA request inputs. If NOP is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Ah offset = 4000_801Ah

Bit	7	6	5	4	3	2	1	0
Read	0	0				0		
Write	NOP	CAER	0			CERQ		
Reset	0	0	0	0	0	0	0	0

DMA*x*_CERQ field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAER	Clear All Enable Requests 0 Clear only the ERQ bit specified in the CERQ field 1 Clear all bits in ERQ
5–3 Reserved	This field is reserved.
CERQ	Clear Enable Request Clears the corresponding bit in ERQ.

20.3.8 Set Enable Request Register (DMA*x*_SERQ)

The SERQ provides a simple memory-mapped mechanism to set a given bit in the ERQ to enable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be set. Setting the SAER bit provides a global set function, forcing the entire contents of ERQ to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Bh offset = 4000_801Bh

Bit	7	6	5	4	3	2	1	0
Read	0	0				0		
Write	NOP	SAER	0			SERQ		
Reset	0	0	0	0	0	0	0	0

DMA*x*_SERQ field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAER	Set All Enable Requests 0 Set only the ERQ bit specified in the SERQ field 1 Set all bits in ERQ
5–3 Reserved	This field is reserved.
SERQ	Set Enable Request Sets the corresponding bit in ERQ.

20.3.9 Clear DONE Status Bit Register (DMAx_CDNE)

The CDNE provides a simple memory-mapped mechanism to clear the DONE bit in the TCD of the given channel. The data value on a register write causes the DONE bit in the corresponding transfer control descriptor to be cleared. Setting the CADN bit provides a global clear function, forcing all DONE bits to be cleared. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Ch offset = 4000_801Ch

Bit	7	6	5	4	3	2	1	0
Read	0	0				0		
Write	NOP	CADN	0			CDNE		
Reset	0	0	0	0	0	0	0	0

DMAx_CDNE field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CADN	Clears All DONE Bits 0 Clears only the TCDn_CSR[DONE] bit specified in the CDNE field 1 Clears all bits in TCDn_CSR[DONE]
5–3 Reserved	This field is reserved.
CDNE	Clear DONE Bit Clears the corresponding bit in TCDn_CSR[DONE]

20.3.10 Set START Bit Register (DMAx_SSRT)

The SSRT provides a simple memory-mapped mechanism to set the START bit in the TCD of the given channel. The data value on a register write causes the START bit in the corresponding transfer control descriptor to be set. Setting the SAST bit provides a global set function, forcing all START bits to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Dh offset = 4000_801Dh

Bit	7	6	5	4	3	2	1	0
Read	0	0				0		
Write	NOP	SAST	0			SSRT		
Reset	0	0	0	0	0	0	0	0

DMAx_SSRT field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAST	Set All START Bits (activates all channels) 0 Set only the TCDn_CSR[START] bit specified in the SSRT field 1 Set all bits in TCDn_CSR[START]
5–3 Reserved	This field is reserved.
SSRT	Set START Bit Sets the corresponding bit in TCDn_CSR[START]

20.3.11 Clear Error Register (DMAx_CERR)

The CERR provides a simple memory-mapped mechanism to clear a given bit in the ERR to disable the error condition flag for a given channel. The given value on a register write causes the corresponding bit in the ERR to be cleared. Setting the CAEI bit provides a global clear function, forcing the ERR contents to be cleared, clearing all channel error indicators. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Eh offset = 4000_801Eh

Bit	7	6	5	4	3	2	1	0
Read	0	0				0		
Write	NOP	CAEI	0			CERR		
Reset	0	0	0	0	0	0	0	0

DMAx_CERR field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEI	Clear All Error Indicators 0 Clear only the ERR bit specified in the CERR field 1 Clear all bits in ERR
5–3 Reserved	This field is reserved.
CERR	Clear Error Indicator Clears the corresponding bit in ERR

20.3.12 Clear Interrupt Request Register (DMA*x*_CINT)

The CINT provides a simple, memory-mapped mechanism to clear a given bit in the INT to disable the interrupt request for a given channel. The given value on a register write causes the corresponding bit in the INT to be cleared. Setting the CAIR bit provides a global clear function, forcing the entire contents of the INT to be cleared, disabling all DMA interrupt requests. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Fh offset = 4000_801Fh

Bit	7	6	5	4	3	2	1	0
Read	0	0				0		
Write	NOP	CAIR	0			CINT		
Reset	0	0	0	0	0	0	0	0

DMAx_CINT field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAIR	Clear All Interrupt Requests 0 Clear only the INT bit specified in the CINT field 1 Clear all bits in INT
5–3 Reserved	This field is reserved.
CINT	Clear Interrupt Request Clears the corresponding bit in INT

20.3.13 Interrupt Request Register (DMAx_INT)

The INT register provides a bit map for the 8 channels signaling the presence of an interrupt request for each channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion. The outputs of this register are directly routed to the interrupt controller. During the interrupt-service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. Typically, a write to the CINT register in the interrupt service routine is used for this purpose.

The state of any given channel's interrupt request is directly affected by writes to this register; it is also affected by writes to the CINT register. On writes to INT, a 1 in any bit position clears the corresponding channel's interrupt request. A zero in any bit position has no affect on the corresponding channel's current interrupt status. The CINT register is provided so the interrupt request for a single channel can easily be cleared without the need to perform a read-modify-write sequence to the INT register.

Address: 4000_8000h base + 24h offset = 4000_8024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMAx_INT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 INT7	Interrupt Request 7 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
6 INT6	Interrupt Request 6 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
5 INT5	Interrupt Request 5

Table continues on the next page...

DMAx_INT field descriptions (continued)

Field	Description
	0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
4 INT4	Interrupt Request 4 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
3 INT3	Interrupt Request 3 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
2 INT2	Interrupt Request 2 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
1 INT1	Interrupt Request 1 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
0 INT0	Interrupt Request 0 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

20.3.14 Error Register (DMAx_ERR)

The ERR provides a bit map for the channels, signaling the presence of an error for each channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate bit in this register. The outputs of this register are enabled by the contents of the EEI, and then routed to the interrupt controller. During the execution of the interrupt-service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. Typically, a write to the CERR in the interrupt-service routine is used for this purpose. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when an error is detected.

The contents of this register can also be polled because a non-zero value indicates the presence of a channel error regardless of the state of the EEI. The state of any given channel's error indicators is affected by writes to this register; it is also affected by writes to the CERR. On writes to the ERR, a one in any bit position clears the corresponding channel's error status. A zero in any bit position has no affect on the corresponding channel's current error status. The CERR is provided so the error indicator for a single channel can easily be cleared.

Address: 4000_8000h base + 2Ch offset = 4000_802Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								ERR7	ERR6	ERR5	ERR4	ERR3	ERR2	ERR1	ERR0
W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMAx_ERR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 ERR7	Error In Channel 7 0 An error in this channel has not occurred 1 An error in this channel has occurred
6 ERR6	Error In Channel 6 0 An error in this channel has not occurred 1 An error in this channel has occurred
5 ERR5	Error In Channel 5 0 An error in this channel has not occurred 1 An error in this channel has occurred
4 ERR4	Error In Channel 4 0 An error in this channel has not occurred 1 An error in this channel has occurred
3 ERR3	Error In Channel 3 0 An error in this channel has not occurred 1 An error in this channel has occurred
2 ERR2	Error In Channel 2 0 An error in this channel has not occurred 1 An error in this channel has occurred
1 ERR1	Error In Channel 1 0 An error in this channel has not occurred 1 An error in this channel has occurred
0 ERR0	Error In Channel 0 0 An error in this channel has not occurred 1 An error in this channel has occurred

20.3.15 Hardware Request Status Register (DMAx_HRS)

The HRS register provides a bit map for the DMA channels, signaling the presence of a hardware request for each channel. The hardware request status bits reflect the current state of the register and qualified (via the ERQ fields) DMA request signals as seen by the DMA's arbitration logic. This view into the hardware request signals may be used for debug purposes.

NOTE

These bits reflect the state of the request as seen by the arbitration logic. Therefore, this status is affected by the ERQ bits.

Address: 4000_8000h base + 34h offset = 4000_8034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								HRS7	HRS6	HRS5	HRS4	HRS3	HRS2	HRS1	HRS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMAx_HRS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 HRS7	Hardware Request Status Channel 7 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 7 is not present 1 A hardware service request for channel 7 is present
6 HRS6	Hardware Request Status Channel 6 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 6 is not present 1 A hardware service request for channel 6 is present
5 HRS5	Hardware Request Status Channel 5

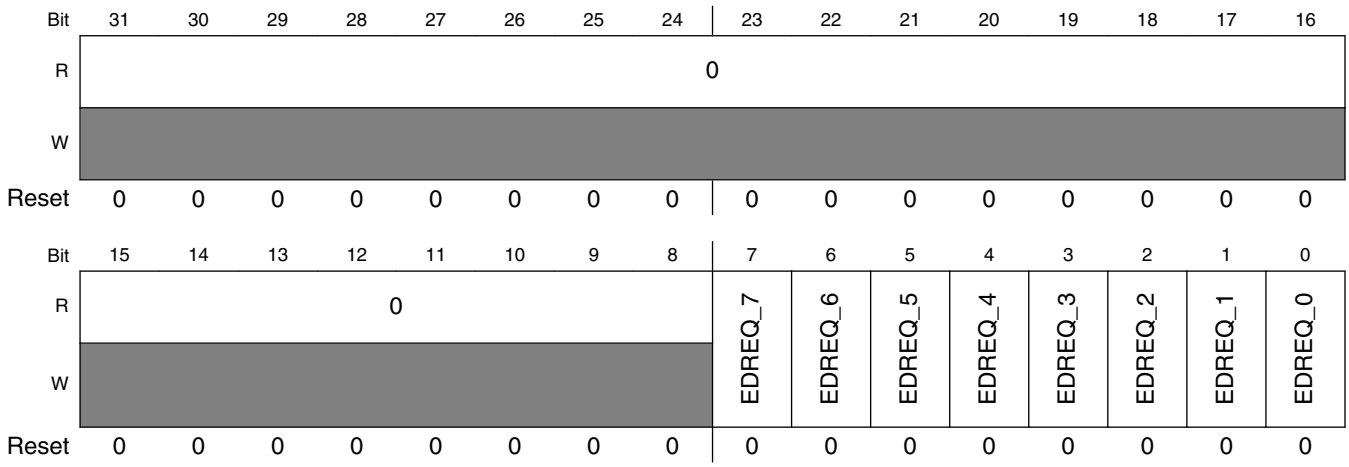
Table continues on the next page...

DMAx_HRS field descriptions (continued)

Field	Description
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 5 is not present 1 A hardware service request for channel 5 is present
4 HRS4	Hardware Request Status Channel 4 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 4 is not present 1 A hardware service request for channel 4 is present
3 HRS3	Hardware Request Status Channel 3 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 3 is not present 1 A hardware service request for channel 3 is present
2 HRS2	Hardware Request Status Channel 2 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 2 is not present 1 A hardware service request for channel 2 is present
1 HRS1	Hardware Request Status Channel 1 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 1 is not present 1 A hardware service request for channel 1 is present
0 HRS0	Hardware Request Status Channel 0 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 0 is not present 1 A hardware service request for channel 0 is present

20.3.16 Enable Asynchronous Request in Stop Register (DMAx_EARS)

Address: 4000_8000h base + 44h offset = 4000_8044h



DMAx_EARS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 EDREQ_7	Enable asynchronous DMA request in stop mode for channel 7 0 Disable asynchronous DMA request for channel 7. 1 Enable asynchronous DMA request for channel 7.
6 EDREQ_6	Enable asynchronous DMA request in stop mode for channel 6 0 Disable asynchronous DMA request for channel 6. 1 Enable asynchronous DMA request for channel 6.
5 EDREQ_5	Enable asynchronous DMA request in stop mode for channel 5 0 Disable asynchronous DMA request for channel 5. 1 Enable asynchronous DMA request for channel 5.
4 EDREQ_4	Enable asynchronous DMA request in stop mode for channel 4 0 Disable asynchronous DMA request for channel 4. 1 Enable asynchronous DMA request for channel 4.
3 EDREQ_3	Enable asynchronous DMA request in stop mode for channel 3. 0 Disable asynchronous DMA request for channel 3. 1 Enable asynchronous DMA request for channel 3.
2 EDREQ_2	Enable asynchronous DMA request in stop mode for channel 2. 0 Disable asynchronous DMA request for channel 2. 1 Enable asynchronous DMA request for channel 2.

Table continues on the next page...

DMAx_EARS field descriptions (continued)

Field	Description
1 EDREQ_1	Enable asynchronous DMA request in stop mode for channel 1. 0 Disable asynchronous DMA request for channel 1 1 Enable asynchronous DMA request for channel 1.
0 EDREQ_0	Enable asynchronous DMA request in stop mode for channel 0. 0 Disable asynchronous DMA request for channel 0. 1 Enable asynchronous DMA request for channel 0.

20.3.17 Channel n Priority Register (DMA*x*_DCHPRI*n*)

When fixed-priority channel arbitration is enabled (CR[ERCA] = 0), the contents of these registers define the unique priorities associated with each channel. The channel priorities are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. Software must program the channel priorities with unique values; otherwise, a configuration error is reported. The range of the priority value is limited to the values of 0 through 7.

Address: 4000_8000h base + 100h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read	ECP	DPA	0			CHPRI		
Write								
Reset	0	0	0	0	0	*	*	*

* Notes:
- CHPRI field: See bit field description.

DMA*x*_DCHPRI*n* field descriptions

Field	Description
7 ECP	Enable Channel Preemption. 0 Channel n cannot be suspended by a higher priority channel's service request. 1 Channel n can be temporarily suspended by the service request of a higher priority channel.
6 DPA	Disable Preempt Ability. 0 Channel n can suspend a lower priority channel. 1 Channel n cannot suspend any channel, regardless of channel priority.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CHPRI	Channel n Arbitration Priority

Table continues on the next page...

DMA*x*_DCHPRI*n* field descriptions (continued)

Field	Description
	Channel priority when fixed-priority arbitration is enabled **NOTE:** Reset value for the channel priority field, CHPRI, is equal to the corresponding channel number for each priority register, that is, DCHPRI7[CHPRI] = 0b111.

20.3.18 TCD Source Address (DMA*x*_TCD*n*_SADDR)

Address: 4000_8000h base + 1000h offset + (32d × i), where i=0d to 7d

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

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_SADDR field descriptions

Field	Description
SADDR	Source Address Memory address pointing to the source data.

20.3.19 TCD Signed Source Address Offset (DMA*x*_TCD*n*_SOFF)

Address: 4000_8000h base + 1004h offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SOFF															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_SOFF field descriptions

Field	Description
SOFF	Source address signed offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

20.3.20 TCD Transfer Attributes (DMA*x*_TCD*n*_ATTR)

Address: 4000_8000h base + 1006h offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SMOD					SSIZE			DMOD					DSIZE		
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_ATTR field descriptions

Field	Description
15–11 SMOD	Source Address Modulo 0 Source address modulo feature is disabled ≠0 This value defines a specific address range specified to be the value after SADDR + SOFF calculation is performed on the original register value. Setting this field provides the ability to implement a circular data queue easily. For data queues requiring power-of-2 size bytes, the queue should start at a 0-modulo-size address and the SMOD field should be set to the appropriate value for the queue, freezing the desired number of upper address bits. The value programmed into this field specifies the number of lower address bits allowed to change. For a circular queue application, the SOFF is typically set to the transfer size to implement post-increment addressing with the SMOD function constraining the addresses to a 0-modulo-size range.
10–8 SSIZE	Source data transfer size **NOTE:** Using a Reserved value causes a configuration error. 000 8-bit 001 16-bit 010 32-bit 011 Reserved 100 16-byte 101 32-byte 110 Reserved 111 Reserved
7–3 DMOD	Destination Address Modulo See the SMOD definition
DSIZE	Destination data transfer size See the SSIZE definition

20.3.21 TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA*x*_TCD*n*_NBYTES_MLNO)

This register, or one of the next two registers (TCD_NBYTES_MLOFFNO, TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is disabled (CR[EMLM] = 0)

If minor loop mapping is enabled, see the TCD_NBYTES_MLOFFNO and TCD_NBYTES_MLOFFYES register descriptions for the definition of TCD word 2.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 7d

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

* Notes:

- x = Undefined at reset.

DMA*x*_TCD*n*_NBYTES_MLNO field descriptions

Field	Description
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed. **NOTE:** An NBYTES value of 0x0000_0000 is interpreted as a 4 GB transfer.

20.3.22 TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA*x*_TCD*n*_NBYTES_MLOFFNO)

One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- SMLOE = 0 and DMLOE = 0

If minor loop mapping is enabled and SMLOE or DMLOE is set, then refer to the TCD_NBYTES_MLOFFYES register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SMLOE	DMLOE	NBYTES													
W																
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
R	NBYTES															
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_NBYTES_MLOFFNO field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

20.3.23 TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA*x*_TCD*n*_NBYTES_MLOFFYES)

One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFNO), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- Minor loop offset is enabled (SMLOE or DMLOE = 1)

If minor loop mapping is enabled and SMLOE and DMLOE are cleared, then refer to the TCD_NBYTES_MLOFFNO register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 7d

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_NBYTES_MLOFFYES field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion.

Table continues on the next page...

DMA*x*_TCD*n*_NBYTES_MLOFFYES field descriptions (continued)

Field	Description
	0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
29–10 MLOFF	If SMLOE or DMLOE is set, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

20.3.24 TCD Last Source Address Adjustment (DMA*x*_TCD*n*_SLAST)

Address: 4000_8000h base + 100Ch offset + (32d × i), where i=0d to 7d

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

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_SLAST field descriptions

Field	Description
SLAST	Last Source Address Adjustment Adjustment value added to the source address at the completion of the major iteration count. This value can be applied to restore the source address to the initial value, or adjust the address to reference the next data structure. This register uses two's complement notation; the overflow bit is discarded.

20.3.25 TCD Destination Address (DMA*x*_TCD*n*_DADDR)

Address: 4000_8000h base + 1010h offset + (32d × i), where i=0d to 7d

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

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_DADDR field descriptions

Field	Description
DADDR	Destination Address Memory address pointing to the destination data.

20.3.26 TCD Signed Destination Address Offset (DMA*x*_TCD*n*_DOFF)

Address: 4000_8000h base + 1014h offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	DOFF															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_DOFF field descriptions

Field	Description
DOFF	Destination Address Signed Offset Sign-extended offset applied to the current destination address to form the next-state value as each destination write is completed.

20.3.27 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA*x*_TCD*n*_CITER_ELINKYES)

If TCDn_CITER[ELINK] is set, the TCDn_CITER register is defined as follows.

Address: 4000_8000h base + 1016h offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	0			LINKCH			CITER
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMAx_TCDn_CITER_ELINKYES field descriptions

Field	Description
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–12 Reserved	This field is reserved.
11–9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request to the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

20.3.28 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA*x*_TCD*n*_CITER_ELINKNO)

If TCDn_CITER[ELINK] is cleared, the TCDn_CITER register is defined as follows.

Address: 4000_8000h base + 1016h offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	CITER						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_CITER_ELINKNO field descriptions

Field	Description
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

20.3.29 TCD Last Destination Address Adjustment/Scatter Gather Address (DMA*x*_TCD*n*_DLASTSGA)

Address: 4000_8000h base + 1018h offset + (32d × i), where i=0d to 7d

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

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_DLASTSGA field descriptions

Field	Description
DLASTSGA	Destination last address adjustment or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo-32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo-32-byte, otherwise a configuration error is reported.

20.3.30 TCD Control and Status (DMA*x*_TCD*n*_CSR)

Address: 4000_8000h base + 101Ch offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8
Read	BWC					MAJORLINKCH		
Write			0					
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read	DONE	ACTIVE	MAJORELINK	ESG	DREQ	INTHALF	INTMAJOR	START
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_CSR field descriptions

Field	Description
15–14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces the eDMA to stall after the completion of each read/write access to control the bus request bandwidth seen by the crossbar switch. 00 No eDMA engine stalls. 01 Reserved 10 eDMA engine stalls for 4 cycles after each R/W. 11 eDMA engine stalls for 8 cycles after each R/W.
13–11 Reserved	This field is reserved.
10–8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
7 DONE	Channel Done This flag indicates the eDMA has completed the major loop. The eDMA engine sets it as the CITER count reaches zero. The software clears it, or the hardware when the channel is activated. **NOTE:** This bit must be cleared to write the MAJORELINK or ESG bits.
6 ACTIVE	Channel Active This flag signals the channel is currently in execution. It is set when channel service begins, and is cleared by the eDMA as the minor loop completes or when any error condition is detected.
5 MAJORELINK	Enable channel-to-channel linking on major loop complete As the channel completes the major loop, this flag enables the linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. **NOTE:** To support the dynamic linking coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0 The channel-to-channel linking is disabled. 1 The channel-to-channel linking is enabled.
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo-32 address containing a 32-byte data structure loaded as the transfer control descriptor into the local memory. **NOTE:** To support the dynamic scatter/gather coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0 The current channel's TCD is normal format. 1 The current channel's TCD specifies a scatter gather format. The DLASTSGA field provides a memory pointer to the next TCD to be loaded into this channel after the major loop completes its execution.

Table continues on the next page...

DMA*x*_TCD*n*_CSR field descriptions (continued)

Field	Description
3 DREQ	Disable Request If this flag is set, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches zero. 0 The channel's ERQ bit is not affected. 1 The channel's ERQ bit is cleared when the major loop is complete.
2 INTHALF	Enable an interrupt when major counter is half complete. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT register when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER == (BITER >> 1)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE:** If BITER = 1, do not use INTHALF. Use INTMAJOR instead. 0 The half-point interrupt is disabled. 1 The half-point interrupt is enabled.
1 INTMAJOR	Enable an interrupt when major iteration count completes. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT when the current major iteration count reaches zero. 0 The end-of-major loop interrupt is disabled. 1 The end-of-major loop interrupt is enabled.
0 START	Channel Start If this flag is set, the channel is requesting service. The eDMA hardware automatically clears this flag after the channel begins execution. 0 The channel is not explicitly started. 1 The channel is explicitly started via a software initiated service request.

20.3.31 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA*x*_TCD*n*_BITER_ELINKYES)

If the TCDn_BITER[ELINK] bit is set, the TCDn_BITER register is defined as follows.

Address: 4000_8000h base + 101Eh offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	0			LINKCH			BITER
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_BITER_ELINKYES field descriptions

Field	Description
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–12 Reserved	This field is reserved.
11–9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.
BITER	Starting major iteration count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.

Table continues on the next page...

DMA*x*_TCD*n*_BITER_ELINKYES field descriptions (continued)

Field	Description
	NOTE: When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

20.3.32 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA*x*_TCD*n*_BITER_ELINKNO)

If the TCDn_BITER[ELINK] bit is cleared, the TCDn_BITER register is defined as follows.

Address: 4000_8000h base + 101Eh offset + (32d × i), where i=0d to 7d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	BITER						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA*x*_TCD*n*_BITER_ELINKNO field descriptions

Field	Description
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.

Table continues on the next page...

DMA*x*_TCD*n*_BITER_ELINKNO field descriptions (continued)

Field	Description
	NOTE: When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field is reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

20.4 Functional description

The operation of the eDMA is described in the following subsections.

20.4.1 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

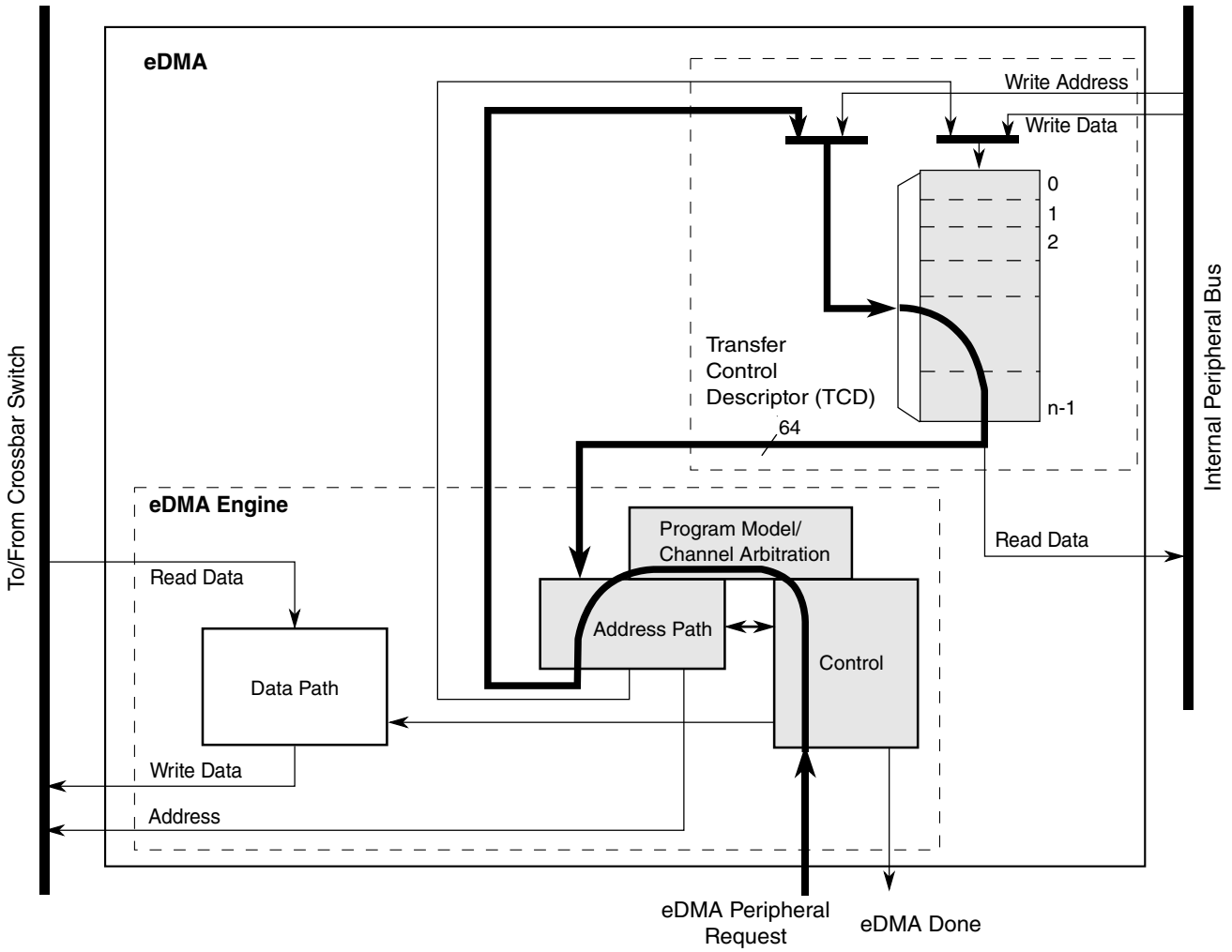


Figure 20-2. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] bit follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration. In the next cycle, the channel arbitration performs, using the fixed-priority or round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor read from the local memory and loaded into the eDMA engine address path channel x or y registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path channel x or y registers.

The following diagram illustrates the second part of the basic data flow:

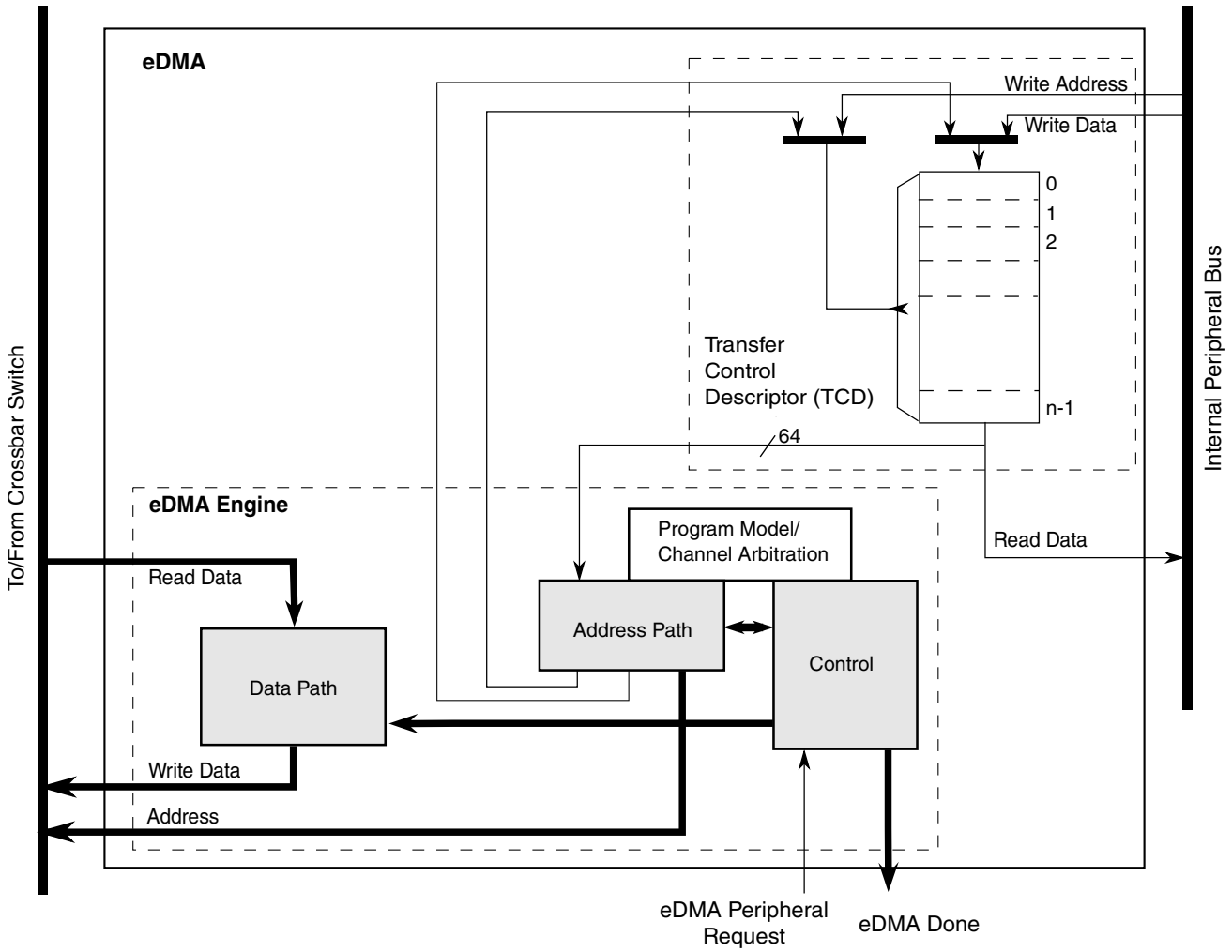


Figure 20-3. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) sequence through the required source reads and destination writes to perform the actual data movement. The source reads are initiated and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the minor byte count has transferred.

After the minor byte count has moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD, for example, SADDR, DADDR, CITER. If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

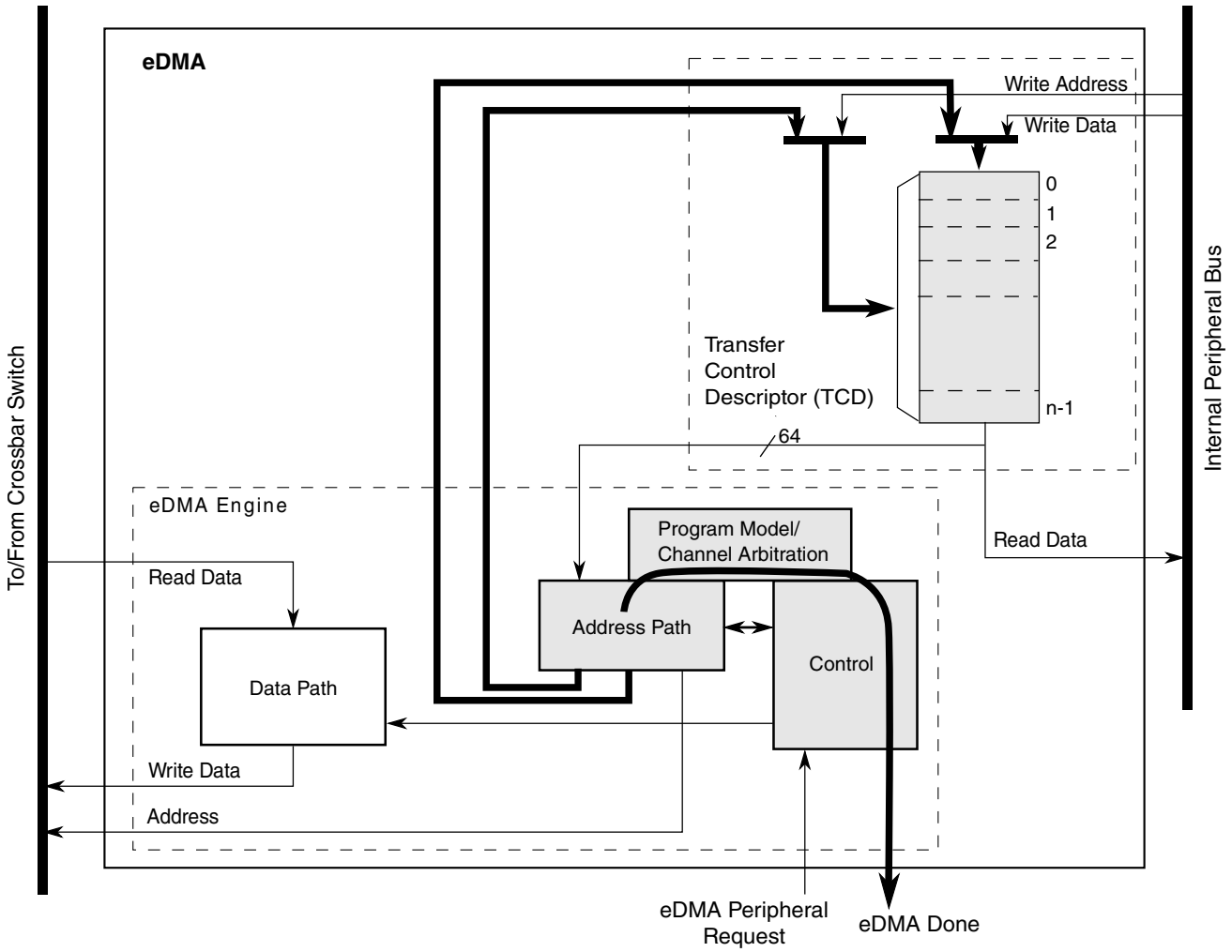


Figure 20-4. eDMA operation, part 3

20.4.2 Fault reporting and handling

Channel errors are reported in the Error Status register (DMAx_ES) and can be caused by:

- A configuration error, which is an illegal setting in the transfer-control descriptor or an illegal priority register setting in Fixed-Arbitration mode, or
- An error termination to a bus master read or write cycle

A configuration error is reported when the starting source or destination address, source or destination offsets, minor loop byte count, or the transfer size represent an inconsistent state. Each of these possible causes are detailed below:

- The addresses and offsets must be aligned on 0-modulo-transfer-size boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.

- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size respectively.
- In fixed arbitration mode, a configuration error is caused by any two channel priorities being equal. All channel priority levels must be unique when fixed arbitration mode is enabled.

NOTE

When two channels have the same priority, a channel priority error exists and will be reported in the Error Status register. However, the channel number will not be reported in the Error Status register. When all of the channel priorities within a group are not unique, the channel number selected by arbitration is undetermined.

To aid in Channel Priority Error (CPE) debug, set the Halt On Error bit in the DMA's Control Register. If all of the channel priorities within a group are not unique, the DMA will be halted after the CPE error is recorded. The DMA will remain halted and will not process any channel service requests. Once all of the channel priorities are set to unique numbers, the DMA may be enabled again by clearing the Halt bit.

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[E_LINK] bit does not equal the TCDn_BITER[E_LINK] bit.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, report as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion when properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error

occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

A transfer may be cancelled by software with the CR[CX] bit. When a cancel transfer request is recognized, the DMA engine stops processing the channel. The current read-write sequence is allowed to finish. If the cancel occurs on the last read-write sequence of a major or minor loop, the cancel request is discarded and the channel retires normally.

The error cancel transfer is the same as a cancel transfer except the Error Status register (DMAx_ES) is updated with the cancelled channel number and ECX is set. The TCD of a cancelled channel contains the source and destination addresses of the last transfer saved in the TCD. If the channel needs to be restarted, you must re-initialize the TCD because the aforementioned fields no longer represent the original parameters. When a transfer is cancelled by the error cancel transfer mechanism, the channel number is loaded into DMA_ES[ERRCHN] and ECX and VLD are set. In addition, an error interrupt may be generated if enabled.

NOTE

The cancel transfer request allows the user to stop a large data transfer in the event the full data transfer is no longer needed. The cancel transfer bit does not abort the channel. It simply stops the transferring of data and then retires the channel through its normal shutdown sequence. The application software must handle the context of the cancel. If an interrupt is desired (or not), then the interrupt should be enabled (or disabled) before the cancel request. The application software must clean up the transfer control descriptor since the full transfer did not occur.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel bit in the eDMA error register is asserted. At the same time, the details of the error condition are loaded into the Error Status register (DMAx_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request, are not affected when an error is detected. After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

20.4.3 Channel preemption

Channel preemption is enabled on a per-channel basis by setting the DCHPRIn[ECP] bit. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher priority channel. After the preempting channel has completed all its minor loop data transfers, the preempted channel is restored and resumes execution. After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended and the higher priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted. Preemption is available only when fixed arbitration is selected.

A channel's ability to preempt another channel can be disabled by setting DCHPRIn[DPA]. When a channel's preempt ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This allows for a pool of low priority, large data-moving channels to be defined. These low priority channels can be configured to not preempt each other, thus preventing a low priority channel from consuming the preempt slot normally available to a true, high priority channel.

20.4.4 Performance

This section addresses the performance of the eDMA module, focusing on two separate metrics:

- In the traditional data movement context, performance is best expressed as the peak data transfer rates achieved using the eDMA. In most implementations, this transfer rate is limited by the speed of the source and destination address spaces.
- In a second context where device-paced movement of single data values to/from peripherals is dominant, a measure of the requests that can be serviced in a fixed time is a more relevant metric. In this environment, the speed of the source and destination address spaces remains important. However, the microarchitecture of the eDMA also factors significantly into the resulting metric.

20.4.4.1 Peak transfer rates

The peak transfer rates for several different source and destination transfers are shown in the following tables. These tables assume:

- Internal SRAM can be accessed with zero wait-states when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states, when viewed from the system bus data phase
- All internal peripheral bus accesses are 32-bits in size

NOTE

All architectures will not meet the assumptions listed above. See the SRAM configuration section for more information.

This table compares peak transfer rates based on different possible system speeds. Specific chips/devices may not support all system speeds listed.

Table 20-4. eDMA peak transfer rates (Mbytes/sec)

System Speed, Width	Internal SRAM-to-Internal SRAM	32 bit internal peripheral bus-to-Internal SRAM	Internal SRAM-to-32 bit internal peripheral bus
66.7 MHz, 32 bit	133.3	66.7	53.3
83.3 MHz, 32 bit	166.7	83.3	66.7
100.0 MHz, 32 bit	200.0	100.0	80.0
133.3 MHz, 32 bit	266.7	133.3	106.7
150.0 MHz, 32 bit	300.0	150.0	120.0

Internal-SRAM-to-internal-SRAM transfers occur at the core's datapath width. For all transfers involving the internal peripheral bus, 32-bit transfer sizes are used. In all cases, the transfer rate includes the time to read the source plus the time to write the destination.

20.4.4.2 Peak request rates

The second performance metric is a measure of the number of DMA requests that can be serviced in a given amount of time. For this metric, assume that the peripheral request causes the channel to move a single internal peripheral bus-mapped operand to/from internal SRAM. The same timing assumptions used in the previous example apply to this calculation. In particular, this metric also reflects the time required to activate the channel.

The eDMA design supports the following hardware service request sequence. Note that the exact timing from Cycle 7 is a function of the response times for the channel's read and write accesses. In the case of an internal peripheral bus read and internal SRAM write, the combined data phase time is 4 cycles. For an SRAM read and internal peripheral bus write, it is 5 cycles.

Table 20-5. Hardware service request process

Cycle		Description
With internal peripheral bus read and internal SRAM write	With SRAM read and internal peripheral bus write	
1		eDMA peripheral request is asserted.
2		The eDMA peripheral request is registered locally in the eDMA module and qualified. TCD*n*_CSR[START] bit initiated requests start at this point with the registering of the user write to TCD*n* word 7.
3		Channel arbitration begins.
4		Channel arbitration completes. The transfer control descriptor local memory read is initiated.
5–6		The first two parts of the activated channel's TCD is read from the local memory. The memory width to the eDMA engine is 64 bits, so the entire descriptor can be accessed in four cycles
7		The first system bus read cycle is initiated, as the third part of the channel's TCD is read from the local memory. Depending on the state of the crossbar switch, arbitration at the system bus may insert an additional cycle of delay here.
8–11	8–12	The last part of the TCD is read in. This cycle represents the first data phase for the read, and the address phase for the destination write.
12	13	This cycle represents the data phase of the last destination write.
13	14	The eDMA engine completes the execution of the inner minor loop and prepares to write back the required TCD*n* fields into the local memory. The TCD*n* word 7 is read and checked for channel linking or scatter/gather requests.
14	15	The appropriate fields in the first part of the TCD*n* are written back into the local memory.
15	16	The fields in the second part of the TCD*n* are written back into the local memory. This cycle coincides with the next channel arbitration cycle start.
16	17	The next channel to be activated performs the read of the first part of its TCD from the local memory. This is equivalent to Cycle 4 for the first channel's service request.

Assuming zero wait states on the system bus, DMA requests can be processed every 9 cycles. Assuming an average of the access times associated with internal peripheral bus-to-SRAM (4 cycles) and SRAM-to-internal peripheral bus (5 cycles), DMA requests can be processed every 11.5 cycles (4 + (4+5)/2 + 3). This is the time from Cycle 4 to Cycle x +5. The resulting peak request rate, as a function of the system frequency, is shown in the following table.

Table 20-6. eDMA peak request rate (MReq/sec)

System frequency (MHz)	Request rate with zero wait states	Request rate with wait states
66.6	7.4	5.8
83.3	9.2	7.2
100.0	11.1	8.7
133.3	14.8	11.6
150.0	16.6	13.0

A general formula to compute the peak request rate with overlapping requests is:

PEAKreq = freq / [entry + (1 + read_ws) + (1 + write_ws) + exit]

where:

Table 20-7. Peak request formula operands

Operand	Description
PEAKreq	Peak request rate
freq	System frequency
entry	Channel startup (4 cycles)
read_ws	Wait states seen during the system bus read data phase
write_ws	Wait states seen during the system bus write data phase
exit	Channel shutdown (3 cycles)

20.4.4.3 eDMA performance example

Consider a system with the following characteristics:

- Internal SRAM can be accessed with one wait-state when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states viewed from the system bus data phase
- System operates at 150 MHz

For an SRAM to internal peripheral bus transfer,

PEAKreq = 150 MHz / [4 + (1 + 1) + (1 + 3) + 3] cycles = 11.5 Mreq/sec

For an internal peripheral bus to SRAM transfer,

PEAKreq = 150 MHz / [4 + (1 + 2) + (1 + 1) + 3] cycles = 12.5 Mreq/sec

Assuming an even distribution of the two transfer types, the average peak request rate would be:

PEAKreq = (11.5 Mreq/sec + 12.5 Mreq/sec) / 2 = 12.0 Mreq/sec

The minimum number of cycles to perform a single read/write, zero wait states on the system bus, from a cold start where no channel is executing and eDMA is idle are:

- 11 cycles for a software, that is, a TCD*n*_CSR[START] bit, request
- 12 cycles for a hardware, that is, an eDMA peripheral request signal, request

Two cycles account for the arbitration pipeline and one extra cycle on the hardware request resulting from the internal registering of the eDMA peripheral request signals. For the peak request rate calculations above, the arbitration and request registering is absorbed in or overlaps the previous executing channel.

Note

When channel linking or scatter/gather is enabled, a two cycle delay is imposed on the next channel selection and startup. This allows the link channel or the scatter/gather channel to be eligible and considered in the arbitration pool for next channel selection.

20.5 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

20.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the CR if a configuration other than the default is desired.
2. Write the channel priority levels to the DCHPRI*n* registers if a configuration other than the default is desired.
3. Enable error interrupts in the EEI register if so desired.
4. Write the 32-byte TCD for each channel that may request service.

5. Enable any hardware service requests via the ERQ register.
6. Request channel service via either:
   - Software: setting the TCD*n*_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in the following table, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, as defined by TCD*n*_SADDR, to the destination, as defined by TCD*n*_DADDR, continue until the number of bytes specified by TCD*n*_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, further post processing executes, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 20-8. TCD Control and Status fields

TCD*n*_CSR field name	Description
START	Control bit to start channel explicitly when using a software initiated DMA service (Automatically cleared by hardware)
ACTIVE	Status bit indicating the channel is currently in execution
DONE	Status bit indicating major loop completion (cleared by software when using a software initiated DMA service)
D_REQ	Control bit to disable DMA request at end of major loop completion when using a hardware initiated DMA service
BWC	Control bits for throttling bandwidth control of a channel
E_SG	Control bit to enable scatter-gather feature
INT_HALF	Control bit to enable interrupt when major loop is half complete
INT_MAJ	Control bit to enable interrupt when major loop completes

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

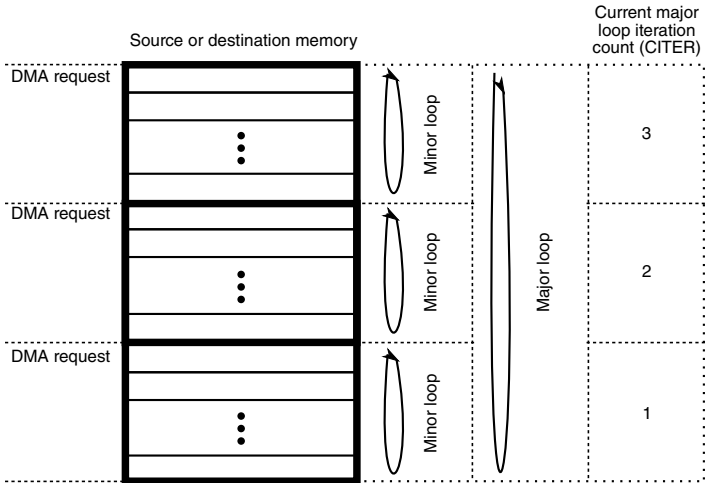


Figure 20-5. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

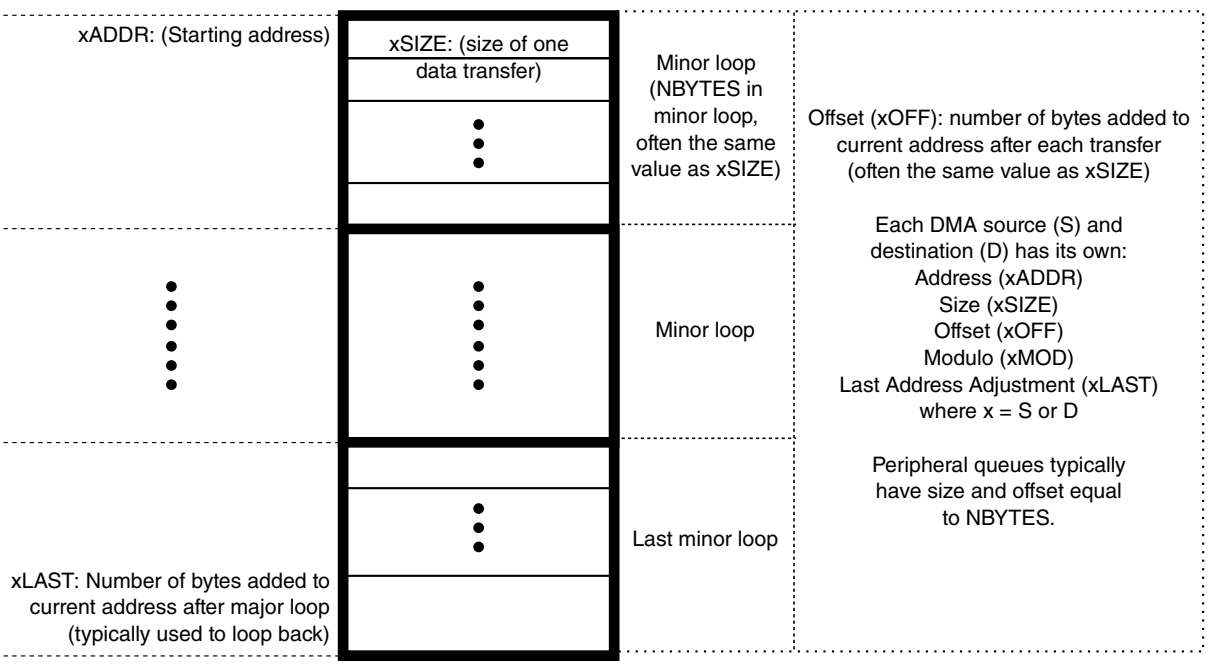


Figure 20-6. Memory array terms

20.5.2 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data. Most programming errors are reported on a per channel basis with the exception of channel priority error (ES[CPE]).

For all error types other than channel priority error, the channel number causing the error is recorded in the Error Status register (DMAx_ES). If the error source is not removed before the next activation of the problem channel, the error is detected and recorded again.

If priority levels are not unique, when any channel requests service, a channel priority error is reported. The highest channel priority with an active request is selected, but the lowest numbered channel with that priority is selected by arbitration and executed by the eDMA engine. The hardware service request handshake signals, error interrupts, and error reporting is associated with the selected channel.

20.5.3 Arbitration mode considerations

This section discusses arbitration considerations for the eDMA.

20.5.3.1 Fixed channel arbitration

In this mode, the channel service request from the highest priority channel is selected to execute.

20.5.3.2 Round-robin channel arbitration

Channels are serviced starting with the highest channel number and rotating through to the lowest channel number without regard to the channel priority levels.

20.5.4 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

20.5.4.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the TCD*n*_CSR[DONE] bit is set and an interrupt generates if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INT_MAJ] = 1
TCDn_CSR[START] = 1 (Should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] bit requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
    b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
    c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
    d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
    e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
    f. Write 32-bits to location 0x2008 → third iteration of the minor loop.
    g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.

h. Write 32-bits to location 0x200C → last iteration of the minor loop → major loop complete.

6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[*n*] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

20.5.4.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop transferring 16 bytes per iteration. After the channel's hardware requests are enabled in the ERQ register, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware, that is, eDMA peripheral, request for channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source to destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.

f. Write 32-bits to location 0x2008 → third iteration of the minor loop.

g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.

h. Write 32-bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0.
8. The channel retires → one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware, that is, eDMA peripheral, requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: channel TCD data from local memory to internal register file.
13. The source to destination transfers are executed as follows:

    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.

    b. Write 32-bits to location 0x2010 → first iteration of the minor loop.

    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.

    d. Write 32-bits to location 0x2014 → second iteration of the minor loop.

    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.

    f. Write 32-bits to location 0x2018 → third iteration of the minor loop.

    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.

    h. Write 32-bits to location 0x201C → last iteration of the minor loop → major loop complete.

14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).

15. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[n] = 1.
16. The channel retires → major loop complete. The eDMA goes idle or services the next channel.

20.5.4.3 Using the modulo feature

The modulo feature of the eDMA provides the ability to implement a circular data queue in which the size of the queue is a power of 2. MOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value while the 28 upper address bits (0x1234567*x*) retain their original value. In this example the source address is set to 0x12345670, the offset is set to 4 bytes and the MOD field is set to 4, allowing for a 2^4 byte (16-byte) size queue.

Table 20-9. Modulo example

Transfer Number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

20.5.5 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

20.5.5.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software initiated service requests. The first is to read the TCD*n*_CITER field and test for a change. Another method may be extracted from the sequence shown below. The second method is

to test the TCD*n*_CSR[START] bit and the TCD*n*_CSR[ACTIVE] bit. The minor-loop-complete condition is indicated by both bits reading zero after the TCD*n*_CSR[START] was set. Polling the TCD*n*_CSR[ACTIVE] bit may be inconclusive, because the active status may be missed if the channel execution is short in duration.

The TCD status bits execute the following sequence for a software activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	1	0	0	Channel service request via software
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

The best method to test for minor-loop completion when using hardware, that is, peripheral, initiated service requests is to read the TCD*n*_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status bits execute the following sequence for a hardware-activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	0	0	0	Channel service request via hardware (peripheral request asserted)
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

For both activation types, the major-loop-complete status is explicitly indicated via the TCD*n*_CSR[DONE] bit.

The TCD*n*_CSR[START] bit is cleared automatically when the channel begins execution regardless of how the channel activates.

20.5.5.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_NBYTES values if read while a channel executes. The true values of the SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file and not the values in the TCD local memory for that channel. The addresses, SADDR and

DADDR, and NBYTES, which decrement to zero as the transfer progresses, can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

20.5.5.3 Checking channel preemption status

Preemption is available only when fixed arbitration is selected as the channel arbitration mode. A preemptive situation is one in which a preempt-enabled channel runs and a higher priority request becomes active. When the eDMA engine is not operating in fixed channel arbitration mode, the determination of the actively running relative priority outstanding requests become undefined. Channel priorities are treated as equal, that is, constantly rotating, when Round-Robin Arbitration mode is selected.

The TCD*n*_CSR[ACTIVE] bit for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended while the preempting channel executes one major loop iteration. If two TCD*n*_CSR[ACTIVE] bits are set simultaneously in the global TCD map, a higher priority channel is actively preempting a lower priority channel.

20.5.6 Channel Linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD*n*_CSR[START] bit of another channel (or itself), therefore initiating a service request for that channel. When properly enabled, the EDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCD*n*_CITER[E_LINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, the initial fields of:

```
TCDn_CITER[E_LINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJOR_E_LINK] = 1
TCDn_CSR[MAJOR_LINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] bit

2. Minor loop done → set TCD12_CSR[START] bit
3. Minor loop done → set TCD12_CSR[START] bit
4. Minor loop done, major loop done→ set TCD7_CSR[START] bit

When minor loop linking is enabled (TCD*n*_CITER[E_LINK] = 1), the TCD*n*_CITER[CITER] field uses a nine bit vector to form the current iteration count. When minor loop linking is disabled (TCD*n*_CITER[E_LINK] = 0), the TCD*n*_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCD*n*_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

Note

The TCD*n*_CITER[E_LINK] bit and the TCD*n*_BITER[E_LINK] bit must equal or a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop, half-way done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, i.e, use another channel's TCD, at the end of a loop.

Table 20-10. Channel Linking Parameters

Desired Link Behavior	TCD Control Field Name	Description
Link at end of Minor Loop	CITER[E_LINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of Major Loop	CSR[MAJOR_E_LINK]	Enable channel-to-channel linking on major loop completion
	CSR[MAJOR_LINKCH]	Link channel number when linking at end of major loop

20.5.7 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

20.5.7.1 Dynamically changing the channel priority

The following two options are recommended for dynamically changing channel priority levels:

1. Switch to Round-Robin Channel Arbitration mode, change the channel priorities, then switch back to Fixed Arbitration mode,
2. Disable all the channels, change the channel priorities, then enable the appropriate channels.

20.5.7.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCD.major.e_link bit during channel execution (see the diagram in TCD structure). This bit is read from the TCD local memory at the end of channel execution, thus allowing the user to enable the feature during channel execution.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic channel link by enabling the TCD.major.e_link bit at the same time the eDMA engine is retiring the channel. The TCD.major.e_link would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

The following coherency model is recommended when executing a dynamic channel link request.

1. Write 1 to the TCD.major.e_link bit.
2. Read back the TCD.major.e_link bit.
3. Test the TCD.major.e_link request status:
   - If TCD.major.e_link = 1, the dynamic link attempt was successful.
   - If TCD.major.e_link = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCD.major.e_link bit to zero on any writes to a channel's TCD.word7 after that channel's TCD.done bit is set, indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link bit. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

20.5.7.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources.When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in eDMA programmer's model, thus replacing the current descriptor.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic scatter/gather operation by enabling the TCD.e_sg bit at the same time the eDMA engine is retiring the channel. The TCD.e_sg would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods for this coherency model are shown in the following subsections. Method 1 has the advantage of reading the major.linkch field and the e_sg bit with a single read. For both dynamic channel linking and scatter/gather requests, the TCD local memory controller forces the TCD.major.e_link and TCD.e_sg bits to zero on any writes to a channel's TCD.word7 if that channel's TCD.done bit is set indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link or TCD.e_sg bits. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

20.5.7.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCD.major.e_link bit is zero, the TCD.major.linkch field is not used by the eDMA. In this case, the TCD.major.linkch bits may be used for other purposes. This method uses the TCD.major.linkch field as a TCD indentification (ID).

1. When the descriptors are built, write a unique TCD ID in the TCD.major.linkch field for each TCD associated with a channel using dynamic scatter/gather.
2. Write 1b to the TCD.d_req bit.

Should a dynamic scatter/gather attempt fail, setting the TCD.d_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offest value.

3. Write the TCD.dlast_sga field with the scatter/gather address.
4. Write 1b to the TCD.e_sg bit.
5. Read back the 16 bit TCD control/status field.
6. Test the TCD.e_sg request status and TCD.major.linkch value:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b and the major.linkch (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If e_sg = 0b and the major.linkch (ID) changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

20.5.7.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCD.dlast_sga field as a TCD indentification (ID).

1. Write 1b to the TCD.d_req bit.

   Should a dynamic scatter/gather attempt fail, setting the d_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offest value.

2. Write theTCD.dlast_sga field with the scatter/gather address.
3. Write 1b to the TCD.e_sg bit.
4. Read back the TCD.e_sg bit.
5. Test the TCD.e_sg request status:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b, read the 32 bit TCD dlast_sga field.

   If e_sg = 0b and the dlast_sga did not change, the attempted dynamic link did not succeed (the channel was already retiring).

If e_sg = 0b and the dlast_sga changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

Chapter 21
Smart Card Interface Module (EMV SIM)

21.1 Introduction

The Euro, MasterCard, Visa Subscriber Identification Module (EMV SIM) is designed to facilitate communication to Smart Cards compatible to the EMV ver4.3 standard (Book 1) and Smart Cards compatible with ISO/IEC 7816-3 Standard.

21.1.1 Features

The EMV SIM module supports the following features:

- Supports Smart Cards based on the EMV Standard v4.3 and ISO 7816-3 standard
- Independent clock for SIM logic (transmitter + receiver) and independent clock for register read-write interface
- 4 byte deep FIFO for transmitter and receiver
- Automatic NACK generation on parity error and receiver FIFO overflow error
- Support for both Inverse and Direct conventions
- Re-transmission of byte upon Smart Card NACK request with programmable threshold of re-transmissions
- Auto detection of Initial Character in receiver and setting of data format (inverse or direct)
- NACK detection in receiver
- Independent timers to measure character wait time, block wait time and block guard time
- Two general purpose counters available for use by software application with programmable clock selection for the counters
- DMA support available to transfer data to/from FIFOs. Programmable option available to select interrupt or DMA feature

- Programmable Prescaler to generate the desired frequency for Card Clock and Baud Rate Divisor to generate the internal ETU clocks for transmitter and receiver for any F/D ratio
- Deep sleep wake-up via Smart Card presence detect interrupt
- Manual control of all Smart Card interface signals
- Automatic power down of port logic on Smart Card presence detect
- Support for 8-bit LRC and 16-bit CRC generation for bytes sent out from transmitter and checking incoming message checksum for receiver

21.2 Block Diagram

Figure below shows the block diagram for the EMV SIM module.

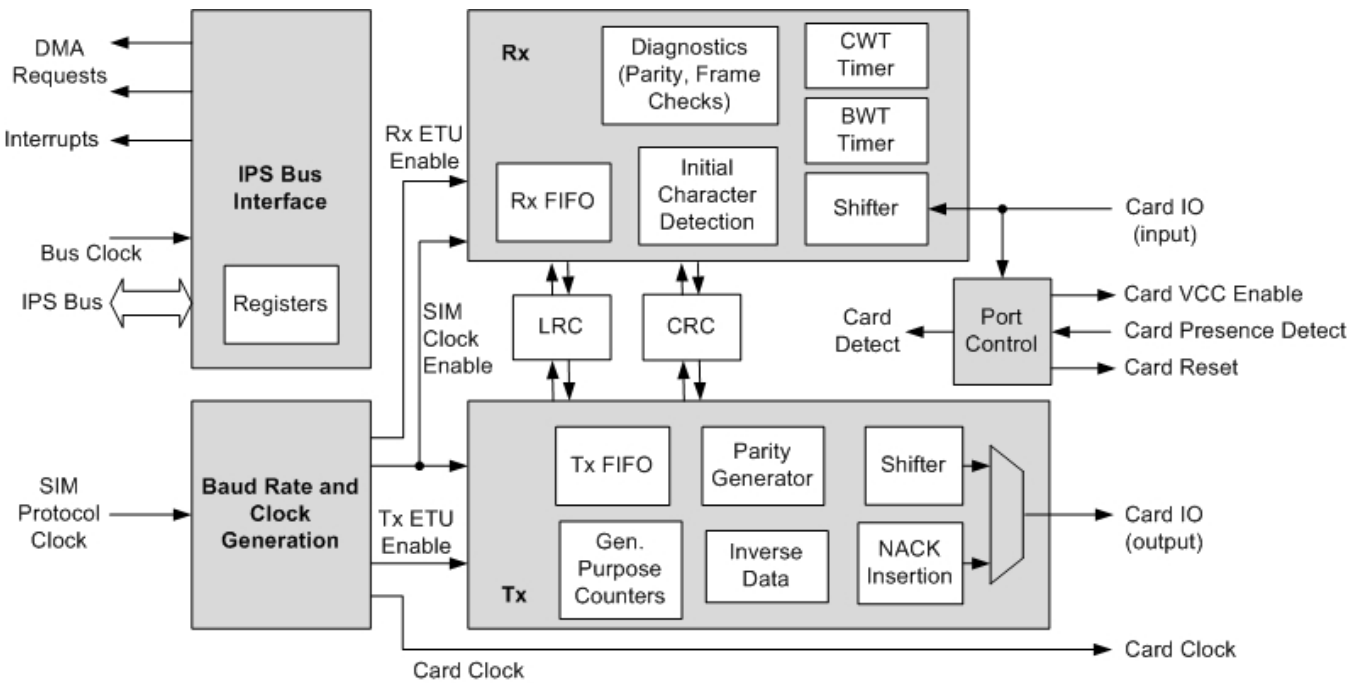


Figure 21-1. EMV SIM Block Diagram

21.3 Design Overview

The EMV SIM is designed to be compliant to the EMV ICC Specifications ver. 4.3 (dated Nov 2011) and the ISO-7816-3 standard.

The EMV SIM module is a block that is configured via the device's peripheral bus interface. The module has 32-bit aligned memory mapped registers that are used to configure and control the operations of the module. The software can write the transmit bytes or read the received bytes using either interrupt operation or DMA operation. The module can generate necessary DMA requests for the operation.

The module interfaces to the external card (Smart Card) via card clock, card reset, card V_{CC} enable and IO pins. The transmitter output from the module and the receiver input to the module are connected to the same IO pin and used in the open drain configuration. A pull-up resistor must be connected to the IO pin on board to allow proper functioning.

The key functional blocks in the module are clocking, transmitter and receiver which control the interactions with the Smart Card. These blocks function on a clock that is different and asynchronous to the register read/write clock. In order for the transmitter and receiver to function properly, the respective configuration must be done before enabling the transmitter or receiver.

The clocking module generates the card clock (output to Smart Card), and the internal clocks for the transmitter and receiver. The internal clocks control the ETU period and the sampling and driving of data on the IO line. The receive operation uses an internal over-sampled clock that oversamples the received input at 16x times the ETU rate. The prescaler generates the card clock in the range of 1 to 5 MHz with near 50% duty cycle and the baud divider generates the internal ETU clocks for transmitter and receiver.

The transmitter comprises of a 4-byte deep FIFO to store the bytes to be sent to the Smart Card. These bytes are sent to the Smart Card serially by the transmitter. The data format (i.e. inverse or direct conversion) is supported and the byte is converted into the right data format before being shifted out serially. The transmitter also supports retransmission of the current byte on the reception of a NACK. The transmitter automatically inserts NACK when the receiver indicates a parity or FIFO overflow error, if configured to do so by software. The transmitter is not required to be enabled for this. The transmitter also includes timer to insert the programmable guard time between the transmitter bytes and general purpose timers to allow software performs certain measurements. The status of the transmitter operation is intimated to software via maskable interrupts.

The receiver comprises of a 4-byte deep FIFO to store the correctly received bytes from the Smart Card. Software must ensure that it reads out the data from FIFO to prevent any overflow when more bytes are sent from the card. The receiver can be configured to detect the initial character. When configured to do so, the data format (direct or inverse convention) is also determined by the receiver and appropriate bit is set in the register map. Internal timers in the receiver measure the various wait times (Character, Block and Guard). On reception of a byte, the receiver checks the byte for Parity Error (in T=0 mode) and Framing Error. It can then direct the transmitter to insert NACK on parity error detection. For a block of bytes received, the receiver checks for LRC or CRC errors

(in T=1 mode). Other errors are stored as status for software to be aware of their occurrence. The status of the receiver operation is intimated to software via maskable interrupts.

In addition to the above blocks, the module also contains the 8-bit LRC and the 16-bit CRC blocks that are common to the transmitter and receiver. Both transmitter and receiver can use the LRC and CRC blocks and can insert the LRC or CRC into the byte stream or check the validity of the input stream using the LRC and CRC values received. Since transmit and receive operations are mutually exclusive, common LRC and CRC blocks for the transmitter and receiver can be used.

The Smart Card interface signals, except IO, which is controlled by the transmitter and receiver; are generated by the port control block which controls the assertion and de-assertion of the clock, reset and VCC signals to Smart Card. It can do so manually (software needs to explicitly write the necessary bits) or using auto power down wherein the module can itself power down the Smart Card when configured to do so by software. The port control block can also detect the Smart Card presence and generate necessary interrupts to indicate the same.

21.4 Signal Description

The following table shows the user-accessible signals available on EMV SIM. See the chip-level specification to find out which signals are actually connected to the external pins.

Table 21-1. EMV SIM Signal Description

Signal	Description	I/O
EMVSIM_SCLK	Card Clock. Clock to Smart Card.	O
EMVSIM_SRST	Card Reset. Reset signal to Smart Card	O
EMVSIM_VCC_EN	Card Power Enable. This signal controls the power to Smart Card	O
EMVSIM_IO	Card Data Line. Bi-directional data line.	I/O
EMVSIM_PD	Card Presence Detect. Signal indicating presence or removal of card	I

21.5 Memory Map and Registers

The memory map comprises of 32-bit aligned registers which can be accessed via 8-bit, 16-bit, or 32-bit accesses. Write access to reserved locations will generate transfer error. Read access to reserved locations will also generate transfer error and the read data bus will show all 0s.

The module will not check for correctness of programmed values in the registers and software must ensure that correct values are being written.

EMVSIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_E000	Version ID Register (EMVSIM0_VER_ID)	32	R	0000_0000h	21.5.1/480
4004_E004	Parameter Register (EMVSIM0_PARAM)	32	R	0000_0404h	21.5.2/480
4004_E008	Clock Configuration Register (EMVSIM0_CLKCFG)	32	R/W	0000_0000h	21.5.3/481
4004_E00C	Baud Rate Divisor Register (EMVSIM0_DIVISOR)	32	R/W	0000_0174h	21.5.4/482
4004_E010	Control Register (EMVSIM0_CTRL)	32	R/W	0100_0006h	21.5.5/483
4004_E014	Interrupt Mask Register (EMVSIM0_INT_MASK)	32	R/W	0000_FFFFh	21.5.6/487
4004_E018	Receiver Threshold Register (EMVSIM0_RX_THD)	32	R/W	0000_0001h	21.5.7/490
4004_E01C	Transmitter Threshold Register (EMVSIM0_TX_THD)	32	R/W	0000_000Fh	21.5.8/490
4004_E020	Receive Status Register (EMVSIM0_RX_STATUS)	32	w1c	0000_0000h	21.5.9/492
4004_E024	Transmitter Status Register (EMVSIM0_TX_STATUS)	32	w1c	0000_00B8h	21.5.10/495
4004_E028	Port Control and Status Register (EMVSIM0_PCSR)	32	R/W	0100_0000h	21.5.11/498
4004_E02C	Receive Data Read Buffer (EMVSIM0_RX_BUF)	32	R	0000_0000h	21.5.12/500
4004_E030	Transmit Data Buffer (EMVSIM0_TX_BUF)	32	W (always reads 0)	0000_0000h	21.5.13/501
4004_E034	Transmitter Guard ETU Value Register (EMVSIM0_TX_GETU)	32	R/W	0000_0000h	21.5.14/501
4004_E038	Character Wait Time Value Register (EMVSIM0_CWT_VAL)	32	R/W	0000_FFFFh	21.5.15/502
4004_E03C	Block Wait Time Value Register (EMVSIM0_BWT_VAL)	32	R/W	FFFF_FFFFh	21.5.16/502
4004_E040	Block Guard Time Value Register (EMVSIM0_BGT_VAL)	32	R/W	0000_0000h	21.5.17/503
4004_E044	General Purpose Counter 0 Timeout Value Register (EMVSIM0_GPCNT0_VAL)	32	R/W	0000_FFFFh	21.5.18/503
4004_E048	General Purpose Counter 1 Timeout Value (EMVSIM0_GPCNT1_VAL)	32	R/W	0000_FFFFh	21.5.19/504

21.5.1 Version ID Register (EMVSIMx_VER_ID)

Version ID for the IP. It corresponds to the version of IP being used.

Address: 4004_E000h base + 0h offset = 4004_E000h

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

EMVSIMx_VER_ID field descriptions

Field	Description
VER	Version ID of the module EMV SIM Version Number used on the chip 31:16 - Major Version (example: 01.00) 15:0 - Minor Version (example: 01.00)

21.5.2 Parameter Register (EMVSIMx_PARAM)

This register provides details on the parameter settings that were used while including this module in the chip.

Address: 4004_E000h base + 4h offset = 4004_E004h

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

EMVSIMx_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 TX_FIFO_ DEPTH	Transmit FIFO Depth Value of parameter for Transmit FIFO Depth (in Bytes)
RX_FIFO_ DEPTH	Receive FIFO Depth Value of parameter for Receive FIFO Depth (in Bytes)

21.5.3 Clock Configuration Register (EMVSIMx_CLKCFG)

This register provides configuration details to enable the right clock frequency of clocks used by EMV SIM module.

Address: 4004_E000h base + 8h offset = 4004_E008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0				GPCNT0_CLK_SEL		GPCNT1_CLK_SEL		CLK_PRSC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

EMVSIMx_CLKCFG field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–10 GPCNT0_CLK_SEL	General Purpose Counter 0 Clock Select Selects which clock source is used by EMV SIM Module general purpose counter 0. The only way to reset the counter is to set these bits to zero. The counter will begin counting as soon as the clock input is selected and the clocks are active. These input clocks are enabled through other register bits of the EMV SIM module (KILL_CLOCK, RCV_EN, and XMT_EN). Counter is active while RCV_EN or XMT_EN are set. 00 Disabled / Reset (default) 01 Card Clock 10 Receive Clock 11 ETU Clock (transmit clock)
9–8 GPCNT1_CLK_SEL	General Purpose Counter 1 Clock Select Selects which clock source is used by EMV SIM Module general purpose counter 1. The only way to reset the counter is to set these bits to zero. The counter will begin counting as soon as the clock input is selected and the clocks are active. These input clocks are enabled through other register bits of the EMV SIM module (KILL_CLOCK, RCV_EN, and XMT_EN). Counter is active while RCV_EN or XMT_EN are set. 00 Disabled / Reset (default) 01 Card Clock 10 Receive Clock 11 ETU Clock (transmit clock)
CLK_PRSC	Clock Prescaler Value The value written to this register will determine the desired card clock frequency. The Card Clock frequency is the EMV SIM Protocol (or logic) clock divided by this prescaler value. The prescaler value should be updated only when card clock is disabled and transmitter and receiver are not operational. 0, 1 Max Divide value used (default 0 is used)

Table continues on the next page...

EMVSIMx_CLKCFG field descriptions (continued)

Field	Description
	2 Divide by 2 >2 Non-zero value will divide clock accordingly

21.5.4 Baud Rate Divisor Register (EMVSIMx_DIVISOR)

This regiter configures the divisor value to generate the baud clock which will drive the card clock and also generate the transmit and receive clocks and respective ETUs.

Address: 4004_E000h base + Ch offset = 4004_E00Ch

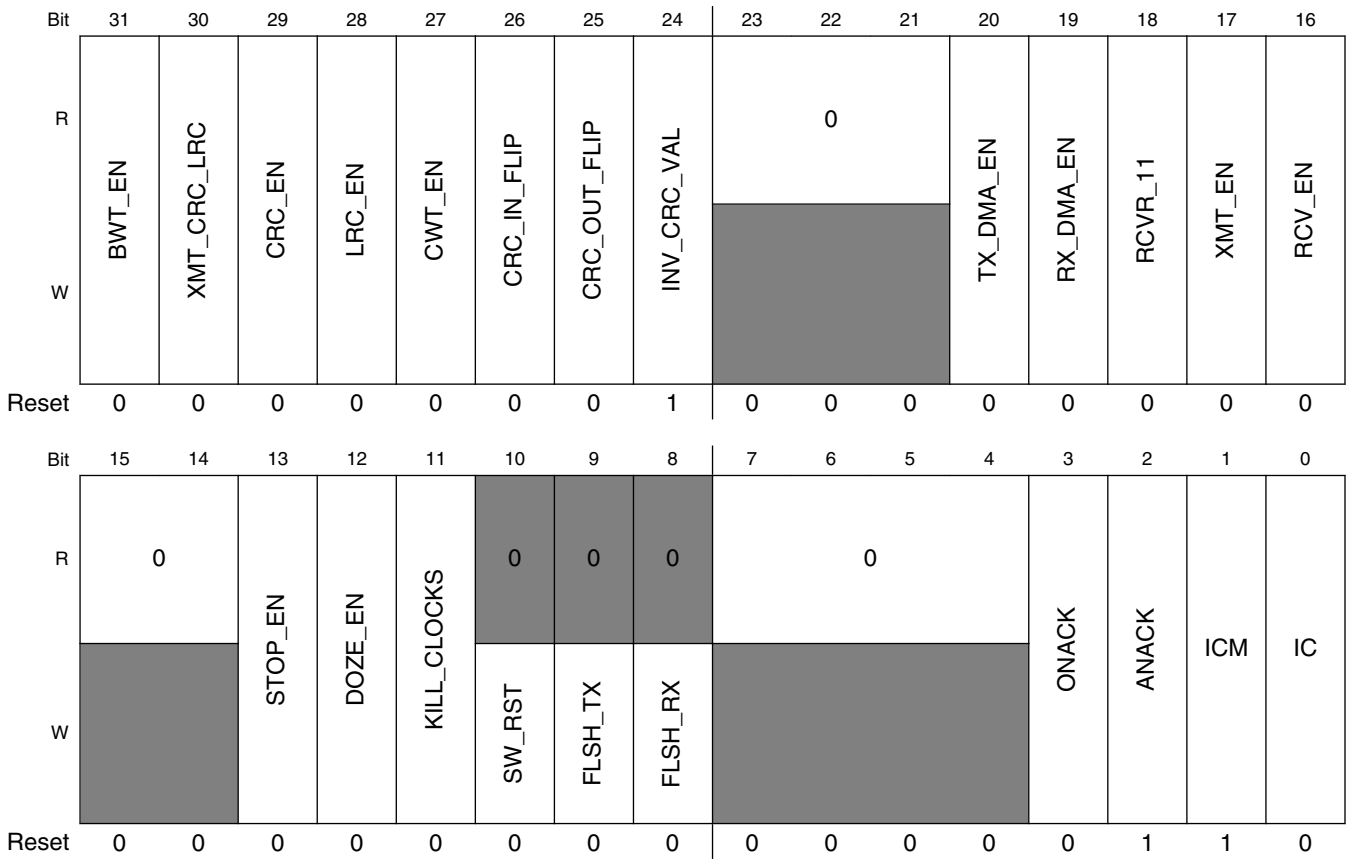
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DIVISOR_VALUE																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	0

EMVSIMx_DIVISOR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIVISOR_ VALUE	Divisor (F/D) Value The value written to this register will be used to generate the ETU bit period that will be used by the transmitter and receiver. The divisor value is the integer result of F/D value; where F and D are the clock rate conversion integer and baud rate adjustment integer, respectively, that required for the desired operation. The value in this register can be changed when no transaction is active with the smart card. When programming this field, the fractional part of F/D should be rounded off the nearest integer. 0 - 4 Invalid. As per ISO 7816 specification, minimum value of F/D is 5 372 Divisor value for F = 372 and D = 1 (default) others Integer value of result of F/D

21.5.5 Control Register (EMVSIMx_CTRL)

Address: 4004_E000h base + 10h offset = 4004_E010h



EMVSIMx_CTRL field descriptions

Field	Description
31 BWT_EN	Block Wait Time Counter Enable Writing a '1' to this bit will enable the BWT and BGT functions. The BWT and BGT functions can then be individually selected using the interrupt mask. 0 Disable BWT, BGT Counters (default) 1 Enable BWT, BGT Counters
30 XMT_CRC_LRC	Transmit CRC or LRC Enable This bit specifies whether or not to transmit the redundancy checking data (LRC or CRC) at the end of a transmission (that is, when the FIFO becomes empty). 0 No CRC or LRC value is transmitted (default) 1 Transmit LRC or CRC info when FIFO empties (whichever is enabled)
29 CRC_EN	CRC Enable

Table continues on the next page...

EMVSIMx_CTRL field descriptions (continued)

Field	Description
	This bit enables the calculation of the 16-bit CRC value for both receiver and transmitter. The result of the calculation is continuously compared to the expected remainder and reflected in the CRC_OK bit in the RX_STATUS register. Clearing this bit resets the current CRC residual value in the EMV SIM hardware. 0 16-bit Cyclic Redundancy Checking disabled (default) 1 16-bit Cyclic Redundancy Checking enabled
28 LRC_EN	LRC Enable This bit enables the calculation of the 8-bit LRC value for both receiver and transmitter. The result of the calculation is continuously compared to zero and reflected in the LRC_OK bit in the RX_STATUS register. Clearing this bit resets the current LRC value in the EMV SIM hardware. 0 8-bit Linear Redundancy Checking disabled (default) 1 8-bit Linear Redundancy Checking enabled
27 CWT_EN	Character Wait Time Counter Enable Enables the character wait time counter. Clearing this bit resets the counter to zero. 0 Character Wait time Counter is disabled (default) 1 Character Wait time counter is enabled
26 CRC_IN_FLIP	CRC Input Byte's Bit Reversal or Flip Control This bit control whether the bits in the CRC input byte will be reversed before CRC calculation or not. When set to '1', the bits within the input byte for CRC will change from 7:0 to 0:7. For CCITT CRC calculation, this bit should be set to '1'. For any other CRC using same polynomial, this bit can be set accordingly. 0 Bits in the input byte will not be reversed (i.e. 7:0 will remain 7:0) before the CRC calculation (default) 1 Bits in the input byte will be reversed (i.e. 7:0 will become 0:7) before CRC calculation
25 CRC_OUT_FLIP	CRC Output Value Bit Reversal or Flip This bit control whether the bits in the CRC output bytes will be reversed or not. When set to '1', the bits within the two bytes for CRC Output will change from 15:0 to {8:15,0:7}. For CCITT CRC calculation, this bit should be set to '1'. For any other CRC using same polynomial, this bit can be set accordingly. 0 Bits within the CRC output bytes will not be reversed i.e. 15:0 will remain 15:0 (default) 1 Bits within the CRC output bytes will be reversed i.e. 15:0 will become {8:15,0:7}
24 INV_CRC_VAL	Invert bits in the CRC Output Value This bit control whether the bits within the CRC Output value will be inverted (1's complement) or not. For CCITT CRC calculation, this bit should be set to '1'. For any other CRC using same polynomial, this bit can be set accordingly. 0 Bits in CRC Output value will not be inverted. 1 Bits in CRC Output value will be inverted. (default)
23–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 TX_DMA_EN	Transmit DMA Enable Enables assertion of DMA write request when Transmit FIFO is empty. Request is held asserted till Transmit FIFO reaches the programmed data threshold value. Transmit Data Threshold Interrupt will not be generated when this bit is asserted.

Table continues on the next page...

EMVSIMx_CTRL field descriptions (continued)

Field	Description
	0 No DMA Write Request asserted for Transmitter (default) 1 DMA Write Request asserted for Transmitter
19 RX_DMA_EN	Receive DMA Enable Enables assertion of DMA read request when Receive FIFO reaches the programmed data threshold value. Request is held asserted till all the programmed threshold bytes are read out. Receiver Data Interrupt will not be generated when this bit is asserted. 0 No DMA Read Request asserted for Receiver (default) 1 DMA Read Request asserted for Receiver
18 RCVR_11	Receiver 11 ETU Mode Enable Used to configure the EMV SIM module receiver for 11 ETU operation (that is, 1 Stop bit). This bit is provided for support of T=1 cards. 0 Receiver configured for 12 ETU operation mode (default) 1 Receiver configured for 11 ETU operation mode
17 XMT_EN	Transmitter Enable Used to enable/disable the EMV SIM transmitter block. It can be set to 0 during the auto power down sequence. **Note:** Setting this bit (transition from 0 to 1) will reset the CRC and LRC values. 0 EMV SIM Transmitter disabled (default) 1 EMV SIM Transmitter enabled
16 RCV_EN	Receiver Enable Used to enable/disable the EMV SIM receiver block. Once the transmitter has completed its operation, the software must enable the receiver using this bit. It can be set to 0 during the auto power down sequence. 0 EMV SIM Receiver disabled (default) 1 EMV SIM Receiver enabled
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 STOP_EN	STOP Enable Used to configure the operation of the EMV SIM module when a processor STOP instruction is executed. This bit is added to provide support for Smart Cards that do not allow the Smart Card clock to be stopped while power is applied. 0 STOP instruction shuts down all EMV SIM clocks (default) 1 STOP instruction shuts down all clocks except for the Smart Card Clock (SCK) (clock provided to Smart Card)
12 DOZE_EN	Doze Enable Used to configure the operation of the EMV SIM module when a processor DOZE instruction is executed. 0 DOZE instruction will gate all internal EMV SIM clocks as well as the Smart Card clock when the transmit FIFO is empty (default) 1 DOZE instruction has no effect on EMV SIM module
11 KILL_CLOCKS	Kill all internal clocks

Table continues on the next page...

EMVSIMx_CTRL field descriptions (continued)

Field	Description
	Used to enable/disable the clock input to the EMV SIM module. This bit will gate all clocks including the Smart Card clock regardless of the state of the STOP bit described above. **Note:** This bit will have no effect on the register read write clock. Only EMV SIM logic clock is gated. 0 EMV SIM input clock enabled (default) 1 EMV SIM input clock is disabled
10 SW_RST	Software Reset Bit Used to reset the entire EMV SIM module. This acts the same as a hardware reset for the EMV SIM module. This bit is self-clearing and always reads 0. **Note:** Software should allow a minimum of 4 Protocol clock cycles before attempting to access the EMV SIM module after a software reset. 0 EMV SIM Normal operation (default) 1 EMV SIM held in Reset
9 FLSH_TX	Flush Transmitter Bit This bit operates as an EMV SIM transmitter reset. The receive portion of the EMV SIM module is not affected. This bit clears automatically and always reads 0. 0 EMV SIM Transmitter normal operation (default) 1 EMV SIM Transmitter held in Reset
8 FLSH_RX	Flush Receiver Bit This bit operates as an EMV SIM receiver reset. The transmit portion of the EMV SIM module is not affected. This bits clears automatically and always reads 0. 0 EMV SIM Receiver normal operation (default) 1 EMV SIM Receiver held in Reset
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ONACK	Overrun NACK Enable Enables NACK generation when Rx FIFO is full and another message is ready for writing into the FIFO. 0 NACK generation on overrun is disabled (default) 1 NACK generation on overrun is enabled
2 ANACK	Auto NACK Enable Enables NACK generation for parity errors in receieved messages or when invalid initial characters are received in ICM mode. 0 NACK generation on errors disabled 1 NACK generation on errors enabled (default)
1 ICM	Initial Character Mode Enables initial character mode. Will be automatically cleared by hardware once a valid initial character is received. 0 Initial Character Mode disabled 1 Initial Character Mode enabled (default)

Table continues on the next page...

EMVSIMx_CTRL field descriptions (continued)

Field	Description
0 IC	Inverse Convention Used to configure the EMV SIM to use either inverse convention or direct convention for its data format. The IC bit can be controlled by software, but it is normally set by hardware as a result of the interpretation of the initial character when in ICM mode. 0 Direction convention transfers enabled (default) 1 Inverse convention transfers enabled

21.5.6 Interrupt Mask Register (EMVSIMx_INT_MASK)

Address: 4004_E000h base + 14h offset = 4004_E014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R W	PEF_IM	RX_DATA_IM	GPCNT1_IM	BGT_ERR_IM	BWT_ERR_IM	RNACK_IM	CWT_ERR_IM	GPCNT0_IM	TDT_IM	TFF_IM	TNACK_IM	TFE_IM	ETC_IM	RFO_IM	TC_IM	RDT_IM
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

EMVSIMx_INT_MASK field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 PEF_IM	Parity Error Interrupt Mask Used to enable/disable the ability of the PEF flag in the RX_STATUS register to generate EMV SIM interrupts. 0 PEF interrupt enabled 1 PEF interrupt masked (default)
14 RX_DATA_IM	Receive Data Interrupt Mask Used to enable/disable the ability of the RX_DATA flag in the RX_STATUS register to generate EMV SIM interrupts. 0 RX_DATA interrupt enabled 1 RX_DATA interrupt masked (default)

Table continues on the next page...

EMVSIMx_INT_MASK field descriptions (continued)

Field	Description
13 GPCNT1_IM	General Purpose Counter 1 Timeout Interrupt Mask Used to enable/disable the ability of the GPCNT1_TO flag in the TX_STATUS register to generate EMV SIM interrupts. 0 GPCNT1_TO interrupt enabled 1 GPCNT1_TO interrupt masked (default)
12 BGT_ERR_IM	Block Guard Time Error Interrupt Used to enable/disable the ability of the BGT_ERR flag in the RX_STATUS register to generate EMV SIM interrupts. 0 BGT_ERR interrupt enabled 1 BGT_ERR interrupt masked (default)
11 BWT_ERR_IM	Block Wait Time Error Interrupt Mask Used to enable/disable the ability of the BWT_ERR flag in the RX_STATUS register to generate EMV SIM interrupts. 0 BWT_ERR interrupt enabled 1 BWT_ERR interrupt masked (default)
10 RNACK_IM	Receiver NACK Threshold Interrupt Mask Used to enable/disable the ability of the RTE flag in the RX_STATUS register to generate EMV SIM interrupts. 0 RTE interrupt enabled 1 RTE interrupt masked (default)
9 CWT_ERR_IM	Character Wait Time Error Interrupt Mask Used to enable/disable the ability of the CWT_ERR flag in the RX_STATUS register to generate EMV SIM interrupts. 0 CWT_ERR interrupt enabled 1 CWT_ERR interrupt masked (default)
8 GPCNT0_IM	General Purpose Timer 0 Timeout Interrupt Mask Used to enable/disable the ability of the GPCNT0_TO flag in the TX_STATUS register to generate EMV SIM interrupts. 0 GPCNT0_TO interrupt enabled 1 GPCNT0_TO interrupt masked (default)
7 TDT_IM	Transmit Data Threshold Interrupt Mask Used to enable/disable the ability of the TDTF flag in the TX_STATUS register to generate EMV SIM interrupts. 0 TDTF interrupt enabled 1 TDTF interrupt masked (default)
6 TFF_IM	Transmit FIFO Full Interrupt Mask Used to enable/disable the ability of the TFF flag in the TX_STATUS register to generate EMV SIM interrupts.

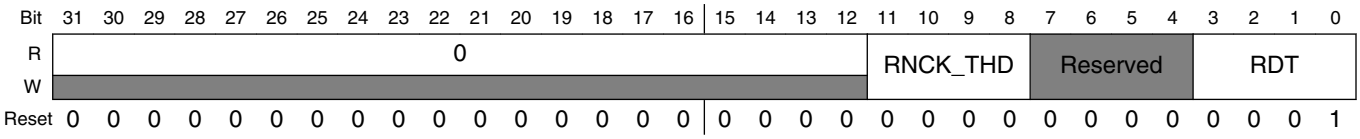
Table continues on the next page...

EMVSIMx_INT_MASK field descriptions (continued)

Field	Description
	0 TFF interrupt enabled 1 TFF interrupt masked (default)
5 TNACK_IM	Transmit NACK Threshold Interrupt Mask Used to enable/disable the ability of the TNTE flag in the TX_STATUS register to generate EMV SIM interrupts. 0 TNTE interrupt enabled 1 TNTE interrupt masked (default)
4 TFE_IM	Transmit FIFO Empty Interrupt Mask Used to enable/disable the ability of the TFE flag in the TX_STATUS register to generate EMV SIM interrupts. 0 TFE interrupt enabled 1 TFE interrupt masked (default)
3 ETC_IM	Early Transmit Complete Interrupt Mask Used to enable/disable the ability of the ETC flag in the TX_STATUS register to generate EMV SIM interrupts. 0 ETC interrupt enabled 1 ETC interrupt masked (default)
2 RFO_IM	Receive FIFO Overflow Interrupt Mask Used to enable/disable the ability of the RFO flag in the RX_STATUS register to generate EMV SIM interrupts. 0 RFO interrupt enabled 1 RFO interrupt masked (default)
1 TC_IM	Transmit Complete Interrupt Mask Used to enable/disable the ability of the TCF flag in the TX_STATUS register to generate EMV SIM interrupts. 0 TCF interrupt enabled 1 TCF interrupt masked (default)
0 RDT_IM	Receive Data Threshold Interrupt Mask Used to enable/disable the ability of the RDTF flag in the RX_STATUS register to generate EMV SIM interrupts. 0 RDTF interrupt enabled 1 RDTF interrupt masked (default)

21.5.7 Receiver Threshold Register (EMVSIMx_RX_THD)

Address: 4004_E000h base + 18h offset = 4004_E018h

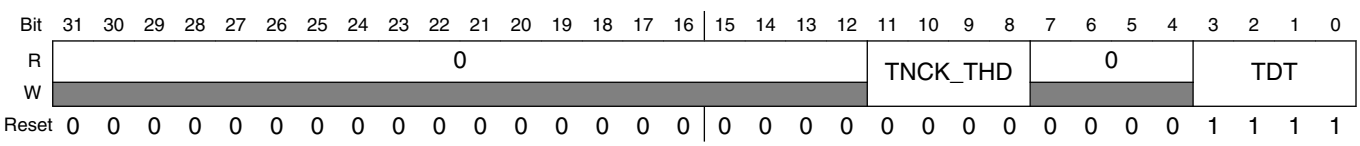


EMVSIMx_RX_THD field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 RNCK_THD	Receiver NACK Threshold Value Used to specify the number of consecutive NACK's transmitted by the EMV SIM module, for a given character, before the receive threshold error (RTE) flag is triggered. A value of 0 indicates that RTE is never set. When a valid character is received by the EMV SIM, the internal counter keeping track of the NACK count resets to zero for the subsequent byte being received. If the ANACK bit is clear in the CNTL register, RNCK_THD has no effect. 0 Zero Threshold. RTE will not be set >0 RTE will set after programmed value of NACKs are received
7–4 Reserved	This field is reserved.
RDT	Receiver Data Threshold Value Determines the number of bytes that must exist in the Receive FIFO to trigger the receive data threshold interrupt flag (RDTF). Value must not exceed the total bytes that can be stored in the Receive FIFO.

21.5.8 Transmitter Threshold Register (EMVSIMx_TX_THD)

Address: 4004_E000h base + 1Ch offset = 4004_E01Ch



EMVSIMx_TX_THD field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 TNCK_THD	Transmitter NACK Threshold Value Used to set the NACK threshold for the transmitter. Once the threshold number set by TNCK_THD has been reached for the current byte being transmitted, the error flag TNTE in the TX_STATUS register will

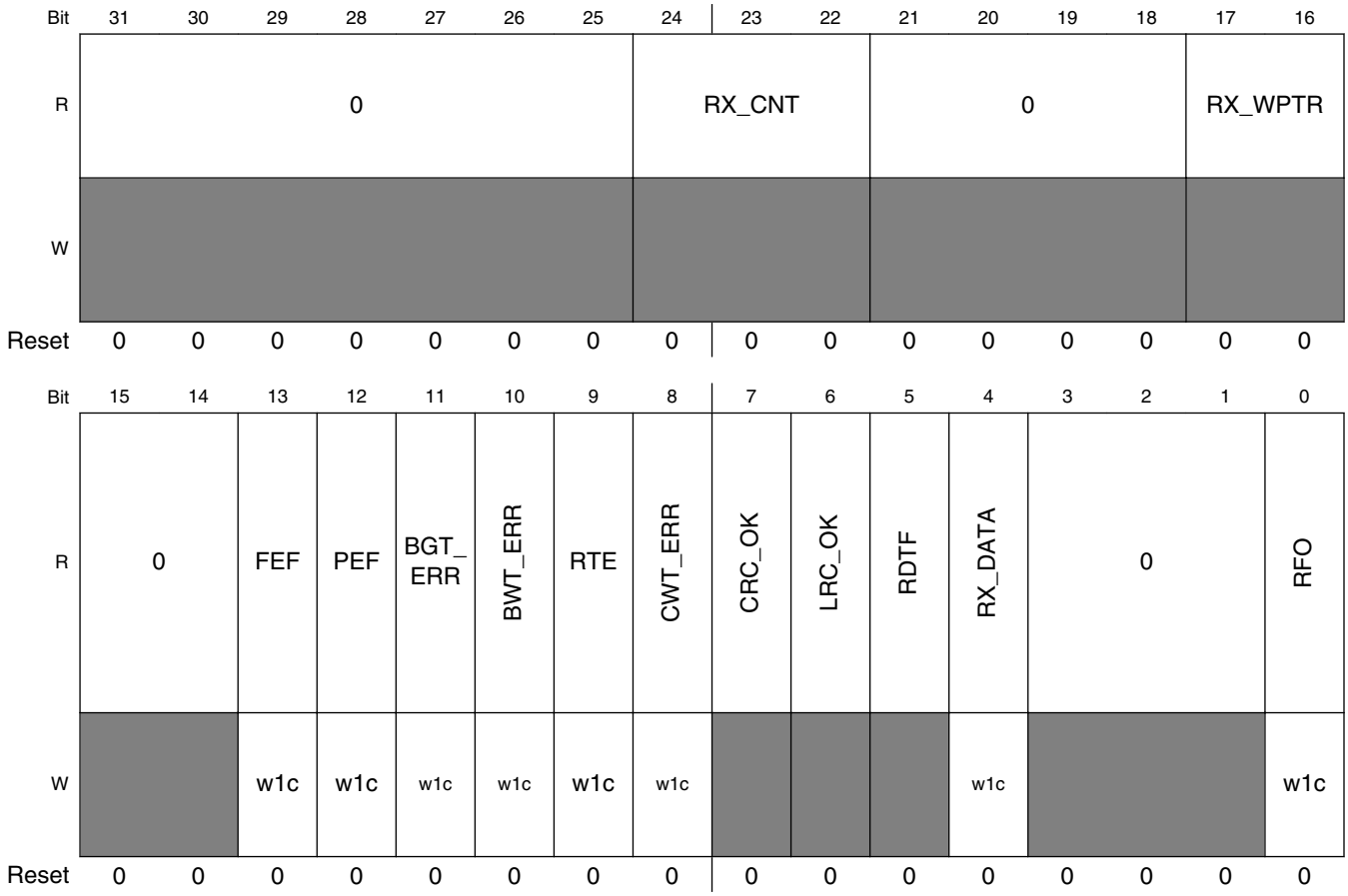
Table continues on the next page...

EMVSIMx_TX_THD field descriptions (continued)

Field	Description
	be set. Setting of TNTE causes the remaining transmissions queued in the transmit FIFO to be aborted and no more transmissions to occur until software clears TNTE. To trigger TNTE, a given byte being transmitted must reach the TNCK_THD threshold itself. Transmit NACKs accumulated on one byte are not carried over to the next. 0 TNTE will never be set; retransmission after NACK reception is disabled. 1 TNTE will be set after 1 nack is received; 0 retransmissions occurs. 2 TNTE will be set after 2 nacks are received; at most 1 retransmission occurs. 3 TNTE will be set after 3 nacks are received; at most 2 retransmissions occurs. . . . . . . N TNTE will be set after N nacks are received; at most (N - 1) retransmissions occurs. . . . . . . 15 TNTE will be set after 15 nacks are received; at most 14 retransmissions occurs.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TDT	Transmitter Data Threshold Value Used to set the threshold value for the Transmit FIFO at which the TDTF bit in the TX_STATUS register will be set. When the number of bytes in the Transmit FIFO is less than or equal to TDT[3:0], TDTF will be set.

21.5.9 Receive Status Register (EMVSIMx_RX_STATUS)

Address: 4004_E000h base + 20h offset = 4004_E020h



EMVSIMx_RX_STATUS field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24–22 RX_CNT	Receive FIFO Byte Count These bits indicate the number of bytes stored in the receive FIFO. 0 FIFO is emtpy >0 Total bytes available in FIFO
21–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–16 RX_WPTR	Receive FIFO Write Pointer Value Value of write pointer of Receive FIFO
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

EMVSIMx_RX_STATUS field descriptions (continued)

Field	Description
13 FEF	Frame Error Flag Used to indicate if the current received byte did not have proper STOP bits. This will not cause a NACK transmission. This bit will not cause an interrupt assertion. It is simply an indication that a framing error has occured. 0 No frame error detected 1 Frame error detected
12 PEF	Parity Error Flag Used to indicate if a received byte had a parity error or not. A parity error will cause transmission of NACK if ANACK bit is set in CTRL register and PEF bit will not be asserted. When ANACK is *not* set, PEF will be asserted. This bit will also not assert when ICM bit is set; however parity is checked in ICM = 1 mode. It is simply an indication that a parity error has occured. 0 No parity error detected 1 Parity error detected
11 BGT_ERR	Block Guard Time Error Flag Used to indicate if the block guard time was too small. The threshold is set by the block guard time register. 0 Block guard time was sufficient 1 Block guard time was too small
10 BWT_ERR	Block Wait Time Error Flag Used to indicate if the block wait time has been exceeded. The threshold is set by the block wait time registers. 0 Block wait time not exceeded 1 Block wait time was exceeded
9 RTE	Received NACK Threshold Error Flag Used to indicate whether the number of consecutive NACK's generated by the EMV SIM module in response to receive parity errors, for the byte being received, equals the value programmed in the RTH[3:0] in the RX_THRESHOLD register. This bit will never set unless the ANACK bit is set in the CNTL register. The SAPDx bit in the CNTL resister must be set to enable the threshold error to trigger the auto power down sequence. RTE is a write once to clear bit. Clearing this bit also resets the internal counter for consecutive NACK's being transmitted for a given byte. 0 Number of NACKs generated by the receiver is less than the value programmed in RTH[3:0] 1 Number of NACKs generated by the receiver is equal to the value programmed in RTH[3:0]
8 CWT_ERR	Character Wait Time Error Flag Used to indicate when the time between received characters is equal to or greater than the value programmed in the CHAR_WAIT register. 0 No CWT violation has occurred (default). 1 Time between two consecutive characters has exceeded the value in CHAR_WAIT.
7 CRC_OK	CRC Check OK Flag

Table continues on the next page...

EMVSIMx_RX_STATUS field descriptions (continued)

Field	Description
	Used to indicate when the calculated 16-bit CRC value matches the expected value for the current input data stream. The value is calculated across all received characters from the point the CRC_EN bit is set in the CTRL register. The current CRC residual can be reset by three mechanisms: • Clear CRC_EN bit in CTRL register • Set XMT_EN bit in ENABLE register • Automatically by hardware when ETC flag is set at the end of a transmission. 0 Current CRC value does not match remainder. 1 Current calculated CRC value matches the expected result.
6 LRC_OK	LRC Check OK Flag Used to indicate when the calculated 8-bit LRC value is correct for the current input data stream. The value is calculated across all received characters from the point the LRC_EN bit is set in the CTRL register. The current LRC residual can be reset by three mechanisms: • Clear LRC_EN bit in CTRL register • Set XMT_EN bit in ENABLE register • Automatically by hardware when ETC flag is set at the end of a transmission. 0 Current LRC value does not match remainder. 1 Current calculated LRC value matches the expected result (i.e. zero).
5 RDTF	Receive Data Threshold Interrupt Flag Interrupt flag asserted when total bytes in the Receive FIFO equal or is greater than the programmed receive threshold RDT[3:0]. The RDTF flag will be set any time the number of unread bytes in the receive FIFO is equal to or greater than the value set by RDT[3:0]. The flag can be cleared by reading enough bytes out of the receive FIFO so as to bring the number of bytes left in the FIFO below the RDT[3:0] level. Another way to clear the flag is to set the RDT[3:0] level higher than the number of unread bytes currently in the FIFO. The RDTF flag will create an interrupt if the RDT_IM bit in the INT_MASK register is cleared. 0 Number of unread bytes in receive FIFO less than the value set by RDT[3:0] (default). 1 Number of unread bytes in receive FIFO greater or than equal to value set by RDT[3:0].
4 RX_DATA	Receive Data Interrupt Flag Interrupt asserted when a new data byte is received and entered into the Receive FIFO. 0 No new byte is received 1 New byte is received ans stored in Receive FIFO
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 RFO	Receive FIFO Overflow Flag Used to indicate that the EMV SIM was unable to store received data due to already unread bytes in the FIFO (FIFO full or almost full). It does not necessarily indicate that data has been lost. If the ONACK control bit in the CNTL register is set, there will be a NACK pulse generated on bytes that would otherwise cause a loss of data due to a full FIFO. These bytes should be retransmitted by the Smart Card which implies that no data has actually been lost. In this case, the RFO flag is just an indicator that this situation has occurred which may be helpful in system debug. For the case where ONACK is not set, a set RFO flag does indicate a loss of data since all bytes received with the OEF flag set will indeed be lost (including the byte that caused the bit to be set). The RFO flag will cause an interrupt if the RFO_IM bit in the INT_MASK register. The RFO flag is a write-one-to-clear bit.

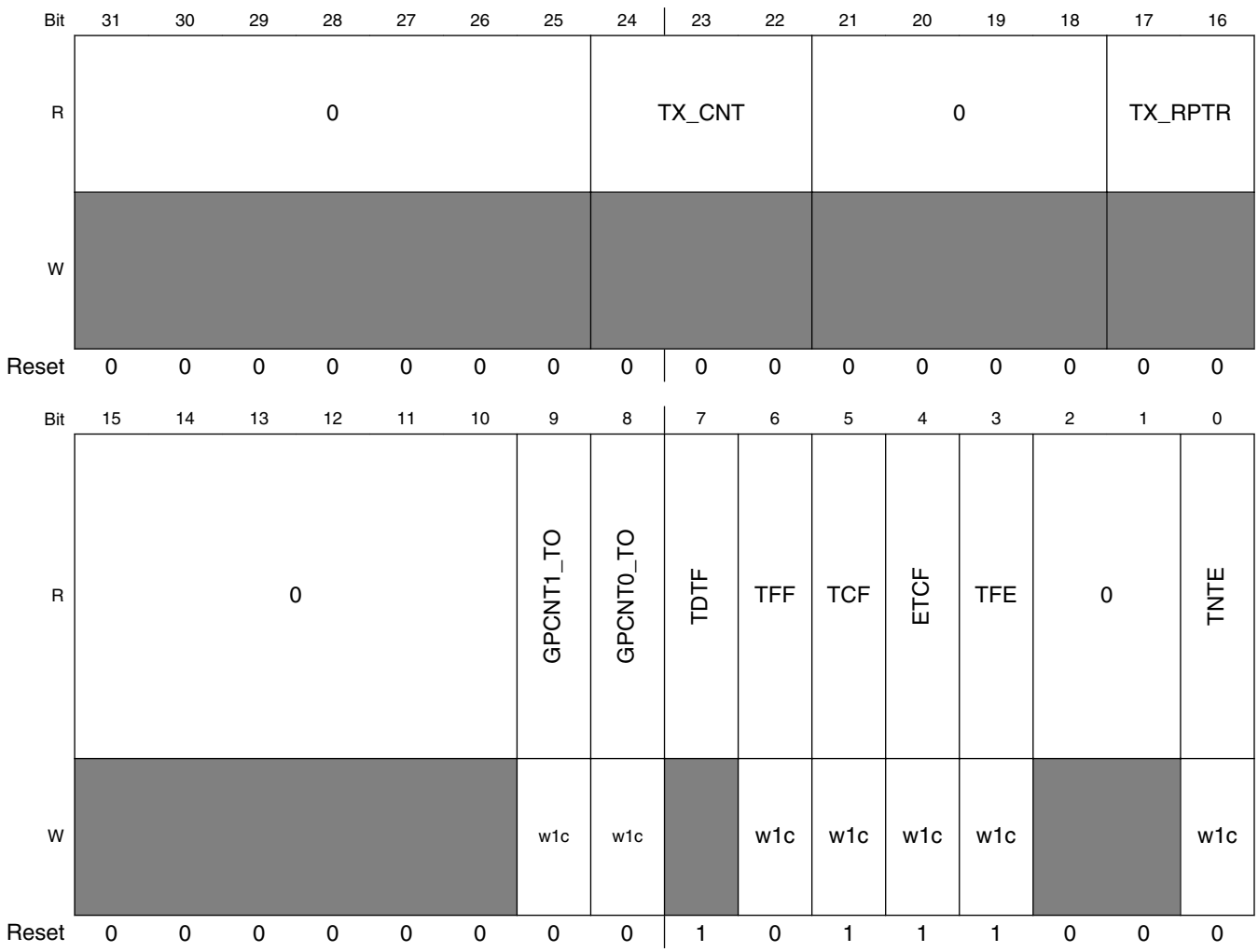
Table continues on the next page...

EMVSIMx_RX_STATUS field descriptions (continued)

Field	Description
	0 No overrun error has occurred (default) 1 A byte was received when the received FIFO was already full

21.5.10 Transmitter Status Register (EMVSIMx_TX_STATUS)

Address: 4004_E000h base + 24h offset = 4004_E024h



EMVSIMx_TX_STATUS field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24–22 TX_CNT	Transmit FIFO Byte Count These bits indicate the number of bytes stored in the transmit FIFO.

Table continues on the next page...

EMVSIMx_TX_STATUS field descriptions (continued)

Field	Description
	0 FIFO is emtpy >0 Total bytes in FIFO
21–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–16 TX_RPTR	Transmit FIFO Read Pointer Value of the read pointer of transmit FIFO.
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 GPCNT1_TO	General Purpose Counter 1 Timeout Flag Used to indicate when the General Purpose Counter 1 has reached the GPCNT1_VAL value in the GPCNT_VAL register. 0 GPCNT1_VAL time not reached, or bit has been cleared. (default) 1 General Purpose counter has reached the GPCNT1_VAL value
8 GPCNT0_TO	General Purpose Counter 0 Timeout Flag Used to indicate when the General Purpose Counter 0 has reached the GPCNT0_VAL value in the GPCNT_VAL register. 0 GPCNT0_VAL time not reached, or bit has been cleared. (default) 1 General Purpose counter has reached the GPCNT0_VAL value
7 TDTF	Transmit Data Threshold Flag Interrupt flag asserted when total bytes in the Transmit FIFO is less than or equal to the programmed transmit data threshold TDT[3:0]. The TDTF flag will be set any time the number of bytes in the transmit FIFO is less than or equal to the value set by TDT[3:0]. The flag can be cleared by writing enough bytes into the transmit FIFO so as to take the number of bytes in the FIFO above the TDT[3:0] level. Another way to clear the flag is to set the TDT[3:0] level lower than the number of bytes currently in the FIFO. The TDTF flag will create an interrupt if the TDT_IM bit in the INT_MASK register is cleared. 0 Number of bytes in FIFO is greater than TDT[3:0], or bit has been cleared 1 Number of bytes in FIFO is less than or equal to TDT[3:0] (default)
6 TFF	Transmit FIFO Full Flag Used to indicate when the Transmit FIFO has been written with maximum number of bytes that it can store. Subsequent writes may get ignored. The TFF bit can only be cleared by writing 1 to this bit or setting the FLUSH_XMT or SOFT_RESET bit in the RESET_CNTL register. 0 Transmit FIFO Full condition has *not* occurred (default) 1 A Transmit FIFO Full condition has occurred
5 TCF	Transmit Complete Flag Used to indicate whether the EMV SIM transmitter is ready for a new transmission. The TC flag becomes set when the guard time has expired after the last byte in the transmit FIFO (if XMT_CRC_LRC = 0) or the LRC or CRC byte (if XMT_CRC_LRC bit = 1) has been transmitted. The TC flag will create an interrupt if TC_IM in the INT_MASK register is low. The TC bit is a write-one-to-clear bit. 0 Transmit pending or in progress 1 Transmit complete (default)

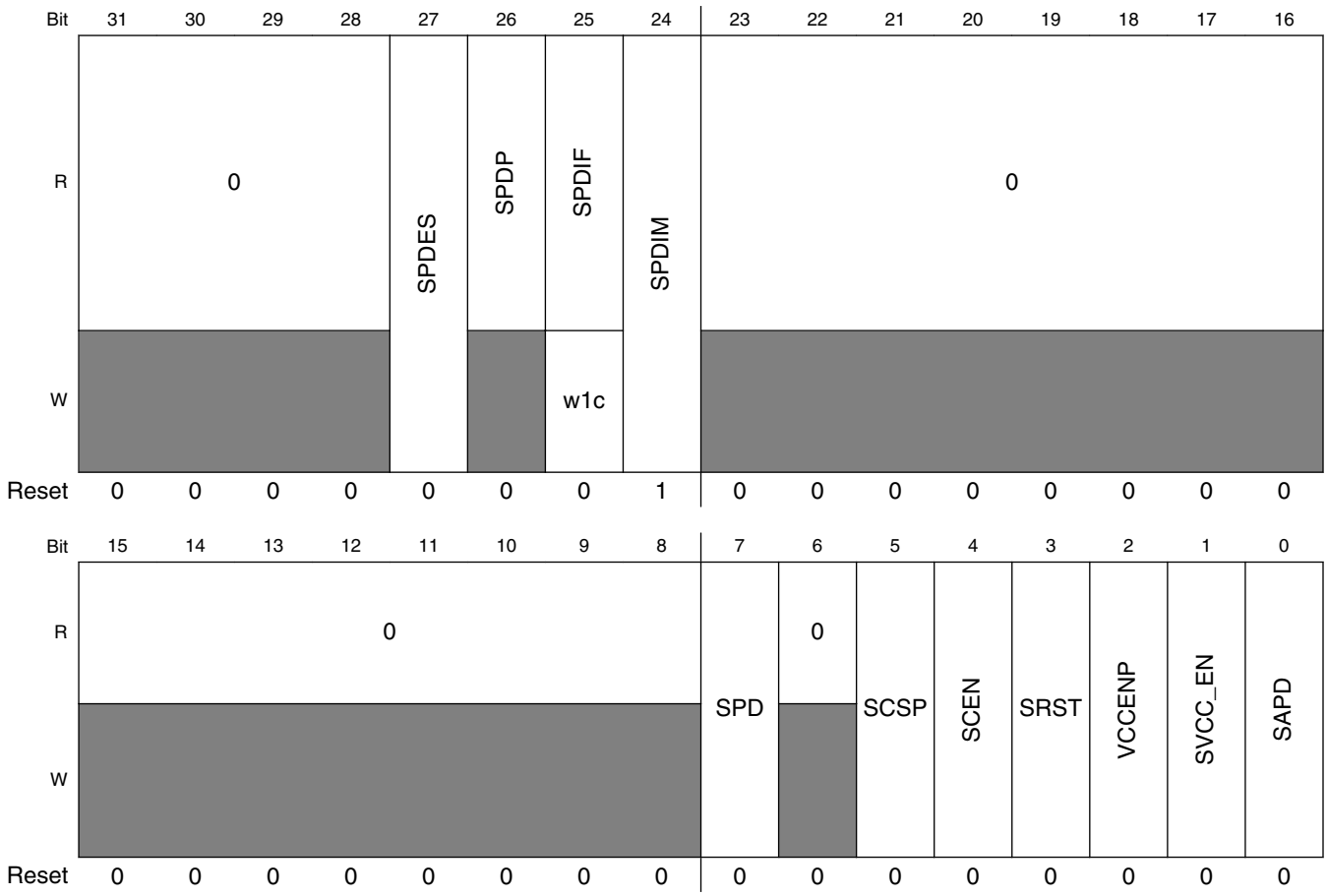
Table continues on the next page...

EMVSIMx_TX_STATUS field descriptions (continued)

Field	Description
4 ETCF	Early Transmit Complete Flag Used to indicate that the EMV SIM transmitter has finished sending the last byte in transmit FIFO (if XMT_CRC_LRC = 0) or finished sending the LRC or CRC byte (if XMT_CRC_LRC = 1). This bit differs from the TC bit in that it is set before the guard time of the last byte has elapsed. The ETC flag will create an interrupt if ETC_IM in the INT_MASK register is low. The ETC bit is a write-one-to-clear bit. 0 Transmit pending or in progress 1 Transmit complete (default)
3 TFE	Transmit FIFO Empty Flag Used to indicate that the EMV SIM transmit FIFO has been emptied. This bit will be set when the last byte in the transmit FIFO has been transferred out of the EMV SIM transmitter to the Smart Card successfully. TFE will also be set when the TNTE flag is set. The TFE flag will create an interrupt if TFE_IM in the INT_MASK register is low. The TFE bit is a write-one-to-clear bit. 0 Transmit FIFO is not empty 1 Transmit FIFO is empty (default)
2–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TNTE	Transmit NACK Threshold Error Flag Used to indicate the transmit NACK threshold has been reached. When TNTE is high, no further transmissions will be done until the TNTE flag is cleared. Any data transmissions still pending in the transmit FIFO will be aborted, and the TC, ETC, and TFE flags will be set. The TNTE flag will create an interrupt if TNACK_IM in the INT_MASK register is low. The TNTE bit is a write-one-to-clear bit. 0 Transmit NACK threshold has not been reached (default) 1 Transmit NACK threshold reached; transmitter frozen

21.5.11 Port Control and Status Register (EMVSIMx_PCSR)

Address: 4004_E000h base + 28h offset = 4004_E028h



EMVSIMx_PCSR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 SPDES	SIM Presence Detect Edge Select Controls which edge of the Smart Card Presence Detect pin is used to detect the presence of the Smart Card. 0 Falling edge on the pin (default) 1 Rising edge on the pin
26 SPDP	Smart Card Presence Detect Pin Status This bit reflects the state of the Smart Card Presence Detect pin. It is not a latched register bit, but instead a synchronized version of the state of the pin itself. 0 SIM Presence Detect pin is logic low 1 SIM Presence Detectpin is logic high

Table continues on the next page...

EMVSIMx_PCSR field descriptions (continued)

Field	Description
25 SPDIF	Smart Card Presence Detect Interrupt Flag Status flag to indicate that an insertion or removal of a Smart Card has been detected on port. Can create an interrupt to the MCU if SPDIM is low and/or initiate the auto power down sequence if SAPD bit is '1'. Write a '1' to this bit to clear. 0 No insertion or removal of Smart Card detected on Port (default) 1 Insertion or removal of Smart Card detected on Port
24 SPDIM	Smart Card Presence Detect Interrupt Mask Interrupt mask for the presence detect interrupt flag (SPDIF). 0 SIM presence detect interrupt is enabled 1 SIM presence detect interrupt is masked (default)
23–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 SPD	Auto Power Down Control Writing a '1' to this location will start the autopower down sequence provided the auto power down feature is enabled by writing '1' to SAPD bit of this register. The SPD bit will autoclear when power down is complete. 0 No effect (default) 1 Start Auto Powerdown or Power Down is in progress
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SCSP	Smart Card Clock Stop Polarity Used to control the polarity of the idle EMV SIM clock when the clock is disabled by SCEN. 0 Clock is logic 0 when stopped by SCEN 1 Clock is logic 1 when stopped by SCEN
4 SCEN	Clock Enable for Smart Card Used to enable/disable the clock to the Smart Card. It can be forced low by hardware during the auto power down sequence. 0 Smart Card Clock Disabled 1 Smart Card Clock Enabled
3 SRST	Reset to Smart Card Used to control state of reset line to the Smart Card. It can be forced low by hardware during the auto power down sequence. Smart Card reset signals are active low. 0 Smart Card Reset is asserted (default) 1 Smart Card Reset is de-asserted
2 VCCENP	VCC Enable Polarity Control Used to control the polarity of the SVEN output pad via the SVCC_EN bit. When set to '1', this bit will invert the value of SVCC_EN bit of this register and output to SVEN output pad of device.

Table continues on the next page...

EMVSIMx_PCSR field descriptions (continued)

Field	Description
	0 VCC_EN is active high. Polarity of SVCC_EN is unchanged. 1 VCC_EN is active low. Polarity of SVCC_EN is inverted.
1 SVCC_EN	Vcc Enable for Smart Card Used to control the state of the SVEN pin on Smart Card port. The SVEN pin controls the Smart Card Vcc enable in the power management chip. It can be forced low by hardware during the auto power down sequence. This bit is XORed with the VCCENP bit to control the polarity of the SVEN output pad. 0 Smart Card Voltage disabled (default) 1 Smart Card Voltage enabled
0 SAPD	Auto Power Down Enable Used to enable/disable the auto power down feature. This bit must be set prior to the auto power conditions are met. This bit will be return to 0 at the end of the auto power down sequence. This bit controls the auto power down function which can be initiated by: • Setting the SPD bit in this register by software • Assertion of the RTE bit in the RX_STATUS register • Assertion of the SPDIF bit when the interrupt is used for detecting card removal Manual Power Down via software by writing to the SCEN, SRST, and VCC_EN can still be carried out irrespective of the value of this bit. 0 Auto power down disabled (default) 1 Auto power down enabled

21.5.12 Receive Data Read Buffer (EMVSIMx_RX_BUF)

Address: 4004_E000h base + 2Ch offset = 4004_E02Ch

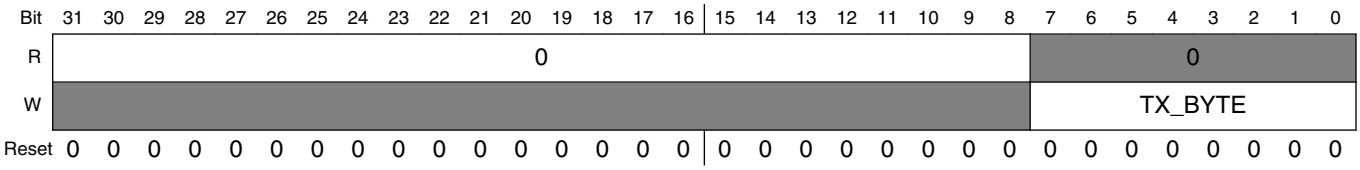
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																								RX_BYTE							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

EMVSIMx_RX_BUF field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RX_BYTE	Receive Data Byte Read Provides the byte value from the top of the receive FIFO. Each read access to this register will increment the read pointer of the Receive FIFO.

21.5.13 Transmit Data Buffer (EMVSIMx_TX_BUF)

Address: 4004_E000h base + 30h offset = 4004_E030h

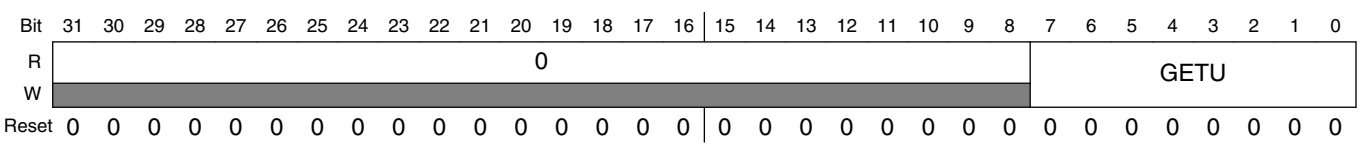


EMVSIMx_TX_BUF field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TX_BYTE	Transmit Data Byte Write to this register to fill the transmit FIFO with the bytes to be transmitted. The Tx FIFO can be written to only when the transmitter is enabled. Writing to this register while transmitter is disabled will lead to ignoring of all write accesses to this register. Reads to this register will return zeros. **NOTE:** Writing more data to the transmit FIFO than it can hold, will cause a Transmit FIFO Full (TFF) error.

21.5.14 Transmitter Guard ETU Value Register (EMVSIMx_TX_GETU)

Address: 4004_E000h base + 34h offset = 4004_E034h



EMVSIMx_TX_GETU field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GETU	Transmitter Guard Time Value in ETU Used to control the number of additional Elementary Time Units (ETUs) inserted between bytes transmitted by the EMV SIM transmitter. An ETU is equivalent to one bit time at the given baud rate (for example, the length of a START bit). The guard time has no effect on the EMV SIM receiver. A value of 0x00 inserts no additional ETUs, while a value of 0xFE inserts 254 additional ETUs. A value of 0xFF subtracts one ETU by reducing the number of STOP bits from two to one. 0 no additional ETUs inserted (default) 1 1 additional ETU inserted ...

Table continues on the next page...

EMVSIMx_TX_GETU field descriptions (continued)

Field	Description
	254 254 additional ETUs inserted 255 Subtracts one ETU by reducing the number of STOP bits from two to one

21.5.15 Character Wait Time Value Register (EMVSIMx_CWT_VAL)

Address: 4004_E000h base + 38h offset = 4004_E038h

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

EMVSIMx_CWT_VAL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CWT	Character Wait Time Value The value written to this register will specify the number of ETU times allowed between characters. Default is 0xFFFF

21.5.16 Block Wait Time Value Register (EMVSIMx_BWT_VAL)

Address: 4004_E000h base + 3Ch offset = 4004_E03Ch

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

EMVSIMx_BWT_VAL field descriptions

Field	Description
BWT	Block Wait Time Value The time from START bit of last byte sent from the EMV SIM module to the START bit of the first byte sent from the Smart Card must be less than the value in this register. If it is not, then the BWT_TO flag is set.

21.5.17 Block Guard Time Value Register (EMVSIMx_BGT_VAL)

Address: 4004_E000h base + 40h offset = 4004_E040h

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

EMVSIMx_BGT_VAL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
BGT	Block Guard Time Value The value in this register is the block guard time. Time from START bit of last byte sent from the EMV SIM module to the START bit of the first byte sent from the SmartCard must be greater than this value. If it is not, then the BGT flag will be set.

21.5.18 General Purpose Counter 0 Timeout Value Register (EMVSIMx_GPCNT0_VAL)

Address: 4004_E000h base + 44h offset = 4004_E044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																GPCNT0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

EMVSIMx_GPCNT0_VAL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GPCNT0	General Purpose Counter 0 Timeout Value The value written to this register will be used to compare to the general purpose counter 0 in the EMV SIM module. Once the General purpose counter reaches this value, the GPCNT0 flag in the TX_STATUS register will be set. This counter is intended to be used for any events that must be monitored for duration based on the card clock, receiver sample rate, or ETU rate (transmit clock). Example: ATR arrival time and ATR duration.

21.5.19 General Purpose Counter 1 Timeout Value (EMVSIMx_GPCNT1_VAL)

Address: 4004_E000h base + 48h offset = 4004_E048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																GPCNT1																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

EMVSIMx_GPCNT1_VAL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
GPCNT1	General Purpose Counter 1 Timeout Value The value written to this register will be used to compare to the general purpose counter in the EMV SIM module. Once the General purpose counter reaches this value, the GPCNT1 flag in the XMT_STATUS register will be set. This counter is intended to be used for any events that must be monitored for duration based on the card clock, receiver sample rate, or ETU rate (transmit clock). Example: ATR arrival time and ATR duration.

21.6 Functional Description

21.6.1 Initialization

21.6.1.1 Configuring EMV SIM

The first operation that must be done is to configure the EMV SIM interface signals to allow the EMV SIM Transmit or Receive operation to commence. The following steps describe a typical way to enable the Smart Card:

- Configure prescaler by programming the desired value in PRSC register to generate the card clock. The card clock frequency is calculated as:
  - f_{SIM} / (PRSC[7:0]); where f_{SIM} is the EMV SIM Clock Frequency
  - Prescaler value must be set such that the maximum card clock frequency specification set by EMV & ISO-7816 specifications are not violated
- Configure the DIVISOR_VALUE in the Baud Rate Divisor Register. This value determines the ETU period to be used by the transmitter or receiver
- Enable power to the Smart Card by setting the VCC_EN bit in PCSR register
- Enable the Smart Card clock by setting the SCEN bit in PCSR register

- Release Smart Card reset using the SRST bit in PCSR register to bring the Smart Card out of reset
- Program the power down functionality i.e. if Auto Power Down of Smart Card is required or not
- Select the Data Format type for the EMV SIM
  - Either set the IC bit to the desired data format (inverse or direct) OR
  - Set the ICM bit to allow automatic detection of Initial Character. IC bit will be updated to desired data format upon valid Initial Character detection
- Configure if the EMV SIM module is enabled to run in STOP and DOZE modes by setting the STOP_EN and DOZE_EN bits as desired
- Enable DMA request generation for Transmitter or Receiver as desired

Once the above configuration is complete, the receiver or transmitter can be configured depending on the operation to be carried out on the Smart Card.

NOTE

XMT_EN and RCV_EN bits should not be changed in the same write cycle. Software should attempt separate write accesses to CTRL register for updating these bits, one by one. For example, if XMT_EN = 1 and RCV_EN = 0 and in order to enable the receiver, first write to CTRL register should make XMT_EN = 0 and second write to CTRL register should make RCV_EN = 1.

21.6.1.2 Configuring Receiver

The following is a list of configurations that need to be performed (after the "Configuring EMV SIM" step) in order to configure the EMV SIM receiver for operation:

- Enable NACK generation capability
  - To generate NACK on parity errors and invalid initial characters, set the ANACK bit in CTRL register
  - To generate NACK on receiver FIFO overflow error, set the ONACK bit in CTRL register
- Configure the desired threshold for NACK generation and data threshold in the RX_THD register. These are the thresholds at which the RTE and RDTF flags will be set, respectively.
  - If auto power down is desired on RTE flag assertion, then write to SAPD bit PCSR to initiate the Smart Card power down
- Configure the timeout values for Character Wait Time Counter, the Block Wait Timer Counter and the Block Guard Time Counter
- Enable necessary interrupts by clearing respective bits in the INT_MASK register

- Configure the General Purpose Counter as needed and select desired clock source (GPCNT0/1_CLKSEL in the CLKCFG register)
- Enable Receiver by setting RCV_EN bit to 1 in CTRL register
- Receiver will enter Initial Character Detection mode if ICM bit is set or enter normal receive mode if ICM bit is not set
- Enable LRC or CRC as desired

21.6.1.3 Configuring Transmitter

The following is a list of configurations that need to be performed (after the "Configuring EMV SIM" step) in order to configure the EMV SIM transmitter for operation:

- Select desired re-transmission threshold for NACKed characters in TX_THD register
  - This is the threshold at which the TNTE flag will be set by using the TNACK_THD[3:0] bits
- Configure the desired Guard Time between the characters transmitted by writing to the GETU[7:0] bits in TX_GETU register
- Configure the desired data threshold in TX_THD register.
- Enable necessary interrupts by clearing respective bits in the INT_MASK register
- Configure the General Purpose Counter as needed and select desired clock source (GPCNT0/1_CLKSEL in the CLKCFG register)
- Enable Transmitter by setting the XMT_EN in the CTRL register
- Enable LRC & CRC as desired
- Program bytes to be transmitted into the Tx FIFO. If DMA is enabled, DMA request will get asserted automatically.

NOTE

Tx FIFO can be written to after the transmitter is enabled (XMT_EN = 1). Writing to Tx FIFO while transmitter is disabled will not be successful and write access will be ignored.

NOTE

The Transmit and Receive operations are mutually exclusive and should not be enabled together.

21.6.2 Smart Card Interface and Control

The EMV SIM module controls the all the interface signals for the Smart Card. The Smart Card clock is generated using the prescaler (PRSC register) and the ETU periods are generated using the baud rate divisor (DIVISOR register). The Smart Card Reset and Voltage Enable are controlled by software through the PCSR register. The message bytes are transmitted and received on the Card IO device pin which is used in open drain configuration.

21.6.2.1 Smart Card Presence Detect

The Smart Card Presence Detect pin (SIMPD) allows for detection of the insertion or removal of a Smart Card. The SPDES bit in the PCSR register allows the software to configure which edge of the SIMPD pin will cause a presence detect interrupt flag to be asserted.

An internal pull-down is required on SIMPD pin to allow a high to low transition on this pin when the Smart Card is removed.

21.6.2.2 Powering Up

The first step to communicating with a Smart Card is to provide power and a clock signal to the card. Once the card is detected as present (using the presence detect features, or some other method), the Smart Card should be powered up according to the power up sequence specified in the ISO 7816-3 specification.

1. Apply voltage to the Smart Card by setting the VCC_EN bit in the PCSR register.
2. Select the appropriate clock frequency for the Smart Card by programming the CLKCFG and DIVISOR registers.
3. Enable the clock to the Smart Card by setting the SCEN bit in the PCSR register.
4. Remove the card from reset by setting the SRST bit in the PCSR register.

21.6.2.3 Automatic Power Down

Smart Cards require a proper sequence of operations to be followed to allow them to be powered down. The power down sequence as per ISO 7816 specifications is:

- Smart Card Reset (RST) pin transitions from high to low
- Smart Card Clock (CLK) is turned off to low
- Smart Card IO (IO) is transitioned from high impedance to low
- Smart Card Voltage Enable (VCC_EN) to turned off (pin driven low)

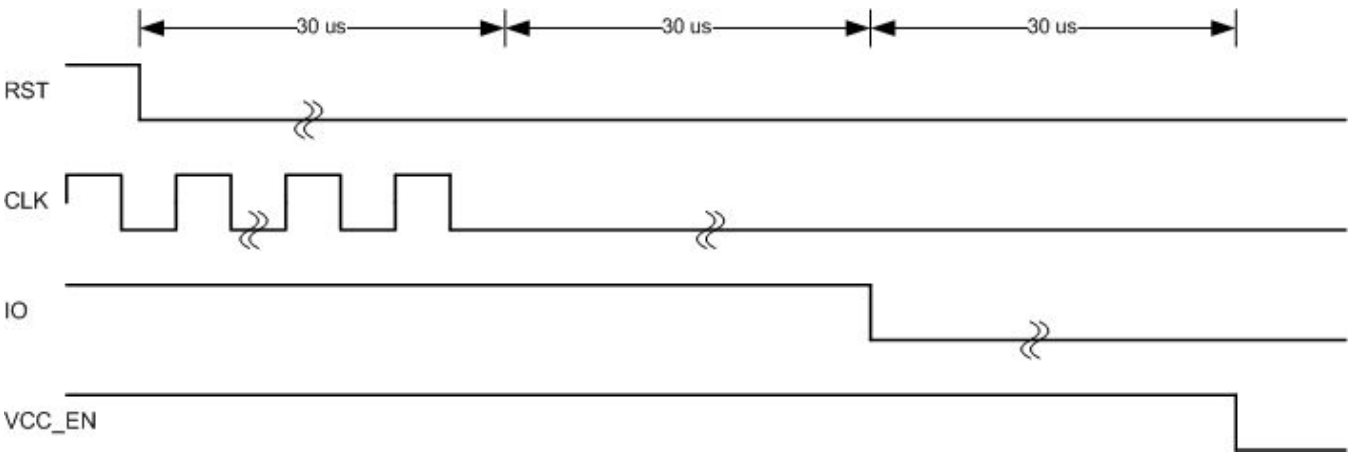


Figure 21-2. Auto Power Down Sequence

These operations can be done manually by software by writing to appropriate bits (SCEN, SRST and VCC_EN) in the PCSR register or automatically by hardware under the following conditions provided the auto power down feature is enabled in the PCSR register.

- Setting the SPD bit in PSCR register by software
- Assertion of RTE bit in RX_STATUS register
- Assertion of SPDIF bit in PCSR register when configured to detect card removal

NOTE

The SPDES bit in PCSR register determines which edge transition of the Smart Card Presence Detect pin is used for Smart Card presence detection. Presence detection can be used to determine if the card has been inserted or removed. The occurrence of the edge specified by SPDES bit will cause the following: SPDIF to be set; if the SPDIM mask is clear, an interrupt to the CPU; and if SAPD in the PCSR register is set, an auto power down sequence to begin. If Smart card insertion is expected, SAPD can be set low to avoid the auto power down sequence from falsely triggering. The bit SPDP can be used to determine the current state of the Smart Card Presence Detect pin.

NOTE

The auto power down operation requires a low frequency clock (usually 32 kHz clock source) to be active on the device.

The auto power down sequence will clear the following register bits upon power down sequence completion:

- SPD bit in PCSR register
- SCEN bit in PCSR register

- SRST bit in PCSR register
- VCC_EN bit in PCSR register
- XMT_EN and RCV_EN bits in CTRL register

21.6.3 EMV SIM Receiver

The EMV SIM Receiver is designed to receive all message bytes and store valid bytes into the receive FIFO. It performs checks on the incoming messages and indicates the transmitter to enable NACK insertion for each message. In addition, the receiver automatically detects NACKs received when the transmitter is sending out message bytes.

21.6.3.1 Data Sampling

The incoming data is internally synchronized and sampled using a 16x oversampled clock, called the EMV SIM logic clock. The frequency of this clock is 16 times the ETU Clock (1/ETU Period). The whole receiver works on this EMV SIM logic clock.

21.6.3.2 Initial Character Detection

The EMV SIM module supports the detection of the Initial Character and determine the data format when it is placed in the initial character mode by setting the ICM bit in CTRL register. When placed in this mode, the EMV SIM module samples in the input line and waits for one of the initial characters to be detected. Once detected, the ICM bit is cleared and the data format bit, IC bit, is set to appropriate value depending on the data format detected using the initial character.

The two possible data formats are inverse convention and direct convention. Figures below illustrate the differences between the two formats. Essentially, inverse convention differs from direct convention in that the order of the bits is flipped (i.e. MSB for LSB) and the data bits and parity bit are logically inverted. When receiving inverse convention data, the transformation of the data back to direct convention format is done in hardware, including the inversion of the data and parity bits.

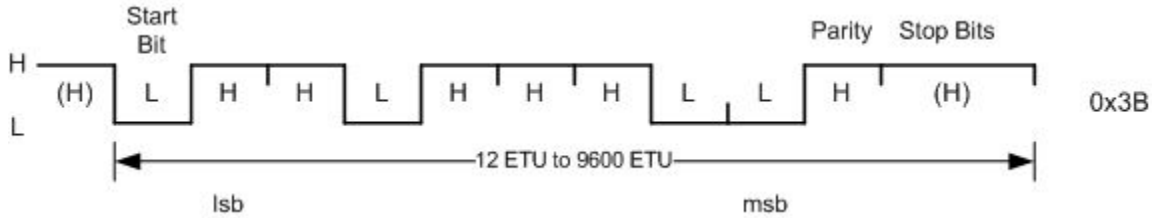


Figure 21-3. Initial Character in Direct Conversion

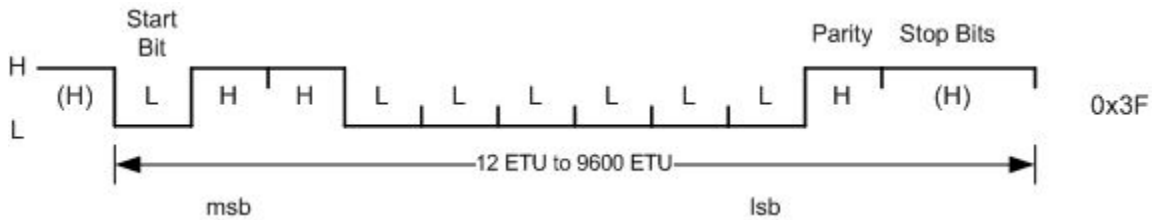


Figure 21-4. Initial Character in Inverse Conversion

To place the EMV SIM into initial character mode, set the ICM bit in the CTRL register. Once a valid initial character is received, the IC bit in the CTRL register will be set accordingly by the hardware, and the ICM bit will be cleared. Software can read the state of the IC bit to determine which mode the EMV SIM is currently using. The 0x3B (as decoded by direct convention) with parity bit high will cause direct convention to be used (IC set to logic 0), whereas a 0x3F (as decoded by inverse convention) with parity bit high will cause inverse convention to be used (IC set to logic 1).

The receiver remains in initial character mode until disabled by software or a valid initial character is detected. Upon detection of valid character, the receiver enters data receive mode. It is recommended to use one of the general purpose counter as a timeout during initial character mode.

When the receiver is in initial character mode, all received bytes will continue to be placed into the receive FIFO whether they be valid initial characters or not. If a valid initial character is received that causes the data format being used to change, all subsequent bytes will be decoded with that format before being placed into the FIFO, including the initial character byte itself. That is, if the IC bit is low, and the correct initial character for setting inverse convention is received, that character and all subsequently received characters will be stored in the FIFO after having been decoded using inverse convention (for example, the initial character will be stored as 0x3F).

NOTE

There is a condition where a parity error in the initial byte for direct convention (0x3B) could be decoded as what appears to be a valid initial character for inverse convention (that is, 0x3F). The EMV SIM module will not recognize this as a valid

initial character for inverse convention and will set the PEF flag.

21.6.3.3 Receiver Diagnostics

The EMV SIM module performs diagnostic checks on the incoming messages. These checks include parity check and message frame checks.

Parity Error Detection: The receiver is responsible for detecting parity errors in the received data. Data is always transmitted with even parity, except when in inverse convention mode. In inverse convention mode, all data bits and the parity bit are complemented making the data appear to be odd parity. The parity bit is defined as the 10th bit of the received data. The parity of the 2nd through 10th received bits is calculated by the receiver parity logic. This logic determines if the parity of the 9 received bits is correct.

When a parity error is detected on a given byte, the parity error flag (PEF bit) is asserted in the RX_STATUS register if NACK generation on errors is disabled (ANACK = 0 in CTRL register). The parity error flag does not generate an interrupt to the system, however, the module can cause the transmitter to transmit a NACK (if ANACK = 1 in CTRL register) without software needed to enable transmitter for this and request re-transmission of the current byte.

Framing Error Detection: The receiver is responsible for detecting framing errors in the received data. Data is always transmitted with 2 STOP bits, except when in 11 ETU mode, where there is one STOP bit. The STOP bits are defined as high state at the 11th and 12th bit positions in the received data. In either mode, the receiver expects to see a high at the 11th bit position in the received data. If a low is detected at the 11th bit position, the receiver indicates a framing error for that byte.

When a framing error is detected on a given byte, the FEF bit is set in the RX_STATUS register. A framing error cannot cause an interrupt, nor can it create a NACK pulse to the Smart Card asking for a retransmission of the corrupted data.

LRC & CRC: The receiver also checks the LRC or CRC on the incoming block of bytes (if LRC or CRC is enabled). The result of the CRC or LRC check is updated in the RX_STATUS register once the complete message block is received.

21.6.3.4 NACK Detection

During transmission of message bytes, the receiver automatically checks for NACKs sent by the Smart Card. The software does not need to enable receiver for this. The NACK is detected by sampling the IO line at 11 ETUs after the falling edge of the START bit. Once a NACK is detected, the transmitter is signaled and the transmitter will retransmit the current byte after two ETU time (as per ISO 7816 specification).

21.6.3.5 Using EMV SIM Receiver with "T=1" Smart Cards

The EMV SIM module provides hardware support for “T=1” type Smart Cards. These type of cards present several requirements above and beyond the standard “T=0” cards. The features provided to meet the requirements that pertain to the EMV SIM receiver are as follows:

- 11 ETU Characters
  - “T=1” cards can transmit with character lengths of 11 ETUs (that is, one STOP bit). The EMV SIM module provides the RCVR11 bit in the CTRL register in order to configure the receiver state machine to accept 11 ETU characters.
- Character Wait Time Counter
  - The character waiting time (CWT) is defined as the time between the start bits of two consecutive characters received from the Smart Card. The value of CWT can range from 12 ETU to 32779 ETU. The EMV SIM module provides a 16-bit counter with programmable comparator clocked at the ETU bit rate to identify when the CWT has been exceeded by the Smart Card.
- Block Wait Time
  - The block waiting time (BWT) is defined as the maximum time between the start bits of the last character of a transmitted block and the first character of the next received block. The block wait time must not exceed a value that is programmable. The EMV SIM module provides a 32-bit Block Wait Timer that can be used to identify when the BWT has been violated.
- Block Guard Time
  - The block guard time (BGT) is defined as the minimum delay between the start bits of the last character of a transmitted block and the first character of the next received block. The block guard must be greater than a value that is programmable. The EMV SIM module provides a 16-bit Block Guard Timer that can be used to identify when the BGT has been violated.
- Error Detection Code
  - “T=1” cards can specify LRC or CRC error detection codes to be used. The EMV SIM module provides hardware support for both the LRC and CRC operations.

21.6.4 EMV SIM Transmitter

The transmitter is responsible for reading the message bytes from the Tx FIFO, adding the LRC or CRC at the end of the message block and re-transmit any byte on reception of a NACK. It assists the receive operation by inserting NACKs when directed by the receiver.

21.6.4.1 Message Transmission

The transmitter operation does not start until the Tx FIFO is empty. Software can configure the EMV SIM module and enable the transmitter but unless a byte is written into the Tx FIFO, no transmission will start. Clearing the XMT_EN bit while the transmitter is in operation, will halt any transmission in progress, flush the transmit FIFO. Refer Message Handling section for more details on using the Tx FIFO.

The transmitter continuously transmits the messages from the Tx FIFO. Upon NACK detection, the transmitter resends the same byte until the Transmitter NACK Threshold limit is reached. When TNTE is set, the EMV SIM transmitter is halted, and all pending transfers are aborted, and the TC, ETC, AND TFE flags are set. All bytes remaining in the transmit FIFO are lost. There is no way to restart the transmission on the next byte in the FIFO. The transmitter remains frozen until TNTE is cleared by software. The only way to clear TNTE is to write a 1 to the TNTE bit in the TX_STATUS register. It is possible to disable the detection of NACKs from the Smart Card by setting TNCK_THD[3:0] to 0x0. By setting TNCK_THD[3:0] to 0x1, it is possible to disable all retransmissions while still setting TNTE on the first NACK received. In general, TNTE is set on the NACK that causes the threshold set by TNCK_THD[3:0] to be reached. This final NACK will not cause a retransmission, whereas all previous NACKs will cause a retransmission.

If all bytes are successfully transmitted and the Tx FIFO is empty, the transmitter send the check byte (i.e. LRC or CRC) if the XMT_CRC_LRC and LRC_EN or CRC_EN bits are set. The transmitter sends each byte to the LRC or CRC block to allow the check byte generation.

At the end of each byte, the transmitter inserts the programmed Guard ETUs. Refer to Protocol Timers section for more details on Guard Time Counter.

21.6.4.2 NACK Insertion

The transmitter is responsible for driving the output low during the STOP bit time to signify an error was detected in the received data from the Smart Card. This logic responds to a NACK request generated by the receiver block.

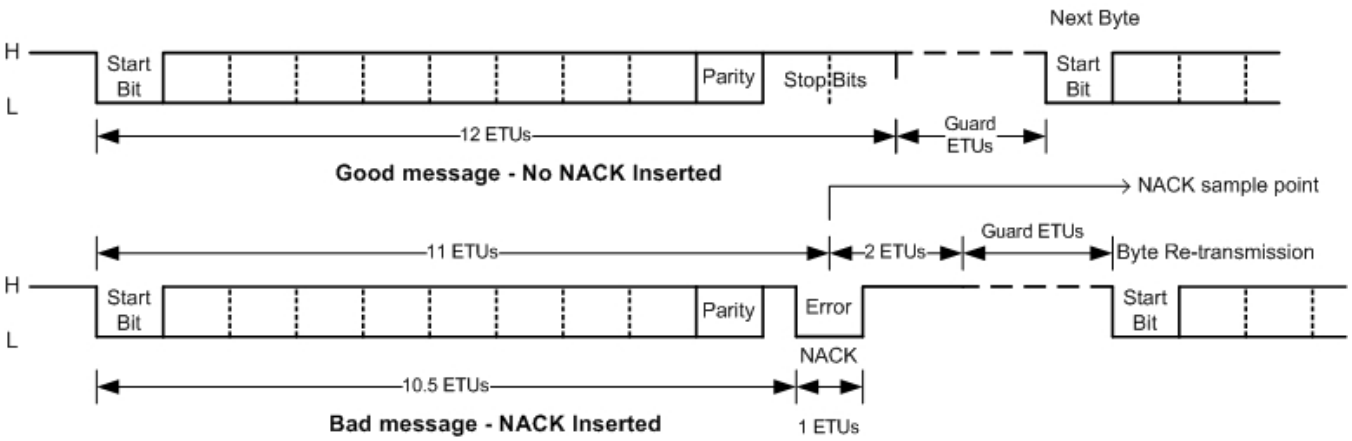


Figure 21-5. NACK Insertion

The NACK generation logic is also responsible for keeping track of the number of NACKs received during a transmit operation. The receiver detects NACKs generated by the Smart Card, and reports them to the transmit NACK logic. Once the number of detected NACKs has reached the programmed threshold (TNCK_THD[3:0]), an interrupt flag (TNTE) is generated, the transmit FIFO is flushed, and the transmitter is disabled.

21.6.4.3 Using EMV SIM Transmitter with "T=1" Smart Cards

The EMV SIM module provides hardware support for “T=1” type Smart Cards. These type of cards present several requirements above and beyond the standard “T=0” cards. The features provided to meet the requirements that pertain to the EMV SIM transmitter are as follows:

- 11 ETU Characters
  - The EMV SIM module transmitter has a programmable guard time register that allows the programmer to specify the number of ETUs between character transmissions. Programming a value of 255 (0xFF) in the GETU[7:0] bits in the GUARD_CNTL register will set the number of ETUs per character transmitted to 11.
- Character Waiting Time
  - The character waiting time (CWT) is defined as the time between the start bits of two consecutive characters. The value of CWT can range from 12 ETU to 32779 ETU. The time between transmitted characters is controlled by the

programmable guard time in the GETU register. However, the time between the last byte in the transmit FIFO, and the next transmitted byte can be largely affected by software response time to the transmit interrupts. The EMV SIM transmitter provides a Transmit FIFO threshold (TDTF) interrupt to signal the system when the expected number of characters have been transmitted from the transmit FIFO. The minimum CWT is achieved only if the software can respond to the TDTF interrupt and write new data to the transmit FIFO before the last character in the Transmit FIFO has been sent.

- Block Waiting Time
  - The block waiting time (BWT) is defined as the maximum time between the start bits of the last character of a transmitted block and the first character of the next received block. The value of BWT is always greater than 1800 ETU. The EMV SIM transmitter provides two General Purpose Counter(s) either of which can be used to track the BWT. The BWT is purely determined by software response time to the transmit interrupts.
- Block Guard Time
  - The block guard time (BGT) is defined as the minimum delay between the start bits of the last character of a transmitted block and the first character of the next received block. The value of BGT is 22 ETU. The EMV SIM module supports the BGT by providing the ability to generate an interrupt when the last byte is received, and transmitting within 2 ETU after the XMT_EN bit is set. The BGT will be determined by the speed at which the software can react to an interrupt and enable the transmitter.
- Error Detection Code
  - “T=1” cards can specify LRC or CRC error detection codes to be used. The EMV SIM module provides hardware support for both the LRC and CRC operation.

21.6.5 LRC and CRC

21.6.5.1 LRC Block

The EMV SIM module provides an 8-bit Linear Redundancy Check (LRC) generator / checker. The block is provided for use with T=1 Smart Cards that support LRC. This block can be enabled through the LRC_EN bit in the CNTL register. This block performs an 8-bit exclusive-OR on all received or transmitted characters. At the end of the reception of a block of characters, the result is expected to be 00. If so, the LRC_OK bit is set in the RX_STATUS register. During transmission, the LRC block Exclusive-ORs

each character that is transmitted with the current value of the LRC. If the XMT_CRC_LRC bit in the CTRL register is set, the LRC value will automatically be sent by the transmitter as the final character when the transmit FIFO empties.

The LRC value can be reset in multiple ways. Clearing the LCR_EN bit in the CTRL register will reset the LRC value. At the end of a transmission (either after the LRC byte is transmitted, or after the last character in the transmit FIFO is sent when XMT_CRC_LRC is clear), the LRC value is automatically reset by the EMV SIM hardware. Finally, when setting the XMT_EN bit, the EMV SIM hardware resets the LRC value.

Summary of configuration needed to use LRC block:

- Use the LRC_EN bit in CTRL register to enable the LRC block
- Use the XMT_CRC_LRC bit to enable the transmission of the LRC Character after the last character in the Transmit FIFO is sent.

21.6.5.2 CRC Block

The EMV SIM module provides an 16-bit Cyclic Redundancy Check (CRC) generator / checker. The block is provided for use with T=1 Smart Cards that support CRC. This block can be enabled through the CRC_EN bit in the CTRL register. This block performs a polynomial based check on all received or transmitted characters. The polynomial used for calculating the CRC is $G(x) = x^{16} + x^{12} + x^5 + 1$. The CRC register is initialized to all "1" before the data is shifted in. The CRC block can be configured to compute using the ISO 7816 CRC method or the 16-bit CCITT CRC method. The software can select the following bits: CRC_IN_FLIP, CRC_OUT_FLIP, and INV_CRC_VAL of the CTRL register.

- To select CCITT CRC method, set
  - CRC_IN_FLIP = 1
  - CRC_OUT_FLIP = 1
  - INV_CRC_VAL = 1
- To select normal CRC method (i.e. ISO 7816 CRC method), set
  - CRC_IN_VAL = 0
  - CRC_OUT_VAL = 0
  - INV_CRC_VAL = 1

During transmission, the CRC block updates the current value of the CRC using each character. If the XMT_CRC_LRC bit in the CTRL register is set, the CRC value will automatically be formatted as per the selected CRC method (CCITT or normal) and sent by the EMV SIM transmitter as the final two characters when the transmit FIFO empties. At the end of the reception of a block of characters, the residual from the CRC calculation is compared and the CRC_OK bit is set in the RX_STATUS register.

NOTE

Setting any other value for CRC_IN_FLIP, CRC_OUT_FLIP and INV_CRC_VAL will cause CRC value to be formatted accordingly; however the CRC will not be compliant to the standard CRC used in Smart Cards.

The CRC value can be reset/initialized in multiple ways. Clearing the CRC_EN bit in the CTRL register will reset the CRC value. At the end of a transmission (either after the CRC characters are transmitted, or after the last character in the transmit FIFO is sent when XMT_CRC_LRC is clear), the CRC value is automatically reset by the EMV SIM hardware. Finally, when setting the XMT_EN bit, the EMV SIM hardware resets the CRC value.

Summary of configuration needed to use CRC block:

- Use CRC_IN_FLIP, CRC_OUT_FLIP and INV_CRC_VAL bits in CTRL register to select the desired CRC method (CCITT or normal)
- Use the CRC_EN bit in CTRL register to enable the CRC block
- Use the XMT_CRC_LRC bit to enable the transmission of the CRC Characters after the last character in the Transmit FIFO is sent.

21.6.6 Message Handling

The EMV SIM module has FIFOs on both transmit and receive side for handling all message bytes.

21.6.6.1 Transmit FIFO

A 4-byte deep FIFO is implemented in the transmitter. The transmit FIFO can be written into by the software but reads to this FIFO will return zeros. To take care of clock synchronization, each write access will have wait states inserted by the module but this is transparent to the software. The Tx FIFO can be written to while the transmitter is enabled, however, the transmitter does not start transmitting while the FIFO is empty. A write to the Tx FIFO will initiate the transmit operation. Software can store more bytes into the Tx FIFO until the programmed threshold is reached. Software can enable DMA operation by setting TX_DMA_EN = 1 in the CTRL register. On setting this bit, a DMA request will be asserted and remain asserted till the number of bytes in FIFO reached the programmed threshold value (TDT[3:0] in the TX_THD register).

The transmit FIFO can be flushed by setting the FLSH_TX bit in the CTRL register. A transmit NACK threshold error (TNTE) will halt the transmitter, and flush the transmit FIFO. The flush operation resets the transmit read and write pointers to equal values. Everything in the transmitter block is reset by the transmit flush operation. This does not include the control registers associated with the transmitter. The Transmit data threshold flag (TDTF) does not get cleared by the FLSH_TX operation.

21.6.6.2 Receive FIFO

A 4-byte deep FIFO is implemented in the receiver. Since more than 4-bytes can be received in the FIFO, the software must ensure that the Rx FIFO is periodically read. The Rx FIFO can be read by software via normal CPU accesses on interrupt (RX_DATA interrupt flag) or via DMA by setting the DMA_RX_EN bit in CTRL register. The software cannot write to this register. A write access may terminate in a transfer error.

The Rx FIFO is loaded by the receiver when it receives an error free message byte with correct parity. The FIFO stores the message byte only. When the total bytes in the Rx FIFO equals the programmed threshold value (RDT[3:0] in RX_THD register), the RDTF bit is asserted in the RX_STATUS register. An interrupt will be asserted if the RDT_IM bit in the INT_MASK is cleared. If DMA access to RX_FIFO is enabled, the DMA request to read the Rx FIFO will be asserted when the threshold flag is set and will clear when all the bytes are read out from the Rx FIFO.

The receive FIFO can be flushed by setting the FLSH_RX bit in the CTRL register. The flush operation resets the receiver except for receiver control registers.

NOTE

Since the register bits and the receiver logic are in different clock domains, the software needs to make sure to allow clock synchronization time when sampling the RDTF bit. One proposed method could be to read one byte at a time (from Rx FIFO) when RDTF is asserted, allow a time of 3 bus clock cycles after Rx FIFO read and then check the status of the RDTF bit and repeat as necessary. Alternatively, one could set the receiver threshold one more than actually needed.

Rx FIFO Overflow Detection: When a byte is received by the receiver and the Rx FIFO is not read and already contains 4 bytes, a FIFO overflow error will be asserted (RFO bit set to 1 in RX_STATUS register). The received byte will be discarded leaving the FIFO with the first 4 bytes received. If the ONACK bit in the CTRL register is asserted, the

transmitter will generate a NACK for the overflow condition every time a new byte is received that cannot be stored in the Rx FIFO. The EMV SIM module will continually send NACKs till the Rx FIFO is read or NACK transmit threshold is reached.

21.6.7 Protocol Timers

The EMV SIM module has several 16-bit and 32-bit timers to allow the software perform protocol timing checks and measurements like:

- Guard Time
- Character Wait Time
- Block Wait Time
- Block Guard Time
- Work Wait Time

In addition to the above, the EMV SIM module also contains two 32-bit general purpose counters which can be used by software for any application specific time measurement.

21.6.7.1 Transmit Guard Time

The Transmit Guard time is the number of ETUs inserted by the transmitter between each character being transmitted. The number of ETUs is controlled by programming the TX_GETU register. The Guard Time is measured as the time (in ETUs) between the STOP bits of previous character and the START bit of next character.

For a Guard Time programmed as 0xFF in GETU[7:0] bits of TX_GETU register, the transmitter inserts one STOP bit instead of two.

21.6.7.2 Character Wait Time

The EMV SIM receiver block includes a 16-bit counter that counts the number of bit times (ETUs) between received characters (i.e. time between start bits in the consecutive characters received). When enabled, the Character Wait Time Counter (CWT) will not start counting until after the START bit(s) of a valid character has been received. The counter is synchronized to the receive character bit positions so that an accurate count of the number of ETUs between characters can be made. The Character Wait Time Counter has a 16-bit programmable comparator. Software can write the expected number of ETUs between characters to the comparator. If the time between characters exceeds this value, an interrupt flag will be set and an interrupt generated if the mask is clear. The CWT can be configured as follows:

- Program the CWT_VAL register to the value that needs to be enforced
- Activate CWT function by setting the CWT_EN bit in the CTRL register
- Enable the CWT_ERR interrupt by clearing the CWT_IM bit in the INT_MASK register
- If an interrupt occurs, then the software should read the RX_STATUS register to determine if the error was caused by a CWT_ERR violation. The software should clear the CWT_ERR interrupt by writing a “1” to the CWT_ERR bit in the RX_STATUS register
- Software can disable the CWT counter by clearing the CWT_EN bit in the CTRL register at any time

21.6.7.3 Block Wait Time and Block Guard Time

A 32-bit timer is used to measure both Block Wait Time (BWT) and Block Guard Time (BGT). The BWT and BGT timer are used to measure the delay between the leading edge of the last character of the block received by the card and the leading edge of the first character of the next block sent by the card. If this time measured is larger than the value in BWT_VAL register, then the BWT_ERR flag will be set and an interrupt generated. If the time is less than the value in the BGT_VAL register, then the BGT_ERR flag will be set and an interrupt generated. The block wait timer can be configured as follows:

- Program the BWT_VAL and BGT_VAL registers to the value that need to be enforced
- Activate the block wait timer by writing '1' to BWT_EN bit in the CTRL register
- Enable the BWT_ERR and BGT_ERR interrupts by clearing the BWT_IM and BGT_IM bits in the INT_MASK register
- If an interrupt occurs, then the software should read the RX_STATUS register to determine if the error was caused by a BWT_ERR or BGT_ERR violation. The software should clear the interrupt status bit by writing a “1” to the BWT_ERR bit or BGT_ERR, respectively, in the RX_STATUS register
- Software can disable the BWT counter by clearing the BWT_EN bit in the CTRL register at any time

NOTE

BWT should be enabled after all bytes are written to in Tx FIFO. This is necessary to avoid the condition where writes to Tx FIFO are very slow (more than 1 ETU time apart) and if BWT is enabled earlier, then BWT error can trigger falsely.

21.6.7.4 Work Wait Time

There is no separate counter to measure the work wait time (WWT). Work wait time is a combination of BWT and CWT. To measure WWT, software must configure both CWT_VAL and BWT_VAL to the value to enforced. When measuring WWT, the BGT timer will not be used so it should be masked (inactive) by the BGT_IM bit. Below is the sequence to program the EMV SIM to send and receive data while checking WWT:

- Program both the CWT_VAL and the BWT_VAL registers to the WWT (work wait time) value that needs to be enforced. Set BGT register to 0 (this is the default value).
- Activate both the CWT and BWT functions by setting the CWT_EN and BWT_EN bit in the CTRL register
- Enable the CWT_ERR and BWT_ERR interrupts by clearing the CWT_IM and BWT_IM bits in the INT_MASK register
- Program the EMV SIM module to enable Receiver or Transmitter as desired by application
- If an interrupt occurs, then the software should consider that a WWT violation if the CWT_ERR or BWT_ERR bits are asserted. The software should clear the CWT_ERR or BWT_ERR interrupt by writing a “1” to the BWT_ERR or CWT_ERR bit in the RX_STATUS register

21.6.7.5 General Purpose Timers

The EMV SIM module provides TWO 16-bit counters for use, when timing events during Smart Card communication. The clock source for the counter is selectable between three sources: Card Clock, Receiver Clock (16 times the ETU clock) or ETU Clock (for Transmitter). The GPCNT0/1_CLKSEL[1:0] bits in the CTRL register are used to select the clock input. The counter is enabled as soon as the input clock is selected. The starting of the counter is immediate once the input clock is running. Software can control the three input clock sources by using the SCEN, KILL_CLOCKS, RCV_EN and XMT_EN bits provided in the CTRL register.

To run the counters from the card clock source the following conditions must be met:

- ‘KILL_CLOCKS = 0’ in the CTRL register
- ‘RCV_EN = 1’ or XMT_EN = 1 in the CTRL register
- 'SCEN = 1' in the PCSR register
- ‘GPCNT_CLKSET[1:0] = 01’ in the CLKCFG register

The counter will begin to count at the card clock rate as soon as these conditions are met.

To run the counters from the receive clock source the following conditions must be met:

- ‘KILL_CLOCKS = 0’ in the CTRL register.

- ‘RCV_EN = 1’ or XMT_EN = 1 in the CTRL register
- ‘GPCNT0/1_CLKSET[1:0] = 10’ in the CLKCFG register

The counter will begin to count at the receive clock rate as soon as these conditions are met.

To run the counters from the ETU (transmit) clock source the following conditions must be met:

- ‘KILL_CLOCKS = 0’ in the CTRL register.
- ‘RCV_EN = 1’ or XMT_EN = 1 in the CTRL register
- ‘GPCNT0/1_CLKSET[1:0] = 11’ in the CLKCFG register

The counters will begin to count at the ETU clock rate as soon as these conditions are met.

The counters can be reset by setting GPCNT0/1_CLKSEL[1:0] to 00. A 16-bit comparator value (GPCNT0/1_VAL register) is provided that allows the software to select a count value at which the interrupt flags (GPCNT0/1_TO bit in TX_STATUS register) can be set and an interrupt generated if the mask (GPCNT0/1_IM bits in INT_MASK register) are clear.

The following sequence should be followed to configure both general purpose timers:

- Program counter comparator using the GPCNT0/1_VAL register
- Enable the conditions described above using the CTRL register
- Select desired clock source for the General Purpose Counter using the CLKCFG register
- Enable interrupts in the INT_MASK register

21.6.8 Answer To Reset (ATR) Detection

The first communication that occurs between the Smart Card and the EMV SIM module will be a block of data sent from the Smart Card to the EMV SIM module after the card is powered up and the card reset is removed (As described in Smart Card Interface and Control Section). This block is called the Answer To Reset (ATR). To receive the ATR, the EMV SIM module should be configured for 12 ETU character reception. According to the ISO 7816-3 spec, both “T=0” and “T=1” cards will communicate initially using 12 ETU character durations.

The following steps provide a suggested approach to configure the EMV SIM module to receive the ATR.

- Clear RCVR11 bit in the CTRL register

- Set ANACK bit in the CTRL register to enable NACK generation. The ISO 7816-3 spec allows the EMV SIM module to NACK any communication errors that occur during the initial communication at 12 ETU.

NOTE

The Europay MasterCard and VISA (EMV) cards are similar to "T=1", but do not allow the EMV SIM module to NACK during the initial communication

- Enable RDTF and RFO interrupts by setting RDT_IM and RFO_IM bits in INT_MASK register. These interrupts are enabled to notify the software when characters are received. A threshold can be set for the number of characters to receive before generating an interrupt.
- Set desired threshold for received characters by writing RDT in RX_THD register

The EMV SIM should be setup to perform in initial character mode during ATR detection. This will cause the hardware to identify the first valid character sent during the ATR as an initial character. This character will automatically configure the hardware for the data convention used by the Smart Card.

- Set Initial Character Mode by setting the ICM bit in the CTRL register

The ISO 7816-3 spec requires that Smart Cards meet certain timing restrictions. One of these is the time from the de-assertion of the card reset to the beginning of the ATR sequence. The EMV SIM module's General Purpose Counter can be used to verify that the Smart Card begins its ATR within the 400 to 40,000 card clock cycle range.

- Set the General Purpose Counter 0 or 1 Timeout Value to 0x9C40 using GPCNT0/1_VAL register.
- Enable General Purpose Counter Interrupt by clearing GPCNT0/1_IM in INT_MASK register.
- Enable General Purpose Counter by programming the GPCNT0/1_CLKSEL[1:0] bits to 01, so that the card clock is used for counting.

The ISO7816-3 spec states that the maximum allowed time between two characters during the ATR is 9600 ETUs (Initial Waiting Time). The Character Wait Time Counter should be setup to detect any errors for this condition.

- Set Character Wait Time Counter Comparator to 9600 using the CWT_VAL register
- Enable the Character Wait Time Counter Interrupt by clearing CWT_ERR_IM in INT_MASK register
- Enable the Character Wait Time Counter by setting the CWT_EN bit in the CTRL register

The last step in preparing for ATR reception is to enable the receiver.

- Set RCV_EN bit in CTRL register

The EMV SIM module will generate interrupts once a threshold number of characters is received. The software should react to these interrupts and read the characters from the receive FIFO (RX_BUF) until the complete ATR has been received. If a General Purpose Counter interrupt occurs before the final ATR character is received, then the card should be deactivated according to the ISO 7816-3 spec (See Section Smart Card Interface and Control). Otherwise, once a valid ATR is received, the software will know from the ATR information the specific characteristics for this card (refer to the ISO 7816-3 spec for details).

21.6.8.1 Programming Considerations for T=0 Smart Cards

If the card is of type T=0, the software should adjust the following parameters according to the information in the ATR:

- Adjust the baud rate by changing the values of DIVISOR register based on the values of 'F' and 'D' received in the ATR.
- Adjust the guard time between characters by changing the value of GETU[7:0] in the TX_GETU register.
- Adjust NACK capability by modifying the values of the ONACK and ANACK bits in the CTRL register.
- Adjust the stop clock polarity by modifying the value of the SCSP bit in the PCSR register.
- Adjust the level of transmit NACK re-transmissions allowed by modifying the value of the TNCK_THD[3:0] bits in the TX_THD register.
- Adjust the level for the Receive NACK threshold by modifying the RNCK_THD[3:0] bits in the RX_THD register.

If a negotiation with the Smart Card is desired, the software sends a PPS response to the Smart Card. To send the response, the following steps should be performed:

- Set the desired transmit FIFO threshold level by writing the TDT[3:0] bits in the TX_THD register.
- Clear all transmit interrupt flags in the TX_STATUS register by writing a one to the bits of the register.
- Enable the transmit interrupts desired by clearing the mask bits in the INT_MASK register. If more than 4 characters are to be sent, it is suggested that the TDTF interrupt be used to signify when to write more characters to the transmit FIFO. This results in the most efficient transfer times to the Smart Card.
- Enable the transmitter by setting the XMT_EN bit in the CTRL register.
- Write the characters to be sent as response (max 4) to the transmit FIFO using the TX_BUF register.

At this point, the EMV SIM module will transmit the characters in the transmit FIFO. If more than 4 characters are to be sent, the transmit threshold interrupt will be set when the threshold number of characters are remaining in the FIFO. The software can then write an additional number of characters to be sent without interrupting transmission to the Smart Card.

Once the transmission is complete, the EMV SIM module should be completely configured for standard operation with the T=0 Smart Card. The software can continue to service RDTF interrupts for received characters, and TDTF interrupts for transmitted characters.

21.6.8.2 Programming Considerations for T=1 Smart Cards

If the card is of type T=1, the software should adjust the following parameters according to the information in the ATR:

- Adjust the baud rate by changing the values of DIVISOR register based on the values of 'F' and 'D' received in the ATR.
- Adjust the guard time between characters by changing the value of GETU[7:0] in the TX_GETU register. Setting GETU[7:0] to 0xFF configures the SIM transmitter for 11 ETU transmissions.
- Disable NACK capability by clearing the ONACK and ANACK bits in the CTRL register. T=1 cards do not allow NACKs.
- Adjust the stop clock polarity by modifying the value of the SCSP bit in the PCSR register.
- Set Character Wait Time Counter Timeout Value to value specified in the ATR by using the CWT_VAL register.
- Enable the Character Wait Time Counter Interrupt by clearing CWT_ERR_IM in INT_MASK register.
- Enable the Character Wait Time Counter by setting the CWT_EN bit in the CTRL register.
- Enable CRC or LRC error checking according to the ATR information by setting either the CRC_EN or LRC_EN bit in the CTRL register. These bits should never be set at the same time!

For T=1 cards, the ATR is sent using a T=0 type of structure (12 ETU, no LRC or CRC). If a negotiation with the Smart Card is desired, the software will send a PPS response to the Smart Card. Otherwise, the protocol is initiated with a block transfer from the EMV SIM module. In order to send the response or the first block, the following steps should be performed:

- Set the desired transmit FIFO threshold level by writing the TDT[3:0] bits in the TX_THD register.

- Clear all transmit interrupt flags in the TX_STATUS register by writing a one to the bits of the register.
- Enable the transmit interrupts desired by clearing the mask bits in the INT_MASK register. If more than 4 characters are to be sent, it is suggested that the TDTF interrupt be used to signify when to write more characters to the transmit FIFO. This results in the most efficient transfer times to the Smart Card.
- Enable the transmission of the error checking characters (LRC or CRC) by setting the XMT_CRC_LRC bit in the CTRL register.

NOTE

If the card supports PPS, the software may not be allowed to send the LRC/CRC information until the PPS exchange is completed. If so, do not set the XMT_CRC_LRC bit during the PPS exchange.

- Enable the transmitter by setting the XMT_EN bit in the CTRL register.
- Write the characters to be sent as response (max 4) to the transmit FIFO using the TX_BUF register.

At this point, the EMV SIM module will transmit the characters in the transmit FIFO. If more than 4 characters are to be sent, the transmit threshold interrupt will be set when the threshold number of characters are remaining in the FIFO. The software can then write an additional number of characters to be sent without interrupting transmission to the Smart Card.

Once the transmission is complete, the EMV SIM module should be completely configured for standard operation with the T=1 Smart Card. The software can continue to service RDTF interrupts for received characters, and TDTF interrupts for transmitted characters.

Chapter 22
Flexible I/O (FlexIO)

22.1 Introduction

22.1.1 Overview

The FlexIO is a highly configurable module providing a wide range of functionality including:

- Emulation of a variety of serial/parallel communication protocols
- Flexible 16-bit timers with support for a variety of trigger, reset, enable and disable conditions
- Programmable logic blocks allowing the implementation of digital logic functions on-chip and configurable interaction of internal and external modules
- Programmable state machine for offloading basic system control functions from CPU

22.1.2 Features

The FlexIO module is capable of supporting a wide range of protocols including, but not limited to:

- UART
- I2C
- SPI
- I2S
- Camera IF
- Motorola 68K/Intel 8080 bus
- PWM/Waveform generation

The following key features are provided:

- Array of 32-bit shift registers with transmit, receive and data match modes

- Double buffered shifter operation for continuous data transfer
- Shifter concatenation to support large transfer sizes
- Automatic start/stop bit generation
- 1, 2, 4, 8, 16 or 32 multi-bit shift widths for parallel interface support
- Interrupt, DMA or polled transmit/receive operation
- Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during stop modes
- Highly flexible 16-bit timers with support for a variety of internal or external trigger, reset, enable and disable conditions
- Programmable logic mode for integrating external digital logic functions on-chip or combining pin/shifter/timer functions to generate complex outputs
- Programmable state machine for offloading basic system control functions from CPU with support for up to 8 states, 8 outputs and 3 selectable inputs per state

22.1.3 Block Diagram

The following diagram gives a high-level overview of the configuration of FlexIO timers and shifters.

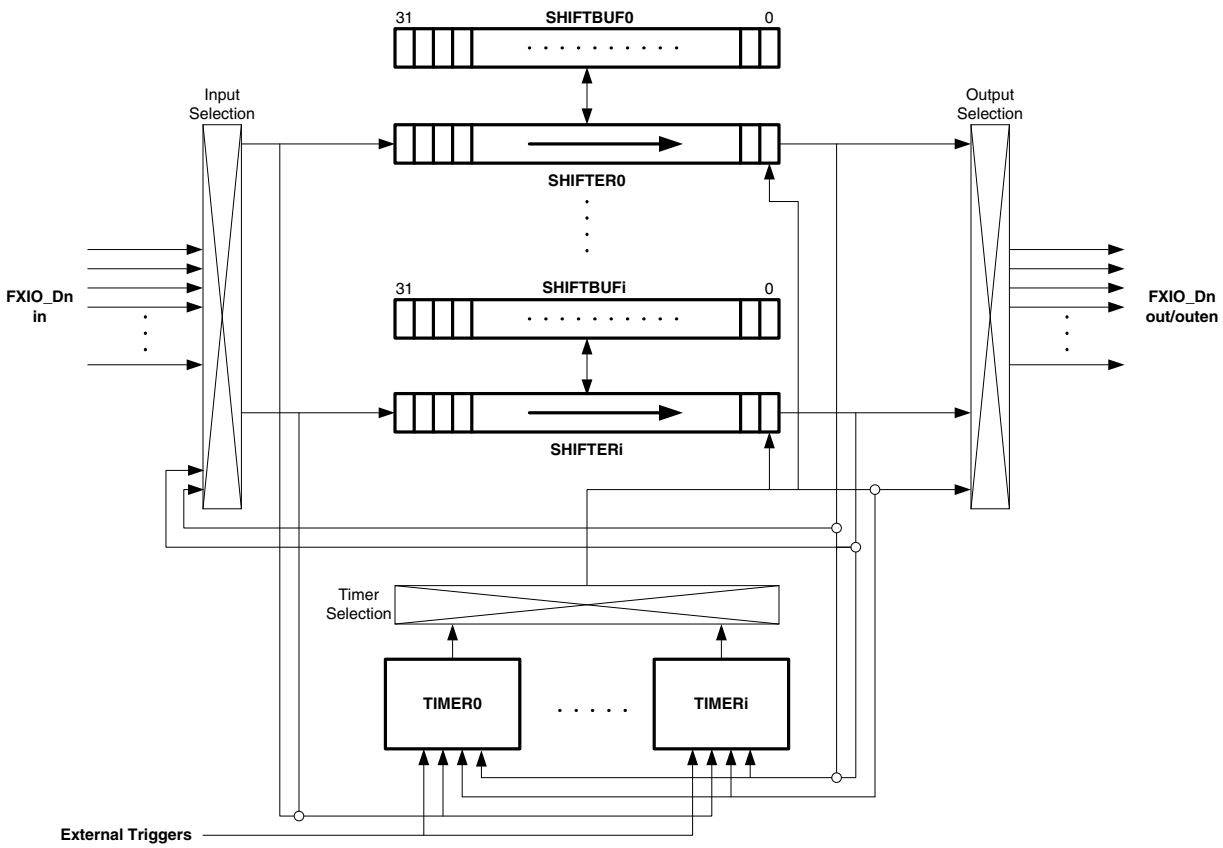


Figure 22-1. FlexIO block diagram

22.1.4 Modes of operation

The FlexIO module supports the chip modes described in the following table.

Table 22-1. Chip modes supported by the FlexIO module

Chip mode	FlexIO Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (CTRL[DOZEN]) is clear and the FlexIO is using an external or internal clock source which remains operating during stop/wait modes.
Low Leakage Stop	The Doze Enable (CTRL[DOZEN]) bit is ignored and the FlexIO will wait for all Timers to complete any pending operation before acknowledging low leakage mode entry.
Debug	Can continue operating provided the Debug Enable bit (CTRL[DBGE]) is set.

22.1.5 FlexIO Signal Descriptions

Signal	Description	I/O
FXIO_Dn (n=0...31)	Bidirectional FlexIO Shifter and Timer pin inputs/outputs	I/O

22.2 Memory Map and Registers

22.2.1 FLEXIO Register Descriptions

22.2.1.1 FLEXIO Memory Map

Offset	Register	Width (In bits)	Access	Reset value
400CA000h	Version ID (VERID)	32	RO	01010001h
400CA004h	Parameter (PARAM)	32	RO	04200808h
400CA008h	FlexIO Control (CTRL)	32	RW	00000000h
400CA00Ch	Pin State (PIN)	32	RO	00000000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
400CA010h	Shifter Status (SHIFTSTAT)	32	W1C	00000000h
400CA014h	Shifter Error (SHIFTERR)	32	W1C	00000000h
400CA018h	Timer Status (TIMSTAT)	32	W1C	00000000h
400CA020h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	00000000h
400CA024h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	00000000h
400CA028h	Timer Interrupt Enable (TIMIEN)	32	RW	00000000h
400CA030h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	00000000h
400CA040h	Shifter State (SHIFTSTATE)	32	RW	00000000h
400CA080h - 400CA09Ch	Shifter Control N (SHIFTCTL0 - SHIFTCTL7)	32	RW	00000000h
400CA100h - 400CA11Ch	Shifter Configuration N (SHIFTCFG0 - SHIFTCFG7)	32	RW	00000000h
400CA200h - 400CA21Ch	Shifter Buffer N (SHIFTBUF0 - SHIFTBUF7)	32	RW	00000000h
400CA280h - 400CA29Ch	Shifter Buffer N Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS7)	32	RW	00000000h
400CA300h - 400CA31Ch	Shifter Buffer N Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS7)	32	RW	00000000h
400CA380h - 400CA39Ch	Shifter Buffer N Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS7)	32	RW	00000000h
400CA400h - 400CA41Ch	Timer Control N (TIMCTL0 - TIMCTL7)	32	RW	00000000h
400CA480h - 400CA49Ch	Timer Configuration N (TIMCFG0 - TIMCFG7)	32	RW	00000000h
400CA500h - 400CA51Ch	Timer Compare N (TIMCMP0 - TIMCMP7)	32	RW	00000000h
400CA680h - 400CA69Ch	Shifter Buffer N Nibble Byte Swapped (SHIFTBUFNBS0 - SHIFTBUFNBS7)	32	RW	00000000h
400CA700h - 400CA71Ch	Shifter Buffer N Half Word Swapped (SHIFTBUFHWS0 - SHIFTBUFHWS7)	32	RW	00000000h
400CA780h - 400CA79Ch	Shifter Buffer N Nibble Swapped (SHIFTBUFNIS0 - SHIFTBUFNIS7)	32	RW	00000000h

22.2.1.2 Version ID (VERID)

22.2.1.2.1 Address

Register	Offset
VERID	400CA000h

22.2.1.2.2 Function

.

22.2.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

22.2.1.2.4 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for the module specification.
23-16	Minor Version Number
MINOR	This read only field returns the minor version number for the module specification.
15-0	Feature Specification Number
FEATURE	This read only field returns the feature set number. 0000000000000000b - Standard features implemented. 0000000000000001b - Supports state, logic and parallel modes.

22.2.1.3 Parameter (PARAM)

22.2.1.3.1 Address

Register	Offset
PARAM	400CA004h

22.2.1.3.2 Function

.

22.2.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRIGGER								PIN							
W																
Reset	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER								SHIFTER							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

22.2.1.3.4 Fields

Field	Function
31-24 TRIGGER	Trigger Number Number of external triggers implemented.
23-16 PIN	Pin Number Number of Pins implemented.
15-8 TIMER	Timer Number Number of Timers implemented.
7-0 SHIFTER	Shifter Number Number of Shifters implemented.

22.2.1.4 FlexIO Control (CTRL)

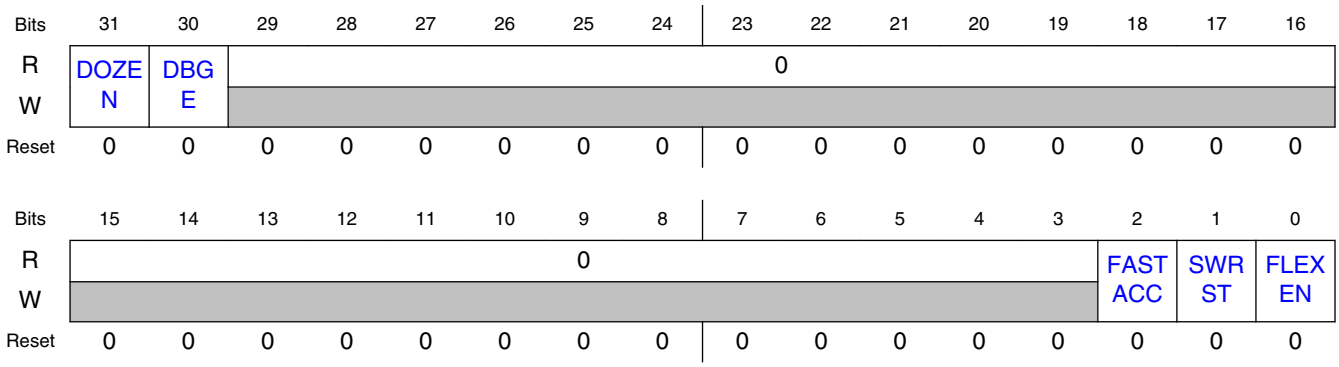
22.2.1.4.1 Address

Register	Offset
CTRL	400CA008h

22.2.1.4.2 Function

.

22.2.1.4.3 Diagram



22.2.1.4.4 Fields

Field	Function
31 DOZEN	Doze Enable Disables FlexIO operation in Doze modes. This field is ignored and the FlexIO always disabled in low-leakage stop modes. 0b - FlexIO enabled in Doze modes. 1b - FlexIO disabled in Doze modes.
30 DBGE	Debug Enable Enables FlexIO operation in Debug mode. 0b - FlexIO is disabled in debug modes. 1b - FlexIO is enabled in debug modes
29-3 —	
2 FASTACC	Fast Access Enables fast register accesses to FlexIO registers, but requires the FlexIO clock to be at least twice the frequency of the bus clock. 0b - Configures for normal register accesses to FlexIO 1b - Configures for fast register accesses to FlexIO
1 SWRST	Software Reset The FlexIO Control Register is not affected by the software reset, all other logic in the FlexIO is affected by the software reset and register accesses are ignored until this bit is cleared. This register bit will remain set until cleared by software, and the reset has cleared in the FlexIO clock domain. 0b - Software reset is disabled 1b - Software reset is enabled, all FlexIO registers except the Control Register are reset.
0 FLEXEN	FlexIO Enable 0b - FlexIO module is disabled. 1b - FlexIO module is enabled.

22.2.1.5 Pin State (PIN)

22.2.1.5.1 Address

Register	Offset
PIN	400CA00Ch

22.2.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.5.3 Fields

Field	Function
31-0 PDI	Pin Data Input Returns the input data on each of the FlexIO pins.

22.2.1.6 Shifter Status (SHIFTSTAT)

22.2.1.6.1 Address

Register	Offset
SHIFTSTAT	400CA010h

22.2.1.6.2 Function

.

22.2.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSF							
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.6.4 Fields

Field	Function
31-8 —	
7-0 SSF	Shifter Status Flag The shifter status flag is updated when one of the following events occurs: For SMOD=Receive, the status flag is set when SHIFTBUF has been loaded with data from Shifter (SHIFTBUF is full), and the status flag is cleared when SHIFTBUF register is read. For SMOD=Transmit, the status flag is set when SHIFTBUF data has been transferred to the Shifter (SHIFTBUF is empty) or when initially configured for SMOD=Transmit, and the status flag is cleared when the SHIFTBUF register is written. For SMOD=Match Store, the status flag is set when a match has occured between SHIFTBUF and Shifter, and the status flag is cleared when the SHIFTBUF register is read. For SMOD=Match Continuous, returns the current match result between the SHIFTBUF and Shifter. For SMOD=State, the status flag for a shifter will set when it is selected by the current state pointer. For SMOD=Logic, returns the current value of the programmable logic block output. The status flag can also be cleared by writing a logic one to the flag for all modes except Match Continuous/State/Logic. 00000000b - Status flag is clear 00000001b - Status flag is set

22.2.1.7 Shifter Error (SHIFTERR)

22.2.1.7.1 Address

Register	Offset
SHIFTERR	400CA014h

22.2.1.7.2 Function

.

22.2.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SEF							
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.7.4 Fields

Field	Function
31-8 —	
7-0 SEF	Shifter Error Flags The shifter error flag is set when one of the following events occurs: For SMOD=Receive, indicates Shifter was ready to store new data into SHIFTBUF before the previous data was read from SHIFTBUF (SHIFTBUF Overrun), or indicates that the received start or stop bit does not match the expected value. For SMOD=Transmit, indicates Shifter was ready to load new data from SHIFTBUF before new data had been written into SHIFTBUF (SHIFTBUF Underrun). For SMOD=Match Store, indicates a match event occured before the previous match data was read from SHIFTBUF (SHIFTBUF Overrun). For SMOD=Match Continuous, the error flag is set when a match has occured between SHIFTBUF and Shifter. For SMOD=Logic, the error flag is set when the output of the programmable logic block has asserted. Can be cleared by writing logic one to the flag. For SMOD=Match Continuous, can also be cleared when the SHIFTBUF register is read. 00000000b - Shifter Error Flag is clear 00000001b - Shifter Error Flag is set

22.2.1.8 Timer Status (TIMSTAT)

22.2.1.8.1 Address

Register	Offset
TIMSTAT	400CA018h

22.2.1.8.2 Function

.

22.2.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TSF							
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.8.4 Fields

Field	Function
31-8 —	
7-0 TSF	Timer Status Flags The timer status flag sets depending on the timer mode, and can be cleared by writing logic one to the flag. In 8-bit counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements, this also causes the counter to reload with the value in the compare register. In 8-bit PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements, this also causes the counter to reload with the value in the compare register.. In 16-bit counter mode, the timer status flag is set when the 16-bit counter equals zero and decrements, this also causes the counter to reload with the value in the compare register.. 00000000b - Timer Status Flag is clear 00000001b - Timer Status Flag is set

22.2.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

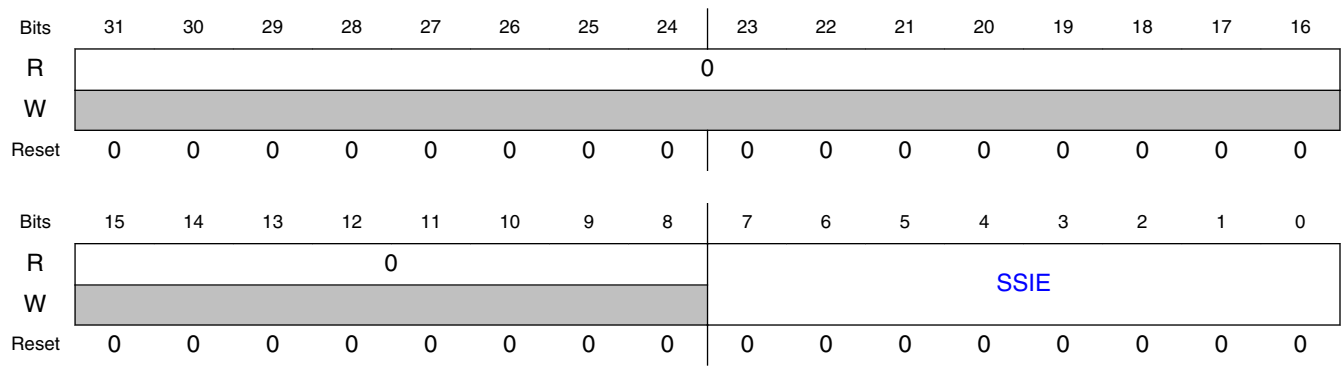
22.2.1.9.1 Address

Register	Offset
SHIFTSIEN	400CA020h

22.2.1.9.2 Function

.

22.2.1.9.3 Diagram



22.2.1.9.4 Fields

Field	Function
31-8 —	
7-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when corresponding SSF is set. 00000000b - Shifter Status Flag interrupt disabled 00000001b - Shifter Status Flag interrupt enabled

22.2.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

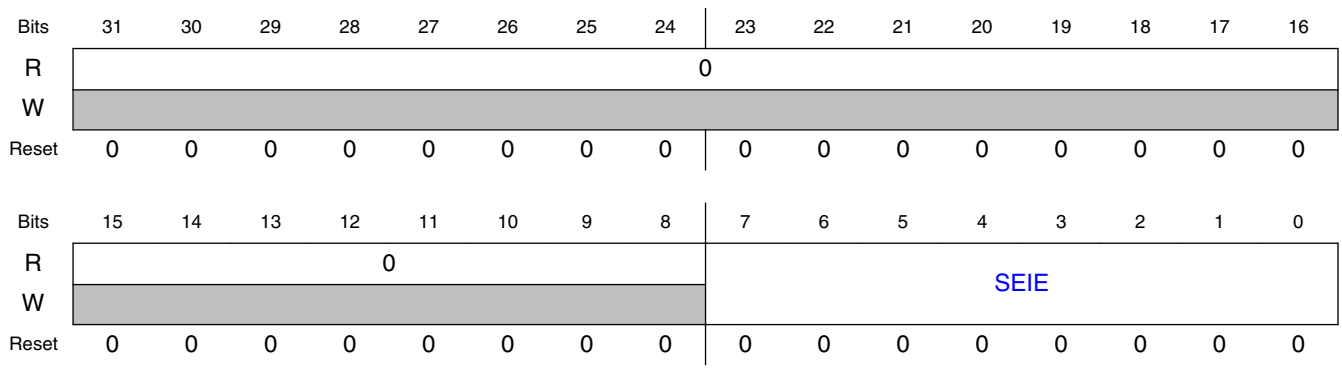
22.2.1.10.1 Address

Register	Offset
SHIFTEIEN	400CA024h

22.2.1.10.2 Function

.

22.2.1.10.3 Diagram



22.2.1.10.4 Fields

Field	Function
31-8 —	
7-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when corresponding SEF is set. 00000000b - Shifter Error Flag interrupt disabled 00000001b - Shifter Error Flag interrupt enabled

22.2.1.11 Timer Interrupt Enable (TIMIEN)

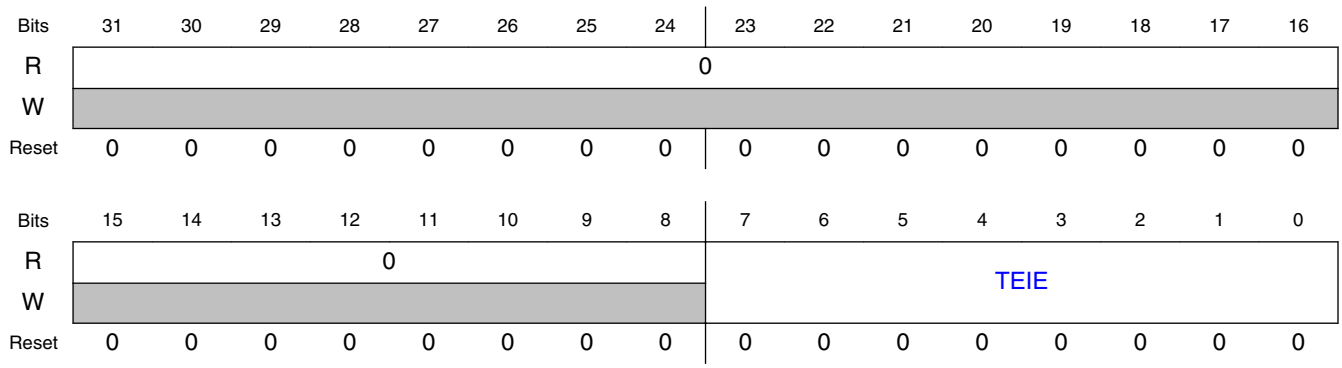
22.2.1.11.1 Address

Register	Offset
TIMIEN	400CA028h

22.2.1.11.2 Function

.

22.2.1.11.3 Diagram



22.2.1.11.4 Fields

Field	Function
31-8 —	
7-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when corresponding TSF is set. 00000000b - Timer Status Flag interrupt is disabled 00000001b - Timer Status Flag interrupt is enabled

22.2.1.12 Shifter Status DMA Enable (SHIFTSDEN)

22.2.1.12.1 Address

Register	Offset
SHIFTSDEN	400CA030h

22.2.1.12.2 Function

.

22.2.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSDE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.12.4 Fields

Field	Function
31-8 —	
7-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when corresponding SSF is set. 00000000b - Shifter Status Flag DMA request is disabled 00000001b - Shifter Status Flag DMA request is enabled

22.2.1.13 Shifter State (SHIFTSTATE)

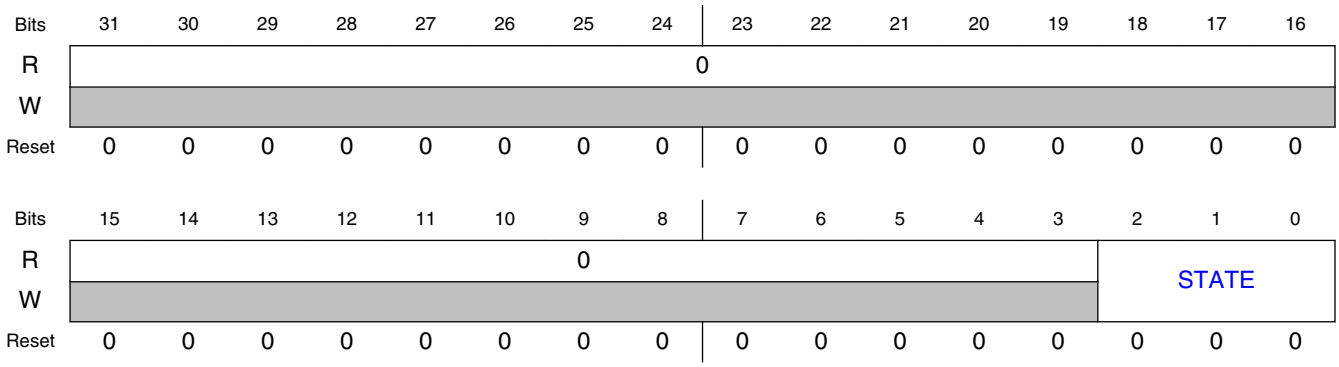
22.2.1.13.1 Address

Register	Offset
SHIFTSTATE	400CA040h

22.2.1.13.2 Function

.

22.2.1.13.3 Diagram



22.2.1.13.4 Fields

Field	Function
31-3 —	
2-0 STATE	Current State Pointer The current state field maintains a pointer to keep track of the current Shifter (configured for State mode) enabled to drive outputs and compute the next state. See 'State Mode' section for more detail.

22.2.1.14 Shifter Control N (SHIFTCTLa)

22.2.1.14.1 Address

Register	Offset
SHIFTCTL0	400CA080h
SHIFTCTL1	400CA084h
SHIFTCTL2	400CA088h
SHIFTCTL3	400CA08Ch
SHIFTCTL4	400CA090h
SHIFTCTL5	400CA094h
SHIFTCTL6	400CA098h
SHIFTCTL7	400CA09Ch

22.2.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					TIMSEL			TIMPOL	0					PINCFG	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PINSEL					PINPOL	0				SMOD		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.14.3 Fields

Field	Function
31-27 —	
26-24 TIMSEL	Timer Select Selects which Timer is used for controlling the logic/shift register and generating the Shift clock. TIMSEL=i will select TIMERi.
23 TIMPOL	Timer Polarity 0b - Shift on posedge of Shift clock 1b - Shift on negedge of Shift clock
22-18 —	
17-16 PINCFG	Shifter Pin Configuration 00b - Shifter pin output disabled 01b - Shifter pin open drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	
12-8 PINSEL	Shifter Pin Select Selects which pin is used by the Shifter input or output. PINSEL=i will select the FXIO_Di pin.
7 PINPOL	Shifter Pin Polarity 0b - Pin is active high 1b - Pin is active low
6-3 —	
2-0 SMOD	Shifter Mode Configures the mode of the Shifter. 000b - Disabled.

Field	Function
	001b - Receive mode. Captures the current Shifter content into the SHIFTBUF on expiration of the Timer. 010b - Transmit mode. Load SHIFTBUF contents into the Shifter on expiration of the Timer. 011b - Reserved. 100b - Match Store mode. Shifter data is compared to SHIFTBUF content on expiration of the Timer. 101b - Match Continuous mode. Shifter data is continuously compared to SHIFTBUF contents. 110b - State mode. SHIFTBUF contents are used for storing programmable state attributes. 111b - Logic mode. SHIFTBUF contents are used for implementing programmable logic look up table.

22.2.1.15 Shifter Configuration N (SHIFTCFGa)

22.2.1.15.1 Address

Register	Offset
SHIFTCFG0	400CA100h
SHIFTCFG1	400CA104h
SHIFTCFG2	400CA108h
SHIFTCFG3	400CA10Ch
SHIFTCFG4	400CA110h
SHIFTCFG5	400CA114h
SHIFTCFG6	400CA118h
SHIFTCFG7	400CA11Ch

22.2.1.15.2 Function

.

22.2.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											PWIDTH				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							INSRC	0	0	SSTOP		0		SSTART	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.15.4 Fields

Field	Function
31-21 —	
20-16 PWIDTH	Parallel Width For all Shifters, this register field configures the number of bits to be shifted on each Shift clock as follows: 1-bit shift for PWIDTH=0 4-bit shift for PWIDTH=1...3 8-bit shift for PWIDTH=4...7 16-bit shift for PWIDTH=8...15 32-bit shift for PWIDTH=16...31 For Shifters which support parallel transmit (SHIFTER0, SHIFTER4) or parallel receive (SHIFTER3, SHIFTER7), this register field, together with SHIFTCTL[PINSEL], also selects the pins to be driven or sampled on each Shift clock as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL] For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail.
15-9 —	
8 INSRC	Input Source Selects the input source for the shifter. 0b - Pin 1b - Shifter N+1 Output
7 —	
6 —	
5-4 SSTOP	Shifter Stop bit For SMOD=Transmit, this field allows automatic stop bit insertion if the selected timer has also enabled a stop bit. For SMOD=Receive or Match Store, this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Stop bit disabled for transmitter/receiver/match store 01b - Reserved for transmitter/receiver/match store 10b - Transmitter outputs stop bit value 0 on store, receiver/match store sets error flag if stop bit is not 0 11b - Transmitter outputs stop bit value 1 on store, receiver/match store sets error flag if stop bit is not 1
3-2 —	

Table continues on the next page...

Field	Function
1-0 SSTART	Shifter Start bit For SMOD=Transmit, this field allows automatic start bit insertion if the selected timer has also enabled a start bit. For SMOD=Receive or Match Store, this field allows automatic start bit checking if the selected timer has also enabled a start bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on enable 01b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on first shift 10b - Transmitter outputs start bit value 0 before loading data on first shift, receiver/match store sets error flag if start bit is not 0 11b - Transmitter outputs start bit value 1 before loading data on first shift, receiver/match store sets error flag if start bit is not 1

22.2.1.16 Shifter Buffer N (SHIFTBUFa)

22.2.1.16.1 Address

Register	Offset
SHIFTBUF0	400CA200h
SHIFTBUF1	400CA204h
SHIFTBUF2	400CA208h
SHIFTBUF3	400CA20Ch
SHIFTBUF4	400CA210h
SHIFTBUF5	400CA214h
SHIFTBUF6	400CA218h
SHIFTBUF7	400CA21Ch

22.2.1.16.2 Function

.

22.2.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.16.4 Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Shift buffer data is used for a variety of functions depending on the SMOD setting: For SMOD=Receive, Shifter data is transferred into SHIFTBUF at the expiration of Timer. For SMOD=Transmit, SHIFTBUF data is transferred into the Shifter before the Timer begins. For SMOD=Match Store/Continuous, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents. The Match is checked either continuosly (Match Continous mode) or when the Timer expires (Match Store mode). SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). In Match Store mode, Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. For SMOD=Logic, SHIFTBUF[31:0] is used to implement a 5-input, 32-bit programmable logic look-up table. See 'Logic Mode' section for more detail. For SMOD=State, SHIFTBUF[31:24] is used to drive the output value when this shifter is selected by the current state pointer and SHIFTBUF[23:0] is used to configure the value of the next state transition. See 'State Mode' section for more detail.

22.2.1.17 Shifter Buffer N Bit Swapped (SHIFTBUFBISa)

22.2.1.17.1 Address

Register	Offset
SHIFTBUFBIS0	400CA280h
SHIFTBUFBIS1	400CA284h
SHIFTBUFBIS2	400CA288h
SHIFTBUFBIS3	400CA28Ch
SHIFTBUFBIS4	400CA290h
SHIFTBUFBIS5	400CA294h

Table continues on the next page...

Register	Offset
SHIFTBUFBIS6	400CA298h
SHIFTBUFBIS7	400CA29Ch

22.2.1.17.2 Function

.

22.2.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.17.4 Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped. Reads return SHIFTBUF[0:31].

22.2.1.18 Shifter Buffer N Byte Swapped (SHIFTBUFBYSa)

22.2.1.18.1 Address

Register	Offset
SHIFTBUFBYS0	400CA300h
SHIFTBUFBYS1	400CA304h
SHIFTBUFBYS2	400CA308h
SHIFTBUFBYS3	400CA30Ch
SHIFTBUFBYS4	400CA310h

Table continues on the next page...

Register	Offset
SHIFTBUFBYS5	400CA314h
SHIFTBUFBYS6	400CA318h
SHIFTBUFBYS7	400CA31Ch

22.2.1.18.2 Function

.

22.2.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.18.4 Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are byte swapped. Reads return { SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24] }.

22.2.1.19 Shifter Buffer N Bit Byte Swapped (SHIFTBUFBBSa)

22.2.1.19.1 Address

Register	Offset
SHIFTBUFBBS0	400CA380h
SHIFTBUFBBS1	400CA384h
SHIFTBUFBBS2	400CA388h
SHIFTBUFBBS3	400CA38Ch

Table continues on the next page...

Register	Offset
SHIFTBUFBBS4	400CA390h
SHIFTBUFBBS5	400CA394h
SHIFTBUFBBS6	400CA398h
SHIFTBUFBBS7	400CA39Ch

22.2.1.19.2 Function

.

22.2.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.19.4 Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped within each byte. Reads return { SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7] }.

22.2.1.20 Timer Control N (TIMCTLa)

22.2.1.20.1 Address

Register	Offset
TIMCTL0	400CA400h
TIMCTL1	400CA404h
TIMCTL2	400CA408h

Table continues on the next page...

Register	Offset
TIMCTL3	400CA40Ch
TIMCTL4	400CA410h
TIMCTL5	400CA414h
TIMCTL6	400CA418h
TIMCTL7	400CA41Ch

22.2.1.20.2 Function

.

22.2.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		TRGSEL						TRGPOL	TRGSRC	0				PINCFG	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PINSEL					PINPOL	0					TIMOD	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.20.4 Fields

Field	Function
31-30 —	
29-24 TRGSEL	Trigger Select The valid values for TRGSEL will depend on the FLEXIO_PARAM register. • When TRGSRC = 1, the valid values for N will depend on PIN, TIMER, SHIFTER fields in the FLEXIO_PARAM register. • When TRGSRC = 0, the valid values for N will depend on TRIGGER field in FLEXIO_PARAM register. Refer to the chip configuration section for external trigger selection. **NOTE:** For a pin, N=0 to 31. For a Shifter/Timer, N=0 to 7. The internal trigger selection is configured as follows: • 4*N - Pin 2*N input • 4*N+1 - Shifter N status flag • 4*N+2 - Pin 2*N+1 input • 4*N+3 - Timer N trigger output
23	Trigger Polarity

Table continues on the next page...

Field	Function
TRGPOL	0b - Trigger active high 1b - Trigger active low
22 TRGSRC	Trigger Source 0b - External trigger selected 1b - Internal trigger selected
21-18 —	
17-16 PINCFG	Timer Pin Configuration 00b - Timer pin output disabled 01b - Timer pin open drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output
15-13 —	
12-8 PINSEL	Timer Pin Select Selects which pin is used by the Timer input or output. PINSEL=i will select the FXIO_Di pin.
7 PINPOL	Timer Pin Polarity 0b - Pin is active high 1b - Pin is active low
6-2 —	
1-0 TIMOD	Timer Mode In 8-bit counter mode, the lower 8-bits of the counter and compare register are used to configure the baud rate of the timer shift clock and the upper 8-bits are used to configure the shifter bit count. In 8-bit PWM mode, the lower 8-bits of the counter and compare register are used to configure the high period of the timer shift clock and the upper 8-bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit counter mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. 00b - Timer Disabled. 01b - Dual 8-bit counters baud/bit mode. 10b - Dual 8-bit counters PWM mode. 11b - Single 16-bit counter mode.

22.2.1.21 Timer Configuration N (TIMCFGa)

22.2.1.21.1 Address

Register	Offset
TIMCFG0	400CA480h
TIMCFG1	400CA484h

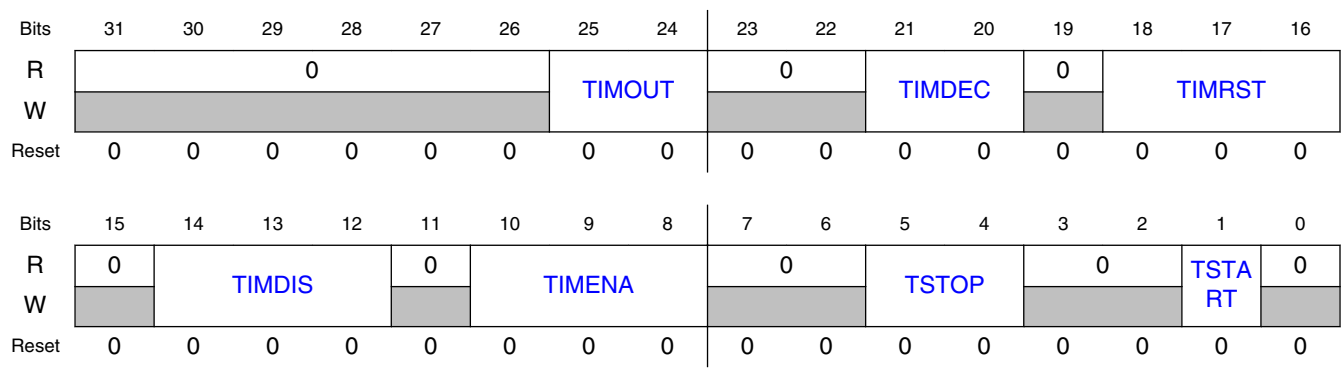
Table continues on the next page...

Register	Offset
TIMCFG2	400CA488h
TIMCFG3	400CA48Ch
TIMCFG4	400CA490h
TIMCFG5	400CA494h
TIMCFG6	400CA498h
TIMCFG7	400CA49Ch

22.2.1.21.2 Function

The options to enable or disable the timer using the Timer N-1 enable or disable are reserved when N is evenly divisible by 4 (eg: Timer 0).

22.2.1.21.3 Diagram



22.2.1.21.4 Fields

Field	Function
31-26 —	
25-24 TIMOUT	Timer Output Configures the initial state of the Timer Output and whether it is affected by the Timer reset. 00b - Timer output is logic one when enabled and is not affected by timer reset 01b - Timer output is logic zero when enabled and is not affected by timer reset 10b - Timer output is logic one when enabled and on timer reset 11b - Timer output is logic zero when enabled and on timer reset
23-22 —	
21-20 TIMDEC	Timer Decrement Configures the source of the Timer decrement and the source of the Shift clock. 00b - Decrement counter on FlexIO clock, Shift clock equals Timer output.

Table continues on the next page...

Field	Function
	01b - Decrement counter on Trigger input (both edges), Shift clock equals Timer output. 10b - Decrement counter on Pin input (both edges), Shift clock equals Pin input. 11b - Decrement counter on Trigger input (both edges), Shift clock equals Trigger input.
19 —	
18-16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset will only reset the lower 8-bits that configure the baud rate. In all other modes, the timer reset will reset the full 16-bits of the counter. 000b - Timer never reset 001b - Reserved 010b - Timer reset on Timer Pin equal to Timer Output 011b - Timer reset on Timer Trigger equal to Timer Output 100b - Timer reset on Timer Pin rising edge 101b - Reserved 110b - Timer reset on Trigger rising edge 111b - Timer reset on Trigger rising or falling edge
15 —	
14-12 TIMDIS	Timer Disable Configures the condition that causes the Timer to be disabled and stop decrementing. 000b - Timer never disabled 001b - Timer disabled on Timer N-1 disable 010b - Timer disabled on Timer compare 011b - Timer disabled on Timer compare and Trigger Low 100b - Timer disabled on Pin rising or falling edge 101b - Timer disabled on Pin rising or falling edge provided Trigger is high 110b - Timer disabled on Trigger falling edge 111b - Reserved
11 —	
10-8 TIMENA	Timer Enable Configures the condition that causes the Timer to be enabled and start decrementing. 000b - Timer always enabled 001b - Timer enabled on Timer N-1 enable 010b - Timer enabled on Trigger high 011b - Timer enabled on Trigger high and Pin high 100b - Timer enabled on Pin rising edge 101b - Timer enabled on Pin rising edge and Trigger high 110b - Timer enabled on Trigger rising edge 111b - Timer enabled on Trigger rising or falling edge
7-6 —	
5-4 TSTOP	Timer Stop Bit The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters will output the contents of the stop bit when the timer is disabled. When stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00b - Stop bit disabled 01b - Stop bit is enabled on timer compare

Table continues on the next page...

Field	Function
	10b - Stop bit is enabled on timer disable 11b - Stop bit is enabled on timer compare and timer disable
3-2 —	
1 TSTART	Timer Start Bit When start bit is enabled, configured shifters will output the contents of the start bit when the timer is enabled and the timer counter will reload from the compare register on the first rising edge of the shift clock. 0b - Start bit disabled 1b - Start bit enabled
0 —	

22.2.1.22 Timer Compare N (TIMCMPa)

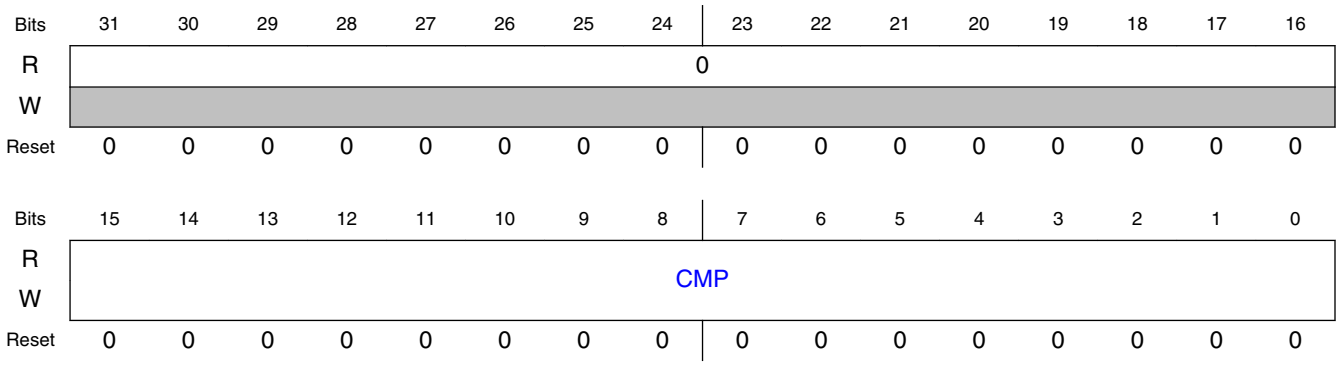
22.2.1.22.1 Address

Register	Offset
TIMCMP0	400CA500h
TIMCMP1	400CA504h
TIMCMP2	400CA508h
TIMCMP3	400CA50Ch
TIMCMP4	400CA510h
TIMCMP5	400CA514h
TIMCMP6	400CA518h
TIMCMP7	400CA51Ch

22.2.1.22.2 Function

.

22.2.1.22.3 Diagram



22.2.1.22.4 Fields

Field	Function
31-16 —	
15-0 CMP	Timer Compare Value The timer compare value is loaded into the timer counter when the timer is first enabled, when the timer is reset and when the timer decrements down to zero. In dual 8-bit counters baud/bit mode, the lower 8-bits configures the baud rate divider equal to (CMP[7:0] + 1) * 2. The upper 8-bits configure the number of bits in each word equal to (CMP[15:8] + 1) / 2. In dual 8-bit counters PWM mode, the lower 8-bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2.

22.2.1.23 Shifter Buffer N Nibble Byte Swapped (SHIFTBUFNBSa)

22.2.1.23.1 Address

Register	Offset
SHIFTBUFNBS0	400CA680h
SHIFTBUFNBS1	400CA684h
SHIFTBUFNBS2	400CA688h
SHIFTBUFNBS3	400CA68Ch
SHIFTBUFNBS4	400CA690h
SHIFTBUFNBS5	400CA694h
SHIFTBUFNBS6	400CA698h
SHIFTBUFNBS7	400CA69Ch

22.2.1.23.2 Function

.

22.2.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.23.4 Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped within each byte. Reads return { SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4] }.

22.2.1.24 Shifter Buffer N Half Word Swapped (SHIFTBUFHWSa)

22.2.1.24.1 Address

Register	Offset
SHIFTBUFHWS0	400CA700h
SHIFTBUFHWS1	400CA704h
SHIFTBUFHWS2	400CA708h
SHIFTBUFHWS3	400CA70Ch
SHIFTBUFHWS4	400CA710h
SHIFTBUFHWS5	400CA714h
SHIFTBUFHWS6	400CA718h
SHIFTBUFHWS7	400CA71Ch

22.2.1.24.2 Function

.

22.2.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFHWS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFHWS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.24.4 Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are half word swapped. Reads return { SHIFTBUF[15:0], SHIFTBUF[31:24] }.

22.2.1.25 Shifter Buffer N Nibble Swapped (SHIFTBUFNISa)

22.2.1.25.1 Address

Register	Offset
SHIFTBUFNIS0	400CA780h
SHIFTBUFNIS1	400CA784h
SHIFTBUFNIS2	400CA788h
SHIFTBUFNIS3	400CA78Ch
SHIFTBUFNIS4	400CA790h
SHIFTBUFNIS5	400CA794h
SHIFTBUFNIS6	400CA798h
SHIFTBUFNIS7	400CA79Ch

22.2.1.25.2 Function

.

22.2.1.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22.2.1.25.4 Fields

Field	Function
31-0 SHIFTBUFNIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped. Reads return { SHIFTBUF[3:0], SHIFTBUF[7:4], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[27:24], SHIFTBUF[31:28] }.

22.3 Functional description

22.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of the FlexIO. The timing of shift, load and store events are controlled by the Timer assigned to the Shifter via the SHIFTCTL[TIMSEL] register. The Shifters are designed to support either DMA, interrupt or polled operation. The following block diagram provides a detailed view of the Shifter microarchitecture.

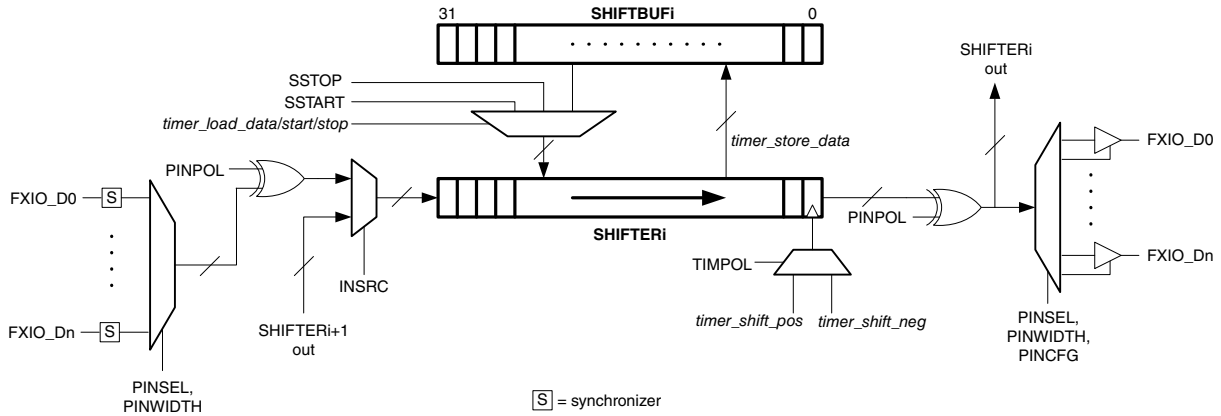


Figure 22-2. Shifter Microarchitecture

22.3.1.1 Transmit Mode

When configured for Transmit mode (SHIFTCTL[SMOD]=Transmit), the shifter will load data from the SHIFTBUF register and shift data out when a load event is signalled by the assigned Timer. An optional start/stop bit can also be automatically loaded before/after SHIFTBUF data by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Note that the shifter will immediately load a stop bit when the Shifter is initially configured for Transmit mode if a stop bit is enabled.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when data has been loaded from the SHIFTBUF register into the Shifter or when the Shifter is initially configured into Transmit mode. The flag will clear when new data has been written into the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). The flag can be cleared by writing it with logic 1.

22.3.1.2 Receive Mode

When configured for Receive mode (SHIFTCTL[SMOD]=Receive), the shifter will shift data in and store data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when data has been stored into the SHIFTBUF register from the Shifter. The flag will clear when the data has been read from the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

22.3.1.3 Match Store Mode

When configured for Match Store mode (SHIFTCTL[SMOD]=Match Store), the shifter will shift data in, check for a match result and store matched data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when a match occurs and matched data has been stored into the SHIFTBUF register from the Shifter. The flag will clear when the matched data has been read from the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

22.3.1.4 Match Continuous Mode

When configured for Match Continuous mode (SHIFTCTL[SMOD]=Match Continuous), the shifter will shift data in and continuously check for a match result whenever a shift event is signalled by the assigned Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when a match occurs. The flag will clear automatically as soon as there is no longer a match between Shifter data and SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when a match occurs. The flag will clear when there is a read from the SHIFTBUF register or it written with logic 1.

22.3.1.5 State Mode

Using State mode enables the user to implement any state machine with up to 8 states, 8 outputs and 3 selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU, which could potentially remain in a STOP/VLPS low power mode.

When configured for State mode (SHIFTCTL[SMOD]=State), the SHIFTBUF register is used to drive the output and compute next state values when the Shifter has been selected by the current state pointer (SHIFTSTATE[STATE]). The following diagram provides a detailed view of Shifter microarchitecture when configured for State mode.

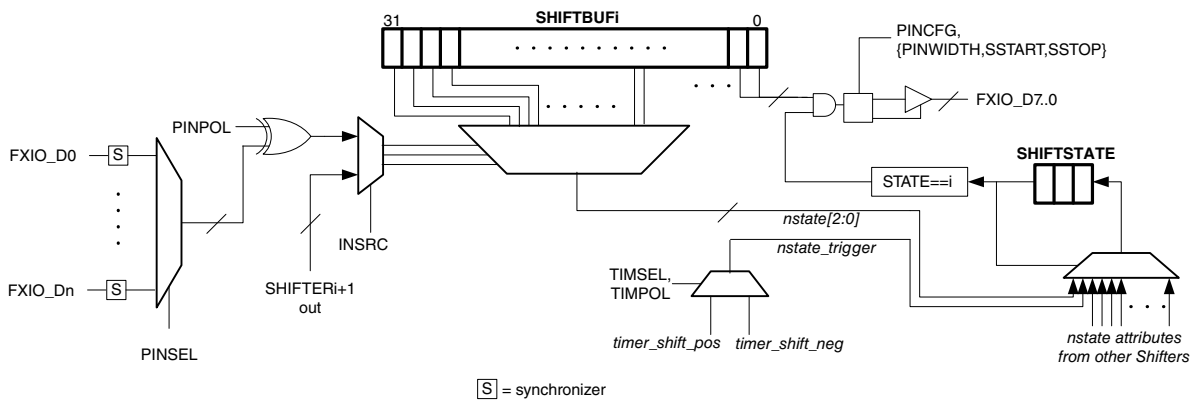


Figure 22-3. State Microarchitecture

When Shifter i has been selected by the current state pointer, output pins FXIO_D[7:0] will be driven by SHIFTBUFi[31:24] using the configuration set by SHIFTCTLi[PINCFG]. When set, SHIFTCFGi{PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} are respectively used to disable the output drive on pins FXIO_D[7:0] for state machine applications which require less than 8 output pins.

The next state value is computed using the 3 input pins selected by SHIFTCTLi[PINSEL] together with SHIFTBUFi[23:0]. Note that each state could potentially use a different set of 3 input pins. The following table details how the next state value is computed when the current state pointer is pointing to Shifter i.

Table 22-2. Next State computation for SHIFTSTATE[STATE]=i

FXIO_D[PINSEL+2]	FXIO_D[PINSEL+1]	FXIO_D[PINSEL]	Next State Value

Table continues on the next page...

Table 22-2. Next State computation for SHIFTSTATE[STATE]=i (continued)

0	0	0	SHIFTBUFi[2:0]
0	0	1	SHIFTBUFi[5:3]
0	1	0	SHIFTBUFi[8:6]
0	1	1	SHIFTBUFi[11:9]
...	...	...	...
1	1	1	SHIFTBUFi[23:21]

Note that other shifters/timers could potentially be configured to drive the input pins of a given state, allowing the user to create complex combinations of shifters/timers as desired e.g. the output of a Shifter configured for logic mode could potentially be used to drive a state machine input.

The next state transition is triggered using the Timer output selected by SHIFTCTLi[TIMSEL] with polarity controlled by SHIFTCTLi[TIMPOL]. Note that each state could potentially use a different Timer to trigger each next state transition, allowing a variety of internal/external trigger sources and clocking configurations to be used (see Timer section for more detail).

The current state pointer defaults to Shifter 0 at reset, however it can be written by the user to select a different Shifter for the initial state. If the current state pointer selects a Shifter which is not configured for State mode, then outputs will not be driven and a next state transition is never triggered.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set whenever the Shifter has been selected by the current state pointer. The flag will clear when the current state pointer is updated to a different Shifter.

22.3.1.6 Logic Mode

Using logic mode enables the user to implement a small amount of programmable digital logic within a FlexIO Shifter. This feature allows board-level glue logic to be integrated on-chip using FlexIO hardware.

When configured for Logic mode (SHIFTCTL[SMOD]=Logic), the SHIFTBUF register is used to implement a 5-input, 32-bit programmable logic look-up table. The following diagram provides a detailed view of Shifter microarchitecture when configured for Logic mode.

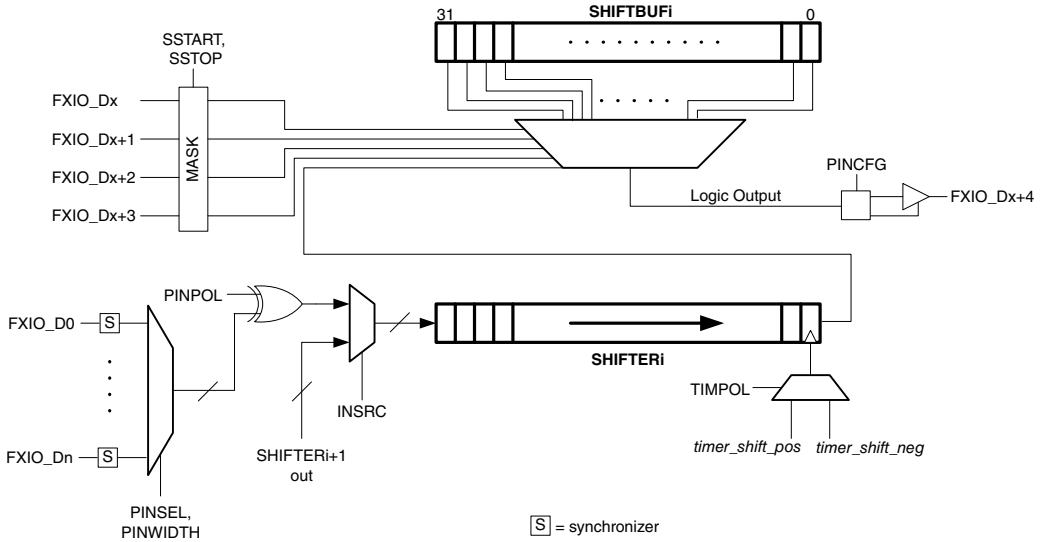


Figure 22-4. Logic Microarchitecture

The look-up table is driven using 4 pin inputs (maskable using SHIFTCFG[SSTOP] and SHFITCFG[SSTART]) plus 1 input from the internal shifter and can be configured to drive an output pin using the SHIFTCTL[PINCFG] field. Pin inputs and outputs are fixed for each logic look-up table and are not selectable. The following table lists the logic output value selected by the look-up table for Shifter 'i'.

Table 22-3. Logic Look-up table for Shifter 'i'

SHIFTERi[0]	FXIO_D[x+3][1]	FXIO_D[x+2]	FXIO_D[x+1]	FXIO_D[x]	Logic Output to FXIO_D[x+4]
0	0	0	0	0	SHIFTBUFi[0]
0	0	0	0	1	SHIFTBUFi[1]
0	0	0	1	0	SHIFTBUFi[2]
0	0	0	1	1	SHIFTBUFi[3]
...	...	...	...	...	...
1	1	1	1	1	SHIFTBUFi[31]

1. for Shifter i=0...3, x=i

   for Shifter i=4...7, x=i+4

To minimize output glitches, SHIFTCFG[SSTOP] and SHIFTCFG[SSTART] can be used to mask unused input pins. When set, {SSTOP[1:0], SSTART[1:0]} will mask FXIO_D[x+3]...FXIO_D[x] inputs respectively, so that any transitions on these pins will not cause the logic output to glitch.

Note that other shifters/timers could potentially be configured to drive the input pins of a given look-up table (without synchronization), allowing the user to concatenate look-up tables or create complex combinations of shifters/timers as desired.

SHIFTCFG[PWIDTH] will control the number of delay stages introduced by the internal shifter input (SHIFTERi[0]). For example, when configured for 1-bit shift (PWIDTH=0), the internal shifter will introduce a 32 Shift clock delay before passing its input (selected by SHIFTCTL[PINSEL]) to the look-up table. When configured for 32-bit shift (PWIDTH=16...31), the internal shifter will introduce a 1 Shift clock delay to its input.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set whenever the output pin allocated to the logic look-up table has a value of 1 (after being synchronized to the FlexIO clock). The flag will clear when the output pin has a value of 0. This also allows the SSF flag to be used as a trigger to a Timer if desired.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when the output pin allocated to the logic look-up table has a value of 1. The flag can be cleared by writing it with logic 1.

22.3.2 Timer operation

The FlexIO 16-bit timers control the loading, shifting and storing of the shift registers, the counters load the contents of the compare register and decrement down to zero on the FlexIO clock. They can perform generic timer functions such as generating a clock or select output or a PWM waveform. Timers can be configured to enable in response to a trigger, pin or shifter condition; decrement always or only on a trigger or pin edge; reset in response to a trigger or pin condition; and disable on a trigger or pin condition or on a timer compare. Timers can optionally include a start condition and/or stop condition.

Each timer operates independently, although a timer can be configured to enable or disable at the same time as the previous timer (eg: timer1 can enable or disable at the same time as timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently and can be configured to be a timer output, shifter status flag, pin input or an external trigger input (refer to the chip configuration section for details on the external trigger connections). The trigger configuration is separate from the pin configuration, which can be configured for input, output data or output enable.

The Timer Configuration Register (TIMCFGn) should be configured before setting the Timer Mode (TIMOD). Once the TIMOD is configured for the desired mode, when the condition configured by timer enable (TIMENA) is detected then the following events occur.

- Timer counter will load the current value of the Compare Register and start decrementing as configured by TIMDEC.

- Timer output will set depending on the TIMOUT configuration.
- Transmit shifters controlled by this timer will either output their start bit value, or load the shift register from the shift buffer and output the first bit, as configured by SSTART.

The Timer will then generate the timer output and timer shift clock depending on the TIMOD and TIMDEC fields. The shifter clock is either equal to the timer output (when TIMDEC=00 or 01) or equal to the decrement clock (when TIMDEC=10 or 11). When TIMDEC is configured to decrement from a pin or trigger, the timer will decrement on both rising and falling edges.

When the Timer is configured to reset as configured in the TIMRST field then the Timer counter will load the current value of the Compare Register again, the timer output may also be affected by the reset as configured in TIMOUT.

If the Timer start bit is enabled, the timer counter will reload with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMOUT=1), a shifter that is configured to shift on falling edge and load on the first shift will not load correctly.

When configured for 8-bit counter mode, whenever the lower 8-bit counter decrements to zero the timer output will toggle, the lower 8-bit counter register will reload from the compare register and the upper 8-bit counter will decrement. For 8-bit PWM mode, the lower 8-bit counter will only decrement when the output is high and the upper 8-bit counter will only decrement when the output is low. The timer output will toggle whenever either lower or upper 8-bit counter decrements to zero.

When the timer decrements to zero, a compare event occurs depending on the timer mode. For 8-bit counter or PWM modes, both halves of the counter must equal zero and the upper half must decrement for the timer compare event to occur, while in 16-bit mode the entire counter must equal zero and decrement. The timer compare event will cause the timer status flag to set, the timer counter to load the contents of the timer compare register, the timer output to toggle, any configured transmit shift registers to load and any configured receive shift registers to store .

When the is Timer is configured to add a stop bit on each compare, the following additional events will occur.

- Transmit shifters controlled by this timer will output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer will store the contents of the shift register in their shift buffer, as configured by SSTOP.
- On the first rising edge of the shifter clock after the compare, the timer counter will reload the current value of the Compare Register.

Transmit shifters must be configured to load on the first shift when the timer is configured to insert a stop bit on each compare.

When the condition configured by timer disable (TIMDIS) is detected, the following events occur.

- Timer counter will reload the current value of the Compare Register and start decrementing as configured by TIMDEC.
- Timer output will clear.
- Transmit shifters controlled by this timer will output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer will store the contents of the shift register in their shift buffer, as configured by SSTOP.

If the timer stop bit is enabled, the timer counter will continue decrementing until the next rising edge of the shift clock is detected, at which point it will finish. A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). Receive shift registers will stop bit enabled will store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge while the timer is in the stop state condition. If there is no configured edge between the timer disable and the next rising edge of the shift clock then the final store and verify do not occur.

22.3.3 Pin operation

The pin configuration for each timer and shifter can be configured to use any FlexIO pin with either polarity. Each timer and shifter can be configured as an input, output data, output enable or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity, since the output enable assertion would cause logic zero to be output on the pin) or to control the enable on the bidirectional output. Any timer or shifter could be configured to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

In addition, shifters can also be configured to use multiple FlexIO pins in parallel using the SHIFTCFG[PWIDTH] field. PWIDTH is used to configure the following settings of a shifter:

1. Number of bits shifted per Shift clock.
2. Number of pins driven by the shifter per Shift clock (only on shifters supporting parallel transmit i.e. SHIFTER0, SHIFTER4).
3. Number of pins sampled by the shifter per Shift clock (only on Shifter supporting parallel receive i.e. SHIFTER3, SHIFTER7).

When configured for parallel shift, either 4, 8, 16 or 32-bits can be shifted on every Shift clock. If an adjacent shifter is selected as the input source (SHIFTCFG[INSRC]=1), the least significant 4, 8, 16 or 32-bits from the adjacent shifter will be sampled on each Shift clock.

For shifters supporting parallel receive (SHIFTER3, SHIFTER7), the shifter can be configured to sample multiple pins (SHIFTCFG[INSRC]=0), with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant bits will be masked with 0 e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[31:24] will sample {0,0,FXIO_D[12:7]} on each Shift clock.

For shifters supporting parallel transmit (SHIFTER0, SHIFTER4), the shifter can be configured to drive multiple pins using SHIFTCTL[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant pins will not be driven e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[5:0] will drive only FXIO_D[12:7] on each Shift clock.

When configuring a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized to the FlexIO clock before the signal is used by a timer or shifter. This introduces a small latency of between 0.5 to 1.5 FlexIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FlexIO clock cycles.

If an input is used by more than one timer or shifter then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

Note that FlexIO pins are also connected internally, configuring a FlexIO shifter or timer to output data on an unused pin will make an internal connection that allows other shifters and timer to use this pin as an input. This allows a shifter output to be used to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized to the FlexIO clock and therefore incurs a 1 cycle latency.

So when using a Pin input as a Timer Trigger, Timer Clock or Shifter Data Input, the following synchronization delays occur:

1. 0.5 – 1.5 FlexIO clock cycles for external pin
2. 1 FlexIO clock cycle for an internally driven pin

For timing considerations such as output valid time and input setup time for specific applications (SPI Master, SPI Slave, I2C Master, I2S Master, I2S Slave) please refer to the FlexIO Application Information Section.

22.4 Application Information

This section provides examples for a variety of FlexIO module applications.

22.4.1 UART Transmit

UART transmit can be supported using one Timer, one Shifter and one Pin (two Pins if supporting CTS). The start and stop bit insertion is handled automatically and multiple transfers can be supported using DMA controller. The timer status flag can be used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention, before transmitting a break or idle character the SSTART and SSTOP fields should be altered to transmit the required state and the data to transmit must equal 0xFF or 0x00. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the number of bits to transmit). Note that when performing byte writes to SHIFTBUFn (or SHIFTBUFBIS for transmitting MSB first), the rest of the register remains unaltered allowing an address mark bit or additional stop bit to remain undisturbed.

FlexIO does not support automatic insertion of parity bits.

Table 22-4. UART Transmit Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open drain by setting PINPOL=0x1 and PINCFG=0x1.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_2222	Configure start bit, stop bit, enable on trigger low and disable on compare. Can support CTS by configuring TIMEN=0x3.
TIMCTLn	0x01C0_0001	Configure dual 8-bit counter using Shifter 0 status flag as inverted internal trigger source. Can support CTS by configuring PINSEL=0x1 (for Pin 1) and PINPOL=0x1.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit

Table 22-4. UART Transmit Configuration

Register	Value	Comments
		transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS[7:0] register instead.

22.4.2 UART Receive

UART receive can be supported using one Timer, one Shifter and one Pin (two Timers and two Pins if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers can be supported using the DMA controller. The timer status flag can be used to indicate when the stop bit of each word is received.

Triple voting of the received data is not supported by FlexIO, data is sampled only once in the middle of each bit. Another timer can be used to implement a glitch filter on the incoming data, another Timer can also be used to detect an idle line of programmable length. Break characters will cause the error flag to set and the shifter buffer register will return 0x00.

FlexIO does not support automatic verification of parity bits.

Table 22-5. UART Receiver Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2422	Configure start bit, stop bit, enable on pin posedge and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x0000_0081	Configure dual 8-bit counter using inverted Pin 0 input.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be

Table 22-5. UART Receiver Configuration

Register	Value	Comments
		read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

The UART Receiver with RTS configuration uses a 2nd Timer to generate the RTS output. The RTS will assert when the start bit is detected and negate when the data is read from the shifter buffer register. No start bit will be detected while the RTS is asserted, the received data is simply ignored.

Table 22-6. UART Receiver with RTS Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2522	Configure start bit, stop bit, enable on pin posedge with trigger low and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x03C0_0081	Configure dual 8-bit counter using inverted Pin 0 input. Trigger is internal using inverted Pin 1 input.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0030_6100	Enable on Timer N enable and disable on trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(n+1)	0x0143_0083	Configure 16-bit counter and output on Pin 1. Trigger is internal using Shifter 0 flag.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

22.4.3 SPI Master

SPI master mode can be supported using two Timers, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of 1 clock cycle between the slave select negating and before the next transfer. Writing to the transmit buffer by either core or DMA is used to initiate each transfer.

Due to synchronization delays, the setup time for the serial input data is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 22-7. SPI Master (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on negedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on posedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0. Set PINPOL to invert the output shift clock.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can

Table continues on the next page...

Table 22-7. SPI Master (CPHA=0) Configuration (continued)

Register	Value	Comments
		support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 22-8. SPI Master (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0021	Start bit loads data on first shift.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on negedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0. Set PINPOL to invert the output shift clock. Set TIMDIS=3 to keep slave select asserted for as long as there is data in the transmit buffer.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.

Table continues on the next page...

Table 22-8. SPI Master (CPHA=1) Configuration (continued)

Register	Value	Comments
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

22.4.4 SPI Slave

SPI slave mode can be supported using one Timer, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The transmit data must be written to the transmit buffer register before the external slave select asserts, otherwise the shifter error flag will be set.

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

Table 22-9. SPI Slave (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on falling edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on rising edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_6000	Configure enable on trigger rising edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can

Table continues on the next page...

Table 22-9. SPI Slave (CPHA=0) Configuration (continued)

Register	Value	Comments
		support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 22-10. SPI Slave (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_6602	Configure start bit, enable on trigger rising edge, disable on trigger falling edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

22.4.5 I2C Master

I2C master mode can be supported using two Timers, two Shifters and two Pins. One timer is used to generate the SCL output and one timer is used to control the shifters. The two shifters are used to transmit and receive for every word, when receiving the transmitter must transmit 0xFF to tristate the output. FlexIO inserts a stop bit after every word to generate/verify the ACK/NACK. FlexIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag will set on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (START to Repeated START/ STOP), so the compare register needs to be programmed with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin equal to output (although this increases both the clock high and clock low periods by at least 1 FlexIO clock cycle each). The second timer uses the SCL input pin to control the transmit/receive shift registers, this enforces an SDA data hold time by an extra 2 FlexIO clock cycles.

Both the transmit and receive shifters need to be serviced for each word in the transfer, the transmit shifter must transmit 0xFF when receiving and the receive shifter returns the data actually present on the SDA pin. The transmit shifter will load 1 additional word on the last falling edge of SCL pin, this word should be 0x00 if generating a STOP condition or 0xFF if generating a repeated START condition. During the last word of a master-receiver transfer, the transmit SSTOP bit should be set by software to generate a NACK.

The receive shift register will assert an error interrupt if a NACK is detected, but software is responsible for generating the STOP or repeated START condition. If a NACK is detected during master-transmit, the interrupt routine should immediately write the transmit shifter register with 0x00 (if generating STOP) or 0xFF (if generating repeated START). Software should then wait for the next rising edge on SCL and then disable both timers. The transmit shifter should then be disabled after waiting the setup delay for a repeated START or STOP condition.

Due to synchronization delays, the data valid time for the transmit output is 2 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The I2C master data valid is delayed 2 cycles because the clock output is passed through a synchronizer before clocking the transmit/receive shifter (to guarantee some SDA hold time). Since the SCL output is synchronous with FlexIO clock, the synchronization delay is 1 cycle and then 1 cycle to generate the output.

Table 22-11. I2C Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTLn	0x0101_0082	Configure transmit using Timer 1 on rising edge of clock with inverted output enable (open drain output) on Pin 0.
SHIFTCFG(n+1)	0x0000_0020	Start bit disabled and stop bit enabled (logic 0) for ACK/NACK detection.
SHIFTCTL(n+1)	0x0180_0001	Configure receive using Timer 1 on falling edge of clock with input data on Pin 0.
TIMCMPn	0x0000_2501	Configure 2 word transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of words x 18) + 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0102_2222	Configure start bit, stop bit, enable on trigger high, disable on compare, reset if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTLn	0x01C1_0101	Configure dual 8-bit counter using Pin 1 output enable (SCL open drain), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_000F	Configure 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_1112	Enable when Timer 0 is enabled, disable when Timer 0 is disabled, enable start bit and stop bit at end of each word, decrement on pin input.
TIMCTL(n+1)	0x01C0_0183	Configure 16-bit counter using inverted Pin 1 input (SCL).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0], use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

22.4.6 I2S Master

I2S master mode can be supported using two Timers, two Shifters and four Pins. One timer is used to generate the bit clock and control the shifters and one timer is used to generate the frame sync. FlexIO waits for the first write to the transmit data buffer before

enabling bit clock and frame sync generation. Data transfers can be supported using the DMA controller and the shifter error flag will set on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FlexIO clock frequency, and the initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure the frame sync is generated one clock cycle before the first output data.

Due to synchronization delays, the setup time for the receiver input is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 22-12. I2S Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Load transmit data on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_0202	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (bit clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(n+1)	0x0000_007F	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (number of bits x baud rate divider) - 1.
TIMCFG(n+1)	0x0000_0100	Enable when Timer 0 is enabled and never disable.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter using inverted Pin 3 output (as frame sync).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.

Table continues on the next page...

Table 22-12. I2S Master Configuration (continued)

Register	Value	Comments
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

22.4.7 I2S Slave

I2S slave mode can be supported using two Timers, two Shifters and four Pins (for single transmit and single receive, other combinations of transmit and receive are possible).

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag will be set.

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The output valid time of I2S slave is max 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization plus 1 cycle to output the data

Table 22-13. I2S Slave Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0103_0002	Configure transmit using Timer 1 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0180_0101	Configure receive using Timer 1 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_007D	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] = (number of bits x 4) - 3.
TIMCFGn	0x0030_2400	Configure enable on pin rising edge (inverted frame sync) and disable on compare, initial clock state is logic 1 and decrement on trigger input (bit clock).
TIMCTLn	0x0440_0383	Configure 16-bit counter using inverted Pin 3 input (frame sync), with Pin 2 input (bit clock) as the trigger.

Table continues on the next page...

Table 22-13. I2S Slave Configuration (continued)

Register	Value	Comments
TIMCMP(n+1)	0x0000_003F	Configure 32-bit transfers. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_3500	Configure enable on pin rising edge with trigger high and disable on compare with trigger low, initial clock state is logic 0 and decrement on pin input.
TIMCTL(n+1)	0x0340_0203	Configure 16-bit counter using Pin 2 input (bit clock), with Timer 0 output as the trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

22.4.8 Camera Interface

Camera Interface can be supported using one Timer, one or more Shifters and multiple Pins. Multiple transfers can be supported using DMA controller.

The example below describes FlexIO configuration for interfacing to an 8-bit CMOS sensor with PCLK, VSYNC, HREF and D[7:0] outputs. The example uses a 128-bit buffer to capture 16-pixels of image data before interrupt or DMA transfer, however a bigger or smaller buffer may be used depending on system DMA performance and FlexIO resource usage by other applications. Note that additional timers may be used to track number of pixels per row and number of rows per frame or HREF/VSYNC may be assigned as GPIO interrupts for software tracking.

Table 22-14. Camera Interface Configuration for 8-bit CMOS sensor

Register	Value	Comments
SHIFTCFGn...n+2[1]	0x0007_0100	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFGn+3	0x0007_0000	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTLn...n+3	0x0080_0001	Configure receive using Timer 0 on negedge of clock.

Table continues on the next page...

Table 22-14. Camera Interface Configuration for 8-bit CMOS sensor (continued)

Register	Value	Comments
TIMCMPn	0x0000_001F	Configure 16-pixel (8 bits/pixel x 16 pixels = 128-bits) transfer. Set TIMCMP[15:0] = (number of pixels x 2) - 1.
TIMCFGn	0x0120_6600	Configure enable on trigger (HREF) rising edge and disable on trigger falling edge, initial Shift clock state is logic 0 and decrement on PCLK input.
TIMCTLn	0x12C0_0803	Configure 16-bit counter using FXIO_D[8] input (PCLK), with FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUFn...n+3	Data to receive	Received data can be read from SHIFTBUFn...n+3, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

1. n=0 or 4

22.4.9 Motorola 68K/Intel 8080 Bus Interface

The Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by Smart/Asynchronous LCD controllers. In conjunction with GPIO, FlexIO is able to drive these interfaces using one Timer and one Shifter, although additional Shifters could be used to support large transfers via the DMA controller.

The configuration below provides an example of how to drive a 16-bit 68K or 8080 bus. For a 8080 bus, two GPIO are used to drive the nCS and RS pins. For a 68K bus, an additional GPIO is required to drive the RDWR pin.

Table 22-15. Motorola 68K/Intel 8080 Write Configuration

Register	Value	Comments
SHIFTCFG0...7	0x000F_0100	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with data output to FXIO_D[15:0].
SHIFTCTL1...7	0x0000_0002	Configure transmit using Timer 0 on posedge of clock.
TIMCMP0	0x0000_0101 (1-beat) 0x0000_1F01 (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.

Table continues on the next page...

Table 22-15. Motorola 68K/Intel 8080 Write Configuration (continued)

Register	Value	Comments
TIMCFG0	0x0000_2200	Configure enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	0x01C3_1001 (Motorola 68K, 1-beat) 0x1DC3_1001 (Motorola 68K, 16-beats) 0x01C3_1081 (Intel 8080, 1-beat) 0x1DC3_1081 (Intel 8080, 16-beats)	Configure dual 8-bit counter using FXIO_D[16] output (EN pin for 68K, nWR pin for 8080), with Shifter 0 (1-beat) or Shifter 7 (16-beats) flag as the inverted trigger.
SHIFTBUF0...7	Data to transmit	Transmit data can be written to SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats) to initiate a transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table 22-16. Motorola 68K/Intel 8080 Read Configuration

Register	Value	Comments
SHIFTCFG0...6	0x000F_0100	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCFG7	0x000F_0000	Configure 16-bit parallel shift in from pin.
SHIFTCTL0...7	0x0080_0001	Configure receive using Timer 0 on negedge of clock with data input from FXIO_D[15:0].
TIMCMP0	0x0000_0101 (1-beat) 0x0000_1F01 (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_2220	Configure stop_bit, enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	0x1DC3_1001 (Motorola 68K, 1 beat) 0x01C3_1001 (Motorola 68K, 16 beats) 0x1DC3_1181 (Intel 8080, 1 beat) 0x01C3_1181 (Intel 8080, 16 beats)	Configure dual 8-bit counter using either FXIO_D[16] output (EN pin for 68K) or FXIO_D[17] output (nRD pin for 8080), with Shifter 7 flag (1-beat) or Shifter 0 flag (16-beats) as the inverted trigger.
SHIFTBUF0...7	Data received	Received data can be read from SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats), use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

In general, any operation to a 68K/8080 bus slave will begin with a register write cycle followed by one or more data read or write cycles. To accomplish this, the following program flow should be used:

1. Configure FlexIO with 1-beat write configuration
2. Configure GPIO to assert nCS, RS pins (and deassert RDWR pin for 68K)
3. Write register index data to SHIFTBUF0[15:0]
4. Configure GPIO to deassert RS pin (and assert RDWR pin for 68K data read)
5. Configure FlexIO with desired read or write configuration (e.g. 1 or 16-beats)
6. Use the Shifter Status Flag to trigger interrupt or DMA driven data transfers to/from SHIFTBUF registers
7. Configure GPIO to deassert nCS pin

22.4.10 Low Power State Machine

The configuration below details a hypothetical state machine example to illustrate the flexibility allowed when using Shifter state mode.

In this example, FlexIO waits for the FXIO_D[2] pin to assert and then drives a complementary square wave output at a frequency of FLEXIO_CLK/131072 on the FXIO_D[1:0] pins while the comparator output is asserted (assumes comparator is connected to external trigger 15, see Chip Configuration chapter for actual FlexIO trigger mappings). Throughout this operation, the CPU can be kept in a STOP/VLPS mode, by clearing the CTRL[DOZEN] bit and ensuring the FLEXIO_CLK is enabled. The state diagram below shows the states and transitions implemented by this example.

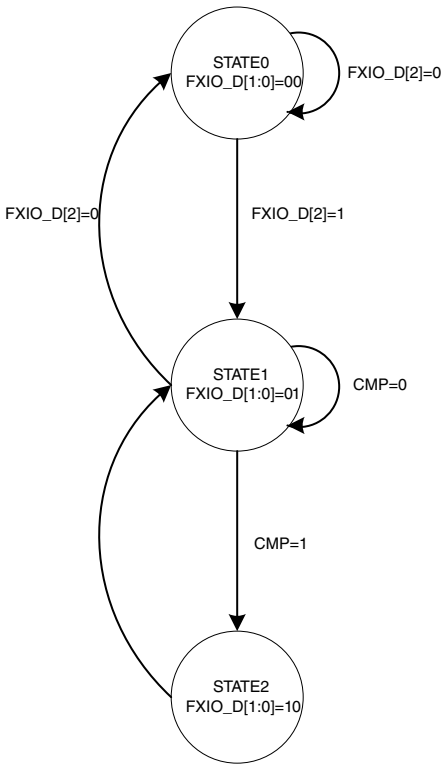


Figure 22-5. State Diagram

Table 22-17. State Machine Configuration

Register	Value	Comments
SHIFTCFG0...2	0x0000_0003	Enable FXIO_D[1:0] as outputs.
SHIFTCTL0	0x0080_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF0	0x0020_8208	State0: Drive FXIO_D[1:0]=00, transition to State0 if FXIO_D[2]=0, State1 if FXIO_D[2]=1.
SHIFTCTL1	0x0000_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output high to trigger next state.
SHIFTBUF1	0x0140_8408	State1: Drive FXIO_D[1:0]=01, transition to State0 if FXIO_D[2]=0, State1 if CMP=0, State2 if CMP=1 (FXIO_D[3]=1)
SHIFTCTL2	0x0080_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF2	0x0224_9249	State2: Drive FXIO_D[1:0]=10, transition to State1 when Timer0 output low

Table continues on the next page...

Table 22-17. State Machine Configuration (continued)

Register	Value	Comments
TIMCMP0	0x0000_FFFF	Configure baud rate of divide by 131072 of the FlexIO clock. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_0000	Configure timer always enabled.
TIMCTL0	0x0000_0003	Configure single 16-bit counter.
TIMCFG1	0x0010_7600	Configure timer enabled on trigger rising edge, disabled on trigger falling edge, decrement on trigger edge.
TIMCTL1	0x0F03_0303	Configure timer output enabled on FXIO_D[3], external trigger 15 (comparator output) selected.

Chapter 23
Flash Memory Controller (FMC)

23.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The Flash Memory Controller (FMC) is a memory acceleration unit that provides:

- an interface between the device and the nonvolatile memory.
- buffers that can accelerate flash memory transfers.

23.1.1 Overview

The Flash Memory Controller manages the interface between the device and the flash memory. The FMC receives status information detailing the configuration of the memory and uses this information to ensure a proper interface. The following table shows the supported read/write operations.

Flash memory type	Read	Write
Program flash memory	8-bit, 16-bit, and 32-bit reads	—[1]

1. A write operation to program flash memory results in a bus error.

The FMC has a 32-bit interface to the device. The flash memory has a 2-bank structure, with independent lower and upper banks (both banks are 64-bits wide). The FMC has 2 separate 64-bit interfaces to the flash memory, with one 64-bit interface to each bank.

In addition, the FMC provides 2 separate mechanisms for accelerating the interface between the device and the flash memory. A 64-bit speculation buffer can prefetch the next 64-bit flash memory location, and a shared 4-way, 4-set cache can store previously accessed flash memory data for quick access times.

23.1.2 Features

Interface features between the device and the flash memory include:

- 8-bit, 16-bit, and 32-bit read operations to program flash memory.
- Read accesses to consecutive 32-bit spaces in memory *that do not hit speculation buffer or cache* return the 2nd read data with no wait states.
- Crossbar master access protection for setting no access, read-only access, write-only access, or read/write access for each crossbar master.
- 64-bit prefetch speculation buffer with controls for instruction/data access per master
- Invalidation control for the speculation buffer
- 128 bytes = 4-way by 4-set by 64 bits per entry cache, with controls for caching instructions and/or data
- Enable and invalidation controls for cache
- Read accesses that hit a valid speculation or cache entry return the read data with no wait states

23.2 Modes of operation

The FMC only operates when a bus master accesses the flash memory.

In terms of device power modes, the FMC only operates in run and wait modes, including VLPR and VLPW modes.

For any device power mode where the flash memory cannot be accessed, the FMC is disabled.

23.3 External signal description

The FMC has no external signals.

23.4 Memory map and register descriptions

In this device, the PFC = Platform Flash Controller does not have any program model; therefore, there are no register definitions in the PFC chapter. All PFC operating controls are in the Platform Control Register (MCMx_PLACR) in the MCM module.

23.5 Flash Access Control (FAC) Function

The Flash Access Control (FAC) is a configurable memory protection scheme optimized to allow end users to use software libraries while offering programmable restrictions to these libraries. The flash memory is divided into *equal size segments* that provide protection to proprietary software libraries. The protection of these segments is controlled: the FAC provides a cycle-by-cycle evaluation of the access rights for each transaction routed to the on-chip flash memory. Two levels of vendors can add their proprietary software to a device; FAC protection of segments for each level are defined once, using the PGMONCE command.

Flash access control aligns to the 3 privilege levels supported by ARM Cortex-M family products:

- Most secure state is supervisor/privileged secure: allows execute-only and provides supervisor-only access control.
- Mid-level state is execute-only.
- Unsecure state is where no access control states are set.

Features:

- Lightweight access control logic for on-chip flash memory
- Flash address space divided into (32 or 64) equal-sized segments (segment size is defined as flash_size [bytes]/(32 or 64))
- Separate control bits for supervisor-only access and execute-only access per segment
- Access control evaluated on each bus cycle routed to the flash
- Access violation errors terminate the bus cycle and return zeroes for read data
- Programming model allows 2 levels of protected segments

23.5.1 Memory map and register definitions

The FAC registers are documented in the FTFA chapter.

23.5.2 FAC functional description

The access control functionality is implemented in 2 separate blocks within the SoC. The Flash Management Unit (FMU) includes non-volatile configuration information that is retrieved during reset and and sent to the platform to control access to the flash array during normal operation.

There are (4) 64-bit NVM storage locations to support access control features. These NVM locations are summarized in the table below.

Table 23-1. NVM Locations

NVM location	Description	
NVSACC1, NVSACC2	Two locations are ANDed together and loaded during reset into the *x*_SACC register to provide access configuration.	Segment-wise control for supervisor-only access vs. supervisor and user access
NVXACC1, NVXACC2	Two locations are ANDed together and loaded during reset into the *x*_XACC register to provide access configuration.	Segment-wise control for execute-only vs. data and execute

Each of these NVM locations is programmable through a Program Once flash command and can be programmed one time. These NVM locations are unaffected by Erase All Blocks flash command and debug interface initiated mass erase operations. Since the 2 NVXACCx fields are ANDed, the access protection can only be increased. A segment's access controls can be changed from data read and execute (XA*n* =1) to execute-only (XA*n* =0), or from supervisor and user mode (SAn = 1) to supervisor-only mode (SAn = 0).

The flash is released from reset early while the core continues to be held in reset. The FMU captures the NVM access control information in internal registers. The FMU ANDs the multiple execute-only fields to create a single execute-only field. This execute-only field driven to the platform is static prior to the core being released from reset. The supervisor-only field is handled in the same manner.

The FMU includes the FAC registers that provide control access to the flash address space. During the address phase of every attempted flash transfer, the supervisor access (SA*n*) and execute access (XA*n*) bits are examined to either allow or deny access. If access is denied, then the access is aborted and terminates with a bus error; the read data is also zeroed.

The next table shows segment assignments relative to the flash location.

Table 23-2. Flash Protection Ranges

SA*n* and XA*n* Bit	Protected Segment Address Range	Segment Size (Fraction of total Flash)
64 Segment Encodings		
0	0x0_0000_0000 – (Flash_size/64-1)	1/64
1	(Flash_size/64) – 2*(Flash_size/64-1)	1/64
.....		
63	63*(Flash_size/64) – 62*(Flash_size/64-1)	1/64
32 Segment Encodings		

Table continues on the next page...

Table 23-2. Flash Protection Ranges (continued)

SA*n* and XA*n* Bit	Protected Segment Address Range	Segment Size (Fraction of total Flash)
0	0x0_0000_0000 – (Flash_size/32-1)	1/32
1	(Flash_size/32) – 2*(Flash_size/32-1)	1/32
.....		
31	31*(Flash_size/32) – 30*(Flash_size/32-1)	1/32

Individual segments within the flash memory can be independently protected from user access and data access. Protection is controlled by the individual bits within the *x*_SACC and *x*_XACC registers, as shown in the next figure.

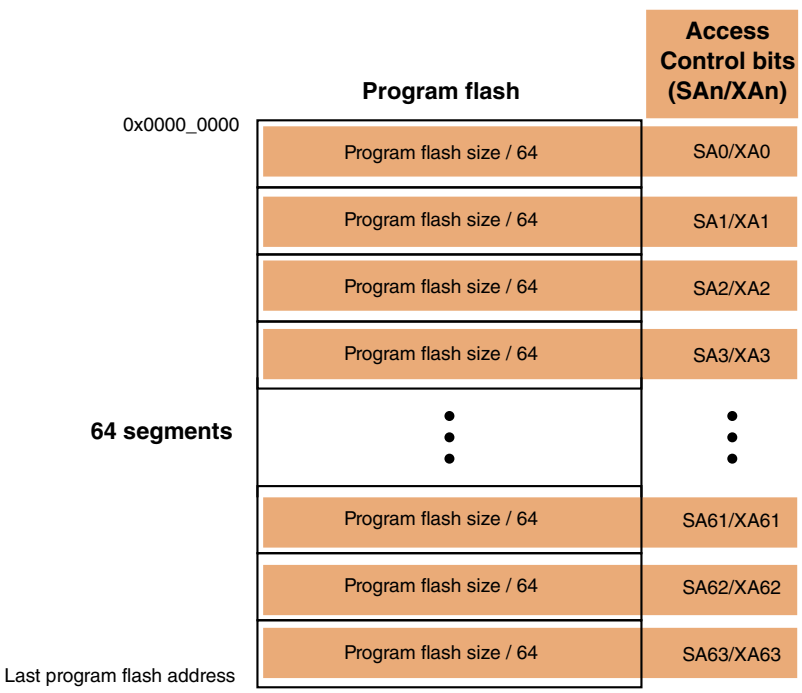


Figure 23-1. Program flash protection (64 segments)

23.5.2.1 Interface Signals

Table 23-3. Interface Signals

Signal	Width	From	To	Description
xacc	64 or 32	FMU	Platform	Direct xacc (execute-only access control) register
sacc	64 or 32	FMU	Platform	Direct sacc (supervisor access control) register
numsg	8	FMU	Platform	NUMSG bit field - Binary encoded number of segments 0x40 for 64 segments

Table continues on the next page...

Table 23-3. Interface Signals (continued)

Signal	Width	From	To	Description
				0x20 for 32 segments
fac_enable	1	SIM	FMU	SIM Option bit - derived from an IFR bit and captured in SIM_SOPTx. A way to disable the flash access control for phantom devices without this feature. fac_enable==1 - Access Control feature is enabled fac_enable==0 - Access Control feature is disabled • During the reset sequence, XACC registers are written to all "1"s. • During the reset sequence, SACC registers are written to all 1"s. • Implied protection based on XACC registers is turned off.

23.5.2.2 Flash Command Impact

Program Longword/Phrase/ Section	If the targeted flash location is in an execute-only protected segment, then these program commands are not allowed unless a Read 1s All Blocks command is executed and returns with a pass code (which means the part has been fully erased). After the Read 1s All Blocks command is executed with a pass code returned, then the protected segment is open to program commands. To close off programmability to execute-only spaces once again, the device must be reset or a Read 1s All Blocks command is executed with a fail result. Attempts to program in a protected segment *when not open to program commands* causes a Protection Violation flag.
PGMCHK	The FMU will not execute the PGMCHK on a segment that has been configured as execute-only. The Flash Protection Violation flag is set if an attempt is made to execute PGMCHK command on an execute-only address.
Erase Flash Sector	If the targeted flash sector is in an execute-only protected segment, then the Erase Flash Sector command is not allowed, and sets the Protection Violation flag. The only means of erasing protected space is by an Erase All operation.
ERSALL	The Erase All Blocks command is not affected by Access Control. An Erase All Blocks command will erase any libraries that have been programmed in any execute-only segment. The programmed execute-only assignment is not erased as part of the Erase All Blocks command, and access control regions remain as previously programmed. **NOTE:** The ERSALL command may be used for field upgrades. Access control states remain programmed. Software must plan accordingly, possibly making extra space available for future use.

23.5.2.3 Core Platform Impact

Platform core caches (Flash and LMEM caches)	If any segment is marked as *execute-only*, then the caches are hidden from the user. The tag is read-only and cannot be written, and the data caches cannot be read or written. Writes to the tag and data arrays are ignored, and reads of the data array return 0's. This will impact debug breakpoints. See the debug section for details.

Table continues on the next page...

Debug	The debugger is a non-processor bus master and cannot step, trace or break in execute-only regions. In supervisor-only mode, the debugger is restricted from changing modes. Debug accesses to any segment of flash space marked as execute-only also terminate with a bus error.
PC-relative addressing	The PC-relative addressing issue is still being understood and this section will be updated in the future. PC relative re-entry to execute-only segments will be allowed..... Restrictions will be placed on software for PC relative addressing, because hardware cannot determine if PC relative data references are crossing segment boundaries. • If ifetch is executing in a protected segment, then data references will be allowed. • Hardware cannot track speculative ifetches across boundaries.
Interrupts	If function calls are used to move into an execute-only segment, then this can be tracked by hardware when typical software controls are used (i.e., saving registers and states before executing new code).
Reset Vector	In the ARM core, the reset vector fetch is supervisor data, which poses issues if the reset vector is located in a segment marked execute-only. Additional logic has been implemented to allow supervisor data fetches to execute-only spaces, after reset until the first valid instruction fetch. After the first valid instruction fetch, the FAC logic follows normal checks.

23.5.2.4 Software Impact

As implementation, verification and validation continue, there will be more details on software impact that will need to be communicated to tool and library vendors. The hardware cannot see all states of the ARM core and cannot track the software flow requiring software restrictions to work with the hardware for a robust solution.

Any segment marked as execute-only can see all code in the system. This means one execute-only segment can read the execute-only code in another segment. Therefore, if Freescale is sending pre-loaded code to another vendor, then that vendor will have access to Freescale code. Possibly use NDAs and legal agreements to deal with this issue.

For single pre-loads (for example, if Freescale is pre-loading for a GP market or if a vendor with a blank part is pre-loading proprietary code), then both levels of access control must be programmed, to protect the pre-loaded code.

If any portion of a protected segment is not used by pre-loaded code, then it (the portion of a protected segment that is not used by pre-loaded code) should be programmed with NOPs, to prevent additional code from being programmed in that segment by hackers.

23.5.2.5 Access Check Evaluation

The flash controller FAC provides a cycle-by-cycle evaluation of the access rights for each data transaction routed to the on-chip flash memory.

The entire flash storage capacity is partitioned into equal sized segments. Two registers include a supervisor-only access control indicator and a execute-only access control indicator for each segment.

The FAC logic performs the required access control evaluation using the reference address and a 2-bit attribute (or "protection" field) as inputs from the bus cycle plus the contents of the programming model registers.

The following code example illustrates C code for FAC evaluation:

```
unsigned long long sacc;  // supervisor-only map
unsigned long long xacc;  // execute-only map
unsigned int seg_size;    // 8-bit segment size
unsigned int fac_error;

fac_evaluation (addr, prot)
  unsigned int addr;  // access address
  unsigned int hprot; // encoded 2-bit "protection" field {supv, data}
{
  unsigned int sacc_flag; // sacc flag for this segment
  unsigned int xacc_flag; // xacc flag for this segment
  unsigned int i;         // segment index

  i         = (addr >> (8 + seg_size & 0x0f)) & 0x3f; // form 6-bit segment index
  sacc_flag = (sacc >> i) & 1;              // extract sacc bit for this segment
  xacc_flag = (xacc >> i) & 1;              // extract xacc bit for this segment

  // create a 4-tuple concatenating the 2-bit protection field + {sacc, xacc} flags

  switch ((hprot & 3) << 2 | (sacc_flag << 1) | xacc_flag) {
  // all these combinations are allowed accesses
    case 0x2: // {user, ifetch} && {supv+user, ifetch-only}
    case 0x3: // {user, ifetch} && {supv+user, ifetch+data}
    case 0x7: // {user, data}   && {supv+user, ifetch+data}
    case 0x8: // {supv, ifetch} && {supv-only, ifetch-only}
    case 0x9: // {supv, ifetch} && {supv-only, ifetch+data}
    case 0xa: // {supv, ifetch} && {supv+user, ifetch-only}
    case 0xb: // {supv, ifetch} && {supv+user, ifetch+data}
    case 0xd: // {supv, data}   && {supv-only, ifetch+data}
    case 0xf: // {supv, data}   && {supv+user, ifetch+data}
      fac_error = 00;
      break;

  // all these combinations are unallowed, that is, errored accesses
    case 0x0: // {user, ifetch} && {supv-only, ifetch-only}
    case 0x1: // {user, ifetch} && {supv-only, ifetch+data}
    case 0x4: // {user, data}   && {supv-only, ifetch-only}
    case 0x5: // {user, data}   && {supv-only, ifetch+data}
    case 0x6: // {user, data}   && {supv+user, ifetch-only}
    case 0xc: // {supv, data}   && {supv-only, ifetch-only}
    case 0xe: // {supv, data}   && {supv+user, ifetch-only}
      fac_error = 1;
      break;

   } // switch()

} // fac_evaluation()
```

23.5.2.6 FAC application tips

In one use case, the NVSACC1 and NVXACC1 locations are programmed by Freescale and they protect Freescale libraries that have been programmed into associated flash segments in a device. Later, the NVSACC2 and NVXACC2 NVM locations can optionally be programmed by a third-party vendor who wants to program their proprietary software and to extend the protection of protected flash segments to include their software libraries before supplying it all to their customers.

Their customer would then develop their own code to use the available libraries, and program their code into the remaining available on-chip flash. The device continues to support the end user with standard security features that further limit external access to flash resources.

23.6 Initialization and application information

The FMC does not require user initialization. Flash acceleration features are enabled by default.

The FMC has no visibility into flash memory erase and program cycles, because the Flash Memory module manages them directly. As a result, if an application is executing flash memory commands, then the FMC's cache might need to be disabled and/or flushed, to prevent the possibility of returning stale data. To invalidate the cache in this manner, use the Flash Cache Invalidate control register in the MCM.

Chapter 24
Flash Memory Module (FTFA)

24.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The flash memory module includes the following accessible memory regions:

- Program flash memory for vector space and code store

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The flash memory module includes a memory controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased with security disabled. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that each flash block or sector be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

24.1.1 Features

The flash memory module includes the following features.

NOTE

See the device's Chip Configuration details for the exact amount of flash memory available on your device.

24.1.1.1 Program Flash Memory Features

- Sector size of 2 KB
- Program flash protection scheme prevents accidental program or erase of stored data
- Program flash access control scheme prevents unauthorized access to selected code segments
- Automated, built-in, program and erase algorithms with verify
- Read access to one program flash block is possible while programming or erasing data in the other program flash block

24.1.1.2 Other Flash Memory Module Features

- Internal high-voltage supply generator for flash memory program and erase operations
- Optional interrupt generation upon flash command completion
- Supports MCU security mechanisms which prevent unauthorized access to the flash memory contents

24.1.2 Block Diagram

The block diagram of the flash memory module is shown in the following figure.

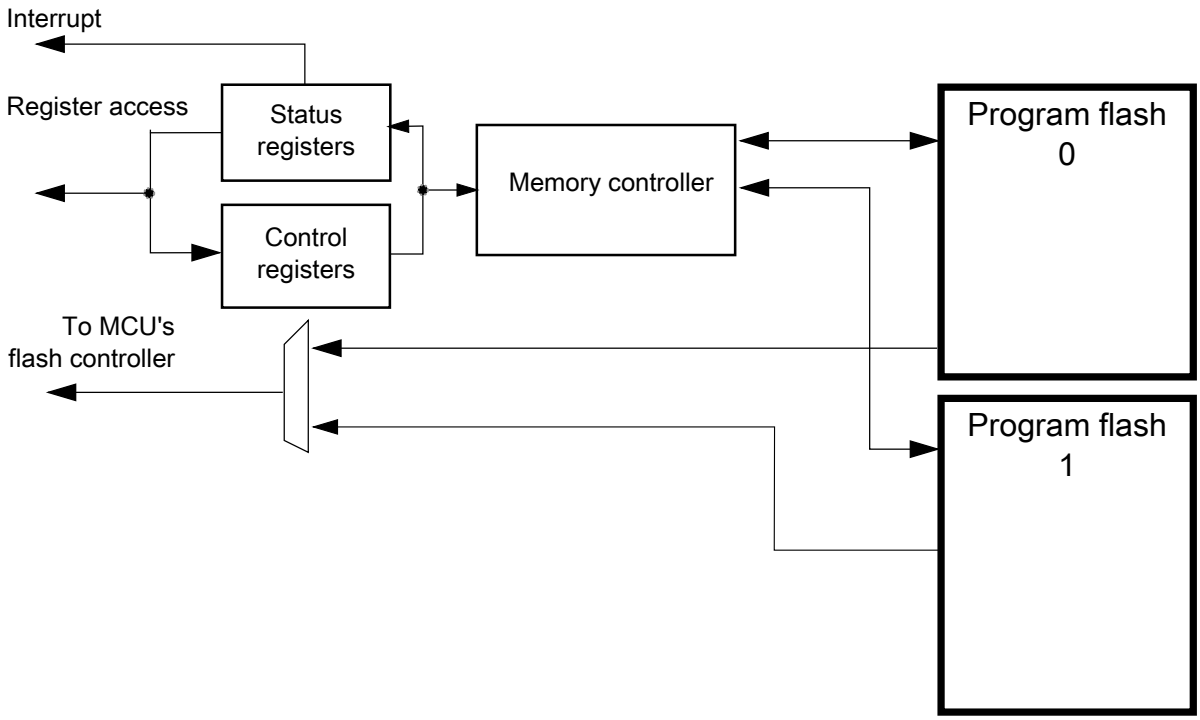


Figure 24-1. Flash Block Diagram

24.1.3 Glossary

Command write sequence — A series of MCU writes to the flash FCCOB register group that initiates and controls the execution of flash algorithms that are built into the flash memory module.

Endurance — The number of times that a flash memory location can be erased and reprogrammed.

FCCOB (Flash Common Command Object) — A group of flash registers that are used to pass command, address, data, and any associated parameters to the memory controller in the flash memory module.

Flash block — A macro within the flash memory module which provides the nonvolatile memory storage.

Flash Memory Module — All flash blocks plus a flash management unit providing high-level control and an interface to MCU buses.

HSRUN — An MCU power mode enabling high-speed access to the memory resources in the flash module. The user has no access to the flash command set when the MCU is in HSRUN mode.

IFR — Nonvolatile information register found in each flash block, separate from the main memory array.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

NVM — Nonvolatile memory. A memory technology that maintains stored data during power-off. The flash array is an NVM using NOR-type flash memory technology.

NVM Normal Mode — An NVM mode that provides basic user access to flash memory module resources. The CPU or other bus masters initiate flash program and erase operations (or other flash commands) using writes to the FCCOB register group in the flash memory module.

NVM Special Mode — An NVM mode enabling external, off-chip access to the memory resources in the flash memory module. A reduced flash command set is available when the MCU is secured. See the Chip Configuration details for information on when this mode is used.

Phrase — 64 bits of data with an aligned phrase having byte-address[2:0] = 000.

Program flash — The program flash memory provides nonvolatile storage for vectors and code store.

Program flash Sector — The smallest portion of the program flash memory (consecutive addresses) that can be erased.

Retention — The length of time that data can be kept in the NVM without experiencing errors upon readout. Since erased (1) states are subject to degradation just like programmed (0) states, the data retention limit may be reached from the last erase operation (not from the programming time).

RWW— Read-While-Write. The ability to simultaneously read from one memory resource while commanded operations are active in another memory resource.

Secure — An MCU state conveyed to the flash memory module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

24.2 External Signal Description

The flash memory module contains no signals that connect off-chip.

24.3 Memory Map and Registers

This section describes the memory map and registers for the flash memory module.

Data read from unimplemented memory space in the flash memory module is undefined. Writes to unimplemented or reserved memory space (registers) in the flash memory module are ignored.

24.3.1 Flash Configuration Field Description

The program flash memory contains a 16-byte flash configuration field that stores default protection settings (loaded on reset) and security information that allows the MCU to restrict access to the flash memory module.

Flash Configuration Field Offset Address	Size (Bytes)	Field Description
0x0_0400–0x0_0407	8	Backdoor Comparison Key. Refer to Verify Backdoor Access Key Command and Unsecuring the Chip Using Backdoor Key Access.
0x0_0408–0x0_040B	4	Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3).
0x0_040F	1	Reserved
0x0_040E	1	Reserved
0x0_040D	1	Flash nonvolatile option byte. Refer to the description of the Flash Option Register (FOPT).
0x0_040C	1	Flash security byte. Refer to the description of the Flash Security Register (FSEC).

24.3.2 Program Flash IFR Map

The program flash IFR is nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see the Read Once, Program Once, and Read Resource commands in Read Once Command, Program Once Command and Read Resource Command).

The contents of the program flash IFR are summarized in the table found here and further described in the subsequent paragraphs.

The program flash IFR is located within the program flash 0 memory block .

Address Range	Size (Bytes)	Field Description
0x00 – 0x9F	160	Reserved
0xA0 – 0xA3	4	Program Once XACCH-1 Field (index = 0x10)
0xA4 – 0xA7	4	Program Once XACCL-1 Field (index = 0x10)
0xA8 – 0xAB	4	Program Once XACCH-2 Field (index = 0x11)
0xAC – 0xAF	4	Program Once XACCL-2 Field (index = 0x11)
0xB0 – 0xB3	4	Program Once SACCH-1 Field (index = 0x12)
0xB4 – 0xB7	4	Program Once SACCL-1 Field (index = 0x12)
0xB8 – 0xBB	4	Program Once SACCH-2 Field (index = 0x13)
0xBC – 0xBF	4	Program Once SACCL-2 Field (index = 0x13)
0xC0 – 0xFF	64	Program Once ID Field (index = 0x00 - 0x0F)

24.3.2.1 Program Once Field

The Program Once Field in the program flash IFR provides 96 bytes of user data storage separate from the program flash main array. The user can program the Program Once Field one time only as there is no program flash IFR erase mechanism available to the user. The Program Once Field can be read any number of times. This section of the program flash IFR is accessed in 4-byte or 8-Byte records using the Read Once and Program Once commands (see Read Once Command and Program Once Command).

24.3.3 Program Flash Erasable IFR Map

The program flash erasable IFR is nonvolatile information memory that can be erased but has limited program capabilities and limited read access. The contents of the program flash erasable IFR fields are summarized in the table found here.

Address Range	Size (Bytes)	Field Description
0xC0 – 0xC3	4	Erasable Program Once Field (index = 0x30)
0xC4 – 0xC7	4	Erasable Program Once Field (index = 0x31)
0xC8 – 0xCB	4	Erasable Program Once Field (index = 0x32)
0xCC – 0xCF	4	Erasable Program Once Field (index = 0x33)

24.3.3.1 Erasable Program Once Field

The Program Once Field in the program flash erasable IFR with indexes 0x30 - 0x33 provides 16 bytes of user data storage that can be programmed using the Program Once command and read using the Read Once command. The program flash erasable IFR is erased using the Erase All Blocks command, Erase All Blocks Unsecure command, and using the external erase all feature (mass erase).

24.3.4 Register Descriptions

The flash memory module contains a set of memory-mapped control and status registers.

NOTE

While a command is running (FSTAT[CCIF]=0), register writes are not accepted to any register except FCNFG and FSTAT. The no-write rule is relaxed during the start-up reset sequence, prior to the initial rise of CCIF. During this initialization period the user may write any register. All register writes are also disabled (except for registers FCNFG and FSTAT) whenever an erase suspend request is active (FCNFG[ERSSUSP]=1).

FTFA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0000	Flash Status Register (FTFA_FSTAT)	8	R/W	00h	24.3.4.1/ 605
4002_0001	Flash Configuration Register (FTFA_FCNFG)	8	R/W	00h	24.3.4.2/ 607
4002_0002	Flash Security Register (FTFA_FSEC)	8	R	Undefined	24.3.4.3/ 608
4002_0003	Flash Option Register (FTFA_FOPT)	8	R	Undefined	24.3.4.4/ 609

Table continues on the next page...

FTFA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0004	Flash Common Command Object Registers (FTFA_FCCOB3)	8	R/W	00h	24.3.4.5/ 610
4002_0005	Flash Common Command Object Registers (FTFA_FCCOB2)	8	R/W	00h	24.3.4.5/ 610
4002_0006	Flash Common Command Object Registers (FTFA_FCCOB1)	8	R/W	00h	24.3.4.5/ 610
4002_0007	Flash Common Command Object Registers (FTFA_FCCOB0)	8	R/W	00h	24.3.4.5/ 610
4002_0008	Flash Common Command Object Registers (FTFA_FCCOB7)	8	R/W	00h	24.3.4.5/ 610
4002_0009	Flash Common Command Object Registers (FTFA_FCCOB6)	8	R/W	00h	24.3.4.5/ 610
4002_000A	Flash Common Command Object Registers (FTFA_FCCOB5)	8	R/W	00h	24.3.4.5/ 610
4002_000B	Flash Common Command Object Registers (FTFA_FCCOB4)	8	R/W	00h	24.3.4.5/ 610
4002_000C	Flash Common Command Object Registers (FTFA_FCCOBB)	8	R/W	00h	24.3.4.5/ 610
4002_000D	Flash Common Command Object Registers (FTFA_FCCOBA)	8	R/W	00h	24.3.4.5/ 610
4002_000E	Flash Common Command Object Registers (FTFA_FCCOB9)	8	R/W	00h	24.3.4.5/ 610
4002_000F	Flash Common Command Object Registers (FTFA_FCCOB8)	8	R/W	00h	24.3.4.5/ 610
4002_0010	Program Flash Protection Registers (FTFA_FPROT3)	8	R/W	Undefined	24.3.4.6/ 611
4002_0011	Program Flash Protection Registers (FTFA_FPROT2)	8	R/W	Undefined	24.3.4.6/ 611
4002_0012	Program Flash Protection Registers (FTFA_FPROT1)	8	R/W	Undefined	24.3.4.6/ 611
4002_0013	Program Flash Protection Registers (FTFA_FPROT0)	8	R/W	Undefined	24.3.4.6/ 611
4002_0018	Execute-only Access Registers (FTFA_XACCH3)	8	R	Undefined	24.3.4.7/ 613
4002_0019	Execute-only Access Registers (FTFA_XACCH2)	8	R	Undefined	24.3.4.7/ 613
4002_001A	Execute-only Access Registers (FTFA_XACCH1)	8	R	Undefined	24.3.4.7/ 613
4002_001B	Execute-only Access Registers (FTFA_XACCH0)	8	R	Undefined	24.3.4.7/ 613
4002_001C	Execute-only Access Registers (FTFA_XACCL3)	8	R	Undefined	24.3.4.7/ 613
4002_001D	Execute-only Access Registers (FTFA_XACCL2)	8	R	Undefined	24.3.4.7/ 613

Table continues on the next page...

FTFA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_001E	Execute-only Access Registers (FTFA_XACCL1)	8	R	Undefined	24.3.4.7/ 613
4002_001F	Execute-only Access Registers (FTFA_XACCL0)	8	R	Undefined	24.3.4.7/ 613
4002_0020	Supervisor-only Access Registers (FTFA_SACCH3)	8	R	Undefined	24.3.4.8/ 614
4002_0021	Supervisor-only Access Registers (FTFA_SACCH2)	8	R	Undefined	24.3.4.8/ 614
4002_0022	Supervisor-only Access Registers (FTFA_SACCH1)	8	R	Undefined	24.3.4.8/ 614
4002_0023	Supervisor-only Access Registers (FTFA_SACCH0)	8	R	Undefined	24.3.4.8/ 614
4002_0024	Supervisor-only Access Registers (FTFA_SACCL3)	8	R	Undefined	24.3.4.8/ 614
4002_0025	Supervisor-only Access Registers (FTFA_SACCL2)	8	R	Undefined	24.3.4.8/ 614
4002_0026	Supervisor-only Access Registers (FTFA_SACCL1)	8	R	Undefined	24.3.4.8/ 614
4002_0027	Supervisor-only Access Registers (FTFA_SACCL0)	8	R	Undefined	24.3.4.8/ 614
4002_0028	Flash Access Segment Size Register (FTFA_FACSS)	8	R	Undefined	24.3.4.9/ 615
4002_002B	Flash Access Segment Number Register (FTFA_FACSN)	8	R	Undefined	24.3.4.10/ 616

24.3.4.1 Flash Status Register (FTFA_FSTAT)

The FSTAT register reports the operational status of the flash memory module.

The CCIF, RDCOLERR, ACCERR, and FPVIOL bits are readable and writable. The MGSTAT0 bit is read only. The unassigned bits read 0 and are not writable.

NOTE

When set, the Access Error (ACCERR) and Flash Protection Violation (FPVIOL) bits in this register prevent the launch of any more commands until the flag is cleared (by writing a one to it).

Address: 4002_0000h base + 0h offset = 4002_0000h

Bit	7	6	5	4	3	2	1	0
Read	CCIF	RDCOLERR	ACCERR	FPVIOL	0			MGSTAT0
Write	w1c	w1c	w1c	w1c				
Reset	0	0	0	0	0	0	0	0

FTFA_FSTAT field descriptions

Field	Description
7 CCIF	Command Complete Interrupt Flag Indicates that a flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command, and CCIF stays low until command completion or command violation. CCIF is reset to 0 but is set to 1 by the memory controller at the end of the reset initialization sequence. Depending on how quickly the read occurs after reset release, the user may or may not see the 0 hardware reset value. 0 Flash command in progress 1 Flash command has completed
6 RDCOLERR	Flash Read Collision Error Flag Indicates that the MCU attempted a read from a flash memory resource that was being manipulated by a flash command (CCIF=0). Any simultaneous access is detected as a collision error by the block arbitration logic. The read data in this case cannot be guaranteed. The RDCOLERR bit is cleared by writing a 1 to it. Writing a 0 to RDCOLERR has no effect. 0 No collision error detected 1 Collision error detected
5 ACCERR	Flash Access Error Flag Indicates an illegal access has occurred to a flash memory resource caused by a violation of the command write sequence or issuing an illegal flash command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR while CCIF is set. Writing a 0 to the ACCERR bit has no effect. 0 No access error detected 1 Access error detected
4 FPVIOL	Flash Protection Violation Flag Indicates an attempt was made to program or erase an address in a protected area of program flash memory during a command write sequence . While FPVIOL is set, the CCIF flag cannot be cleared to launch a command. The FPVIOL bit is cleared by writing a 1 to FPVIOL while CCIF is set. Writing a 0 to the FPVIOL bit has no effect. 0 No protection violation detected 1 Protection violation detected
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 MGSTAT0	Memory Controller Command Completion Status Flag The MGSTAT0 status flag is set if an error is detected during execution of a flash command or during the flash reset sequence. As a status flag, this field cannot (and need not) be cleared by the user like the other error flags in this register.

Table continues on the next page...

FTFA_FSTAT field descriptions (continued)

Field	Description
	The value of the MGSTAT0 bit for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the previous result is discarded and any previous error is cleared.

24.3.4.2 Flash Configuration Register (FTFA_FCNFG)

This register provides information on the current functional state of the flash memory module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. The unassigned bits read as noted and are not writable.

Address: 4002_0000h base + 1h offset = 4002_0001h

Bit	7	6	5	4	3	2	1	0
Read	CCIE	RDCOLLIE	ERSAREQ	ERSSUSP	0	0	0	0
Write								
Reset	0	0	0	0	0	0	0	0

FTFA_FCNFG field descriptions

Field	Description
7 CCIE	Command Complete Interrupt Enable Controls interrupt generation when a flash command completes. 0 Command complete interrupt disabled 1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6 RDCOLLIE	Read Collision Error Interrupt Enable Controls interrupt generation when a flash memory read collision error occurs. 0 Read collision error interrupt disabled 1 Read collision error interrupt enabled. An interrupt request is generated whenever a flash memory read collision error is detected (see the description of FSTAT[RDCOLERR]).
5 ERSAREQ	Erase All Request Issues a request to the memory controller to execute the Erase All Blocks command and release security. ERSAREQ is not directly writable but is under indirect user control. Refer to the device's Chip Configuration details on how to request this command. ERSAREQ sets when an erase all request is triggered external to the flash memory module and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the flash memory module when the operation completes. 0 No request or request complete 1 Request to: 1. run the Erase All Blocks command,

Table continues on the next page...

FTFA_FCNFG field descriptions (continued)

Field	Description
	2. verify the erased state, 3. program the security byte in the Flash Configuration Field to the unsecure state, and 4. release MCU security by setting the FSEC[SEC] field to the unsecure state.
4 ERSSUSP	Erase Suspend Allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing. 0 No suspend requested 1 Suspend the current Erase Flash Sector command execution.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

24.3.4.3 Flash Security Register (FTFA_FSEC)

This read-only register holds all bits associated with the security of the MCU and flash memory module.

During the reset sequence, the register is loaded with the contents of the flash security byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002_0000h base + 2h offset = 4002_0002h

Bit	7	6	5	4	3	2	1	0
Read	KEYEN		MEEN		FSLACC		SEC	
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FSEC field descriptions

Field	Description
7–6 KEYEN	Backdoor Key Security Enable Enables or disables backdoor key access to the flash memory module. 00 Backdoor key access disabled 01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access)

Table continues on the next page...

FTFA_FSEC field descriptions (continued)

Field	Description
	10 Backdoor key access enabled 11 Backdoor key access disabled
5–4 MEEN	Mass Erase Enable Enables and disables mass erase capability of the flash memory module at all times in all NVM modes. 00 Mass erase is enabled 01 Mass erase is enabled 10 Mass erase is disabled 11 Mass erase is enabled
3–2 FSLACC	Factory Security Level Access Code Enables or disables access to the flash memory contents during returned part failure analysis at NXP. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed by NXP factory test must begin with a full erase to unsecure the part. When access is granted (SEC is unsecure, or SEC is secure and FSLACC is granted), NXP factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when SEC is set to secure. When SEC is set to unsecure, the FSLACC setting does not matter. 00 NXP factory access granted 01 NXP factory access denied 10 NXP factory access denied 11 NXP factory access granted
SEC	Flash Security Defines the security state of the MCU. In the secure state, the MCU limits access to flash memory module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the flash memory module is unsecured using backdoor key access, SEC is forced to 10b. 00 MCU security status is secure. 01 MCU security status is secure. 10 MCU security status is unsecure. (The standard shipping condition of the flash memory module is unsecure.) 11 MCU security status is secure.

24.3.4.4 Flash Option Register (FTFA_FOPT)

The flash option register allows the MCU to customize its operations by examining the state of these read-only bits, which are loaded from NVM at reset. The function of the bits is defined in the device's Chip Configuration details.

All bits in the register are read-only .

During the reset sequence, the register is loaded from the flash nonvolatile option byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value. However, the register is written to 0xFF if the contents of the flash nonvolatile option byte are 0x00.

Address: 4002_0000h base + 3h offset = 4002_0003h

Bit	7	6	5	4	3	2	1	0
Read	OPT							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FOPT field descriptions

Field	Description
OPT	Nonvolatile Option These bits are loaded from flash to this register at reset. Refer to the device's Chip Configuration details for the definition and use of these bits.

24.3.4.5 Flash Common Command Object Registers (FTFA_FCCOB*n*)

The FCCOB register group provides 12 bytes for command codes and parameters. The individual bytes within the set append a 0-B hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOBB.

Address: 4002_0000h base + 4h offset + (1d × i), where i=0d to 11d

Bit	7	6	5	4	3	2	1	0
Read Write	CCOBn							
Reset	0	0	0	0	0	0	0	0

FTFA_FCCOB*n* field descriptions

Field	Description
CCOBn	The FCCOB register provides a command code and relevant parameters to the memory controller. The individual registers that compose the FCCOB data set can be written in any order, but you must provide all needed values, which vary from command to command. First, set up all required FCCOB fields and then initiate the command's execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit, which locks all FCCOB parameter fields and they cannot be changed by the user until the command completes (CCIF returns to 1). No command buffering or queueing is provided; the next command can be loaded only after the current command completes. Some commands return information to the FCCOB registers. Any values returned to FCCOB are available for reading after the FSTAT[CCIF] flag returns to 1 by the memory controller. The following table shows a generic flash command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific flash command, typically an address and/or data values.

FTFA_FCCOB*n* field descriptions (continued)

Field	Description
	NOTE: The command parameter table is written in terms of FCCOB Number (which is equivalent to the byte number). This number is a reference to the FCCOB register name and is not the register address.

FCCOB Number	Typical Command Parameter Contents [7:0]
0	FCMD (a code that defines the flash command)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0]
4	Data Byte 0
5	Data Byte 1
6	Data Byte 2
7	Data Byte 3
8	Data Byte 4
9	Data Byte 5
A	Data Byte 6
B	Data Byte 7

FCCOB Endianness and Multi-Byte Access :

The FCCOB register group uses a big endian addressing convention. For all command parameter fields larger than 1 byte, the most significant data resides in the lowest FCCOB register number. The FCCOB register group may be read and written as individual bytes, aligned words (2 bytes) or aligned longwords (4 bytes).

24.3.4.6 Program Flash Protection Registers (FTFA_FPROT*n*)

The FPROT registers define which program flash regions are protected from program and erase operations. Protected flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any flash command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow up to 32 protectable regions. Each bit protects a 1/32 region of the program flash memory except for memory configurations with less than 64 KB of program flash where each assigned bit protects 2 KB . For configurations with 48 KB of program flash memory or less, FPROT0 is not used. For configurations with 32 KB of program flash memory or less, FPROT1 is not used. For configurations with 16 KB of program flash memory, FPROT2 is not used. The bitfields are defined in each register as follows:

Program flash protection register	Program flash protection bits
FPROT0	PROT[31:24]
FPROT1	PROT[23:16]
FPROT2	PROT[15:8]
FPROT3	PROT[7:0]

During the reset sequence, the FPROT registers are loaded with the contents of the program flash protection bytes in the Flash Configuration Field as indicated in the following table.

Program flash protection register	Flash Configuration Field offset address
FPROT0	0x000B
FPROT1	0x000A
FPROT2	0x0009
FPROT3	0x0008

To change the program flash protection that is loaded during the reset sequence, unprotect the sector of program flash memory that contains the Flash Configuration Field. Then, reprogram the program flash protection byte.

Address: 4002_0000h base + 10h offset + (1d × i), where i=0d to 3d

Bit	7	6	5	4	3	2	1	0
Read Write	PROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FPROT*n* field descriptions

Field	Description
PROT	Program Flash Region Protect Each program flash region can be protected from program and erase operations by setting the associated PROT bit. **In NVM Normal mode:** The protection can only be increased, meaning that currently unprotected memory can be protected, but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **In NVM Special mode:** All bits of FPROT are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected. **Restriction:** The user must never write to any FPROT register while a command is running (CCIF=0). Trying to alter data in any protected area in the program flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A full block erase of a program flash block is not possible if it contains any protected region.

FTFA_FPROT*n* field descriptions (continued)

Field	Description
	Each bit in the 32-bit protection register represents 1/32 of the total program flash except for memory configurations with less than 64 KB of program flash where each assigned bit protects 2 KB . 0 Program flash region is protected. 1 Program flash region is not protected

24.3.4.7 Execute-only Access Registers (FTFA_XACC*n*)

The XACC registers define which program flash segments are restricted to data read or execute only or both data and instruction fetches.

The eight XACC registers allow up to 64 restricted segments of equal memory size.

Execute-only access register	Program flash execute-only access bits
XACCH0	XA[63:56]
XACCH1	XA[55:48]
XACCH2	XA[47:40]
XACCH3	XA[39:32]
XACCL0	XA[31:24]
XACCL1	XA[23:16]
XACCL2	XA[15:8]
XACCL3	XA[7:0]

During the reset sequence, the XACC registers are loaded with the logical AND of Program Flash IFR addresses A and B as indicated in the following table.

Execute-only access register	Program Flash IFR address A	Program Flash IFR address B
XACCH0	0xA3	0xAB
XACCH1	0xA2	0xAA
XACCH2	0xA1	0xA9
XACCH3	0xA0	0xA8
XACCL0	0xA7	0xAF
XACCL1	0xA6	0xAE
XACCL2	0xA5	0xAD
XACCL3	0xA4	0xAC

Use the Program Once command to program the execute-only access control fields that are loaded during the reset sequence.

Address: 4002_0000h base + 18h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read	XA							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_XACC*n* field descriptions

Field	Description
XA	Execute-only access control 0 Associated segment is accessible in execute mode only (as an instruction fetch) 1 Associated segment is accessible as data or in execute mode

24.3.4.8 Supervisor-only Access Registers (FTFA_SACC*n*)

The SACC registers define which program flash segments are restricted to supervisor only or user and supervisor access.

The eight SACC registers allow up to 64 restricted segments of equal memory size.

Supervisor-only access register	Program flash supervisor-only access bits
SACCH0	SA[63:56]
SACCH1	SA[55:48]
SACCH2	SA[47:40]
SACCH3	SA[39:32]
SACCL0	SA[31:24]
SACCL1	SA[23:16]
SACCL2	SA[15:8]
SACCL3	SA[7:0]

During the reset sequence, the SACC registers are loaded with the logical AND of Program Flash IFR addresses A and B as indicated in the following table.

Supervisor-only access register	Program Flash IFR address A	Program Flash IFR address B
SACCH0	0xB3	0xBB
SACCH1	0xB2	0xBA
SACCH2	0xB1	0xB9
SACCH3	0xB0	0xB8

Table continues on the next page...

Supervisor-only access register	Program Flash IFR address A	Program Flash IFR address B
SACCL0	0xB7	0xBF
SACCL1	0xB6	0xBE
SACCL2	0xB5	0xBD
SACCL3	0xB4	0xBC

Use the Program Once command to program the supervisor-only access control fields that are loaded during the reset sequence.

Address: 4002_0000h base + 20h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read	SA							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_SACC*n* field descriptions

Field	Description
SA	Supervisor-only access control 0 Associated segment is accessible in supervisor mode only 1 Associated segment is accessible in user or supervisor mode

24.3.4.9 Flash Access Segment Size Register (FTFA_FACSS)

The flash access segment size register determines which bits in the address are used to index into the SACC and XACC bitmaps to get the appropriate permission flags.

All bits in the register are read-only.

The contents of this register are loaded during the reset sequence.

Address: 4002_0000h base + 28h offset = 4002_0028h

Bit	7	6	5	4	3	2	1	0
Read	SGSIZE							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FACSS field descriptions

Field	Description
SGSIZE	Segment Size The segment size is a fixed value based on the available program flash size divided by NUMSG.

Program Flash Size	Segment Size	Segment Size Encoding
64 KBytes	2 KBytes	0x3
128 KBytes	4 KBytes	0x4
160 KBytes	4 KBytes	0x4
256 KBytes	4 KBytes	0x4
512 KBytes	8 KBytes	0x5

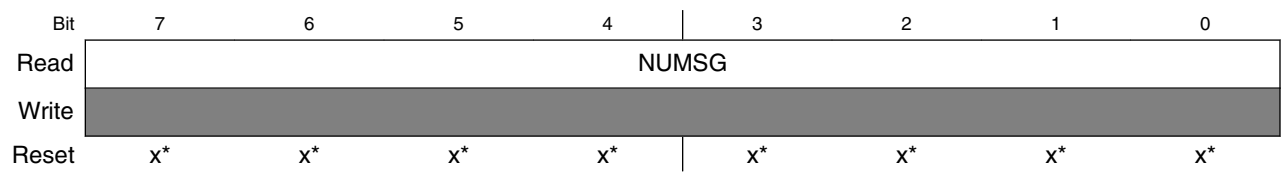
24.3.4.10 Flash Access Segment Number Register (FTFA_FACSN)

The flash access segment number register provides the number of program flash segments that are available for XACC and SACC permissions.

All bits in the register are read-only.

The contents of this register are loaded during the reset sequence.

Address: 4002_0000h base + 2Bh offset = 4002_002Bh



* Notes:
- x = Undefined at reset.

FTFA_FACSN field descriptions

Field	Description
NUMSG	Number of Segments Indicator The NUMSG field indicates the number of equal-sized segments in the program flash. 0x20 Program flash memory is divided into 32 segments (64 Kbytes, 128 Kbytes) 0x28 Program flash memory is divided into 40 segments (160 Kbytes) 0x4x Program flash memory is divided into 64 segments (256 Kbytes, 512 Kbytes)

24.4 Functional Description

The information found here describes functional details of the flash memory module.

24.4.1 Flash Protection

Individual regions within the flash memory can be protected from program and erase operations.

Protection is controlled by the following registers:

- FPROT*n* —
  - For 2^n program flash sizes, four registers typically protect 32 regions of the program flash memory as shown in the following figure

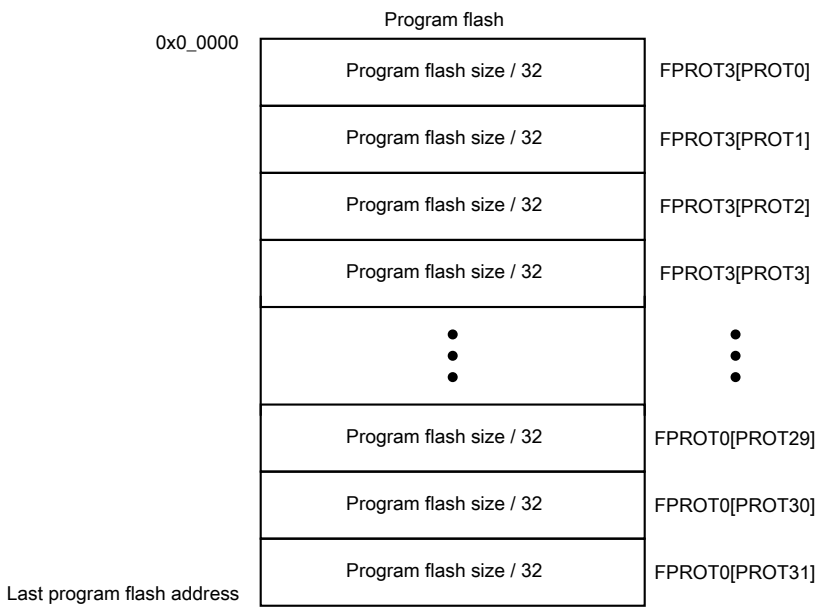


Figure 24-2. Program flash protection

NOTE

Flash protection features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Not all features described in the application note are available on this device.

24.4.2 Flash Access Protection

Individual segments within the program flash memory can be designated for restricted access. Specific flash commands (Program Check, Program Longword, Erase Flash Block, Erase Flash Sector) monitor FXACC contents to protect flash memory but the FSACC contents do not impact flash command operation.

See AN5112: Using the Kinetis Flash Execute-Only Access Control Feature for further details.

Access is controlled by the following registers:

- FTFA_XACC —
  - For 2^n program flash sizes greater than 128KB, eight registers control 64 segments of the program flash memory as shown in the following figure

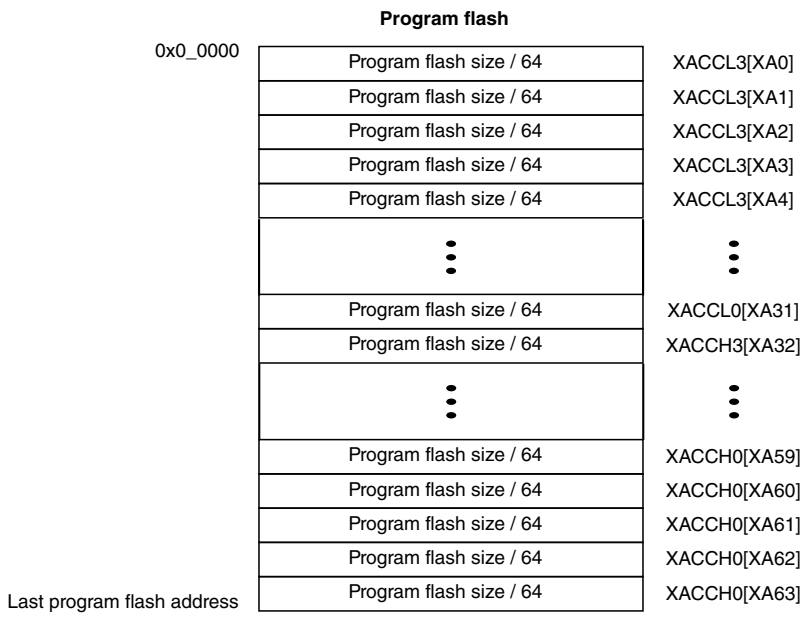


Figure 24-3. Program flash execute-only access control (256KB or 512KB of program flash)

- FTFA_SACC —
  - For 2^n program flash sizes greater than 128KB, eight registers control 64 segments of the program flash memory as shown in the following figure

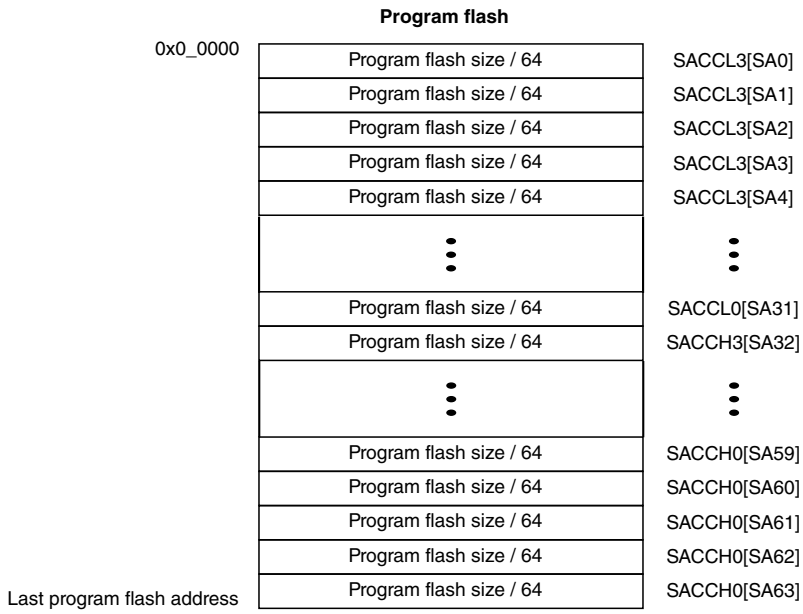


Figure 24-4. Program flash supervisor access control (256KB or 512KB of program flash)

24.4.3 Interrupts

The flash memory module can generate interrupt requests to the MCU upon the occurrence of various flash events.

These interrupt events and their associated status and control bits are shown in the following table.

Table 24-1. Flash Interrupt Sources

Flash Event	Readable Status Bit	Interrupt Enable Bit
Flash Command Complete	FSTAT[CCIF]	FCNFG[CCIE]
Flash Read Collision Error	FSTAT[RDCOLERR]	FCNFG[RDCOLLIE]

Note

Vector addresses and their relative interrupt priority are determined at the MCU level.

Some devices also generate a bus error response as a result of a Read Collision Error event. See the chip configuration information to determine if a bus error response is also supported.

24.4.4 Flash Operation in Low-Power Modes

24.4.4.1 Wait Mode

When the MCU enters wait mode, the flash memory module is not affected. The flash memory module can recover the MCU from wait via the command complete interrupt (see Interrupts).

24.4.4.2 Stop Mode

When the MCU requests stop mode, if a flash command is active (CCIF = 0) the command execution completes before the MCU is allowed to enter stop mode.

CAUTION

The MCU should never enter stop mode while any flash command is running (CCIF = 0).

NOTE

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the flash memory module does not accept flash commands.

24.4.5 Functional Modes of Operation

The flash memory module has two operating modes: NVM Normal and NVM Special.

The operating mode affects the command set availability (see Table 24-2). Refer to the Chip Configuration details of this device for how to activate each mode.

24.4.6 Flash Reads and Ignored Writes

The flash memory module requires only the flash address to execute a flash memory read.

The MCU must not read from the flash memory while commands are running (as evidenced by CCIF=0) on that block. Read data cannot be guaranteed from a flash block while any command is processing within that block. The block arbitration logic detects any simultaneous access and reports this as a read collision error (see the FSTAT[RDCOLERR] bit).

24.4.7 Read While Write (RWW)

The following simultaneous accesses are allowed:

- The user may read from one logical program flash memory space while flash commands are active in the other logical program flash memory space.

Simultaneous operations are further discussed in Allowed Simultaneous Flash Operations.

24.4.8 Flash Program and Erase

All flash functions except read require the user to setup and launch a flash command through a series of peripheral bus writes.

The user cannot initiate any further flash commands until notified that the current command has completed. The flash command structure and operation are detailed in Flash Command Operations.

24.4.9 Flash Command Operations

Flash command operations are typically used to modify flash memory contents.

The next sections describe:

- The command write sequence used to set flash command parameters and launch execution
- A description of all flash commands available

24.4.9.1 Command Write Sequence

Flash commands are specified using a command write sequence illustrated in Figure 24-5. The flash memory module performs various checks on the command (FCCOB) content and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be zero and the CCIF flag must read 1 to verify that any previous command has completed. If CCIF is zero, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

Attempts to launch a flash command in VLP mode will be ignored. Attempts to launch a flash command in HSRUN mode will be trapped with the ACCERR flag being set.

24.4.9.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired flash command. The individual registers that make up the FCCOB data set can be written in any order.

24.4.9.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing FSTAT[CCIF] by writing a '1' to it. FSTAT[CCIF] remains 0 until the flash command completes.

The FSTAT register contains a blocking mechanism that prevents a new command from launching (can't clear FSTAT[CCIF]) if the previous command resulted in an access error (FSTAT[ACCERR]=1) or a protection violation (FSTAT[FPVIOL]=1). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write clears the error flags, the second write clears CCIF.

24.4.9.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The flash memory module reads the command code and performs a series of parameter checks and protection checks, if applicable, which are unique to each command.

If the parameter check fails, the FSTAT[ACCERR] (access error) flag is set. FSTAT[ACCERR] reports invalid instruction codes and out-of bounds addresses. Usually, access errors suggest that the command was not set-up with valid parameters in the FCCOB register group.

Program and erase commands also check the address to determine if the operation is requested to execute on protected areas. If the protection check fails, FSTAT[FPVIOL] (protection error) flag is set.

Command processing never proceeds to execution when the parameter or protection step fails. Instead, command processing is terminated after setting FSTAT[CCIF].

2. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to erase verify, may occur during the execution phase. Run-time errors are reported in FSTAT[MGSTAT0]. A command may have access errors, protection errors, and run-time errors, but the run-time errors are not seen until all access and protection errors have been corrected.
3. Command execution results, if applicable, are reported back to the user via the FCCOB and FSTAT registers.
4. The flash memory module sets FSTAT[CCIF] signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

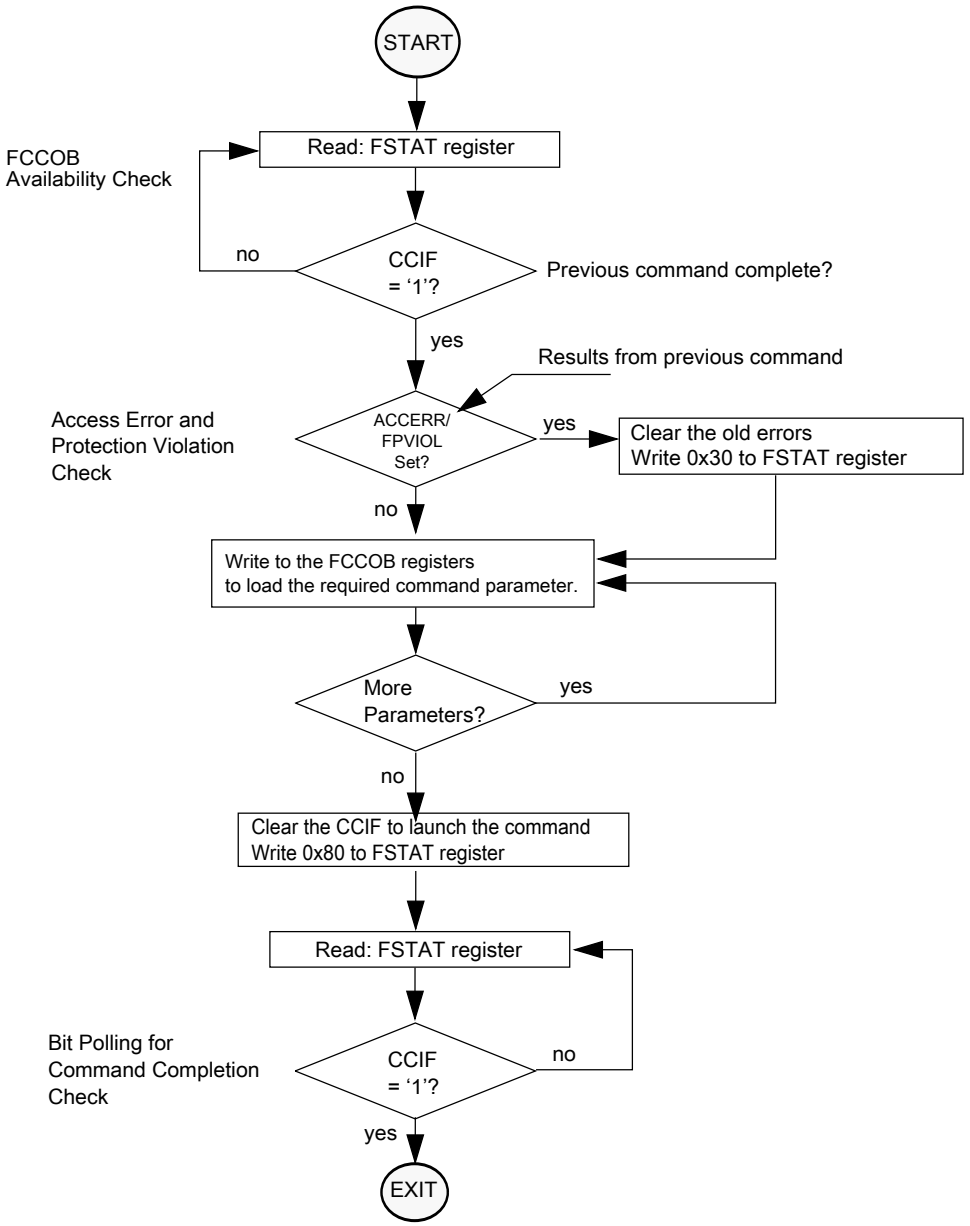


Figure 24-5. Generic flash command write sequence flowchart

24.4.9.2 Flash Commands

The following table summarizes the function of all flash commands.

FCMD	Command	Program flash 0	Program flash 1	Function
0x00	Read 1s Block	×	×	Verify that a program flash block is erased.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Function
0x01	Read 1s Section	×	×	Verify that a given number of program flash locations from a starting address are erased.
0x02	Program Check	×	×	Tests previously-programmed locations at margin read levels.
0x03	Read Resource	IFR, ID	IFR	Read 4 bytes from program flash IFR or version ID.
0x06	Program Longword	×	×	Program 4 bytes in a program flash block.
0x08	Erase Flash Block	×	×	Erase a program flash block. An erase of any flash block is only possible when unprotected.
0x09	Erase Flash Sector	×	×	Erase all bytes in a program flash sector.
0x40	Read 1s All Blocks	×	×	Verify that all program flash blocks are erased then release MCU security.
0x41	Read Once	IFR		Read 4 bytes of a dedicated 64 byte field in the program flash 0 IFR.
0x43	Program Once	IFR		One-time program of 4 bytes of a dedicated 64-byte field in the program flash 0 IFR.
0x44	Erase All Blocks	×	×	Erase all program flash blocks. Then, verify-erase and release MCU security. **NOTE:** An erase is only possible when all memory locations are unprotected.
0x45	Verify Backdoor Access Key	×	×	Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Function
0x49	Erase All Blocks Unsecure	×	×	Erase all program flash blocks, verify-erase, program security byte to unsecure state, release MCU security.
0x4A	Read 1s All Execute-only Segments	×	×	Verify that all program flash execute-only (XA) segments are erased then release flash access control.
0x4B	Erase All Execute-only Segments	×	×	Erase all program flash execute-only (XA) segments then release flash access control.

24.4.9.3 Flash Commands by Mode

The following table shows the flash commands that can be executed in each flash operating mode.

Table 24-2. Flash Commands by Mode

FCMD	Command	NVM Normal			NVM Special		
		Unsecure	Secure	MEEN=10	Unsecure	Secure	MEEN=10
0x00	Read 1s Block	×	×	×	×	—	—
0x01	Read 1s Section	×	×	×	×	—	—
0x02	Program Check	×	×	×	×	—	—
0x03	Read Resource	×	×	×	×	—	—
0x06	Program Longword	×	×	×	×	—	—
0x08	Erase Flash Block	×	×	×	×	—	—
0x09	Erase Flash Sector	×	×	×	×	—	—
0x40	Read 1s All Blocks	×	×	—	×	×	—
0x41	Read Once	×	×	×	×	—	—
0x43	Program Once	×	×	×	×	—	—
0x44	Erase All Blocks	×	×	—	×	×	—
0x45	Verify Backdoor Access Key	×	×	×	×	—	—
0x49	Erase All Blocks Unsecure	×	×	—	×	×	—
0x4A	Read 1s All Execute-only Segments	×	×	×	×	—	—
0x4B	Erase All Execute-only Segments	×	×	×	×	—	—

24.4.9.4 Allowed Simultaneous Flash Operations

Only the operations marked 'OK' in the following table are permitted to run simultaneously on the program flash memories. Some operations cannot be executed simultaneously because certain hardware resources are shared by the memories.

Table 24-3. Allowed Simultaneous Memory Operations

		Program Flash 0			Program Flash 1		
		Read	Program	Sector Erase	Read	Program	Sector Erase
Program flash 0	Read	—				OK	OK
	Program		—		OK		
	Sector Erase			—	OK		
Program flash 1	Read		OK	OK	—		
	Program	OK				—	
	Sector Erase	OK					—

24.4.10 Margin Read Commands

The Read-1s commands (Read 1s All Blocks, Read 1s Block, Read 1s Section, Read 1s All Execute-only Segments) and the Program Check command have a margin choice parameter that allows the user to apply non-standard read reference levels to the program flash array reads performed by these commands. Using the preset 'user' and 'factory' margin levels, these commands perform their associated read operations at tighter tolerances than a 'normal' read. These non-standard read levels are applied only during the command execution. Basic flash array reads use the standard, un-margined, read reference level.

Only the 'normal' read level should be employed during normal flash usage. The non-standard, 'user' and 'factory' margin levels should be employed only in special cases. They can be used during special diagnostic routines to gain confidence that the device is not suffering from the end-of-life data loss customary of flash memory devices.

Erased ('1') and programmed ('0') bit states can degrade due to elapsed time and data cycling (number of times a bit is erased and re-programmed). The lifetime of the erased states is relative to the last erase operation. The lifetime of the programmed states is measured from the last program time.

The 'user' and 'factory' levels become, in effect, a minimum safety margin; i.e. if the reads pass at the tighter tolerances of the 'user' and 'factory' margins, then the 'normal' reads have at least this much safety margin before they experience data loss.

The 'user' margin is a small delta to the normal read reference level. 'User' margin levels can be employed to check that flash memory contents have adequate margin for normal level read operations. If unexpected read results are encountered when checking flash memory contents at the 'user' margin levels, loss of information might soon occur during 'normal' readout.

The 'factory' margin is a bigger deviation from the norm, a more stringent read criteria that should only be attempted immediately (or very soon) after completion of an erase or program command, early in the cycling life. 'Factory' margin levels can be used to check that flash memory contents have adequate margin for long-term data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at 'factory' margin levels, the flash memory contents should be erased and reprogrammed.

CAUTION

Factory margin levels must only be used during verify of the initial factory programming.

24.4.11 Flash Command Description

This section describes all flash commands that can be launched by a command write sequence.

The flash memory module sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in a FCCOB field for the specific commands. Refer to the error handling table provided for each command.

Ensure that FSTAT[ACCERR] and FSTAT[FPVIOL] are cleared prior to starting the command write sequence. As described in Launch the Command by Clearing CCIF, a new command cannot be launched while these error flags are set.

Do not attempt to read a flash block while the flash memory module is running a command (FSTAT[CCIF] = 0) on that same block. The flash memory module may return invalid data to the MCU with the collision error flag (FSTAT[RDCOLERR]) set.

CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

24.4.11.1 Read 1s Block Command

The Read 1s Block command checks to see if an entire program flash block has been erased to the specified margin level. The FCCOB flash address bits determine which flash block is erase-verified.

Table 24-4. Read 1s Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x00 (RD1BLK)
1	Flash address [23:16] in the flash block to be verified
2	Flash address [15:8] in the flash block to be verified
3	Flash address [7:0][1] in the flash block to be verified
4	Read-1 Margin Choice

1. Must be longword aligned (Flash address [1:0] = 00).

After clearing CCIF to launch the Read 1s Block command, the flash memory module sets the read margin for 1s according to Table 24-5 and then reads all locations within the selected program flash block.

Table 24-5. Margin Level Choices for Read 1s Block

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 24-6. Read 1s Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

24.4.11.2 Read 1s Section Command

The Read 1s Section command checks if a section of program flash memory is erased to the specified read margin level. The Read 1s Section command defines the starting address and the number of phrases to be verified.

Table 24-7. Read 1s Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x01 (RD1SEC)
1	Flash address [23:16] of the first phrase to be verified
2	Flash address [15:8] of the first phrase to be verified
3	Flash address [7:0][1] of the first phrase to be verified
4	Number of phrases to be verified [15:8]
5	Number of phrases to be verified [7:0]
6	Read-1 Margin Choice

1. Must be phrase aligned (Flash address [2:0] = 000).

Upon clearing CCIF to launch the Read 1s Section command, the flash memory module sets the read margin for 1s according to Table 24-8 and then reads all locations within the specified section of flash memory. If the flash memory module fails to read all 1s (that is, the flash section is not erased), FSTAT[MGSTAT0] is set. FSTAT[CCIF] sets after the Read 1s Section operation completes.

Table 24-8. Margin Level Choices for Read 1s Section

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 24-9. Read 1s Section Command Error Handling

Error condition	Error bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin code is supplied.	FSTAT[ACCERR]
An invalid flash address is supplied.	FSTAT[ACCERR]
Flash address is not phrase aligned.	FSTAT[ACCERR]
The requested section crosses a Flash block boundary.	FSTAT[ACCERR]
The requested number of phrases is 0.	FSTAT[ACCERR]
Read-1s fails.	FSTAT[MGSTAT0]

24.4.11.3 Program Check Command

The Program Check command tests a previously programmed program flash longword to see if it reads correctly at the specified margin level.

Table 24-10. Program Check Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x02 (PGMCHK)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Margin Choice
8	Byte 0 expected data
9	Byte 1 expected data
A	Byte 2 expected data
B	Byte 3 expected data

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Check command, the flash memory module sets the read margin for 1s according to Table 24-11, reads the specified longword, and compares the actual read data to the expected data provided by the FCCOB. If the comparison at margin-1 fails, FSTAT[MGSTAT0] is set.

The flash memory module then sets the read margin for 0s, re-reads, and compares again. If the comparison at margin-0 fails, FSTAT[MGSTAT0] is set. FSTAT[CCIF] is set after the Program Check operation completes.

The supplied address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10,
- Byte 0 data is programmed to byte address start+0b11.

NOTE

See the description of margin reads, Margin Read Commands

Table 24-11. Margin Level Choices for Program Check

Read Margin Choice	Margin Level Description
0x01	Read at 'User' margin-1 and 'User' margin-0
0x02	Read at 'Factory' margin-1 and 'Factory' margin-0

Table 24-12. Program Check Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]

Table continues on the next page...

Table 24-12. Program Check Command Error Handling (continued)

Error Condition	Error Bit
Flash address is not longword aligned	FSTAT[ACCERR]
An invalid margin choice is supplied	FSTAT[ACCERR]
Flash address is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Either of the margin reads does not match the expected data	FSTAT[MGSTAT0]

24.4.11.4 Read Resource Command

The Read Resource command allows the user to read data from special-purpose memory resources located within the flash memory module. The special-purpose memory resources available include program flash IFR space and the Version ID field. Each resource is assigned a select code as shown in Table 24-14.

Table 24-13. Read Resource Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x03 (RDRSRC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
Returned Values	
4	Read Data [31:24]
5	Read Data [23:16]
6	Read Data [15:8]
7	Read Data [7:0]
User-provided values	
8	Resource Select Code (see Table 24-14)

1. Must be longword aligned (Flash address [1:0] = 00).

Table 24-14. Read Resource Select Codes

Resource Select Code	Description	Resource Size	Local Address Range
0x00	Program Flash 0 IFR	256 Bytes	0x00_0000–0x00_00FF
0x01[1]	Version ID	8 Bytes	0x00_0000–0x00_0007

1. Located in program flash 0 reserved space.

After clearing CCIF to launch the Read Resource command, four consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag sets after the Read Resource operation completes. The Read Resource command exits with an access error if an invalid resource code is provided or if the address for the applicable area is out-of-range.

Table 24-15. Read Resource Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid resource code is entered	FSTAT[ACCERR]
Flash address is out-of-range for the targeted resource.	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]

24.4.11.5 Program Longword Command

The Program Longword command programs four previously-erased bytes in the program flash memory using an embedded algorithm.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 24-16. Program Longword Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x06 (PGM4)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Byte 0 program value
5	Byte 1 program value
6	Byte 2 program value
7	Byte 3 program value

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Longword command, the flash memory module programs the data bytes into the flash using the supplied address. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Longword operation.

The programming operation is unidirectional. It can only move NVM bits from the erased state ('1') to the programmed state ('0'). Erased bits that fail to program to the '0' state are flagged as errors in FSTAT[MGSTAT0]. The CCIF flag is set after the Program Longword operation completes.

The supplied address must be longword aligned (flash address [1:0] = 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10, and
- Byte 0 data is programmed to byte address start+0b11.

Table 24-17. Program Longword Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Flash address points to a protected area	FSTAT[FPVIOL]
Flash address is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

24.4.11.6 Erase Flash Block Command

The Erase Flash Block operation erases all addresses in a single program flash.

Table 24-18. Erase Flash Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x08 (ERSBLK)
1	Flash address [23:16] in the flash block to be erased
2	Flash address [15:8] in the flash block to be erased
3	Flash address [7:0][1] in the flash block to be erased

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Erase Flash Block command, the flash memory module erases the main array of the selected flash block and verifies that it is erased. The Erase Flash Block command aborts and sets the FSTAT[FPVIOL] bit if any region within the block is protected (see the description of the FPROT registers). If the erase verify fails, FSTAT[MGSTAT0] is set. The CCIF flag will set after the Erase Flash Block operation has completed.

Table 24-19. Erase Flash Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Any area of the selected flash block is protected	FSTAT[FPVIOL]
The selected program flash block contains an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s Block command to verify all bits are erased.

24.4.11.7 Erase Flash Sector Command

The Erase Flash Sector operation erases all addresses in a flash sector.

Table 24-20. Erase Flash Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x09 (ERSSCR)
1	Flash address [23:16] in the flash sector to be erased
2	Flash address [15:8] in the flash sector to be erased
3	Flash address [7:0][1] in the flash sector to be erased

1. Must be phrase aligned (flash address [2:0] = 000).

After clearing CCIF to launch the Erase Flash Sector command, the flash memory module erases the selected program flash sector and then verifies that it is erased. The Erase Flash Sector command aborts if the selected sector is protected (see the description of the FPROT registers). If the erase-verify fails the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase Flash Sector operation completes. The Erase Flash Sector command is suspendable (see the FCNFG[ERSSUSP] bit and Figure 24-6).

Table 24-21. Erase Flash Sector Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid Flash address is supplied	FSTAT[ACCERR]
Flash address is not phrase aligned	FSTAT[ACCERR]
The selected program flash sector is protected	FSTAT[FPVIOL]
The selected program flash sector is located in an XA controlled segment and the Erase All Blocks, Erase All Blocks Unsecure or the Read 1s All Blocks command has not successfully completed since the last reset	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s Section command to verify all bits are erased.

24.4.11.7.1 Suspending an Erase Flash Sector Operation

To suspend an Erase Flash Sector operation set the FCNFG[ERSSUSP] bit when CCIF, ACCERR, and FPVIOL are clear and the CCOB command field holds the code for the Erase Flash Sector command. During the Erase Flash Sector operation (see Erase Flash Sector Command), the flash memory module samples the state of the ERSSUSP bit at convenient points. If the flash memory module detects that the ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the flash memory module sets CCIF. While ERSSUSP is set, all writes to flash registers are ignored except for writes to the FSTAT and FCNFG registers.

If an Erase Flash Sector operation effectively completes before the flash memory module detects that a suspend request has been made, the flash memory module clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the flash memory module sets CCIF and leaves the ERSSUSP bit set. While CCIF is set, the ERSSUSP bit can only be cleared to prevent the withdrawal of a suspend request before the flash memory module has acknowledged it.

24.4.11.7.2 Resuming a Suspended Erase Flash Sector Operation

If the ERSSUSP bit is still set when CCIF is cleared to launch the next command, the previous Erase Flash Sector operation resumes. The flash memory module acknowledges the request to resume a suspended operation by clearing the ERSSUSP bit. A new suspend request can then be made by setting ERSSUSP. A single Erase Flash Sector operation can be suspended and resumed multiple times.

There is a minimum elapsed time limit of 4.3 msec between the request to resume the Erase Flash Sector operation (CCIF is cleared) and the request to suspend the operation again (ERSSUSP is set). This minimum time period is required to ensure that the Erase Flash Sector operation will eventually complete. If the minimum period is continually

violated, i.e. the suspend requests come repeatedly and too quickly, no forward progress is made by the Erase Flash Sector algorithm. The resume/suspend sequence runs indefinitely without completing the erase.

24.4.11.7.3 Aborting a Suspended Erase Flash Sector Operation

The user may choose to abort a suspended Erase Flash Sector operation by clearing the ERSSUSP bit prior to clearing CCIF for the next command launch. When a suspended operation is aborted, the flash memory module starts the new command using the new FCCOB contents.

Note

Aborting the erase leaves the bitcells in an indeterminate, partially-erased state. Data in this sector is not reliable until a new erase command fully completes.

The following figure shows how to suspend and resume the Erase Flash Sector operation.

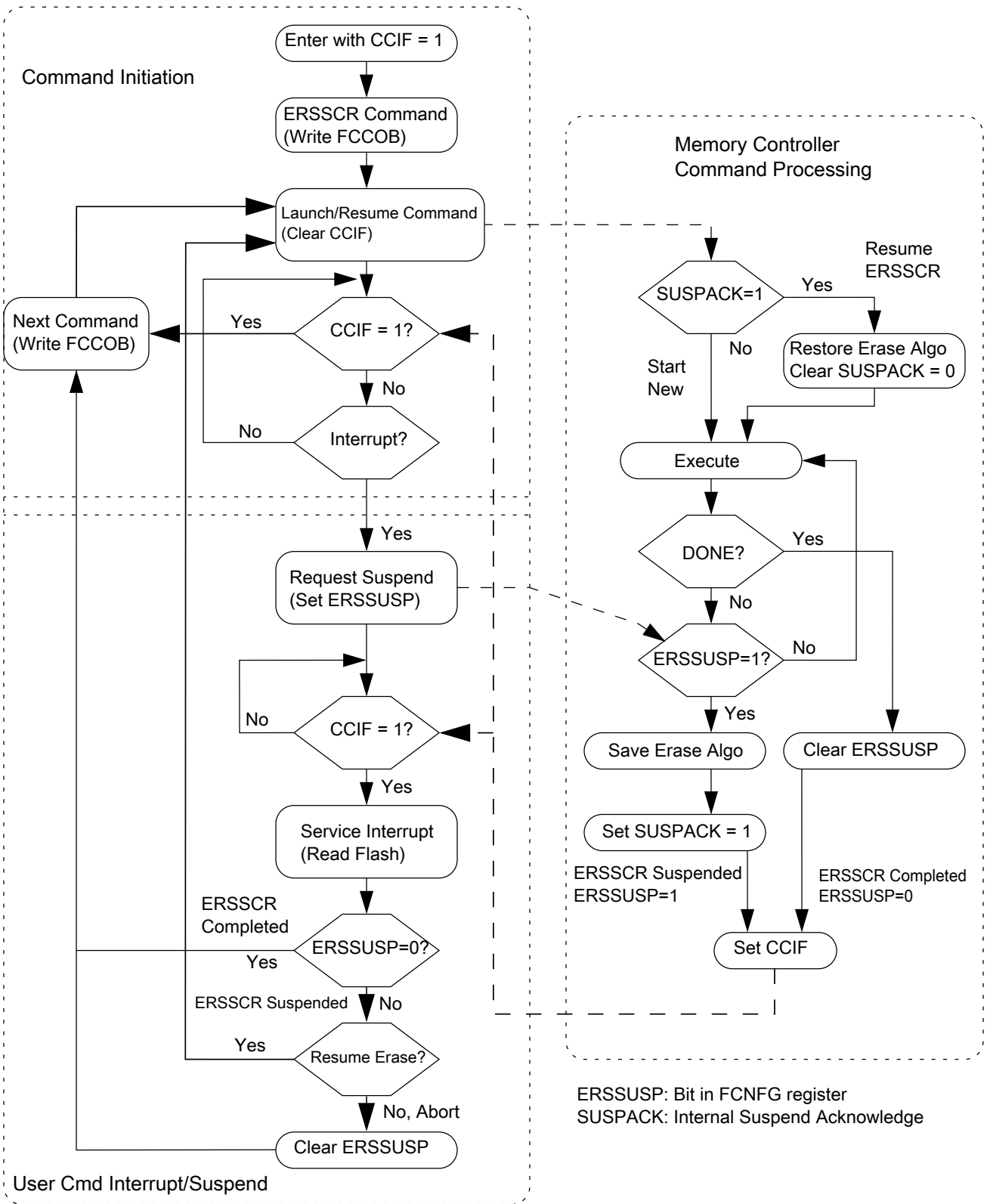


Figure 24-6. Suspend and Resume of Erase Flash Sector Operation

24.4.11.8 Read 1s All Blocks Command

The Read 1s All Blocks command checks if the program flash blocks and program flash erasable IFR have been erased to the specified read margin level, if applicable, and releases security if the readout passes, i.e. all data reads as '1'.

Table 24-22. Read 1s All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x40 (RD1ALL)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Blocks command, the flash memory module :

- sets the read margin for 1s according to Table 24-23,
- checks the contents of the program flash and program flash erasable IFR are in the erased state.

If the flash memory module confirms that these memory resources are erased, access control is disabled and security is released by setting the FSEC[SEC] field to the unsecure state. The security byte in the flash configuration field (see Flash Configuration Field Description) remains unaffected by the Read 1s All Blocks command. If the read fails, i.e. all memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

Table 24-23. Margin Level Choices for Read 1s All Blocks

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 24-24. Read 1s All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

24.4.11.9 Read Once Command

The Read Once command provides read access to special 96-byte fields located in the program flash 0 IFR (see Program Flash IFR Map and Program Once Field). Access to the Program Once ID field is via 16 records (index values 0x00 - 0x0F), each 4 bytes long. Access to the Program Once XACC and SACC fields are via 4 records (index values 0x10 - 0x13), each of which is 8 bytes long. These fields are programmed using the Program Once command described in Program Once Command.

The Read Once command also provides read access to special 4-byte records (index values 0x30 - 0x33) located in the program flash erasable IFR (see Program Flash Erasable IFR Map and Erasable Program Once Field). These fields are programmed using the Program Once command.

Table 24-25. Read Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x41 (RDONCE)
1	Program Once record index (0x00 - 0x13, 0x30 - 0x33)
2	Not used
3	Not used
Returned Values	
4	Program Once byte 0 value
5	Program Once byte 1 value
6	Program Once byte 2 value
7	Program Once byte 3 value
8	Program Once byte 4 value (index 0x10 - 0x13)
9	Program Once byte 5 value (index 0x10 - 0x13)
10	Program Once byte 6 value (index 0x10 - 0x13)
11	Program Once byte 7 value (index 0x10 - 0x13)

After clearing CCIF to launch the Read Once command, a 4-byte or 8-byte Program Once record is read and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. Valid record index values for the Read Once command range from 0x00 - 0x13, 0x30 - 0x33. During execution of the Read Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data. The Read Once command can be executed any number of times.

Table 24-26. Read Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]

24.4.11.10 Program Once Command

The Program Once command enables programming to special 96-byte fields in the program flash 0 IFR (see Program Flash IFR Map and Program Once Field). Access to the Program Once ID field is via 16 records (index values 0x00 - 0x0F), each 4 bytes long. Access to the Program Once XACC and SACC fields are via 4 records (index values 0x10 - 0x13), each of which is 8 bytes long. These records can be read using the Read Once command (see Read Once Command) or using the Read Resource command (see Read Resource Command). These records can be programmed only once since the program flash 0 IFR cannot be erased.

The Program Once command enables programming to special 4-byte records located in the program flash erasable IFR (see Program Flash Erasable IFR Map and Erasable Program Once Field). Record index values 0x30 - 0x33 can be read using the Read Once command. These records can be reprogrammed since the program flash erasable IFR can be erased using the Erase All Blocks command and Erase All Blocks Unsecure command.

Table 24-27. Program Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x43 (PGMONCE)
1	Program Once record index (0x00 - 0x13, 0x30 - 0x33)
2	Not Used
3	Not Used
4	Program Once byte 0 value
5	Program Once byte 1 value
6	Program Once byte 2 value
7	Program Once byte 3 value
8	Program Once byte 4 value (index 0x10 - 0x13)
9	Program Once byte 5 value (index 0x10 - 0x13)
10	Program Once byte 6 value (index 0x10 - 0x13)
11	Program Once byte 7 value (index 0x10 - 0x13)

After clearing CCIF to launch the Program Once command, the flash memory module first verifies that the selected record is erased. If erased, then the selected record is programmed using the values provided. The Program Once command also verifies that the programmed values read back correctly. The CCIF flag is set after the Program Once operation has completed.

Any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. Valid record index values for the Program Once command range from 0x00 - 0x13, 0x30 - 0x33. During execution of the Program Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data.

Table 24-28. Program Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]
The requested record has already been programmed to a non-FFFF value[1]	FSTAT[ACCERR]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

1. If a Program Once record is initially programmed to 0xFFFF_FFFF (0xFFFF_FFFF_FFFF_FFFF for index 0x10 - 0x13), the Program Once command is allowed to execute again on that same record.

24.4.11.11 Erase All Blocks Command

The Erase All Blocks operation erases all flash memory, verifies all memory contents, and releases MCU security.

Table 24-29. Erase All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x44 (ERSALL)

After clearing CCIF to launch the Erase All Blocks command, the flash memory module erases all program flash memory and the program flash erasable IFR space, then verifies that all are erased.

If the flash memory module verifies that all flash memories were properly erased, access control is disabled and security is released by setting the FSEC[SEC] field to the unsecure state. The Erase All Blocks command aborts if any flash region is protected. The security byte and all other contents of the flash configuration field (see Flash Configuration Field Description) are erased by the Erase All Blocks command. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks operation completes.

Access control determined by the contents of the FXACC registers will not block execution of the Erase All Blocks command. While most Flash memory will be erased, the program flash IFR space containing the Program Once XACC and SACC fields will not be erased and, therefore, the contents of the Program Once XACC and SACC fields will not change. The contents of the FXACC and FSACC registers will not be impacted

by the execution of the Erase All Blocks command. After completion of the Erase All Blocks command, access control is disabled until the next reset of the flash module or the Read 1s All Blocks command is executed and fails (FSTAT[MGSTAT0] is set).

Table 24-30. Erase All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any region of the program flash memory is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s All Blocks command to verify all bits are erased.

24.4.11.11.1 Triggering an Erase All External to the Flash Memory Module

The functionality of the Erase All Blocks/Erase All Blocks Unsecure command is also available in an uncommanded fashion outside of the flash memory. Refer to the device's Chip Configuration details for information on this functionality.

Before invoking the external erase all function, the FSTAT[ACCERR and PVIOL] flags must be cleared and the FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory and the program flash erasable IFR space regardless of the protection settings. If the post-erase verify passes, access control determined by the contents of the FXACC registers is disabled and the routine then releases security by setting the FSEC[SEC] field register to the unsecure state. The security byte in the Flash Configuration Field is also programmed to the unsecure state. The status of the erase-all request is reflected in the FCNFG[ERSAREQ] bit. The FCNFG[ERSAREQ] bit is cleared once the operation completes and the normal FSTAT error reporting is available, except FPVIOL, as described in Erase All Blocks Command/ Erase All Blocks Unsecure Command.

24.4.11.12 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command only executes if the mode and security conditions are satisfied (see Flash Commands by Mode). Execution of the Verify Backdoor Access Key command is further qualified by the FSEC[KEYEN] bits. The Verify Backdoor Access Key command releases security if user-supplied keys in the FCCOB match those stored in the Backdoor Comparison Key bytes of the Flash Configuration Field (see Flash Configuration Field Description). The column labelled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

Table 24-31. Verify Backdoor Access Key Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]	Flash Configuration Field Offset Address
0	0x45 (VFYKEY)	
1-3	Not Used	
4	Key Byte 0	0x0_0003
5	Key Byte 1	0x0_0002
6	Key Byte 2	0x0_0001
7	Key Byte 3	0x0_0000
8	Key Byte 4	0x0_0007
9	Key Byte 5	0x0_0006
A	Key Byte 6	0x0_0005
B	Key Byte 7	0x0_0004

After clearing CCIF to launch the Verify Backdoor Access Key command, the flash memory module checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the flash memory module sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the flash memory module compares the key provided in FCCOB to the backdoor comparison key in the Flash Configuration Field. If the backdoor keys match, the FSEC[SEC] field is changed to the unsecure state and security is released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are immediately aborted and the FSTAT[ACCERR] bit is (again) set to 1 until a reset of the flash memory module module occurs. If the entire 8-byte key is all zeros or all ones, the Verify Backdoor Access Key command fails with an access error. The CCIF flag is set after the Verify Backdoor Access Key operation completes.

Table 24-32. Verify Backdoor Access Key Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
The supplied key is all-0s or all-Fs	FSTAT[ACCERR]
An incorrect backdoor key is supplied	FSTAT[ACCERR]
Backdoor key access has not been enabled (see the description of the FSEC register)	FSTAT[ACCERR]
This command is launched and the backdoor key has mismatched since the last power down reset	FSTAT[ACCERR]

24.4.11.13 Erase All Blocks Unsecure Command

The Erase All Blocks Unsecure operation erases all flash memory, verifies all memory contents, programs the security byte in the Flash Configuration Field to the unsecure state, and releases MCU security.

Table 24-33. Erase All Blocks Unsecure Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x49 (ERSALLU)

After clearing CCIF to launch the Erase All Blocks Unsecure command, the flash memory module erases all program flash memory and the program flash erasable IFR space, then verifies that all are erased.

If the flash memory module verifies that all program flash memory and the program flash erasable IFR space was properly erased, access control is disabled, security is released by setting the FSEC[SEC] field to the unsecure state, and the security byte (see Flash Configuration Field Description) is programmed to the unsecure state by the Erase All Blocks Unsecure command. If the erase or program verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks Unsecure operation completes.

Access control determined by the contents of the FXACC registers will not block execution of the Erase All Blocks Unsecure command. While most Flash memory will be erased, the program flash IFR space containing the Program Once XACC and SACC fields will not be erased and, therefore, the contents of the Program Once XACC and SACC fields will not change. The contents of the FXACC and FSACC registers will not be impacted by the execution of the Erase All Blocks Unsecure command. After completion of the Erase All Blocks Unsecure command, access control is disabled until the next reset of the flash module or the Read 1s All Blocks command is executed and fails (FSTAT[MGSTAT0] is set).

Table 24-34. Erase All Blocks Unsecure Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any errors have been encountered during erase or program verify operations	FSTAT[MGSTAT0]

24.4.11.14 Read 1s All Execute-only Segments Command

The Read 1s All Execute-only Segments command checks if the program flash execute-only segments defined by the FXACC registers have been erased to the specified read margin level, if applicable, and releases flash access control if the readout passes, i.e. all data reads as '1'.

Table 24-35. Read 1s All Execute-only Segments Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x4A (RD1XA)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Execute-only Segments command, the flash memory module :

- sets the read margin for 1s according to Table 24-36,
- checks the contents of the program flash execute-only segments are in the erased state.

If the flash memory module confirms that these segments are erased, flash access control is disabled until the next reset or, after programming any of the execute-only segments, the Read 1s All Execute-only Segments command is executed and fails with the FSTAT[MGSTAT0] bit set. If the read fails, i.e. all segments are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The CCIF flag sets after the Read 1s All Execute-only Segments operation has completed.

Table 24-36. Margin Level Choices for Read 1s All Execute-only Segments

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 24-37. Read 1s All Execute-only Segments Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Sector size is larger than segment size	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

24.4.11.15 Erase All Execute-only Segments Command

The Erase All Execute-only Segments operation erases all program flash execute-only segments defined by the FXACC registers, verifies all segments are erased, and releases flash access control.

Table 24-38. Erase All Execute-only Segments Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x4B (ERSXA)

After clearing CCIF to launch the Erase All Execute-only Segments command, the flash memory module erases all program flash execute-only segments, then verifies that all segments are erased.

If the flash memory module verifies that all segments were properly erased, flash access control is disabled until the next reset or, after programming any of the execute-only segments, the Read 1s All Execute-only Segments command is executed and fails with the FSTAT[MGSTAT0] bit set. The Erase All Execute-only Segments command aborts if any XA controlled segment is protected. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Execute-only Segments operation completes.

Access control determined by the contents of the FXACC registers will not block execution of the Erase All Execute-only Segments command. While all XA controlled segments will be erased, the program flash IFR space containing the Program Once XACC fields will not be erased and, therefore, the contents of the Program Once XACC fields will not change. The contents of the FXACC registers will not be impacted by the execution of the Erase All Execute-only Segments command.

Table 24-39. Erase All Execute-only Segments Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Sector size is larger than segment size	FSTAT[ACCERR]
Any XA controlled segment in the program flash memory is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

24.4.12 Security

The flash memory module provides security information to the MCU based on contents of the FSEC security register.

The MCU then limits access to flash memory resources as defined in the device's Chip Configuration details. During reset, the flash memory module initializes the FSEC register using data read from the security byte of the Flash Configuration Field (see Flash Configuration Field Description).

The following fields are available in the FSEC register. The settings are described in the Flash Security Register (FTFA_FSEC) details.

Flash security features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Note that not all features described in the application note are available on this device.

Table 24-40. FSEC register fields

FSEC field	Description
KEYEN	Backdoor Key Access
MEEN	Mass Erase Capability
FSLACC	Factory Security Level Access
SEC	MCU security

24.4.12.1 Flash Memory Access by Mode and Security

The following table summarizes how access to the flash memory module is affected by security and operating mode.

Table 24-41. Flash Memory Access Summary

Operating Mode	Chip Security State	
	Unsecure	Secure
NVM Normal	Full command set	
NVM Special	Full command set	Only the Erase All Blocks, Erase All Blocks Unsecure and Read 1s All Blocks commands.

24.4.12.2 Changing the Security State

The security state out of reset can be permanently changed by programming the security byte of the flash configuration field. This assumes that you are starting from a mode where the necessary program flash erase and program commands are available and that the region of the program flash containing the flash configuration field is unprotected. If the flash security byte is successfully programmed, its new value takes affect after the next chip reset.

24.4.12.2.1 Unsecuring the Chip Using Backdoor Key Access

The chip can be unsecured by using the backdoor key access feature, which requires knowledge of the contents of the 8-byte backdoor key value stored in the Flash Configuration Field (see Flash Configuration Field Description). If the FSEC[KEYEN] bits are in the enabled state, the Verify Backdoor Access Key command (see Verify Backdoor Access Key Command) can be run; it allows the user to present prospective keys for comparison to the stored keys. If the keys match, the FSEC[SEC] bits are changed to unsecure the chip. The entire 8-byte key cannot be all 0s or all 1s; that is, 0000_0000_0000_0000h and FFFF_FFFF_FFFF_FFFFh are not accepted by the Verify Backdoor Access Key command as valid comparison values. While the Verify Backdoor Access Key command is active, program flash memory is not available for read access and returns invalid data.

The user code stored in the program flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN bits are in the enabled state, the chip can be unsecured by the following backdoor key access sequence:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Verify Backdoor Access Key Command
2. If the Verify Backdoor Access Key command is successful, the chip is unsecured and the FSEC[SEC] bits are forced to the unsecure state

An illegal key provided to the Verify Backdoor Access Key command prohibits further use of the Verify Backdoor Access Key command. A reset of the chip is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the chip is unsecured by changing the FSEC[SEC] bits. A successful execution of the Verify Backdoor Access Key command changes the security in the FSEC register only. It does not alter the security byte or the keys stored in the Flash Configuration Field (Flash Configuration Field Description). After the next reset of the chip, the security state of the flash memory

module reverts back to the flash security byte in the Flash Configuration Field. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the program flash protection registers.

If the backdoor keys successfully match, the unsecured chip has full control of the contents of the Flash Configuration Field. The chip may erase the sector containing the Flash Configuration Field and reprogram the flash security byte to the unsecure state and change the backdoor keys to any desired value.

24.4.13 Reset Sequence

On each system reset the flash memory module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FOPT, FSEC, FXACC, FSACC, and FACNFG registers.

FSTAT[CCIF] is cleared throughout the reset sequence. The flash memory module holds off CPU access during the reset sequence. Flash reads are possible when the hold is removed. Completion of the reset sequence is marked by setting CCIF which enables flash user commands.

If a reset occurs while any flash command is in progress, that command is immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed. Commands and operations do not automatically resume after exiting reset.

Chapter 25
Interrupt Multiplexer (INTMUX)

25.1 About this module

25.1.1 Introduction

The Interrupt Multiplexer (INTMUX) expands the number of peripherals that can interrupt the core via 4 channels of the NVIC module.

- The INTMUX module has 4 channels that are assigned to 4 NVIC interrupt slots.
- Each INTMUX channel can provide up to 32 additional interrupt sources, which can be logically OR'd or ANDed.
- The Interrupt Multiplexer (INTMUX) routes the interrupt sources to the interrupt outputs.

25.1.2 Features

INTMUX features:

- Supports 4 multiplex channels
- Each channel receives 32 interrupt sources and has 1 interrupt output
- Each interrupt source can be enabled or disabled
- Each channel supports *Logic AND* or *Logic OR* of all enabled interrupt sources

25.1.3 Block diagram

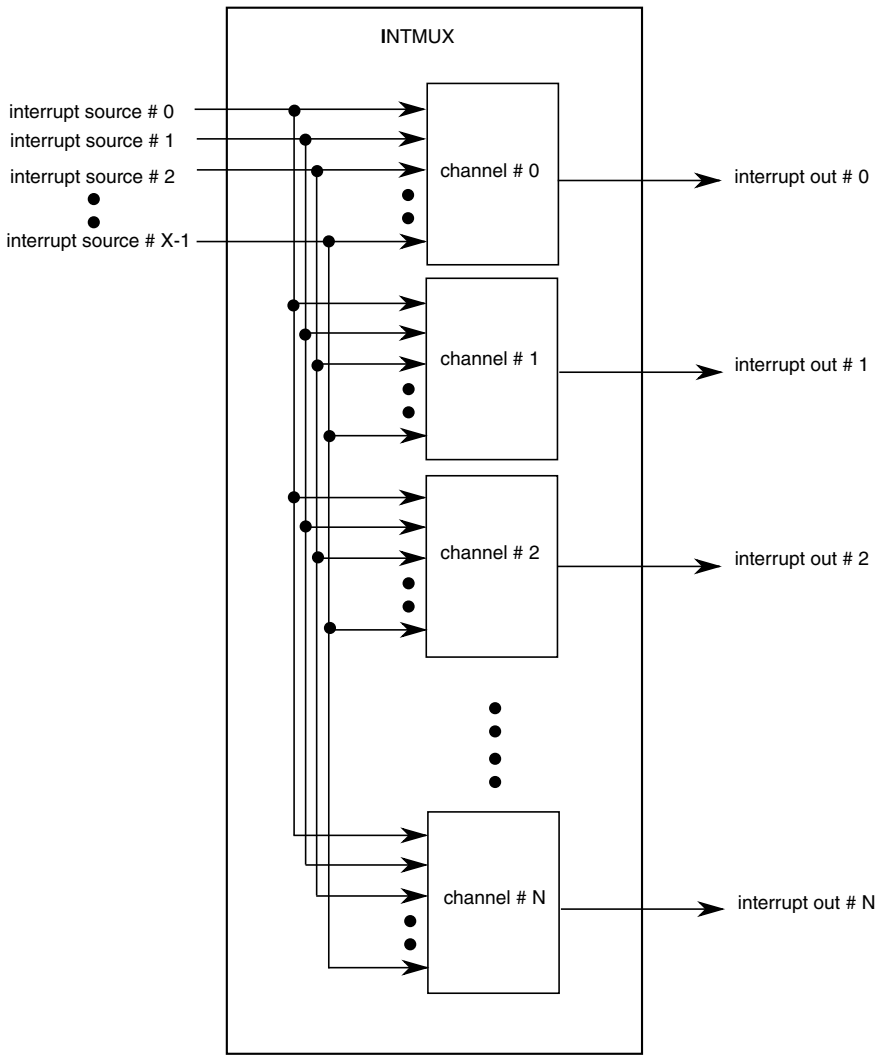


Figure 25-1. INTMUX Block diagram

N: Interrupt Channel Instance Number (N=3)

X: Interrupt Source Number for each channel (X=32)

25.2 Memory Map and register definition

This section includes the module memory map and detailed descriptions of all registers.

INTMUX memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4000	Channel n Control Status Register (INTMUX0_CH0_CSR)	32	R/W	See section	25.2.1/653
4002_4004	Channel n Vector Number Register (INTMUX0_CH0_VEC)	32	R	0000_0000h	25.2.2/654
4002_4010	Channel n Interrupt Enable Register (INTMUX0_CH0_IER_31_0)	32	R/W	0000_0000h	25.2.3/655
4002_4020	Channel n Interrupt Pending Register (INTMUX0_CH0_IPR_31_0)	32	R	0000_0000h	25.2.4/655
4002_4040	Channel n Control Status Register (INTMUX0_CH1_CSR)	32	R/W	See section	25.2.1/653
4002_4044	Channel n Vector Number Register (INTMUX0_CH1_VEC)	32	R	0000_0000h	25.2.2/654
4002_4050	Channel n Interrupt Enable Register (INTMUX0_CH1_IER_31_0)	32	R/W	0000_0000h	25.2.3/655
4002_4060	Channel n Interrupt Pending Register (INTMUX0_CH1_IPR_31_0)	32	R	0000_0000h	25.2.4/655
4002_4080	Channel n Control Status Register (INTMUX0_CH2_CSR)	32	R/W	See section	25.2.1/653
4002_4084	Channel n Vector Number Register (INTMUX0_CH2_VEC)	32	R	0000_0000h	25.2.2/654
4002_4090	Channel n Interrupt Enable Register (INTMUX0_CH2_IER_31_0)	32	R/W	0000_0000h	25.2.3/655
4002_40A0	Channel n Interrupt Pending Register (INTMUX0_CH2_IPR_31_0)	32	R	0000_0000h	25.2.4/655
4002_40C0	Channel n Control Status Register (INTMUX0_CH3_CSR)	32	R/W	See section	25.2.1/653
4002_40C4	Channel n Vector Number Register (INTMUX0_CH3_VEC)	32	R	0000_0000h	25.2.2/654
4002_40D0	Channel n Interrupt Enable Register (INTMUX0_CH3_IER_31_0)	32	R/W	0000_0000h	25.2.3/655
4002_40E0	Channel n Interrupt Pending Register (INTMUX0_CH3_IPR_31_0)	32	R	0000_0000h	25.2.4/655

25.2.1 Channel n Control Status Register (INTMUXx_CH*n*_CSR)

Address: 4002_4000h base + 0h offset + (64d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IRQP	0														
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
R	0				CHIN				0		IRQN		0		AND	RST
W																
Reset	0	0	0	0	*	*	*	*	0	0	0	0	0	0	0	0

* Notes:

- CHIN field: The reset value of CHIN depends on the channel number.

INTMUXx_CHn_CSR field descriptions

Field	Description
31 IRQP	Channel Interrupt Request Pending This bit is available for polling purposes and represents the channel output. 0 No interrupt is pending. 1 The interrupt output of this channel is pending.
30–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 CHIN	Channel Instance Number These bits represent the instance number of this channel. N Channel N
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 IRQN	Channel Input Number These bits represent the interrupt input number of this channel. 00 32 interrupt inputs 01 Reserved 10 Reserved 11 Reserved
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 AND	Logic AND This bit used to Logic AND or Logic OR all enabled interrupt inputs of this channel and generate the interrput output of this channel. 0 Logic OR all enabled interrupt inputs. 1 Logic AND all enabled interrupt inputs.
0 RST	Software Reset This bit is used to software reset this channel. This bit always reads as 0. 0 No operation. 1 Perform a software reset on this channel.

25.2.2 Channel n Vector Number Register (INTMUXx_CHn_VEC)

Address: 4002_4000h base + 4h offset + (64d × i), where i=0d to 3d

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

INTMUXx_CHn_VEC field descriptions

Field	Description
31–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–2 VECN	Vector Number These bits represent the vector number for the highest priority interrupt source of this channel. VECN[13:2] = (CPU vectors + NVIC vectors) + Number of the highest priority interrupt source of this channel. **NOTE:** The interrupt priority is based on the Interrupt Pending Register. The smaller the interrupt source number, the higher the priority.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

25.2.3 Channel n Interrupt Enable Register (INTMUXx_CHn_IER_31_0)

Address: 4002_4000h base + 10h offset + (64d × i), where i=0d to 3d

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

INTMUXx_CHn_IER_31_0 field descriptions

Field	Description
INTE	Interrupt Enable These bits are used to enable the interrupt sources of this channel. 0 Interrupt is disabled. 1 Interrupt is enabled.

25.2.4 Channel n Interrupt Pending Register (INTMUXx_CHn_IPR_31_0)

Address: 4002_4000h base + 20h offset + (64d × i), where i=0d to 3d

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

INTMUX*x*_CH*n*_IPR_31_0 field descriptions

Field	Description
INTP	Interrupt Pending These bits represent the interrupt source pending status of this channel. 0 No interrupt. 1 Interrupt is pending.

25.3 Functional Description

The Interrupt Multiplexer (INTMUX) routes the interrupt sources to the interrupt outputs.

There is one interrupt output for each channel available in the INTMUX. All interrupt sources are available to each channel. There are 4 multiplex channels, and there are 32 interrupt sources.

The INTMUX module provides interrupt status registers to monitor interrupt pending status and vector numbers. This module also implements the ability to logical AND or OR enabled interrupts on a given channel. Software resets can also be performed on a given channel.

See the chip specific Interrupt section for more information regarding the interrupt vectors and sources.

25.3.1 Configuring Output Channels

There is a single set of 32 source interrupts for the INTMUX. Each of the 4 output channels can be configured to output any combination of the source interrupts.

To enable source interrupts on a channel, write a mask value to the Channel n Interrupt Enable Register (CHn_IER_31_0). Each bit of these registers corresponds to one of the 32 source interrupts. Bit n enables source interrupt n on the given channel.

Further control for each output channel is available by selecting the Boolean operation to perform on pending interrupts. The default is logical OR, but logical AND can be selected via the CHn_CSR[AND] register bit.

25.3.2 INTMUX Vectors

The INTMUX is designed so that it integrates with the standard NVIC vector table. Each INTMUX source interrupt has a vector associated with it. These source vectors are intended to immediately follow the NVIC vectors in the vector table pointed to by the CPU's VTOR register. Interrupt service routine addresses can be placed in the INTMUX source vectors, just like NVIC vectors.

The ISR for each of the INTMUX output channels should read the CHn_VEC register to determine the highest priority active source interrupt out of those enabled for the given channel. The CHn_VEC[VECN] bitfield returns the source interrupt number. The VECN field starts at bit 2 of the CHn_VEC register, so reading the register as a whole will return the VECN field multiplied by 4. This is the offset in bytes of the active source vector from the start of the vector table (i.e., VTOR).

If multiple source interrupts become pending simultaneously on a single channel, the CHn_VEC[VECN] field will read as the highest priority vector number. Priority of source interrupts is determined by the source interrupt number. Lower numbers are higher priortity, with source interrupt 0 being highest priority.

NOTE

Unlike the NVIC, the INTMUX does not latch pending source interrupts. This means that the INTMUX output channel ISRs must check for and handle a 0 value of the CHn_VEC register to account for spurious interrupts.

Chapter 26
Low-Leakage Wakeup Unit (LLWU)

26.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The LLWU module allows the user to select up to 32 external pins and up to 8 internal modules as interrupt wake-up sources from low-leakage power modes. It also supports up to 8 internal modules as temporary DMA wake-up sources.

The input sources are described in the device's chip configuration details. Each of the available wake-up sources can be individually enabled.

The $\overline{\text{RESET}}$ pin is an additional source for triggering an exit from low-leakage power modes, and causes the MCU to exit both LLS and VLLS through a reset flow.

The LLWU module also includes four optional digital pin filters for the external wakeup pins.

See AN4503: Power Management for Kinetis MCUs for further details on using the LLWU.

26.1.1 Features

The LLWU module features include:

- Support for up to 32 external input pins and up to 8 internal modules with individual enable bits for MCU interrupt from low leakage modes
- Support for up to 8 internal modules with individual enable bits for DMA wakeup from low leakage modes

- Input sources may be external pins or from internal peripherals capable of running in LLS or VLLS. See the chip configuration information for wakeup input sources for this device.
- External pin wake-up inputs, each of which is programmable as falling-edge, rising-edge, or any change
- Wake-up inputs that are activated after MCU enters a low-leakage power mode
- Optional digital filters provided to qualify an external pin detect. Note that when the LPO clock is disabled, the filters are disabled and bypassed.

26.1.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from LLS, the LLWU is immediately disabled. After recovery from VLLS, the LLWU continues to detect wake-up events until the user has acknowledged the wake-up via a write to PMC_REGSC[ACKISO].

26.1.2.1 LLS mode

Wake-up events due to external pin inputs (LLWU_Px) and internal module interrupt inputs (LLWU_MxIF) result in an interrupt flow when exiting LLS. Wake-up events due to internal module DMA requests (LLWU_MxDF) allow the system to temporarily exit LLS with the CPU clock disabled, allowing the DMA request to be serviced. When the module DMA request negates, the system will re-enter LLS.

NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit Stop mode on an LLS recovery.

26.1.2.2 VLLS modes

All wakeup and reset events result in VLLS exit via a reset flow.

26.1.2.3 Non-low leakage modes

The LLWU is not active in all non-low leakage modes where detection and control logic are in a static state. The LLWU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

When the wake-up pin filters are enabled, filter operation begins immediately. If a low leakage mode is entered within five LPO clock cycles of an active edge, the edge event will be detected by the LLWU.

26.1.2.4 Debug mode

When the chip is in Debug mode and then enters LLS or a VLLSx mode, no debug logic works in the fully-functional low-leakage mode. Upon an exit from the LLS or VLLSx mode, the LLWU becomes inactive.

26.1.3 Block diagram

The following figure is the block diagram for the LLWU module.

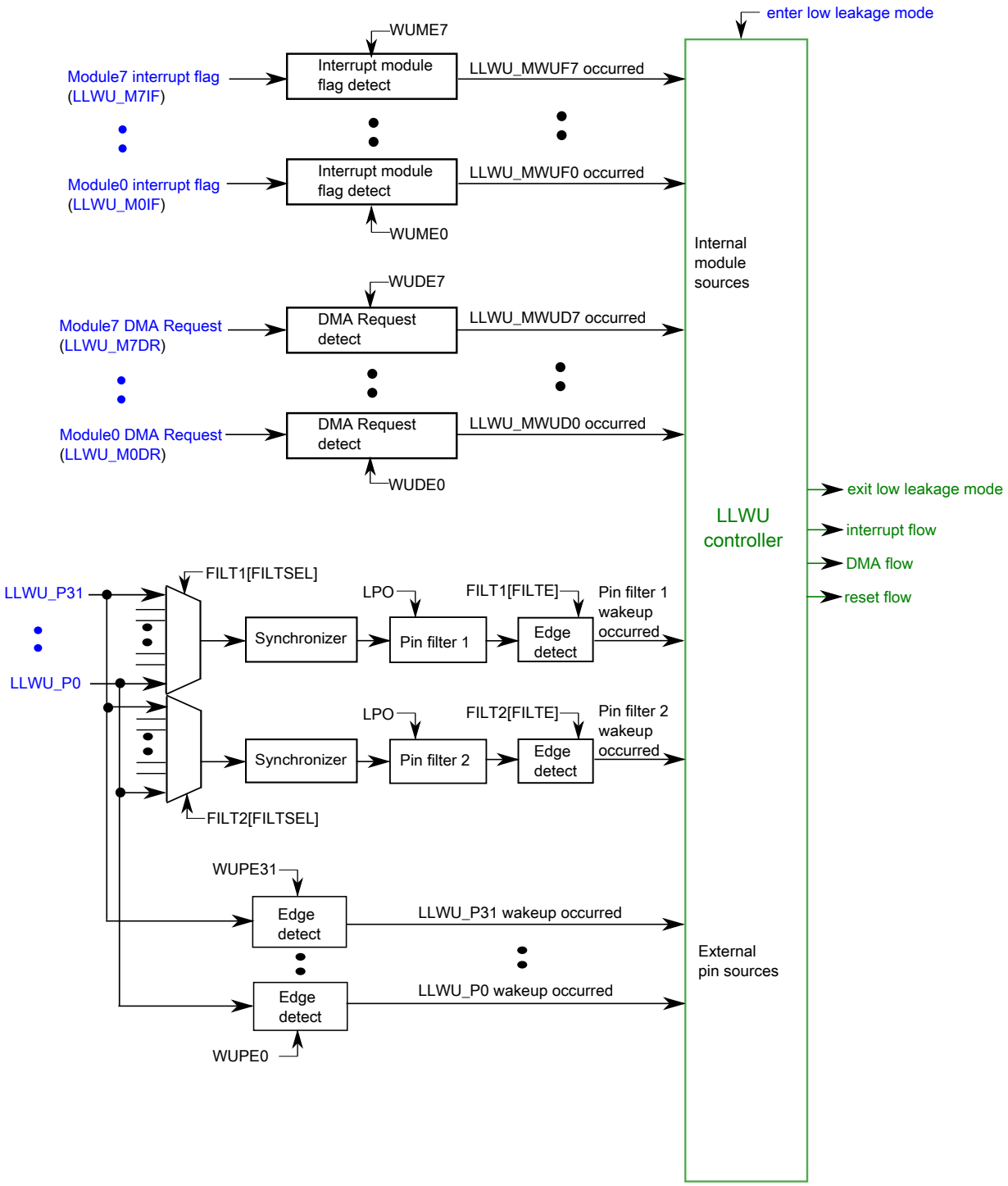


Figure 26-1. LLWU block diagram

26.2 LLWU signal descriptions

The signal properties of LLWU are shown in the table found here.

The external wakeup input pins can be enabled to detect either rising-edge, falling-edge, or on any change.

Table 26-1. LLWU signal descriptions

Signal	Description	I/O
LLWU_Pn	Wakeup inputs (n = 0- 31)	I

26.3 Memory map/register definition

The LLWU includes the following registers:

- Wake-up source enable registers
  - Enable external pin input sources
  - Enable internal peripheral interrupt sources
  - Enable internal peripheral DMA sources
- Wake-up flag registers
  - Indication of wakeup source that caused exit from a low-leakage power mode includes external pin or internal module interrupt
- Wake-up pin filter enable registers

NOTE

The LLWU registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

Registers for Pin/Module sources marked "Reserved" in the LLWU chip configuration information will not be writable and will return 0 when read.

All LLWU registers are reset by Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. Each register's displayed reset value represents this subset of reset types. LLWU registers are unaffected by reset types that do not trigger Chip Reset not VLLS. For more information about the types of reset on this chip, refer to the Introduction details.

LLWU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_1000	Version ID Register (LLWU0_VERID)	32	R	0100_0000h	26.3.1/664
4006_1004	Parameter Register (LLWU0_PARAM)	32	R	2008_0804h	26.3.2/665
4006_1008	LLWU Pin Enable 1 register (LLWU0_PE1)	32	R/W	0000_0000h	26.3.3/665
4006_100C	LLWU Pin Enable 2 register (LLWU0_PE2)	32	R/W	0000_0000h	26.3.4/668
4006_1018	LLWU Module Interrupt Enable register (LLWU0_ME)	32	R/W	0000_0000h	26.3.5/671
4006_101C	LLWU Module DMA Enable register (LLWU0_DE)	32	R/W	0000_0000h	26.3.6/673
4006_1020	LLWU Pin Flag register (LLWU0_PF)	32	R/W	0000_0000h	26.3.7/675
4006_1028	LLWU Module Interrupt Flag register (LLWU0_MF)	32	R	0000_0000h	26.3.8/680
4006_1030	LLWU Pin Filter register (LLWU0_FILT)	32	R/W	0000_0000h	26.3.9/682

26.3.1 Version ID Register (LLWUx_VERID)

Address: 4006_1000h base + 0h offset = 4006_1000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LLWUx_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0000 Standard features implemented

26.3.2 Parameter Register (LLWUx_PARAM)

Address: 4006_1000h base + 4h offset = 4006_1004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PINS								MODULES								DMAS								FILTERS							
W																																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

LLWUx_PARAM field descriptions

Field	Description
31–24 PINS	Pin Number Number of Pin wakeup sources supported
23–16 MODULES	Module Number Number of Module wakeup sources supported
15–8 DMAS	DMA Number Number of DMA wakeup sources supported
FILTERS	Filter Number Number of Pin Filters implemented

26.3.3 LLWU Pin Enable 1 register (LLWUx_PE1)

LLWU_PE1 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P15-LLWU_P0.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4006_1000h base + 8h offset = 4006_1008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WUPE15		WUPE14		WUPE13		WUPE12		WUPE11		WUPE10		WUPE9		WUPE8	
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
R W	WUPE7		WUPE6		WUPE5		WUPE4		WUPE3		WUPE2		WUPE1		WUPE0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LLWUx_PE1 field descriptions

Field	Description
31–30 WUPE15	Wakeup Pin Enable For LLWU_P15 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
29–28 WUPE14	Wakeup Pin Enable For LLWU_P14 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
27–26 WUPE13	Wakeup Pin Enable For LLWU_P13 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
25–24 WUPE12	Wakeup Pin Enable For LLWU_P12 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
23–22 WUPE11	Wakeup Pin Enable For LLWU_P11 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
21–20 WUPE10	Wakeup Pin Enable For LLWU_P10 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
19–18 WUPE9	Wakeup Pin Enable For LLWU_P9 Enables and configures the edge detection for the wakeup pin.

Table continues on the next page...

LLWUx_PE1 field descriptions (continued)

Field	Description
	00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
17–16 WUPE8	Wakeup Pin Enable For LLWU_P8 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
15–14 WUPE7	Wakeup Pin Enable For LLWU_P7 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
13–12 WUPE6	Wakeup Pin Enable For LLWU_P6 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
11–10 WUPE5	Wakeup Pin Enable For LLWU_P5 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
9–8 WUPE4	Wakeup Pin Enable For LLWU_P4 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
7–6 WUPE3	Wakeup Pin Enable For LLWU_P3 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection

Table continues on the next page...

LLWUx_PE1 field descriptions (continued)

Field	Description
	10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE2	Wakeup Pin Enable For LLWU_P2 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE1	Wakeup Pin Enable For LLWU_P1 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE0	Wakeup Pin Enable For LLWU_P0 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

26.3.4 LLWU Pin Enable 2 register (LLWUx_PE2)

LLWU_PE2 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P31-LLWU_P16.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4006_1000h base + Ch offset = 4006_100Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WUPE31		WUPE30		WUPE29		WUPE28		WUPE27		WUPE26		WUPE25		WUPE24	
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
R W	WUPE23		WUPE22		WUPE21		WUPE20		WUPE19		WUPE18		WUPE17		WUPE16	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LLWUx_PE2 field descriptions

Field	Description
31–30 WUPE31	Wakeup Pin Enable For LLWU_P31 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
29–28 WUPE30	Wakeup Pin Enable For LLWU_P30 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
27–26 WUPE29	Wakeup Pin Enable For LLWU_P29 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
25–24 WUPE28	Wakeup Pin Enable For LLWU_P28 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
23–22 WUPE27	Wakeup Pin Enable For LLWU_P27 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
21–20 WUPE26	Wakeup Pin Enable For LLWU_P26 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection

Table continues on the next page...

LLWUx_PE2 field descriptions (continued)

Field	Description
	10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
19–18 WUPE25	Wakeup Pin Enable For LLWU_P25 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
17–16 WUPE24	Wakeup Pin Enable For LLWU_P24 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
15–14 WUPE23	Wakeup Pin Enable For LLWU_P23 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
13–12 WUPE22	Wakeup Pin Enable For LLWU_P22 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
11–10 WUPE21	Wakeup Pin Enable For LLWU_P21 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
9–8 WUPE20	Wakeup Pin Enable For LLWU_P20 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

Table continues on the next page...

LLWUx_PE2 field descriptions (continued)

Field	Description
7–6 WUPE19	Wakeup Pin Enable For LLWU_P19 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE18	Wakeup Pin Enable For LLWU_P18 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE17	Wakeup Pin Enable For LLWU_P17 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE16	Wakeup Pin Enable For LLWU_P16 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

26.3.5 LLWU Module Interrupt Enable register (LLWUx_ME)

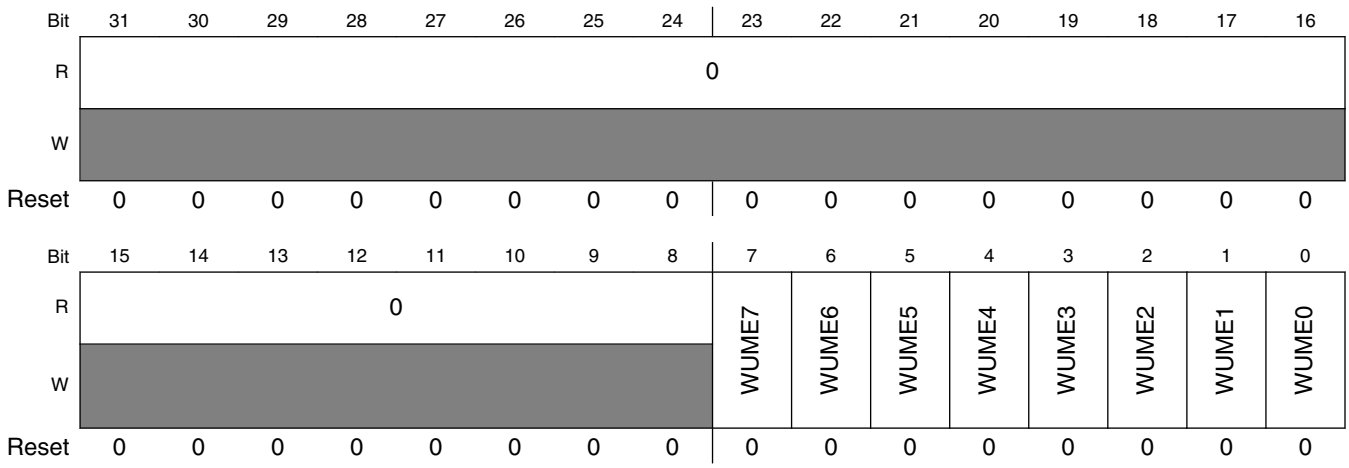
LLWU_ME contains the bits to enable an internal module interrupt as a wakeup source for inputs LLWU_M7IF - LLWU_M0IF.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Memory map/register definition

Address: 4006_1000h base + 18h offset = 4006_1018h



LLWUx_ME field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 WUME7	Wakeup Module Enable For Module 7 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
6 WUME6	Wakeup Module Enable For Module 6 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
5 WUME5	Wakeup Module Enable For Module 5 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
4 WUME4	Wakeup Module Enable For Module 4 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
3 WUME3	Wakeup Module Enable For Module 3 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
2 WUME2	Wakeup Module Enable For Module 2 Enables an internal module as a wakeup source input.

Table continues on the next page...

LLWUx_ME field descriptions (continued)

Field	Description
	0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
1 WUME1	Wakeup Module Enable for Module 1 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
0 WUME0	Wakeup Module Enable For Module 0 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source

26.3.6 LLWU Module DMA Enable register (LLWUx_DE)

LLWU_DE contains the bits to enable an internal module DMA request as a wakeup source for inputs LLWU_M7DR - LLWU_M0DR .

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4006_1000h base + 1Ch offset = 4006_101Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								WUDE7	WUDE6	WUDE5	WUDE4	WUDE3	WUDE2	WUDE1	WUDE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LLWUx_DE field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 WUDE7	DMA Wakeup Enable For Module 7 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
6 WUDE6	DMA Wakeup Enable For Module 6 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
5 WUDE5	DMA Wakeup Enable For Module 5 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
4 WUDE4	DMA Wakeup Enable For Module 4 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
3 WUDE3	DMA Wakeup Enable For Module 3 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
2 WUDE2	DMA Wakeup Enable For Module 2 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
1 WUDE1	DMA Wakeup Enable for Module 1 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source
0 WUDE0	DMA Wakeup Enable For Module 0 Enables an internal module as a DMA wakeup source. 0 Internal module request not used as a DMA wakeup source 1 Internal module request used as a DMA wakeup source

26.3.7 LLWU Pin Flag register (LLWUx_PF)

LLWU_PF contains the wakeup flags indicating which wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4006_1000h base + 20h offset = 4006_1020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WUF31	WUF30	WUF29	WUF28	WUF27	WUF26	WUF25	WUF24	WUF23	WUF22	WUF21	WUF20	WUF19	WUF18	WUF17	WUF16
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WUF15	WUF14	WUF13	WUF12	WUF11	WUF10	WUF9	WUF8	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LLWUx_PF field descriptions

Field	Description
31 WUF31	Wakeup Flag For LLWU_P31 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF31. 0 LLWU_P31 input was not a wakeup source 1 LLWU_P31 input was a wakeup source

Table continues on the next page...

LLWUx_PF field descriptions (continued)

Field	Description
30 WUF30	Wakeup Flag For LLWU_P30 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF30. 0 LLWU_P30 input was not a wakeup source 1 LLWU_P30 input was a wakeup source
29 WUF29	Wakeup Flag For LLWU_P29 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF29. 0 LLWU_P29 input was not a wakeup source 1 LLWU_P29 input was a wakeup source
28 WUF28	Wakeup Flag For LLWU_P28 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF28. 0 LLWU_P28 input was not a wakeup source 1 LLWU_P28 input was a wakeup source
27 WUF27	Wakeup Flag For LLWU_P27 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF27. 0 LLWU_P27 input was not a wakeup source 1 LLWU_P27 input was a wakeup source
26 WUF26	Wakeup Flag For LLWU_P26 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF26. 0 LLWU_P26 input was not a wakeup source 1 LLWU_P26 input was a wakeup source
25 WUF25	Wakeup Flag For LLWU_P25 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF25. 0 LLWU_P25 input was not a wakeup source 1 LLWU_P25 input was a wakeup source
24 WUF24	Wakeup Flag For LLWU_P24 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF24. 0 LLWU_P24 input was not a wakeup source 1 LLWU_P24 input was a wakeup source
23 WUF23	Wakeup Flag For LLWU_P23

Table continues on the next page...

LLWUx_PF field descriptions (continued)

Field	Description
	Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF23. 0 LLWU_P23 input was not a wakeup source 1 LLWU_P23 input was a wakeup source
22 WUF22	Wakeup Flag For LLWU_P22 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF22. 0 LLWU_P22 input was not a wakeup source 1 LLWU_P22 input was a wakeup source
21 WUF21	Wakeup Flag For LLWU_P21 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF21. 0 LLWU_P21 input was not a wakeup source 1 LLWU_P21 input was a wakeup source
20 WUF20	Wakeup Flag For LLWU_P20 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF20. 0 LLWU_P20 input was not a wakeup source 1 LLWU_P20 input was a wakeup source
19 WUF19	Wakeup Flag For LLWU_P19 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF19. 0 LLWU_P19 input was not a wakeup source 1 LLWU_P19 input was a wakeup source
18 WUF18	Wakeup Flag For LLWU_P18 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF18. 0 LLWU_P18 input was not a wakeup source 1 LLWU_P18 input was a wakeup source
17 WUF17	Wakeup Flag For LLWU_P17 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF17. 0 LLWU_P17 input was not a wakeup source 1 LLWU_P17 input was a wakeup source
16 WUF16	Wakeup Flag For LLWU_P16 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF16.

Table continues on the next page...

LLWUx_PF field descriptions (continued)

Field	Description
	0 LLWU_P16 input was not a wakeup source 1 LLWU_P16 input was a wakeup source
15 WUF15	Wakeup Flag For LLWU_P15 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF15. 0 LLWU_P15 input was not a wakeup source 1 LLWU_P15 input was a wakeup source
14 WUF14	Wakeup Flag For LLWU_P14 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF14. 0 LLWU_P14 input was not a wakeup source 1 LLWU_P14 input was a wakeup source
13 WUF13	Wakeup Flag For LLWU_P13 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF13. 0 LLWU_P13 input was not a wakeup source 1 LLWU_P13 input was a wakeup source
12 WUF12	Wakeup Flag For LLWU_P12 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF12. 0 LLWU_P12 input was not a wakeup source 1 LLWU_P12 input was a wakeup source
11 WUF11	Wakeup Flag For LLWU_P11 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF11. 0 LLWU_P11 input was not a wakeup source 1 LLWU_P11 input was a wakeup source
10 WUF10	Wakeup Flag For LLWU_P10 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF10. 0 LLWU_P10 input was not a wakeup source 1 LLWU_P10 input was a wakeup source
9 WUF9	Wakeup Flag For LLWU_P9 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF9. 0 LLWU_P9 input was not a wakeup source 1 LLWU_P9 input was a wakeup source

Table continues on the next page...

LLWUx_PF field descriptions (continued)

Field	Description
8 WUF8	Wakeup Flag For LLWU_P8 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF8. 0 LLWU_P8 input was not a wakeup source 1 LLWU_P8 input was a wakeup source
7 WUF7	Wakeup Flag For LLWU_P7 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF7. 0 LLWU_P7 input was not a wakeup source 1 LLWU_P7 input was a wakeup source
6 WUF6	Wakeup Flag For LLWU_P6 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF6. 0 LLWU_P6 input was not a wakeup source 1 LLWU_P6 input was a wakeup source
5 WUF5	Wakeup Flag For LLWU_P5 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF5. 0 LLWU_P5 input was not a wakeup source 1 LLWU_P5 input was a wakeup source
4 WUF4	Wakeup Flag For LLWU_P4 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF4. 0 LLWU_P4 input was not a wakeup source 1 LLWU_P4 input was a wakeup source
3 WUF3	Wakeup Flag For LLWU_P3 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF3. 0 LLWU_P3 input was not a wakeup source 1 LLWU_P3 input was a wakeup source
2 WUF2	Wakeup Flag For LLWU_P2 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF2. 0 LLWU_P2 input was not a wakeup source 1 LLWU_P2 input was a wakeup source
1 WUF1	Wakeup Flag For LLWU_P1

Table continues on the next page...

LLWUx_PF field descriptions (continued)

Field	Description
	Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF1. 0 LLWU_P1 input was not a wakeup source 1 LLWU_P1 input was a wakeup source
0 WUF0	Wakeup Flag For LLWU_P0 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF0. 0 LLWU_P0 input was not a wakeup source 1 LLWU_P0 input was a wakeup source

26.3.8 LLWU Module Interrupt Flag register (LLWUx_MF)

LLWU_MF contains the wakeup flags indicating which internal module interrupt caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

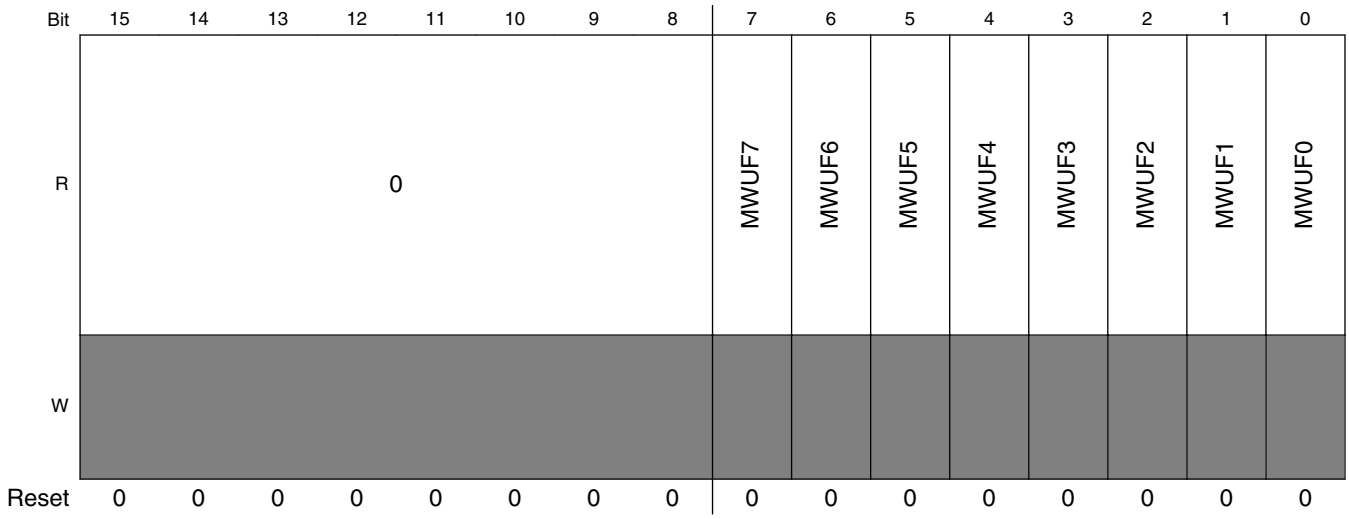
For internal peripherals that are capable of running in a low-leakage power mode, such as a real time clock module or CMP module, the flag from the associated peripheral is accessible as the MWUFx bit. The flag will need to be cleared in the peripheral instead of writing a 1 to the MWUFx bit.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4006_1000h base + 28h offset = 4006_1028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



LLWUx_MF field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 MWUF7	Wakeup flag For module 7 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 7 input was not a wakeup source 1 Module 7 input was a wakeup source
6 MWUF6	Wakeup flag For module 6 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 6 input was not a wakeup source 1 Module 6 input was a wakeup source
5 MWUF5	Wakeup flag For module 5 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 5 input was not a wakeup source 1 Module 5 input was a wakeup source
4 MWUF4	Wakeup flag For module 4 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 4 input was not a wakeup source 1 Module 4 input was a wakeup source
3 MWUF3	Wakeup flag For module 3 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.

Table continues on the next page...

LLWUx_MF field descriptions (continued)

Field	Description
	0 Module 3 input was not a wakeup source 1 Module 3 input was a wakeup source
2 MWUF2	Wakeup flag For module 2 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 2 input was not a wakeup source 1 Module 2 input was a wakeup source
1 MWUF1	Wakeup flag For module 1 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 1 input was not a wakeup source 1 Module 1 input was a wakeup source
0 MWUF0	Wakeup flag For module 0 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 0 input was not a wakeup source 1 Module 0 input was a wakeup source

26.3.9 LLWU Pin Filter register (LLWUx_FILT)

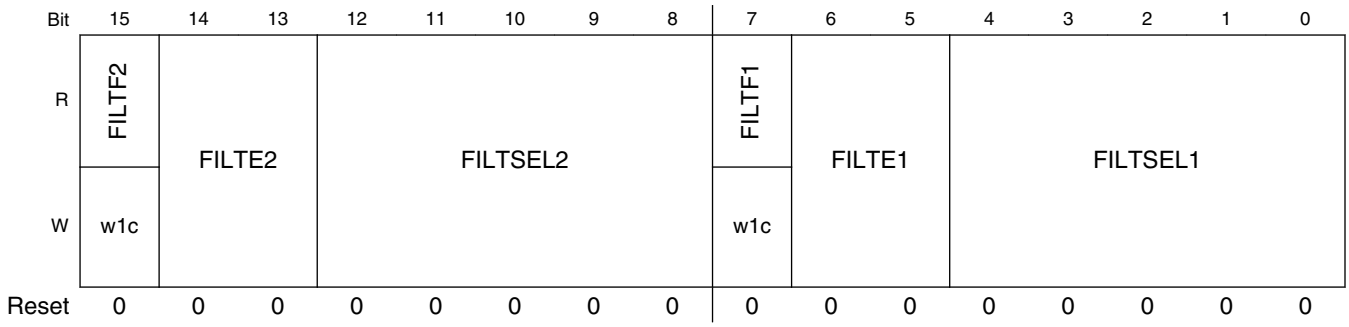
LLWU_FILT is a control and status register that is used to enable/disable the digital filter features for an external pin.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Introduction details for more information.

Address: 4006_1000h base + 30h offset = 4006_1030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FILTF4	FILTE4		FILTSEL4					FILTF3	FILTE3		FILTSEL3				
W	w1c								w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



LLWUx_FILT field descriptions

Field	Description
31 FILTF4	Filter 4 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
30–29 FILTE4	Filter 4 Enable Controls the digital filter 4 options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
28–24 FILTSEL4	Filter 4 Pin Select Selects 1 of the wakeup pins to be muxed into filter 4. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter
23 FILTF3	Filter 3 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
22–21 FILTE3	Filter 3 Enable Controls the digital filter 3 options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
20–16 FILTSEL3	Filter 3 Pin Select Selects 1 of the wakeup pins to be muxed into filter 3.

Table continues on the next page...

LLWUx_FILT field descriptions (continued)

Field	Description
	00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter
15 FILTF2	Filter 2 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
14–13 FILTE2	Filter 2 Enable Controls the digital filter 2 options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
12–8 FILTSEL2	Filter 2 Pin Select Selects 1 of the wakeup pins to be muxed into filter 2. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter
7 FILTF1	Filter 1 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
6–5 FILTE1	Filter 1 Enable Controls the digital filter 1 options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
FILTSEL1	Filter 1 Pin Select Selects 1 of the wakeup pins to be muxed into filter 1. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter

26.4 Functional description

Thie low-leakage wakeup unit (LLWU) module allows internal peripherals and external input pins as a source of wakeup from low-leakage modes.

It is operational only in LLS and VLLSx modes.

The LLWU module contains pin enables for each external pin and internal module. For each external pin, the user can disable or select the edge type for the wakeup with the following options:

- Falling-edge
- Rising-edge
- Either-edge

When an external pin is enabled as a wakeup source, the pin must be configured as an input pin.

The LLWU implements optional 3-cycle glitch filters, based on the LPO clock. A detected external pin is required to remain asserted until the enabled glitch filter times out. Additional latency of up to 2 cycles is due to synchronization, which results in a total of up to 5 cycles of delay before the detect circuit alerts the system to the wakeup or reset event when the filter function is enabled. Four wakeup detect filters are available for selected external pins. Glitch filtering is not provided on the internal modules.

For internal module interrupts, the WUMEx bit enables the associated module interrupt as a wakeup source.

For internal module DMA requests, the WUDEx bit enables the associated module DMA request as a temporary wakeup source.

26.4.1 LLS mode

Wakeup events triggered from either an external pin input or an internal module interrupt, result in a CPU interrupt flow to begin user code execution.

Wakeup events triggered from an internal module DMA request, result in a temporary wake-up from LLS with CPU remaining clock gated to allow the DMA request to be serviced. After the DMA request to the LLWU negates, the system will re-enter LLS mode.

26.4.2 VLLS modes

For any wakeup from VLLS, recovery is always via a reset flow and RCM_SRS[WAKEUP] is set indicating the low-leakage mode was active. State retention data is lost and I/O will be restored after PMC_REGSC[ACKISO] has been written.

A VLLS exit event due to $\overline{\text{RESET}}$ pin assertion causes an exit via a system reset. State retention data is lost and the I/O states immediately return to their reset state. The RCM_SRS[WAKEUP] and RCM_SRS[PIN] bits are set and the system executes a reset flow before CPU operation begins with a reset vector fetch.

26.4.3 Initialization

For an enabled peripheral wakeup input, the peripheral flag must be cleared by software before entering LLS or VLLSx mode to avoid an immediate exit from the mode.

Flags associated with external input pins, filtered and unfiltered, must also be cleared by software prior to entry to LLS or VLLSx mode.

After enabling an external pin filter or changing the source pin, wait at least five LPO clock cycles before entering LLS or VLLSx mode to allow the filter to initialize.

NOTE

After recovering from a VLLS mode, user must restore chip configuration before clearing PMC_REGSC[ACKISO]. In particular, pin configuration for enabled LLWU wake-up pins must be restored to avoid any LLWU flag from being falsely set when PMC_REGSC[ACKISO] is cleared.

The signal selected as a wake-up source pin must be a digital pin, as selected in the pin mux control.

Chapter 27
Low Power Inter-Integrated Circuit (LPI2C)

27.1 Introduction

27.1.1 Overview

The LPI2C is a low power Inter-Integrated Circuit (I2C) module that supports an efficient interface to an I2C bus as a master and/or a slave. The LPI2C can continue operating in stop modes provided an appropriate clock is available and is designed for low CPU overhead with DMA offloading of FIFO register accesses. The LPI2C implements logic support for standard-mode, fast-mode, fast-mode plus and ultra-fast modes of operation. The LPI2C module also complies with the System Management Bus (SMBus) Specification, version 2.

27.1.2 Features

The LPI2C supports the following features of the I2C specification:

- Standard, Fast, Fast+ and Ultra Fast modes are supported.
- HS-mode supported in slave mode.
- HS-mode supported for master mode, provided SCL pin implements current source pull-up (device specific).
- Multi-master support including synchronization and arbitration.
- Clock stretching.
- General call, 7-bit and 10-bit addressing.
- Software reset, START byte and Device ID require software support.

The LPI2C master supports the following features:

- Command/transmit FIFO of 4 words.
- Receive FIFO of 4 words.

- Command FIFO will wait for idle I2C bus before initiating transfer
- Command FIFO can initiate (repeated) START and STOP conditions and one or more master-receiver transfers.
- STOP condition can be generated from command FIFO or automatically when the transmit FIFO is empty.
- Host request input can be used to control the start time of an I2C bus transfer.
- Flexible receive data match can generate interrupt on data match and/or discard unwanted data.
- Flag and optional interrupt to signal Repeated START condition, STOP condition, loss of arbitration, unexpected NACK and command word errors.
- Supports configurable bus idle timeout and pin stuck low timeout.

The LPI2C slave supports the following features:

- Separate I2C slave registers to minimize software overhead due to master/slave switching.
- Support for 7-bit or 10-bit addressing, address range, SMBus alert and general call address.
- Transmit data register supporting interrupt or DMA requests.
- Receive data register supporting interrupt or DMA requests.
- Software controllable ACK or NACK, with optional clock stretching on ACK/NACK bit.
- Configurable clock stretching to avoid transmit FIFO underrun and receive FIFO overrun.
- Flag and optional interrupt at end of packet, STOP condition or bit error detection.

27.1.3 Block Diagram

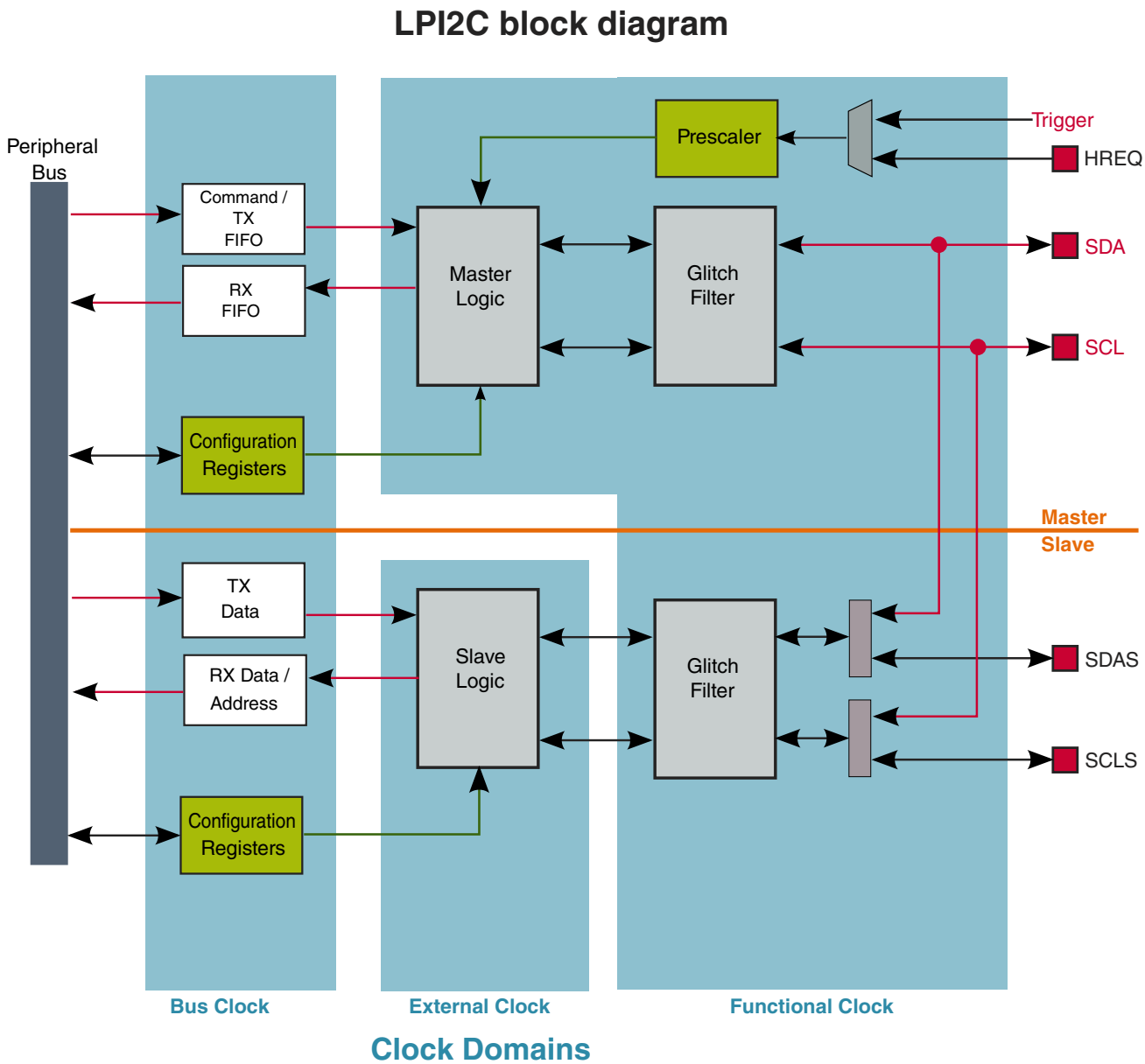


Figure 27-1. LPI2C block diagram

27.1.4 Modes of operation

The LPI2C module supports the chip modes described in the following table.

Table 27-1. Chip modes supported by the LPI2C module

Chip mode	LPI2C Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (MCR[DOZEN]) is set and the LPI2C is using an external or internal clock source which remains operating during stop/wait modes.
Low Leakage Stop	The Doze Enable (MCR[DOZEN]) bit is ignored and the LPI2C will wait for the current transfer to complete any pending operation before acknowledging low leakage mode entry.
Debug	Can continue operating provided the Debug Enable bit (MCR[DBGE]) is set.

27.1.5 Signal Descriptions

Signal	Description	I/O
SCL	LPI2C clock line. In 4-wire mode, this is the SCL input pin.	I/O
SDA	LPI2C data line. In 4-wire mode, this is the SDA input pin.	I/O
HREQ	Host request, can initiate an LPI2C master transfer if asserted and the I2C bus is idle.	I
SCLS	Secondary I2C clock line. In 4-wire mode, this is the SCL output pin. If LPI2C master/slave are configured to use separate pins, this the LPI2C slave SCL pin.	I/O
SDAS	Secondary I2C data line. In 4-wire mode, this is the SDA output pin. If LPI2C master/slave are configured to use separate pins, this the LPI2C slave SDA pin.	I/O

27.2 Memory Map and Registers

LPI2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Version ID Register (LPI2C_VERID)	32	R	See section	27.2.1/692
4	Parameter Register (LPI2C_PARAM)	32	R	See section	27.2.2/692
10	Master Control Register (LPI2C_MCR)	32	R/W	0000_0000h	27.2.3/693

Table continues on the next page...

LPI2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
14	Master Status Register (LPI2C_MSR)	32	R/W	0000_0001h	27.2.4/694
18	Master Interrupt Enable Register (LPI2C_MIER)	32	R/W	0000_0000h	27.2.5/696
1C	Master DMA Enable Register (LPI2C_MDER)	32	R/W	0000_0000h	27.2.6/698
20	Master Configuration Register 0 (LPI2C_MCFGR0)	32	R/W	0000_0000h	27.2.7/699
24	Master Configuration Register 1 (LPI2C_MCFGR1)	32	R/W	0000_0000h	27.2.8/700
28	Master Configuration Register 2 (LPI2C_MCFGR2)	32	R/W	0000_0000h	27.2.9/702
2C	Master Configuration Register 3 (LPI2C_MCFGR3)	32	R/W	0000_0000h	27.2.10/ 703
40	Master Data Match Register (LPI2C_MDMR)	32	R/W	0000_0000h	27.2.11/ 703
48	Master Clock Configuration Register 0 (LPI2C_MCCR0)	32	R/W	0000_0000h	27.2.12/ 704
50	Master Clock Configuration Register 1 (LPI2C_MCCR1)	32	R/W	0000_0000h	27.2.13/ 705
58	Master FIFO Control Register (LPI2C_MFCR)	32	R/W	0000_0000h	27.2.14/ 706
5C	Master FIFO Status Register (LPI2C_MFSR)	32	R	0000_0000h	27.2.15/ 706
60	Master Transmit Data Register (LPI2C_MTDR)	32	W	0000_0000h	27.2.16/ 707
70	Master Receive Data Register (LPI2C_MRDR)	32	R	0000_4000h	27.2.17/ 708
110	Slave Control Register (LPI2C_SCR)	32	R/W	0000_0000h	27.2.18/ 709
114	Slave Status Register (LPI2C_SSR)	32	R/W	0000_0000h	27.2.19/ 710
118	Slave Interrupt Enable Register (LPI2C_SIER)	32	R/W	0000_0000h	27.2.20/ 713
11C	Slave DMA Enable Register (LPI2C_SDER)	32	R/W	0000_0000h	27.2.21/ 714
124	Slave Configuration Register 1 (LPI2C_SCFGR1)	32	R/W	0000_0000h	27.2.22/ 715
128	Slave Configuration Register 2 (LPI2C_SCFGR2)	32	R/W	0000_0000h	27.2.23/ 717
140	Slave Address Match Register (LPI2C_SAMR)	32	R/W	0000_0000h	27.2.24/ 718
150	Slave Address Status Register (LPI2C_SASR)	32	R	0000_4000h	27.2.25/ 719
154	Slave Transmit ACK Register (LPI2C_STAR)	32	R/W	0000_0000h	27.2.26/ 720
160	Slave Transmit Data Register (LPI2C_STDR)	32	W	0000_0000h	27.2.27/ 720

Table continues on the next page...

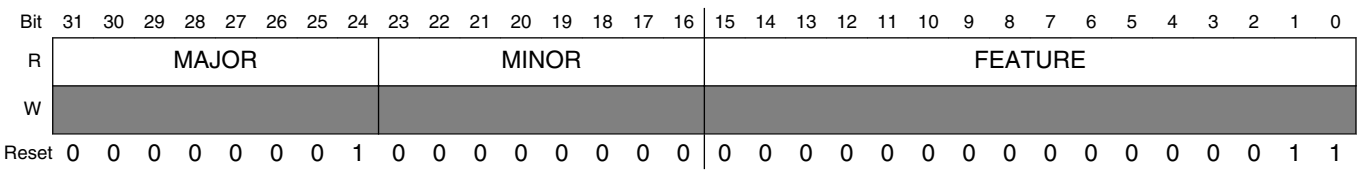
LPI2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
170	Slave Receive Data Register (LPI2C_SRDR)	32	R	0000_4000h	27.2.28/ 721

27.2.1 Version ID Register (LPI2C_VERID)

.

Address: 0h base + 0h offset = 0h



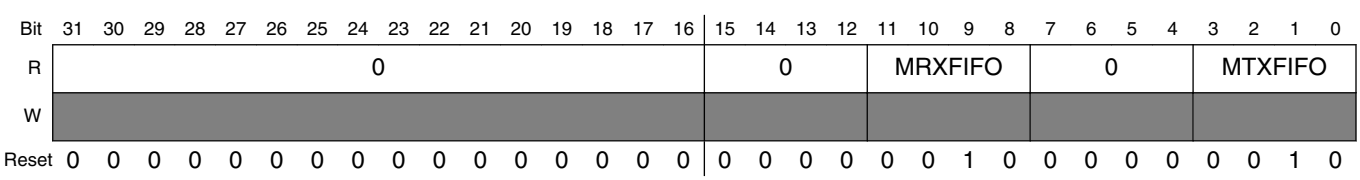
LPI2C_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0002 Master only with standard feature set. 0x0003 Master and slave with standard feature set.

27.2.2 Parameter Register (LPI2C_PARAM)

.

Address: 0h base + 4h offset = 4h



LPI2C_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 MRXFIFO	Master Receive FIFO Size The number of words in the master receive FIFO is 2^MRXFIFO.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MTXFIFO	Master Transmit FIFO Size The number of words in the master transmit FIFO is 2^MTXFIFO.

27.2.3 Master Control Register (LPI2C_MCR)

Address: 0h base + 10h offset = 10h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0						0	0	0				DBGEN	DOZEN	RST	MEN
W							RRF	RTF								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2C_MCR field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

LPI2C_MCR field descriptions (continued)

Field	Description
9 RRF	Reset Receive FIFO 0 No effect. 1 Receive FIFO is reset.
8 RTF	Reset Transmit FIFO 0 No effect. 1 Transmit FIFO is reset.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 DBGEN	Debug Enable 0 Master is disabled in debug mode. 1 Master is enabled in debug mode.
2 DOZEN	Doze mode enable Enables or disables Doze mode for the master. 0 Master is enabled in Doze mode. 1 Master is disabled in Doze mode.
1 RST	Software Reset Reset all internal master logic and registers, except the Master Control Register. Remains set until cleared by software. 0 Master logic is not reset. 1 Master logic is reset.
0 MEN	Master Enable 0 Master logic is disabled. 1 Master logic is enabled.

27.2.4 Master Status Register (LPI2C_MSR)

Address: 0h base + 14h offset = 14h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	MBF	0							
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
R	0	DMF	PLTF	FEF	ALF	NDF	SDF	EPF	0						RDF	TDF
W		w1c	w1c	w1c	w1c	w1c	w1c	w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

LPI2C_MSR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 BBF	Bus Busy Flag 0 I2C Bus is idle. 1 I2C Bus is busy.
24 MBF	Master Busy Flag 0 I2C Master is idle. 1 I2C Master is busy.
23–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields as configured by MATCFG. Received data that is discarded due to CMD field does not cause this flag to set. 0 Have not received matching data. 1 Have received matching data.
13 PLTF	Pin Low Timeout Flag Will set when the SCL and/or SDA input is low for more than PINLOW cycles, even when the LPI2C master is idle. Software is responsible for resolving the pin low condition. This flag cannot be cleared for as long as the pin low timeout continues and must be cleared before the LPI2C can initiate a START condition. 0 Pin low timeout has not occurred or is disabled. 1 Pin low timeout has occurred.
12 FEF	FIFO Error Flag Detects an attempt to send or receive data without first generating a (repeated) START condition. This can occur if the transmit FIFO underflows when the AUTOSTOP bit is set. When this flag is set, the LPI2C master will send a STOP condition (if busy) and will not initiate a new START condition until this flag has been cleared. 0 No error. 1 Master sending or receiving data without START condition.
11 ALF	Arbitration Lost Flag This flag will set if the LPI2C master transmits a logic one and detects a logic zero on the I2C bus, or if it detects a START or STOP condition while it is transmitting data. When this flag sets, the LPI2C master will release the bus (go idle) and will not initiate a new START condition until this flag has been cleared. 0 Master has not lost arbitration. 1 Master has lost arbitration.
10 NDF	NACK Detect Flag This flag will set if the LPI2C master detects a NACK when transmitting an address or data. If a NACK is expected for a given address (as configured by the command word) then the flag will set if a NACK is not generated. When set, the master will transmit a STOP condition and will not initiate a new START condition until this flag has been cleared.

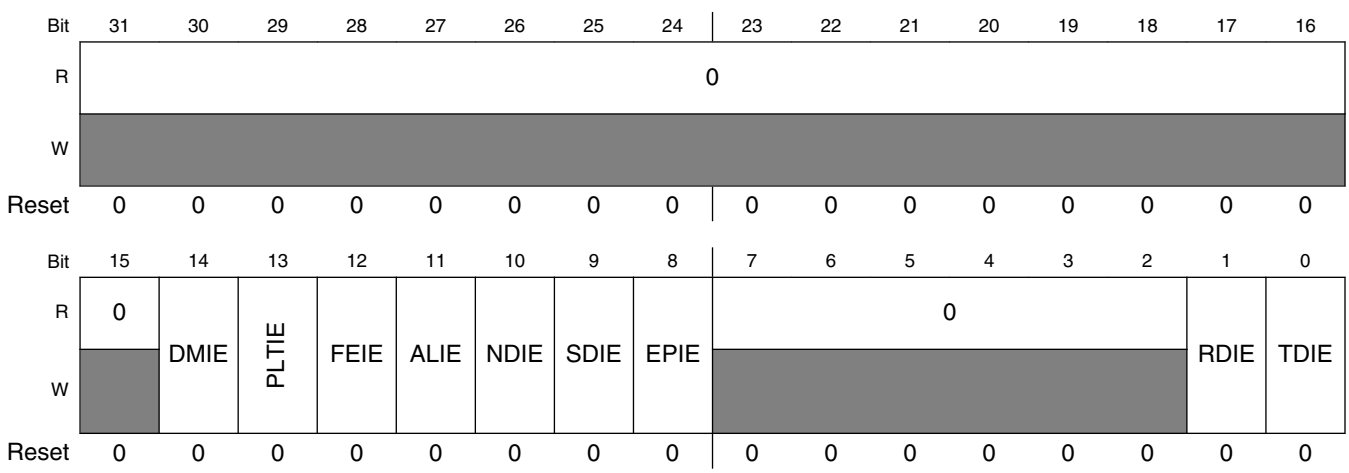
Table continues on the next page...

LPI2C_MSR field descriptions (continued)

Field	Description
	0 Unexpected NACK not detected. 1 Unexpected NACK was detected.
9 SDF	STOP Detect Flag This flag will set when the LPI2C master generates a STOP condition. 0 Master has not generated a STOP condition. 1 Master has generated a STOP condition.
8 EPF	End Packet Flag This flag will set when the LPI2C master generates either a repeated START or a STOP condition. It does not set when the master first generates a START condition. 0 Master has not generated a STOP or Repeated START condition. 1 Master has generated a STOP or Repeated START condition.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. 0 Receive Data is not ready. 1 Receive data is ready.
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. 0 Transmit data not requested. 1 Transmit data is requested.

27.2.5 Master Interrupt Enable Register (LPI2C_MIER)

Address: 0h base + 18h offset = 18h

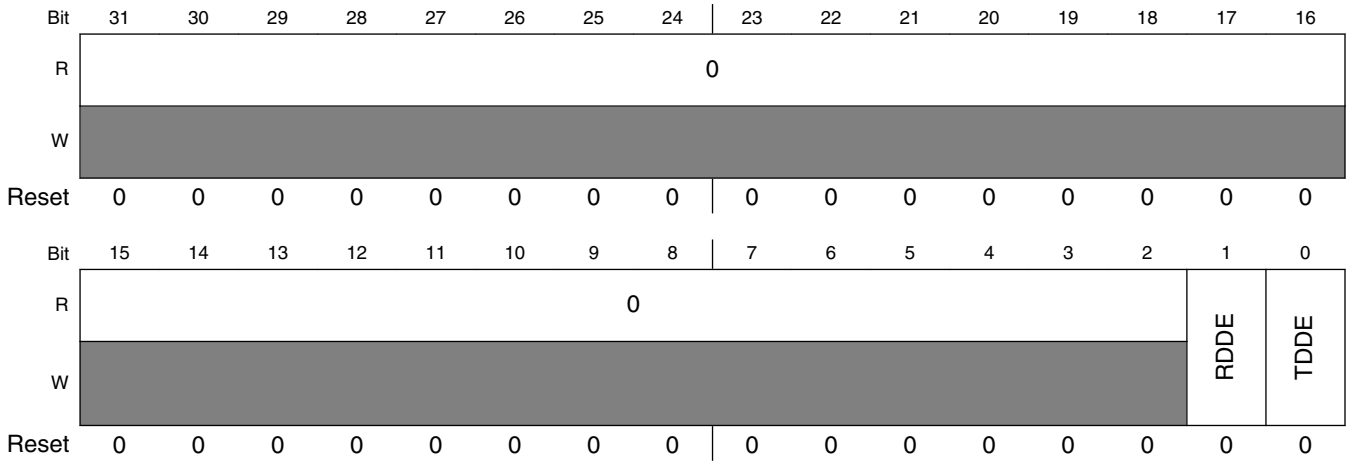


LPI2C_MIER field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DMIE	Data Match Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
13 PLTIE	Pin Low Timeout Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
12 FEIE	FIFO Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
11 ALIE	Arbitration Lost Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
10 NDIE	NACK Detect Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
9 SDIE	STOP Detect Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
8 EPIE	End Packet Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDIE	Receive Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
0 TDIE	Transmit Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled

27.2.6 Master DMA Enable Register (LPI2C_MDER)

Address: 0h base + 1Ch offset = 1Ch

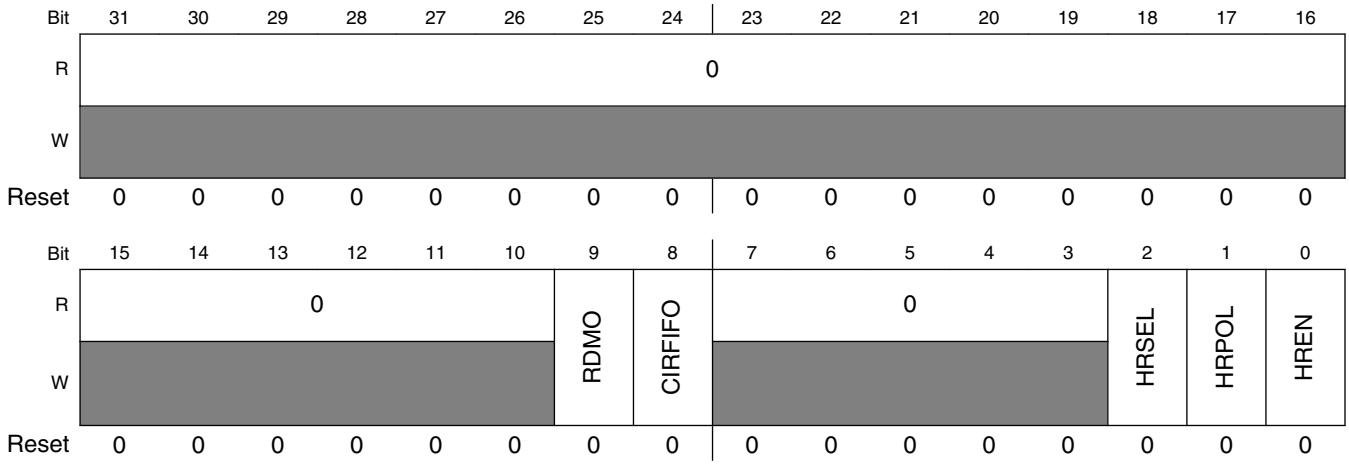


LPI2C_MDER field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDDE	Receive Data DMA Enable 0 DMA request disabled. 1 DMA request enabled.
0 TDDE	Transmit Data DMA Enable 0 DMA request disabled. 1 DMA request enabled

27.2.7 Master Configuration Register 0 (LPI2C_MCFGR0)

Address: 0h base + 20h offset = 20h



LPI2C_MCFGR0 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause DMF to set is discarded. Once DMF is set, the RDMO configuration is ignored. When disabling RDMO, clear RDMO before clearing DMF to ensure no receive data is lost. 0 Received data is stored in the receive FIFO as normal. 1 Received data is discarded unless the RMF is set.
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as normal, but once the LPI2C master is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This will cause the contents of the transmit FIFO to be cycled through repeatedly. If AUTOSTOP is set, a STOP condition will be sent whenever the transmit FIFO is empty and the read pointer is restored. 0 Circular FIFO is disabled. 1 Circular FIFO is enabled.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 HRSEL	Host Request Select Selects the source of the host request input. 0 Host request input is pin LPI2C_HREQ. 1 Host request input is input trigger.
1 HRPOL	Host Request Polarity

Table continues on the next page...

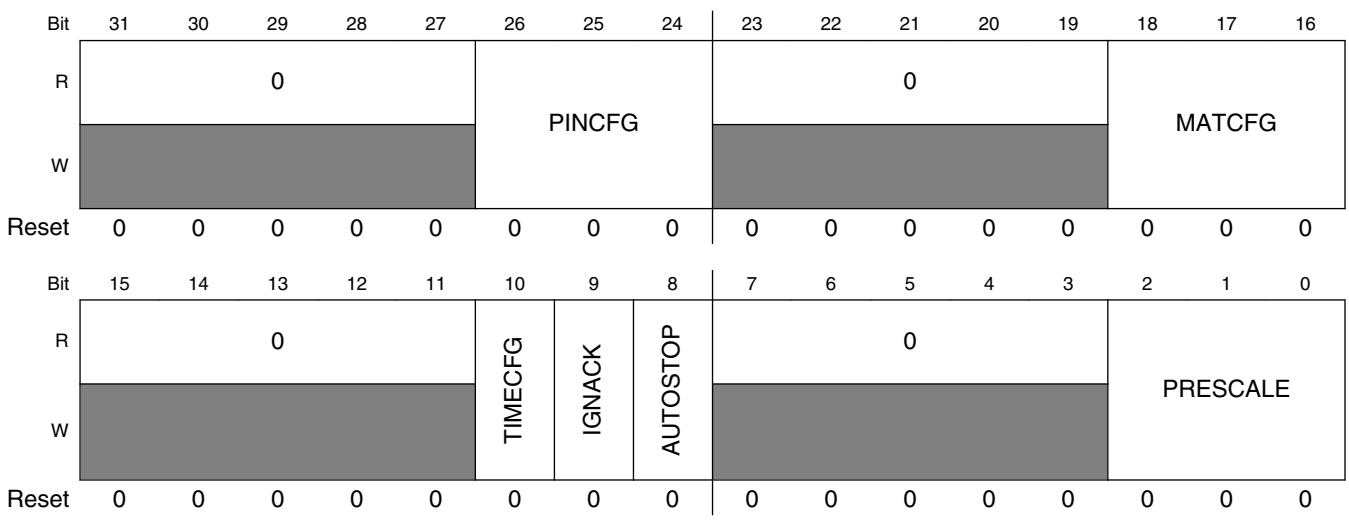
LPI2C_MCFGR0 field descriptions (continued)

Field	Description
	Configures the polarity of the host request input pin. 0 Active low. 1 Active high.
0 HREN	Host Request Enable When enabled, the LPI2C master will only initiate a START condition if the host request input is asserted and the bus is idle. A repeated START is not affected by the host request. 0 Host request input is disabled. 1 Host request input is enabled.

27.2.8 Master Configuration Register 1 (LPI2C_MCFGR1)

The MCFGR1 should only be written when the I2C Master is disabled.

Address: 0h base + 24h offset = 24h



LPI2C_MCFGR1 field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26–24 PINCFG	Pin Configuration Configures the pin mode. 000 LPI2C configured for 2-pin open drain mode. 001 LPI2C configured for 2-pin output only mode (ultra-fast mode). 010 LPI2C configured for 2-pin push-pull mode. 011 LPI2C configured for 4-pin push-pull mode.

Table continues on the next page...

LPI2C_MCFGR1 field descriptions (continued)

Field	Description
	100 LPI2C configured for 2-pin open drain mode with separate LPI2C slave. 101 LPI2C configured for 2-pin output only mode (ultra-fast mode) with separate LPI2C slave. 110 LPI2C configured for 2-pin push-pull mode with separate LPI2C slave. 111 LPI2C configured for 4-pin push-pull mode (inverted outputs).
23–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. 000 Match disabled. 001 Reserved. 010 Match enabled (1st data word equals MATCH0 OR MATCH1). 011 Match enabled (any data word equals MATCH0 OR MATCH1). 100 Match enabled (1st data word equals MATCH0 AND 2nd data word equals MATCH1). 101 Match enabled (any data word equals MATCH0 AND next data word equals MATCH1). 110 Match enabled (1st data word AND MATCH1 equals MATCH0 AND MATCH1). 111 Match enabled (any data word AND MATCH1 equals MATCH0 AND MATCH1).
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 TIMECFG	Timeout Configuration 0 Pin Low Timeout Flag will set if SCL is low for longer than the configured timeout. 1 Pin Low Timeout Flag will set if either SCL or SDA is low for longer than the configured timeout.
9 IGNACK	When set, the received NACK field is ignored and assumed to be ACK. This bit is required to be set in Ultra-Fast Mode. 0 LPI2C Master will receive ACK and NACK normally. 1 LPI2C Master will treat a received NACK as if it was an ACK.
8 AUTOSTOP	Automatic STOP Generation When enabled, a STOP condition is generated whenever the LPI2C master is busy and the transmit FIFO is empty. The STOP condition can also be generated using a transmit FIFO command. 0 No effect. 1 STOP condition is automatically generated whenever the transmit FIFO is empty and LPI2C master is busy.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C master logic, except the digital glitch filters. 000 Divide by 1. 001 Divide by 2. 010 Divide by 4. 011 Divide by 8. 100 Divide by 16. 101 Divide by 32.

Table continues on the next page...

LPI2C_MCFGR1 field descriptions (continued)

Field	Description
	110 Divide by 64. 111 Divide by 128.

27.2.9 Master Configuration Register 2 (LPI2C_MCFGR2)

The MCFGR2 should only be written when the I2C Master is disabled.

Address: 0h base + 28h offset = 28h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FILTSDA				0				FILTSCL				0				BUSIDLE											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

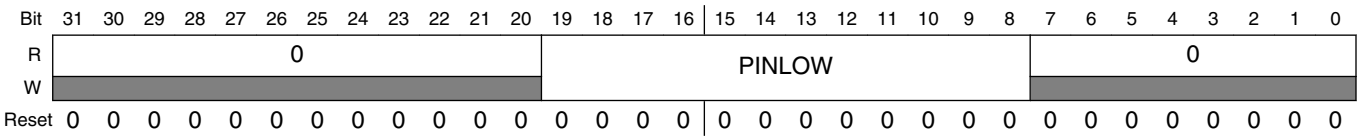
LPI2C_MCFGR2 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 FILTSDA	Glitch Filter SDA Configures the I2C master digital glitch filters for SDA input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSDA cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSDA cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration and is automatically bypassed in High Speed mode.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 FILTSCL	Glitch Filter SCL Configures the I2C master digital glitch filters for SCL input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSCL cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSCL cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration and is automatically bypassed in High Speed mode.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period in clock cycles. If both SCL and SDA are high for longer than BUSIDLE cycles, then the I2C bus is assumed to be idle and the master can generate a START condition. When set to zero, this feature is disabled.

27.2.10 Master Configuration Register 3 (LPI2C_MCFGR3)

The MCFGR3 should only be written when the I2C Master is disabled.

Address: 0h base + 2Ch offset = 2Ch

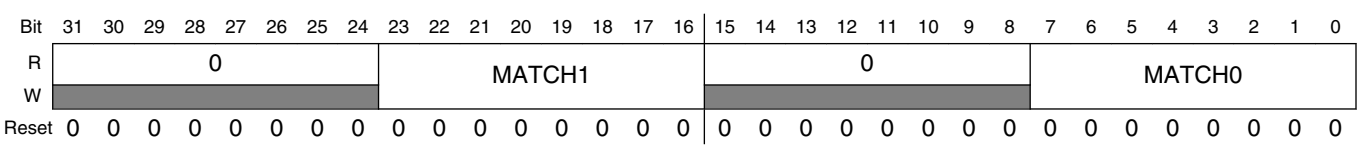


LPI2C_MCFGR3 field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–8 PINLOW	Pin Low Timeout Configures the pin low timeout flag in clock cycles. If SCL and/or SDA is low for longer than (PINLOW * 256) cycles then PLTF is set. When set to zero, this feature is disabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

27.2.11 Master Data Match Register (LPI2C_MDMR)

Address: 0h base + 40h offset = 40h



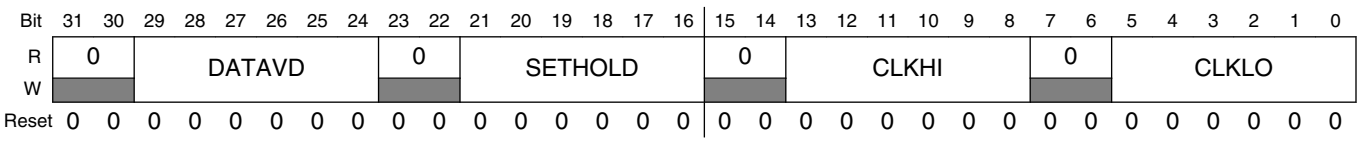
LPI2C_MDMR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 MATCH1	Match 1 Value Compared against the received data when receive data match is enabled.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

27.2.12 Master Clock Configuration Register 0 (LPI2C_MCCR0)

The MCCR0 cannot be changed when the I2C master is enabled and is used for standard, fast, fast-mode plus and ultra-fast transfers.

Address: 0h base + 48h offset = 48h



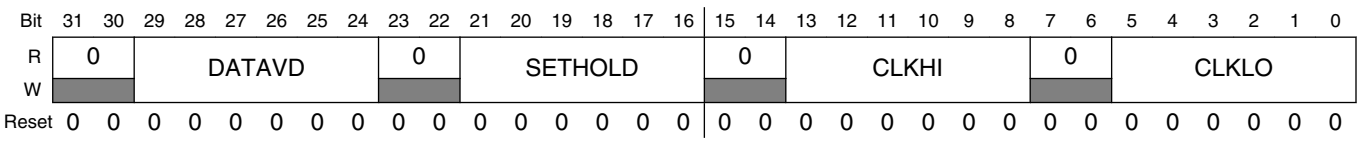
LPI2C_MCCR0 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master as the setup and hold time for a (repeated) START condition and setup time for a STOP condition. The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. This value is also used for the minimum bus free time between a STOP and a START condition.

27.2.13 Master Clock Configuration Register 1 (LPI2C_MCCR1)

The MCCR1 cannot be changed when the I2C master is enabled and is used for high speed mode transfers. The separate clock configuration for high speed mode allows arbitration to take place in Fast mode (with timing configured by MCCR0), before switching to high speed mode (with timing configured by MCCR1).

Address: 0h base + 50h offset = 50h

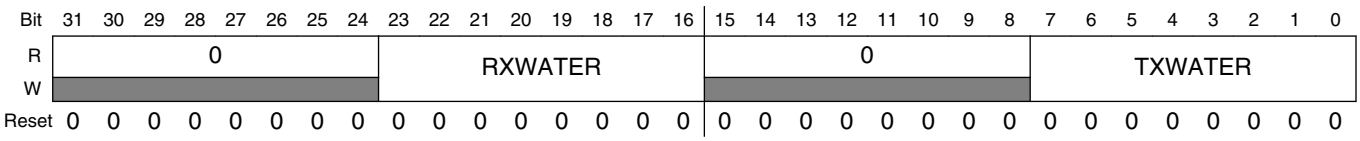


LPI2C_MCCR1 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master as the setup and hold time for a (repeated) START condition and setup time for a STOP condition. The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. This value is also used for the minimum bus free time between a STOP and a START condition.

27.2.14 Master FIFO Control Register (LPI2C_MFCR)

Address: 0h base + 58h offset = 58h

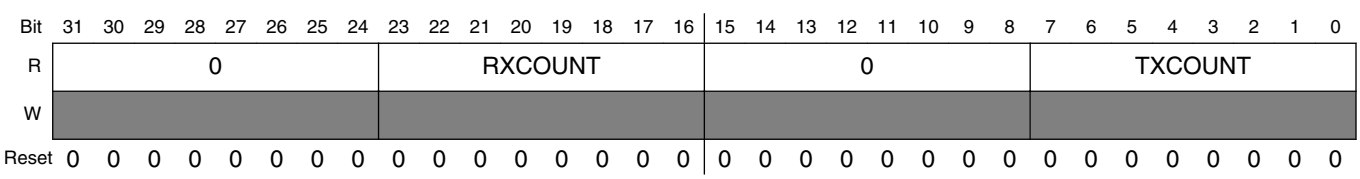


LPI2C_MFCR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size will be truncated.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size will be truncated.

27.2.15 Master FIFO Status Register (LPI2C_MFSR)

Address: 0h base + 5Ch offset = 5Ch



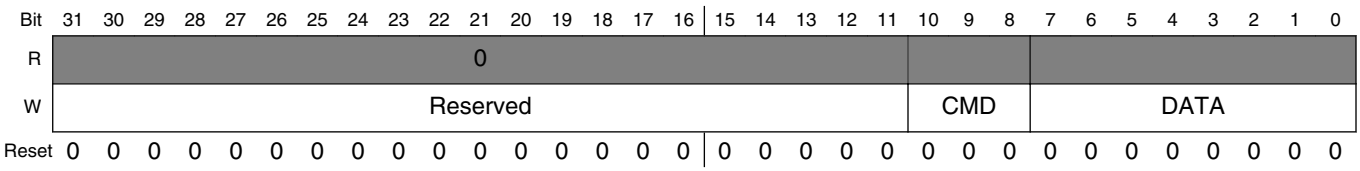
LPI2C_MFSR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

27.2.16 Master Transmit Data Register (LPI2C_MTDR)

An 8-bit write to the CMD field will store the data in the Command FIFO, but does not increment the FIFO write pointer. An 8-bit write to the DATA field will zero extend the CMD field unless the CMD field has been written separately since the last FIFO write, it also increments the FIFO write pointer. A 16-bit or 32-bit will write both the CMD and DATA fields and increment the FIFO.

Address: 0h base + 60h offset = 60h

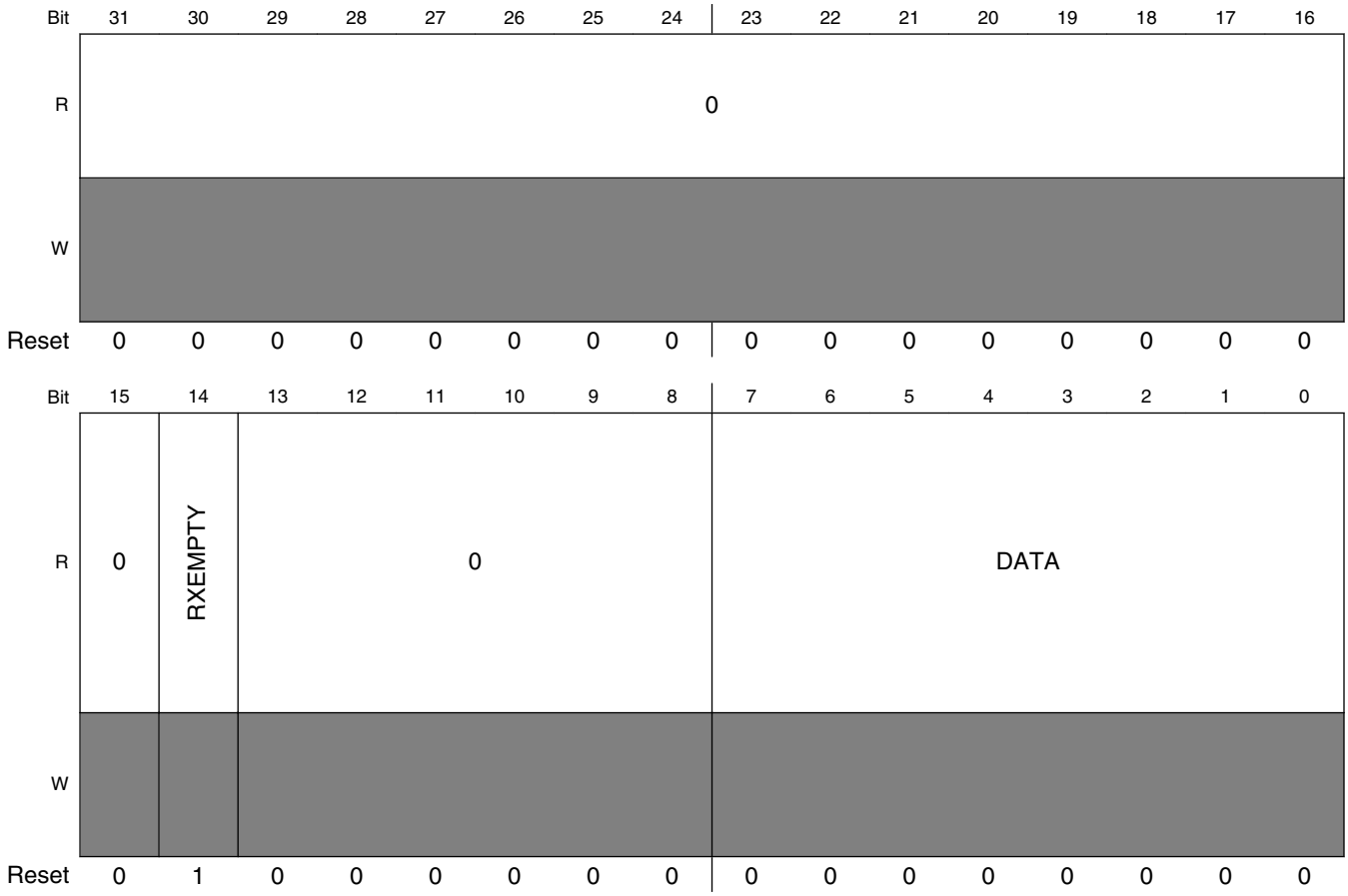


LPI2C_MTDR field descriptions

Field	Description
31–11 Reserved	This field is reserved.
10–8 CMD	Command Data 000 Transmit DATA[7:0]. 001 Receive (DATA[7:0] + 1) bytes. 010 Generate STOP condition. 011 Receive and discard (DATA[7:0] + 1) bytes. 100 Generate (repeated) START and transmit address in DATA[7:0]. 101 Generate (repeated) START and transmit address in DATA[7:0]. This transfer expects a NACK to be returned. 110 Generate (repeated) START and transmit address in DATA[7:0] using high speed mode. 111 Generate (repeated) START and transmit address in DATA[7:0] using high speed mode. This transfer expects a NACK to be returned.
DATA	Transmit Data Performing an 8-bit write to DATA will zero extend the CMD field.

27.2.17 Master Receive Data Register (LPI2C_MRDR)

Address: 0h base + 70h offset = 70h

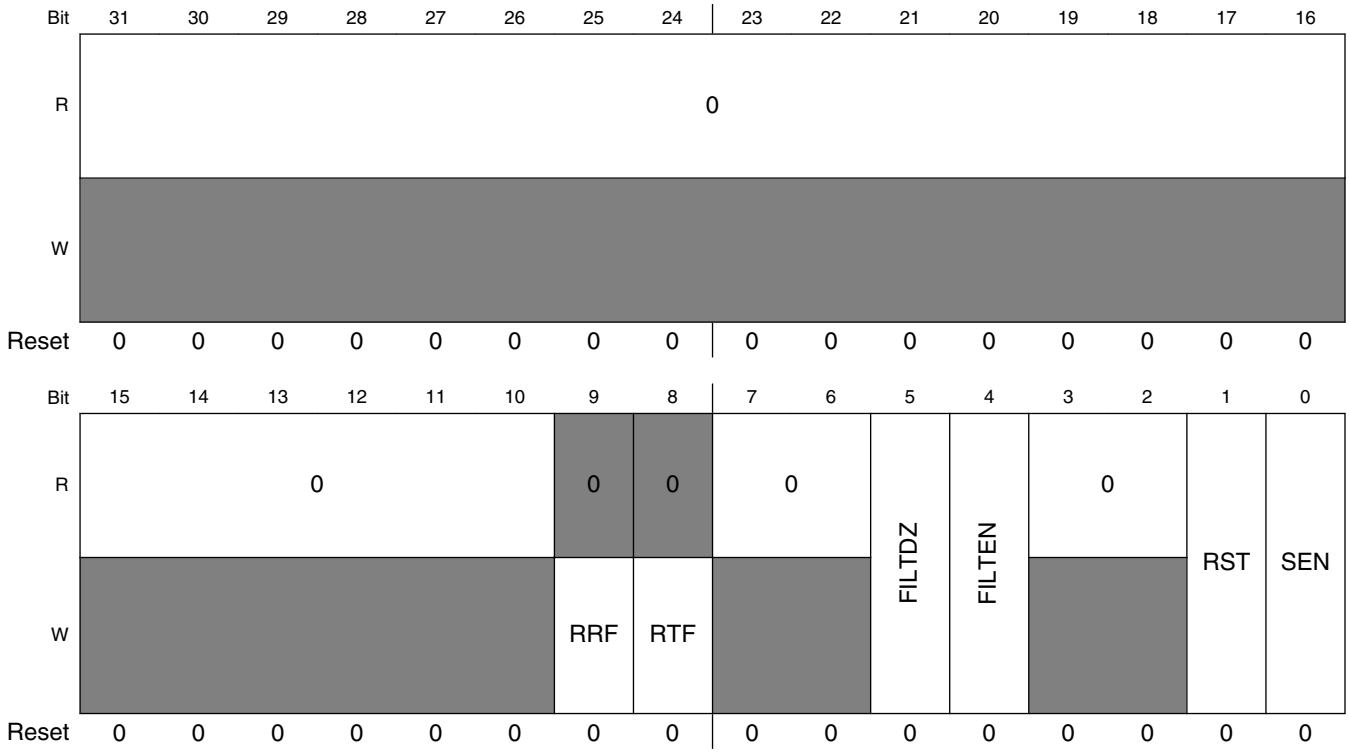


LPI2C_MRDR field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 RXEMPTY	RX Empty 0 Receive FIFO is not empty. 1 Receive FIFO is empty.
13–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA	Receive Data Reading this register returns the data received by the I2C master that has not been discarded. Receive data can be discarded due to the CMD field or the master can be configured to discard non-matching data.

27.2.18 Slave Control Register (LPI2C_SCR)

Address: 0h base + 110h offset = 110h



LPI2C_SCR field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RRF	Reset Receive FIFO 0 No effect. 1 Receive Data Register is now empty.
8 RTF	Reset Transmit FIFO 0 No effect. 1 Transmit Data Register is now empty.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 FILTDZ	Filter Doze Enable 0 Filter remains enabled in Doze mode. 1 Filter is disabled in Doze mode.
4 FILTEN	Filter Enable 0 Disable digital filter and output delay counter for slave mode. 1 Enable digital filter and output delay counter for slave mode.

Table continues on the next page...

LPI2C_SCR field descriptions (continued)

Field	Description
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RST	Software Reset 0 Slave logic is not reset. 1 Slave logic is reset.
0 SEN	Slave Enable 0 Slave mode is disabled. 1 Slave mode is enabled.

27.2.19 Slave Status Register (LPI2C_SSR)

Address: 0h base + 114h offset = 114h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	SBF	0							
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
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					w1c	w1c	w1c	w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2C_SSR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 BBF	Bus Busy Flag

Table continues on the next page...

LPI2C_SSR field descriptions (continued)

Field	Description
	0 I2C Bus is idle. 1 I2C Bus is busy.
24 SBF	Slave Busy Flag 0 I2C Slave is idle. 1 I2C Slave is busy.
23–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 SARF	SMBus Alert Response Flag This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 SMBus Alert Response disabled or not detected. 1 SMBus Alert Response enabled and detected.
14 GCF	General Call Flag This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 Slave has not detected the General Call Address or General Call Address disabled. 1 Slave has detected the General Call Address.
13 AM1F	Address Match 1 Flag Indicates that the received address has matched the ADDR1 field or ADDR0 to ADDR1 range as configured by ADDRCFG. This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 Have not received ADDR1 or ADDR0/ADDR1 range matching address. 1 Have received ADDR1 or ADDR0/ADDR1 range matching address.
12 AM0F	Address Match 0 Flag Indicates that the received address has matched the ADDR0 field as configured by ADDRCFG. This flag is cleared by reading the Address Status Register. This flag cannot generate an asynchronous wakeup. 0 Have not received ADDR0 matching address. 1 Have received ADDR0 matching address.
11 FEF	FIFO Error Flag FIFO error flag can only set when clock stretching is disabled. 0 FIFO underflow or overflow not detected. 1 FIFO underflow or overflow detected.
10 BEF	Bit Error Flag This flag will set if the LPI2C slave transmits a logic one and detects a logic zero on the I2C bus. The slave will ignore the rest of the transfer until the next (repeated) START condition. 0 Slave has not detected a bit error. 1 Slave has detected a bit error.

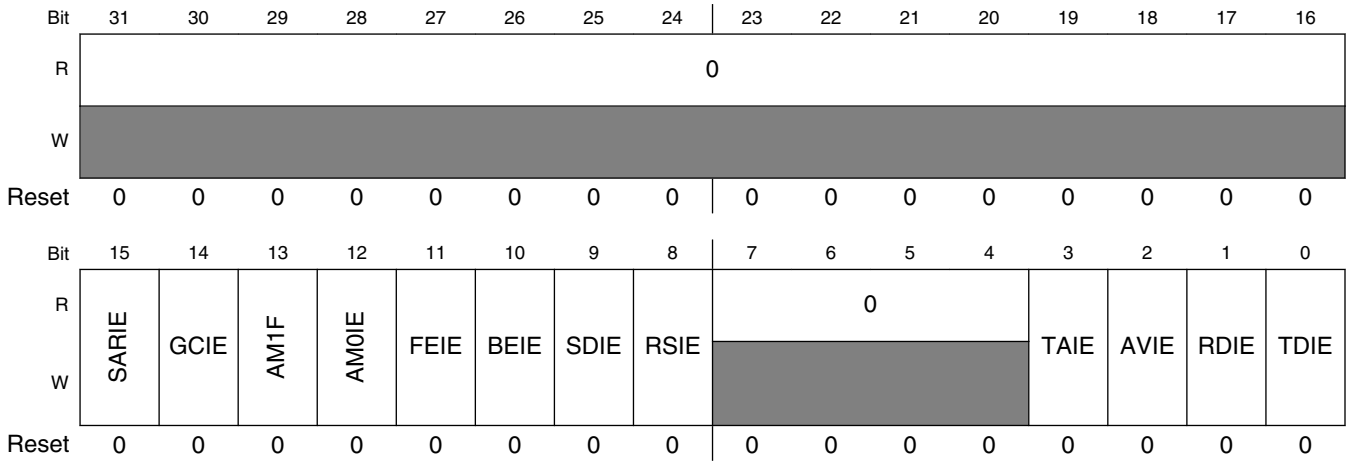
Table continues on the next page...

LPI2C_SSR field descriptions (continued)

Field	Description
9 SDF	STOP Detect Flag This flag will set when the LPI2C slave detects a STOP condition, provided the LPI2C slave matched the last address byte. 0 Slave has not detected a STOP condition. 1 Slave has detected a STOP condition.
8 RSF	Repeated Start Flag This flag will set when the LPI2C slave detects a repeated START condition, provided the LPI2C slave matched the last address byte. It does not set when the slave first detects a START condition. 0 Slave has not detected a Repeated START condition. 1 Slave has detected a Repeated START condition.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TAF	Transmit ACK Flag This flag is cleared by writing the transmit ACK register. 0 Transmit ACK/NACK is not required. 1 Transmit ACK/NACK is required.
2 AVF	Address Valid Flag This flag is cleared by reading the address status register. When RXCFG is set, this flag is also cleared by reading the receive data register. 0 Address Status Register is not valid. 1 Address Status Register is valid.
1 RDF	Receive Data Flag This flag is cleared by reading the receive data register. When RXCFG is set, this flag is not cleared when reading the receive data register and AVF is set. 0 Receive Data is not ready. 1 Receive data is ready.
0 TDF	Transmit Data Flag This flag is cleared by writing the transmit data register. When TXCFG is clear, it is also cleared if a NACK or Repeated START or STOP condition is detected. 0 Transmit data not requested. 1 Transmit data is requested.

27.2.20 Slave Interrupt Enable Register (LPI2C_SIER)

Address: 0h base + 118h offset = 118h



LPI2C_SIER field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 SARIE	SMBus Alert Response Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
14 GCIE	General Call Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
13 AM1F	Address Match 1 Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
12 AM0IE	Address Match 0 Interrupt Enable 0 Interrupt enabled. 1 Interrupt disabled.
11 FEIE	FIFO Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
10 BEIE	Bit Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
9 SDIE	STOP Detect Interrupt Enable

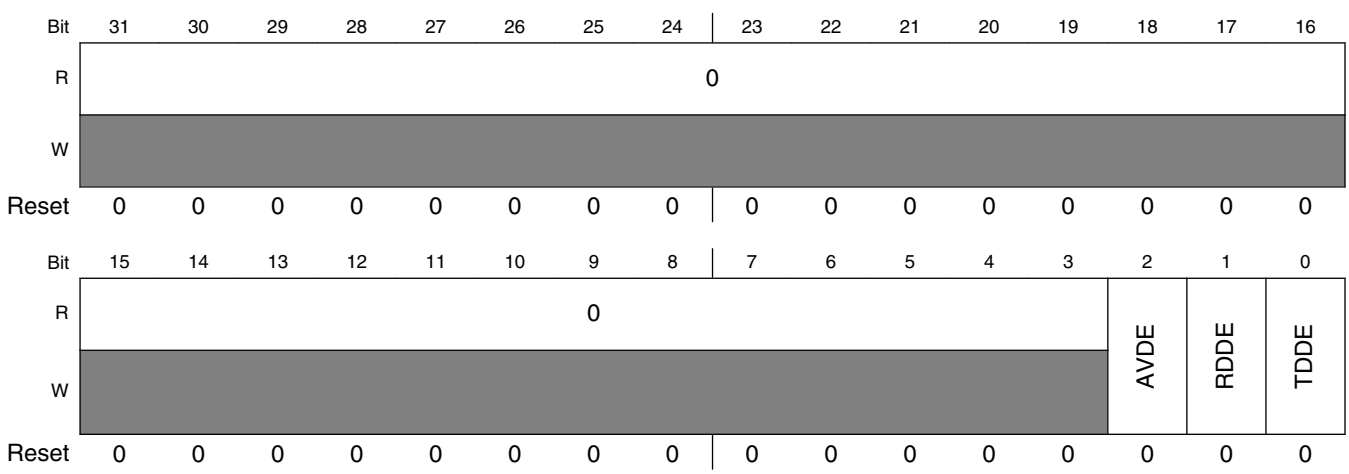
Table continues on the next page...

LPI2C_SIER field descriptions (continued)

Field	Description
	0 Interrupt disabled. 1 Interrupt enabled.
8 RSIE	Repeated Start Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TAIE	Transmit ACK Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
2 AVIE	Address Valid Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
1 RDIE	Receive Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
0 TDIE	Transmit Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled

27.2.21 Slave DMA Enable Register (LPI2C_SDER)

Address: 0h base + 11Ch offset = 11Ch



LPI2C_SDER field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 AVDE	Address Valid DMA Enable The Address Valid DMA request is shared with the Receive Data DMA request. If both are enabled, then set RXCFG to allow the DMA to read the address from the Receive Data Register. 0 DMA request disabled. 1 DMA request enabled.
1 RDDE	Receive Data DMA Enable 0 DMA request disabled. 1 DMA request enabled.
0 TDDE	Transmit Data DMA Enable 0 DMA request disabled. 1 DMA request enabled

27.2.22 Slave Configuration Register 1 (LPI2C_SCFGR1)

The SCFGR1 should only be written when the I2C Slave is disabled.

Address: 0h base + 124h offset = 124h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													ADDRCFG		
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
R	0		HSMEN	IGNACK	RXCFG	TXCFG	SAEN	GCEN	0				ACKSTALL	TXDSTALL	RXSTALL	ADRSTALL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2C_SCFGR1 field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 ADDRCFG	Address Configuration

Table continues on the next page...

LPI2C_SCFGR1 field descriptions (continued)

Field	Description
	Configures the condition that will cause an address to match. 000 Address match 0 (7-bit). 001 Address match 0 (10-bit). 010 Address match 0 (7-bit) or Address match 1 (7-bit). 011 Address match 0 (10-bit) or Address match 1 (10-bit). 100 Address match 0 (7-bit) or Address match 1 (10-bit). 101 Address match 0 (10-bit) or Address match 1 (7-bit). 110 From Address match 0 (7-bit) to Address match 1 (7-bit). 111 From Address match 0 (10-bit) to Address match 1 (10-bit).
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 HSMEN	High Speed Mode Enable Enables detection of the High-speed Mode master code of slave address 0000_1XX, but does not cause an address match on this code. When set and any Hs-mode master code is detected, the FILTEN and ACKSTALL bits are ignored until the next STOP condition is detected. 0 Disables detection of Hs-mode master code. 1 Enables detection of Hs-mode master code.
12 IGNACK	Ignore NACK When set, the LPI2C slave will continue transfers after a NACK is detected. This bit is required to be set in Ultra-Fast Mode. 0 Slave will end transfer when NACK detected. 1 Slave will not end transfer when NACK detected.
11 RXCFG	Receive Data Configuration 0 Reading the receive data register will return receive data and clear the receive data flag. 1 Reading the receive data register when the address valid flag is set will return the address status register and clear the address valid flag. Reading the receive data register when the address valid flag is clear will return receive data and clear the receive data flag.
10 TXCFG	Transmit Flag Configuration The transmit data flag will always assert before a NACK is detected at the end of a slave-transmit transfer. This can cause an extra word to be written to the transmit data FIFO. When TXCFG=0, the transmit data register is automatically emptied when a slave-transmit transfer is detected. This cause the transmit data flag to assert whenever a slave-transmit transfer is detected and negate at the end of the slave-transmit transfer. When TXCFG=1, the transmit data flag will assert whenver the transit data register is empty and negate when the transmit data register is full. This allows the transmit data register to be filled before a slave-transmit transfer is detected, but can cause the transmit data register to be written before a NACK is detected on the last byte of a slave transmit transfer. 0 Transmit Data Flag will only assert during a slave-transmit transfer when the transmit data register is empty. 1 Transmit Data Flag will assert whenever the transmit data register is empty.
9 SAEN	SMBus Alert Enable

Table continues on the next page...

LPI2C_SCFGR1 field descriptions (continued)

Field	Description
	0 Disables match on SMBus Alert. 1 Enables match on SMBus Alert.
8 GCEN	General Call Enable 0 General Call address is disabled. 1 General call address is enabled.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ACKSTALL	ACK SCL Stall Enables SCL clock stretching during slave-transmit address byte(s) and slave-receiver address and data byte(s) to allow software to write the Transmit ACK Register before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the 9th bit and is therefore not compatible with high speed mode. When ACKSTALL is enabled, there is no need to set either RXSTALL or ADRSTALL 0 Clock stretching disabled. 1 Clock stretching enabled.
2 TXDSTALL	TX Data SCL Stall Enables SCL clock stretching when the transmit data flag is set during a slave-transmit transfer. Clock stretching occurs following the 9th bit and is therefore compatible with high speed mode. 0 Clock stretching disabled. 1 Clock stretching enabled.
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when receive data flag is set during a slave-receive transfer. Clock stretching occurs following the 9th bit and is therefore compatible with high speed mode. 0 Clock stretching disabled. 1 Clock stretching enabled.
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when the address valid flag is asserted. Clock stretching only occurs following the 9th bit and is therefore compatible with high speed mode. 0 Clock stretching disabled. 1 Clock stretching enabled.

27.2.23 Slave Configuration Register 2 (LPI2C_SCFGR2)

The SCFGR2 should only be written when the I2C Slave is disabled.

Address: 0h base + 128h offset = 128h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FILTSDA				0				FILTSCL				0		DATAVD						0				CLKHOLD			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2C_SCFGR2 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 FILTSDA	Glitch Filter SDA Configures the I2C slave digital glitch filters for SDA input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSDA cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSDA+3 cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration, and is disabled in high speed mode.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 FILTSCL	Glitch Filter SCL Configures the I2C slave digital glitch filters for SCL input, a configuration of 0 will disable the glitch filter. Glitches equal to or less than FILTSCL cycles long will be filtered out and ignored. The latency through the glitch filter is equal to FILTSCL+3 cycles and must be configured less than the minimum SCL low or high period. The glitch filter cycle count is not affected by the PRESCALE configuration, and is disabled in high speed mode.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 DATAVD	Data Valid Delay Configures the SDA data valid delay time for the I2C slave equal to FILTSCL+DATAVD+3 cycles. This data valid delay must be configured to less than the minimum SCL low period. The I2C slave data valid delay time is not affected by the PRESCALE configuration, and is disabled in high speed mode.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C slave, when clock stretching is enabled. The minimum hold time is equal to CLKHOLD+3 cycles. The I2C slave clock hold time is not affected by the PRESCALE configuration, and is disabled in high speed mode.

27.2.24 Slave Address Match Register (LPI2C_SAMR)

Address: 0h base + 140h offset = 140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					ADDR1										0
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
R	0					ADDR0										0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2C_SAMR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26–17 ADDR1	Address 1 Value Compared against the received address to detect the Slave Address. In 10-bit mode, the first address byte is compared to { 11110, ADDR1[10:9] } and the second address byte is compared to ADDR1[8:1]. In 7-bit mode, the address is compared to ADDR1[7:1].
16–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–1 ADDR0	Address 0 Value Compared against the received address to detect the Slave Address. In 10-bit mode, the first address byte is compared to { 11110, ADDR0[10:9] } and the second address byte is compared to ADDR0[8:1]. In 7-bit mode, the address is compared to ADDR0[7:1].
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

27.2.25 Slave Address Status Register (LPI2C_SASR)

Address: 0h base + 150h offset = 150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0	ANV	0			RADDR										
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPI2C_SASR field descriptions

Field	Description
31–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 ANV	Address Not Valid 0 RADDR is valid. 1 RADDR is not valid.
13–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RADDR	Received Address RADDR updates whenever the AMF is set and the AMF is cleared by reading this register. In 7-bit mode, the address byte is store in RADDR[7:0]. In 10-bit mode, the first address byte is { 11110, RADDR[10:9],

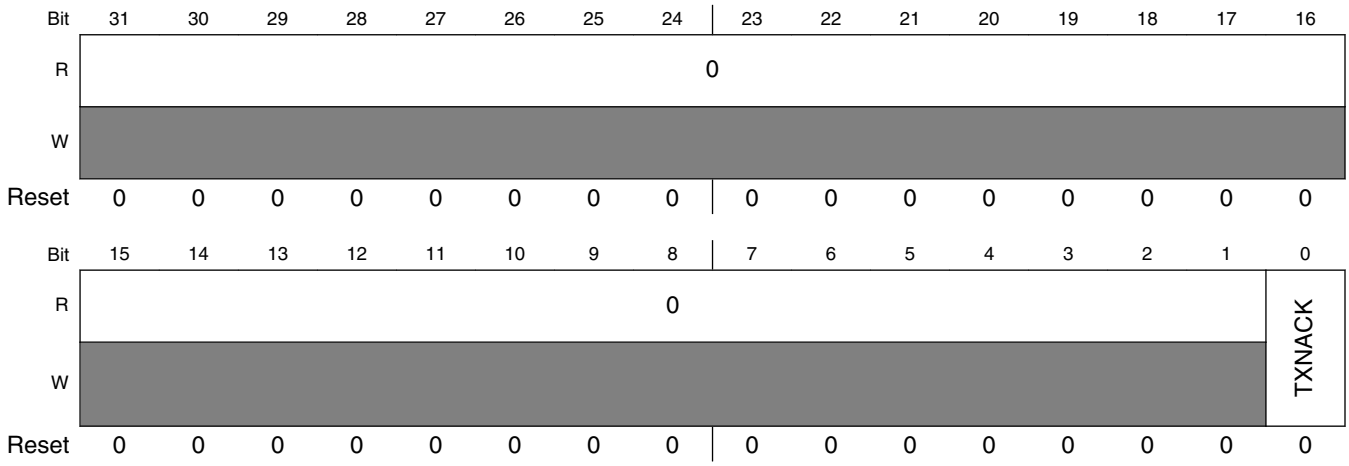
Table continues on the next page...

LPI2C_SASR field descriptions (continued)

Field	Description
	RADDR[0] } and the second address byte is RADDR[8:1]. The R/W bit is therefore always stored in RADDR[0].

27.2.26 Slave Transmit ACK Register (LPI2C_STAR)

Address: 0h base + 154h offset = 154h

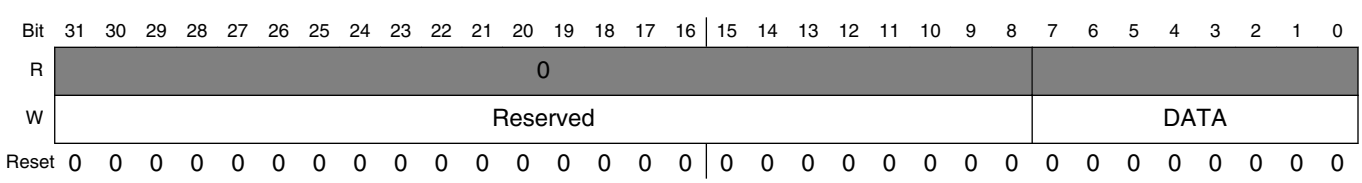


LPI2C_STAR field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TXNACK	Transmit NACK When NACKSTALL is set, must be written once for each matching address byte and each received word. Can also be written when LPI2C Slave is disabled or idle to configure the default ACK/NACK. 0 Transmit ACK for received word. 1 Transmit NACK for received word.

27.2.27 Slave Transmit Data Register (LPI2C_STDR)

Address: 0h base + 160h offset = 160h

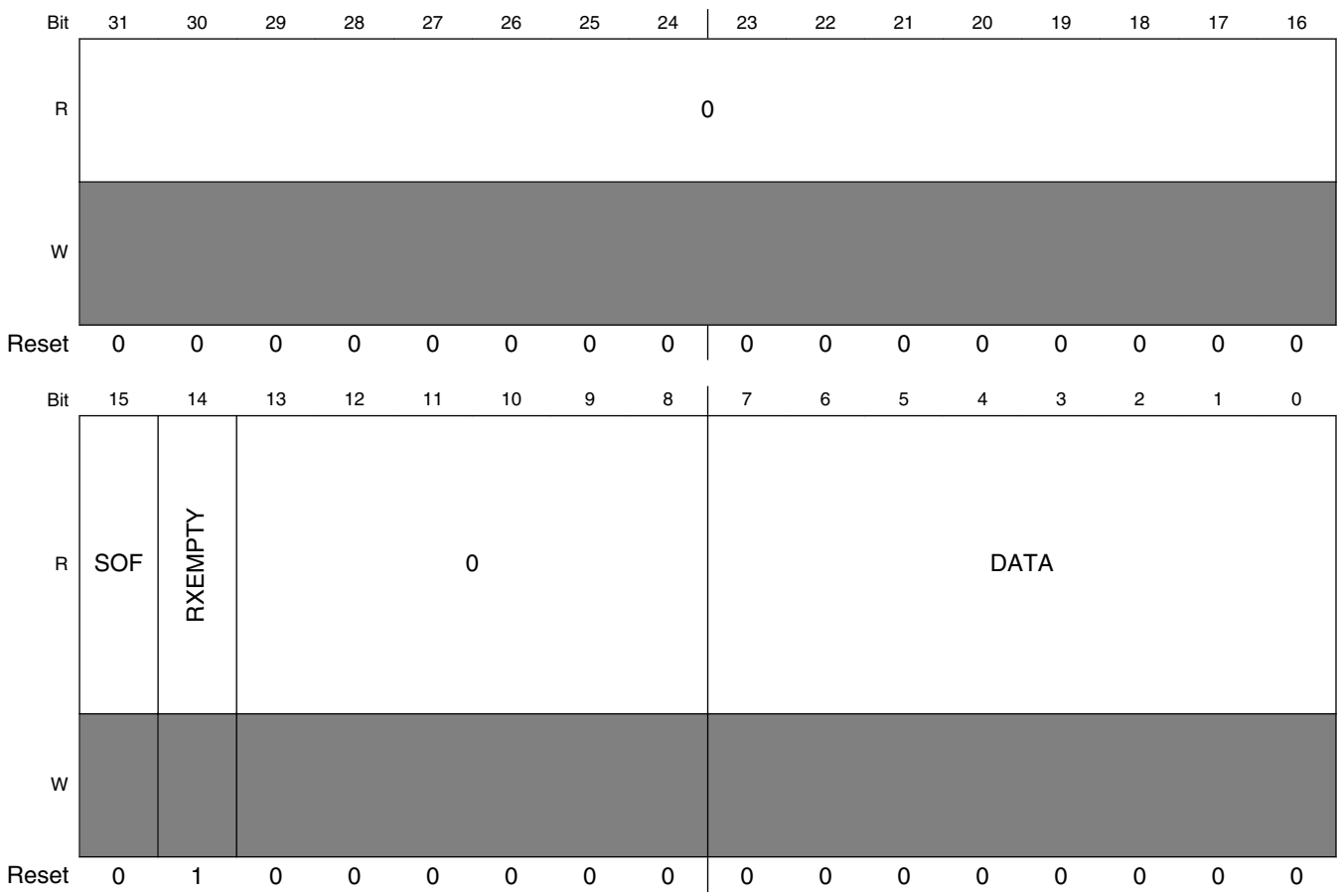


LPI2C_STDR field descriptions

Field	Description
31–8 Reserved	This field is reserved.
DATA	Transmit Data Writing this register will store I2C slave transmit data in the transmit register.

27.2.28 Slave Receive Data Register (LPI2C_SRDR)

Address: 0h base + 170h offset = 170h



LPI2C_SRDR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 SOF	Start Of Frame 0 Indicates this is not the first data word since a (repeated) START or STOP condition. 1 Indicates this is the first data word since a (repeated) START or STOP condition.

Table continues on the next page...

LPI2C_SRDR field descriptions (continued)

Field	Description
14 RXEMPTY	RX Empty 0 The Receive Data Register is not empty. 1 The Receive Data Register is empty.
13–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA	Receive Data Reading this register returns the data received by the I2C slave.

27.3 Functional description

27.3.1 Clocking and Resets

27.3.1.1 Functional clock

The LPI2C functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support I2C bus transfers by the LPI2C master. It is also used by the LPI2C slave to support digital filter and data hold time configurations. The LPI2C master divides the functional clock by a prescaler and the resulting frequency must be at least eight times faster than the I2C bus bandwidth.

27.3.1.2 External clock

The LPI2C slave logic is clocked directly from the external pins LPI2C_SCL and LPI2C_SDA (or LPI2C_SCLS and LPI2C_SDAS if master and slave are implemented on separate pins). This allows the LPI2C slave to remain operational, even when the LPI2C functional clock is disabled. Note that the LPI2C slave digital filter must be disabled if the LPI2C functional clock is disabled and this can effect compliance with some of the timing parameters of the I2C specification, such as the data hold time.

27.3.1.3 Bus clock

The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C master and slave registers.

27.3.1.4 Chip reset

The logic and registers for the LPI2C master and slave are reset to their default state on a chip reset.

27.3.1.5 Software reset

The LPI2C master implements a software reset bit in its Control Register. The MCR[RST] will reset all master logic and registers to their default state, except for the MCR itself.

The LPI2C slave implements a software reset bit in its Control Register. The SCR[RST] will reset all slave logic and registers to their default state, except for the SCR itself.

27.3.1.6 FIFO reset

The LPI2C master implements write-only control bits that resets the transmit FIFO (MCR[RTF] and receive FIFO (MCR[RRF]). A FIFO is empty after being reset.

The LPI2C slave implements write-only control bits that resets the transmit data register (SCR[RTF] and receive data register (SCR[RRF]). A data register is empty after being reset.

27.3.2 Master Mode

The LPI2C master logic operates independently from the slave logic to perform all master mode transfers on the I2C bus.

27.3.2.1 Transmit and Command FIFO

The transmit FIFO stores command data to initiate the various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- START or Repeated START condition with address byte and expecting ACK or NACK.
- Transmit data (this is the default for zero extended byte writes to the transmit FIFO).
- Receive 1-256 bytes of data (can also be configured to discard receive data and not store in receive FIFO).
- STOP condition (can also be configured to send STOP condition when transmit FIFO is empty).

Multiple transmit and receive commands can be inserted between the START condition and STOP conditon, transmit and receive commands must not be interleaved in order to comply with the I2C specification. The receive data command and the receive data and discard command can be interleaved to ensure only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C master supports 10-bit addressing through a (repeated) START condition, followed by a transmit byte with the second address byte, followed by any number of data bytes with the master-transmit data.

A START or Repeated START condition that is expecting a NACK (for example, hs-mode master code) must be followed by a STOP or (repeated) START condition.

27.3.2.2 Master Operation

Whenever the LPI2C is enabled, it monitors the I2C bus to detect when the I2C bus is idle (MSR[BBF]). The I2C bus is no longer considered idle if either SCL or SDA are low and becomes idle if a STOP condition is detected or if a bus idle timeout is detected (as configured by MCFGR2[BUSIDLE]). Once the I2C bus is idle, the transmit FIFO is not empty, and the host request is either asserted or disabled, then the LPI2C master will initiate a transfer on the I2C bus. This involves the following steps:

- Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler.
- Transmit a START condition and address byte using the timing configuration in MCCR0, if a high speed mode transfer is configured then timing configuration from MCCR1 is used instead.
- Perform master-transmit or master-receive transfers, as configured by the transmit FIFO.
- Transmit a Repeated START or STOP condition as configured by the transmit FIFO and/or MCFGR1[AUTOSTOP]. A repeated START can change which timing configuration register is used.

When the LPI2C master is disabled (either due to MCR[MEN] being clear or automatically due to mode entry), the LPI2C will continue to empty the transmit FIFO until a STOP condition is transmitted. However, it will no longer stall the I2C bus waiting for the transmit or receive FIFO and once the transmit FIFO is empty it will generate a STOP condition automatically.

The LPI2C master can stall the I2C bus under certain conditions, this will result in SCL pulled low continuously on the first bit of a byte until the condition is removed:

- LPI2C master is enabled and busy, transmit FIFO is empty, and MCFGR1[AUTOSTOP] is clear.
- LPI2C master is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and receive FIFO is full.

27.3.2.3 Receive FIFO and Data Match

The receive FIFO is used to store receive data during master-receiver transfers. Receive data can also be configured to discard receive data instead of storing in the receive FIFO, this is configured by the command word in the transmit FIFO.

Receive data supports a receive data match function that can match received data against one of two bytes or against a masked data byte. The data match function can also be configured to compare only the first one or two received data words since the last (repeated) START condition. Receive data that is already discarded due to the command word cannot cause the data match to set and will delay the match on first received data word until after the discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF] to allow all subsequent data to be received.

27.3.2.4 Timing Parameters

The following timing parameters can be configured by the LPI2C master. Parameters are configured separately for high speed mode (MCCR1) and other modes (MCCR0). This allows the high speed mode master code to be sent using the regular timing parameters and then switch to the high speed mode timing (following a repeated START) until the next STOP condition.

The LPI2C master timing parameters in LPI2C functional clock cycles are configured as follows. They must be configured to meet the I2C timing specification for the required mode.

- Bus idle time is always (MCCR0[CLKLO] + 1) multiplied by the prescaler. This is extended by the time it takes to detect external SDA rising edge.
- START or repeated START hold time is equal to (MCCR0/1[SETHOLD] + 1) multiplied by the prescaler.
- START, or repeated START, or STOP setup time is equal to (MCCR0/1[SETHOLD] + 1) multiplied by the prescaler. This is extended by the time it takes to detect external SCL rising edge.
- SCL low time (before clock stretching) is equal to (MCCR0/1[CLKLO] + 1) multiplied by the prescaler.
- SCL high time is equal to (MCCR0/1[CLKHI] + 1) multiplied by the prescaler. This is extended by the time it takes to detect external SCL rising edge.
- SDA output delay is equal to (MCCR0/1[DATAVD] + 1) multiplied by the prescaler.

The time taken to detect an external rising edge depends on a number of factors including the bus loading and external pull-up resistor sizing. The minimum delay equals two plus the pin input digital filter setting (which are configured separately for SCL and SDA), divided by the prescaler (since the pin input digital filters are not affected by the prescaler setting).

The following timing restrictions must be enforced to avoid unexpected START or STOP conditions on the I2C bus or unexpected START or STOP conditions detected by the LPI2C master. They can be summarized as SDA cannot change when SCL is high outside of a transmitted (repeated) START or STOP condition.

Table 27-2. Timing Parameters

Timing Parameter	Minimum	Maximum	Comment
CLKLO	0x03	-	CLKLO must also be greater than delay through the SCL filter.
CLKHI	0x01	-	
SETHOLD	0x02	-	
DATAVD	0x01	CLKLO - [(FILTSDA+2) / (2 ^ PRESCALER)]	DATAVD must be less than CLKLO minus delay through the SDA filter.
FILTSCL	0x00	[CLKLO × (2 ^ PRESCALER)] - 3	
FILTSDA	FILTSCL	[CLKLO × (2 ^ PRESCALER)] - 3	Does not apply if compensating for board level skew between SCL and SDA.
BUSIDLE	(CLKLO+SETHOLD+2) × 2	-	Must also be greater than CLKHI+1.

The timing parameters must be configured to meet the requirements of the I2C specification, this will depend on the mode being supported, the frequency of the LPI2C functional clock. Some example configurations are provided below.

Table 27-3. LPI2C Example Timing Configurations

I2C Mode	Clock Frequency	Baud Rate	PRESCALER	FILTSCL/ FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Fast	8 MHz	400 kbps	0x0	0x0/0x0	0x04	0x0B	0x05	0x02
Fast+	8 MHz	1 Mbps	0x0	0x0/0x0	0x02	0x03	0x01	0x01
Fast	48 MHz	400 kbps	0x2	0x1/0x1	0x07	0x11	0x0B	0x03
Fast+	48 MHz	1 Mbps	0x2	0x1/0x1	0x03	0x06	0x04	0x04
Fast+	48 MHz	1 Mbps	0x0	0x1/0x1	0x1D	0x18	0x13	0x0F
HS-mode	48 MHz	3.2 Mbps	0x0	0x0/0x0	0x07	0x08	0x03	0x01
Fast	60 MHz	400 kbps	0x1	0x2/0x2	0x11	0x28	0x21	0x08
Fast+	60 MHz	1 Mbps	0x1	0x2/0x2	0x07	0x0F	0x0B	0x01
HS-mode	60 MHz	3.33 Mbps	0x1	0x0/0x0	0x04	0x03	0x04	0x01
Ultrafast	60 MHz	5 Mbps	0x0	0x0/0x0	0x02	0x05	0x03	0x01

The formula to calculate number of cycles per bit is as follows:

*Baud rate divide = ((CLKLO+CLKHI+2)*2^PRESCALER) + ROUNDDOWN((2+FILTSCL)/2^PRESCALER)*

This assumes SCL will pull high within 1 cycle of the LPI2C functional clock, this will depend on the pullup resistor and loading on the SCL pin.

27.3.2.5 Error Conditions

The LPI2C master will monitor for errors while it is active, the following conditions will generate an error flag and block a new START condition from being sent until the flag is cleared by software:

- START or STOP condition detected and not generated by LPI2C master (sets MSR[ALF]).
- Transmitting data on SDA and different value being received (sets MSR[ALF]).
- NACK detected when transmitting data, provided MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- NACK detected and expecting ACK for address byte, provided MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- ACK detected and expecting NACK for address byte, provided MCFGR1[IGNACK] is clear (sets MSR[NDF]).

- Transmit FIFO requesting to transmit or receive data without a START condition (sets MSR[FEF]).
- SCL (or SDA if MCFGR1[TIMECFG] is set) is low for (MCFGR2[TIMELOW] * 256) prescaler cycles without a pin transition (sets MSR[PLTF]).

Software must respond to the MSR[PTLF] flag to terminate the existing command either cleanly (by clearing MCR[MEN]) or abruptly (by setting MCR[SWRST]).

The MCFGR2[BUSIDLE] field can be used to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE+1) prescaler cycles. The I2C bus is normally considered idle when the LPI2C master is first enabled, but when BUSIDLE is configured greater than zero then SCL and/or SDA must be high for (BUSIDLE+1) prescaler cycles before the I2C bus is first considered idle.

27.3.2.6 Pin Configuration

The LPI2C master defaults to open-drain configuration of the LPI2C_SDA and LPI2C_SCL pins. Support for true open drain is device specific and requires the pins where LPI2C pins are muxed to support true open drain. Support for high speed mode is also device specific and requires the LPI2C_SCL pin to support the current source pull-up required in the I2C specification.

The LPI2C master also supports the output only push-pull function required for I2C ultra-fast mode using the LPI2C_SDA and LPI2C_SCL pins. Support for ultra-fast mode also requires the IGNACK bit to be set.

A push-pull 2 wire configuration is also available to the LPI2C master that may support a partial high speed mode provided the LPI2C is the only master and all I2C pins on the bus are at the same voltage. This will configure the LPI2C_SCL pin as push-pull for every clock except the 9th clock pulse to allow high speed mode compatible slaves to perform clock stretching. In this mode, the LPI2C_SDA pin is tristated for master-receive data bits and master-transmit ACK/NACK bits.

The push-pull 4 wire configuration separates the SCL input and output and the SDA input and output onto separate pins, with SCL/SDA used as the input pins and SCLS/SDAS used as the output pins with configurable polarity. This simplifies the external connections when connecting the I2C bus to external level shifters. The LPI2C master logic and LPI2C slave logic are not able to connect to separate I2C buses when using this configuration.

27.3.3 Slave Mode

The LPI2C slave logic operates independently from the master logic to perform all slave mode transfers on the I2C bus.

27.3.3.1 Address Match

The LPI2C slave can be configured to match one of two addresses using either 7-bit or 10-bit addressing modes for each address, or to match a range of addresses in either 7-bit or 10-bit addressing modes. Separately, it can be configured to match the General Call Address or the SMBus Alert Address and generate appropriate flags. The LPI2C slave can also be configured to detect the high speed mode master code and to disable the digital filters and output valid delay time until the next STOP condition is detected.

Once a valid address is matched, the LPI2C slave will automatically perform slave-transmit or slave-receive transfers until a NACK is detected (unless IGNACK is set), a bit error is detected (the LPI2C slave is driving SDA, but a different value is sampled), or a (repeated) START or STOP condition is detected.

27.3.3.2 Transmit and Receive

The transmit and receive data registers are double buffered and only update during a slave-transmit and slave-receive transfer respectively. The slave address that was received can be configured to be read from either the receive data register (for example, when using DMA to transfer data) or from the address status register. The transmit data register can be configured to only request data once a slave-transmit transfer is detected or to request new data whenever the transmit data register is empty.

The transmit data register should only be written when the transmit data flag is set. The receive data register should only be read when the received data flag is set (or the address valid flag is set and RXCFG=1). The address status register should only be read when the address valid flag is set.

27.3.3.3 Clock Stretching

The LPI2C slave supports many configurable options for when clock stretching is performed. The following conditions can be configured to perform clock stretching.

- During 9th clock pulse of address byte and address valid flag is set.
- During 9th clock pulse of slave-transmit transfer and transmit data flag is set.
- During 9th clock pulse of slave-receive transfer and receive data flag is set.

- During 8th clock pulse of address byte or slave-receive transfer and transmit ACK flag is set. This is disabled in high speed mode.
- Clock stretching can also be extended for CLKHOLD cycles to allow additional setup time to sample the SDA pin externally. This is disabled in high speed mode.

Unless extended by the CLKHOLD configuration, clock stretching will extend for one peripheral bus clock cycle after SDA updates when clock stretching is enabled.

27.3.3.4 Timing Parameters

The LPI2C slave can configure the following timing parameters, these parameters are disabled when SCR[FILTEN] is clear, when SCR[FILTDZ] is set in Doze mode, and when LPI2C slave detects high speed mode. When disabled, the LPI2C slave is clocked directly from the I2C bus and may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

- SDA data valid time from SCL negation to SDA update.
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally.
- SCL glitch filter time.
- SDA glitch filter time.

The LPI2C slave imposes the following restrictions on the timing parameters.

- FILTSDA must be configured to greater than or equal to FILTSCL (unless compensating for board level skew between SDA and SCL).
- DATAVD must be configured less than the minimum SCL low period.

27.3.3.5 Error Conditions

The LPI2C slave can detect the following error conditions.

- Bit error flag will set when the LPI2C slave is driving SDA, but samples a different value than what is expected.
- FIFO error flag will set due to a transmit data underrun or a receive data overrun. Clock stretching can be enabled to eliminate the possibility of underrun and overrun occurring.
- FIFO error flag will also set due to an address overrun when RXCFG is set, otherwise an address overrun is not flagged. Clock stretching can be enabled to eliminate the possibility of overrun occurring.

The LPI2C slave does not implement a timeout due to SCL and/or SDA being stuck low. If this detection is required, the LPI2C master logic should be used and software can reset the LPI2C slave when this condition is detected.

27.3.4 Interrupts and DMA Requests

The LPI2C master and slave interrupts may be combined depending on the device.

The LPI2C master and slave transmit DMA requests may be combined depending on the device.

The LPI2C master and slave receive DMA requests may be combined depending on the device.

27.3.4.1 Master mode

The following table illustrates the master mode sources that can generate the LPI2C master interrupt and LPI2C master transmit/receive DMA requests.

Table 27-4. Master Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
TDF	Data can be written to transmit FIFO, as configured by TXWATER.	Y	TX	Y
RDF	Data can be read from the receive FIFO, as configured by RXWATER.	Y	RX	Y
EPF	Master has transmitted Repeated START or STOP condition.	Y	N	Y
SDF	Master has transmitted STOP condition.	Y	N	Y
NDF	Master detected NACK during address byte when expecting ACK, master detected ACK during address byte and expecting NACK, or master detected NACK during master-transmitter data byte.	Y	N	Y
ALF	Master lost arbitration due to START/STOP condition detected at	Y	N	Y

Table continues on the next page...

Table 27-4. Master Interrupts and DMA Requests (continued)

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
	wrong time, or Master was transmitting data but received different data than what was transmitted.			
FEF	Master expecting START condition in command FIFO and next entry in FIFO is not START condition.	Y	N	Y
PLTF	Pin low timeout is enabled and SCL (or SDA if configured) is low for longer than the configured timeout.	Y	N	Y
DMF	Received data matches the configured data match, and receive data not discarded due to command FIFO entry.	Y	N	Y
MBF	LPI2C master is busy transmitting/receiving data.	N	N	N
BBF	LPI2C master is enabled and activity detected on I2C bus, but STOP condition has not been detected and bus idle timeout (if enabled) has not occurred.	N	N	N

27.3.4.2 Slave mode

The following table illustrates the slave mode sources that can generate the LPI2C slave interrupt and the LPI2C slave transmit/receive DMA requests.

Table 27-5. Slave Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
TDF	Data can be written to transmit data register.	Y	TX	Y
RDF	Data can be read from the receive data register.	Y	RX	Y

Table continues on the next page...

Table 27-5. Slave Interrupts and DMA Requests (continued)

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
AVF	Address can be read from the address status register.	Y	RX	Y
TAF	ACK/NACK can be written to the transmit ACK register.	Y	N	Y
RSF	Slave has detected an address match followed by a Repeated START condition.	Y	N	Y
SDF	Slave has detected an address match followed by a STOP condition.	Y	N	Y
BEF	Slave was transmitting data, but received different data than what was transmitted.	Y	N	Y
FEF	Transmit data underrun, receive data overrun or address status overrun (when RXCFG=1). This flag can only set when clock stretching is disabled.	Y	N	Y
AM0F	Slave detected address match with ADDR0 field.	Y	N	N
AM1F	Slave detected address match with ADDR1 field or address range.	Y	N	N
GCF	Slave detected address match with general call address.	Y	N	N
SARF	Slave detected address match with SMBus alert address.	Y	N	N
SBF	LPI2C slave is busy receiving address byte or transmitting/receiving data.	N	N	N
BBF	LPI2C slave is enabled and START condition detected on I2C bus, but STOP condition has not been detected.	N	N	N

27.3.5 Peripheral Triggers

The connection of the LPI2C peripheral triggers with other peripherals are device specific.

27.3.5.1 Master Output Trigger

The LPI2C master generates an output trigger that can be connected to other peripherals on the device. The master output trigger asserts on both a Repeated START or STOP condition and remains asserted for one cycle of the LPI2C functional clock divided by the prescaler.

27.3.5.2 Slave Output Trigger

The LPI2C slave generates an output trigger that can be connected to other peripherals on the device. The slave output trigger asserts on both a Repeated START or STOP condition that occurs following a slave address match. It remains asserted until the next slave SCL pin negation.

27.3.5.3 Input Trigger

The LPI2C input trigger can be selected in place of the LPI2C_HREQ pin to control the start of a LPI2C master bus transfer. The input trigger must assert for longer than one LPI2C functional clock cycle to be detected.

27.4 Application Information

Chapter 28
Low Power Interrupt Timer (LPIT)

28.1 Introduction

28.1.1 Overview

The Low Power Periodic Interrupt Timer (LPIT) is a multi-channel timer module generating independent pre-trigger and trigger outputs. These timer channels can operate individually or can be chained together. The LPIT can operate in low power modes if configured to do so. The pre-trigger and trigger outputs can be used to trigger other modules on the device.

Each timer channel can be configured to run independently and made to work in either compare or capture modes. In compare mode, the timers decrement when enabled and generate an output pre-trigger and timeout pulse. The trigger output is 1 clock cycle delayed of the pre-trigger pulse. Each timer channel start, reload and restart can be controlled via control bits. The timer can be configured to always decrement, or decrement on selected trigger inputs or previous channel timeout (when channels are chained). By chaining timer channels, applications can achieve larger timeout durations. In capture mode, the timer can be used to perform measurements as the timer value is captured (in the timer value register) when a selected trigger input is asserted. In capture mode, the timer can support once-off or multiple measurements (for example, frequency measurements).

The timer channels operate on an asynchronous clock, which is independent from the register read/write access clock. Clock synchronization between the clock domains ensures normal operations.

28.1.2 Block Diagram

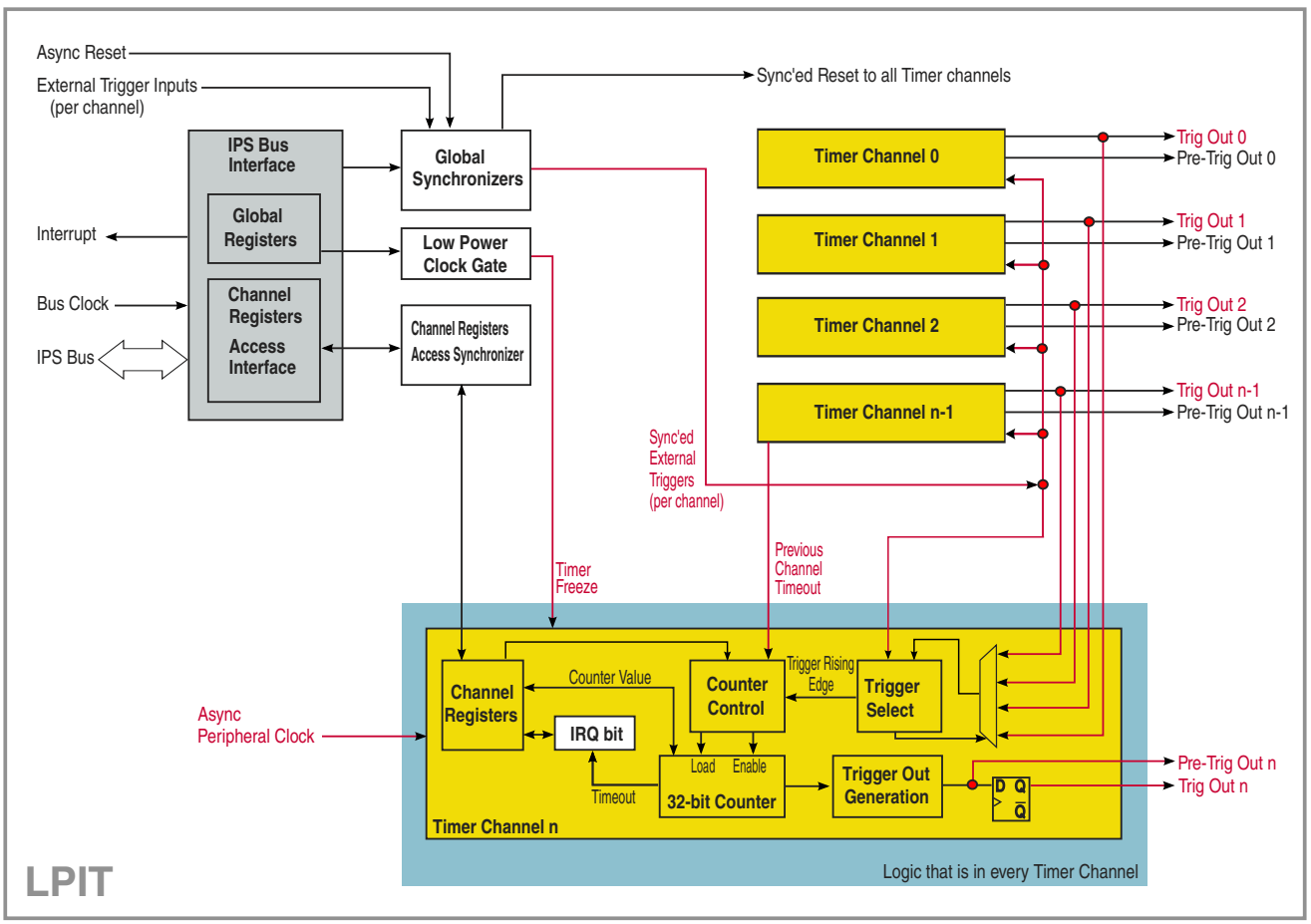


Figure 28-1. Top Level Block Diagram

28.2 Modes of operation

The LPIT module supports the chip modes described in the following table.

Table 28-1. Chip modes supported by the LPIT module

Chip mode	LPIT Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (MCR[DOZE_EN]) is set and the LPIT is using an external or internal clock source which remains operating during stop/wait modes.
Low Leakage Stop	The Doze Enable (MCR[DOZE_EN]) bit is ignored and the LPIT will be disabled for the duration of low leakage mode.
Debug	Can continue operating provided the Debug Enable bit (MCR[DBG_EN]) is set.

28.3 Memory Map and Registers

The memory map comprises of 32-bit aligned registers which can be accessed via 8-bit, 16-bit, or 32-bit accesses. Write access to reserved locations will generate a transfer error. Read access to reserved locations will also generate a transfer error and the read data bus will show all 0s. The Memory Map and complete module is in Big Endian format.

The module will not check for correctness of programmed values in the registers and software must ensure that correct values are being written.

LPIT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_0000	Version ID Register (LPIT0_VERID)	32	R	0100_0000h	28.3.1/738
4003_0004	Parameter Register (LPIT0_PARAM)	32	R	0000_0404h	28.3.2/738
4003_0008	Module Control Register (LPIT0_MCR)	32	R/W	0000_0000h	28.3.3/739
4003_000C	Module Status Register (LPIT0_MSR)	32	w1c	0000_0000h	28.3.4/740
4003_0010	Module Interrupt Enable Register (LPIT0_MIER)	32	R/W	0000_0000h	28.3.5/741
4003_0014	Set Timer Enable Register (LPIT0_SETTEN)	32	R/W	0000_0000h	28.3.6/742
4003_0018	Clear Timer Enable Register (LPIT0_CLRTEN)	32	W (always reads 0)	0000_0000h	28.3.7/743
4003_0020	Timer Value Register (LPIT0_TVAL0)	32	R/W	0000_0000h	28.3.8/744
4003_0024	Current Timer Value (LPIT0_CVAL0)	32	R	FFFF_FFFFh	28.3.9/745
4003_0028	Timer Control Register (LPIT0_TCTRL0)	32	R/W	0000_0000h	28.3.10/ 746
4003_0030	Timer Value Register (LPIT0_TVAL1)	32	R/W	0000_0000h	28.3.8/744
4003_0034	Current Timer Value (LPIT0_CVAL1)	32	R	FFFF_FFFFh	28.3.9/745
4003_0038	Timer Control Register (LPIT0_TCTRL1)	32	R/W	0000_0000h	28.3.10/ 746
4003_0040	Timer Value Register (LPIT0_TVAL2)	32	R/W	0000_0000h	28.3.8/744
4003_0044	Current Timer Value (LPIT0_CVAL2)	32	R	FFFF_FFFFh	28.3.9/745
4003_0048	Timer Control Register (LPIT0_TCTRL2)	32	R/W	0000_0000h	28.3.10/ 746
4003_0050	Timer Value Register (LPIT0_TVAL3)	32	R/W	0000_0000h	28.3.8/744
4003_0054	Current Timer Value (LPIT0_CVAL3)	32	R	FFFF_FFFFh	28.3.9/745
4003_0058	Timer Control Register (LPIT0_TCTRL3)	32	R/W	0000_0000h	28.3.10/ 746

28.3.1 Version ID Register (LPITx_VERID)

Address: 4003_0000h base + 0h offset = 4003_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPITx_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification
FEATURE	Feature Number This read only field returns the feature set number.

28.3.2 Parameter Register (LPITx_PARAM)

This register provides details on the parameter settings that were used while including this module in the device.

Address: 4003_0000h base + 4h offset = 4003_0004h

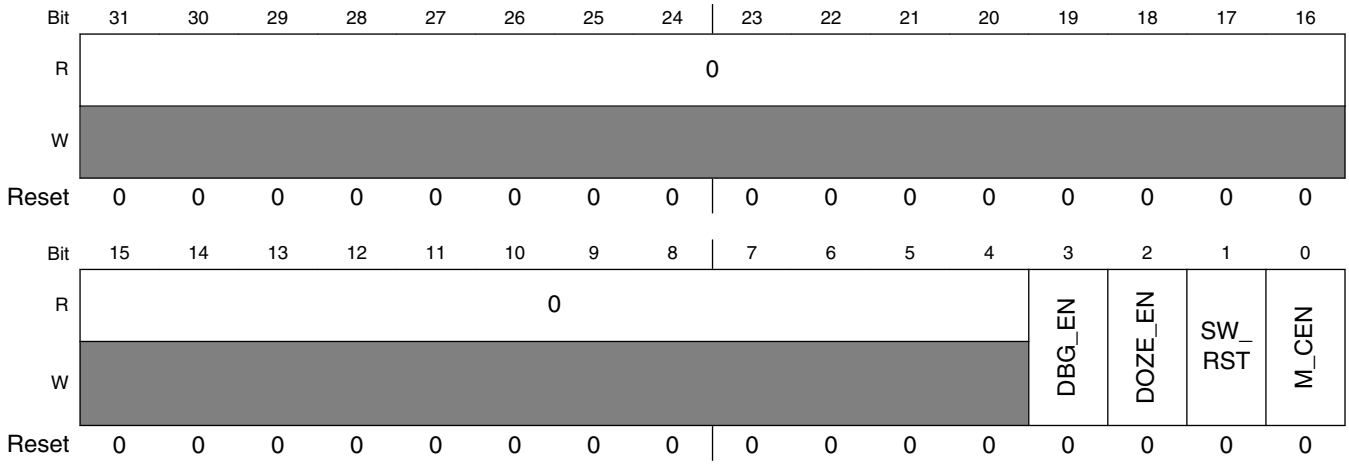
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EXT_TRIG								CHANNEL							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

LPITx_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 EXT_TRIG	Number of External Trigger Inputs Number of external triggers implemented.
CHANNEL	Number of Timer Channels Number of timer channels implemented.

28.3.3 Module Control Register (LPITx_MCR)

Address: 4003_0000h base + 8h offset = 4003_0008h



LPITx_MCR field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 DBG_EN	Debug Enable Bit Allows the timer channels to be stopped when the device enters the Debug mode 0 Timer channels are stopped in Debug mode 1 Timer channels continue to run in Debug mode
2 DOZE_EN	DOZE Mode Enable Bit Allows the timer channels to be stopped or continue to run when the device enters the DOZE mode 0 Timer channels are stopped in DOZE mode 1 Timer channels continue to run in DOZE mode
1 SW_RST	Software Reset Bit Resets all channels and registers, except the Module Control Register. Remains set until cleared by software. 0 Timer channels and registers are not reset 1 Timer channels and registers are reset
0 M_CEN	Module Clock Enable Enables the peripheral clock to the module timers. M_CEN bit must be asserted when writing to timer registers. Both clocks (bus clock and peripheral clock) must be enabled, to allow for clock synchronization and update of register bits. **NOTE:** Writing to the MSR, SETTEN , CLRTEN, TCTRL, and TVAL registers while M_CEN = 0, will lead to the assertion of a transfer error for that bus cycle. Writing to CVAL and reserved registers will always generate a transfer error.

Table continues on the next page...

LPITx_MCR field descriptions (continued)

Field	Description
	0 Protocol clock to timers is disabled 1 Protocol clock to timers is enabled

28.3.4 Module Status Register (LPITx_MSR)

Address: 4003_0000h base + Ch offset = 4003_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0												TIF3	TIF2	TIF1	TIF0
W													w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPITx_MSR field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TIF3	Channel 3 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred
2 TIF2	Channel 2 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred
1 TIF1	Channel 1 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect. 0 Timer has not timed out 1 Timeout has occurred
0 TIF0	Channel 0 Timer Interrupt Flag In compare modes, sets to 1 at the end of the timer period. In capture modes, sets to 1 when the trigger asserts. Writing logic 1 to this flag clears it. Writing 0 has no effect.

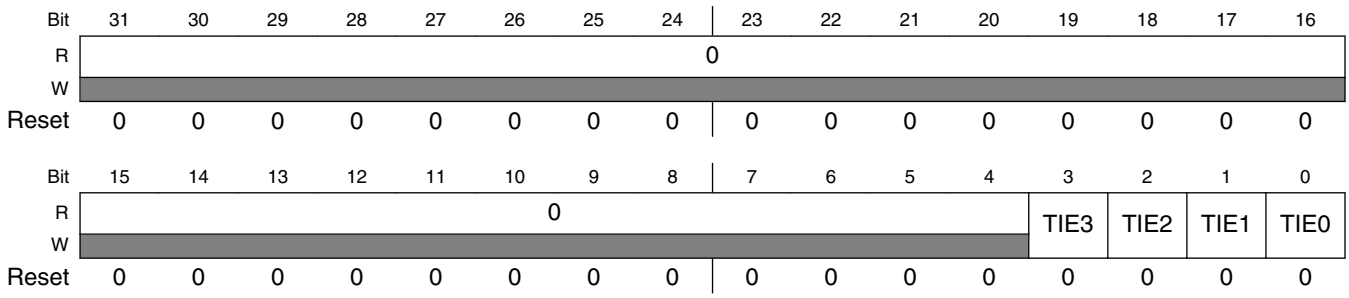
Table continues on the next page...

LPITx_MSR field descriptions (continued)

Field	Description
	0 Timer has not timed out 1 Timeout has occurred

28.3.5 Module Interrupt Enable Register (LPITx_MIER)

Address: 4003_0000h base + 10h offset = 4003_0010h



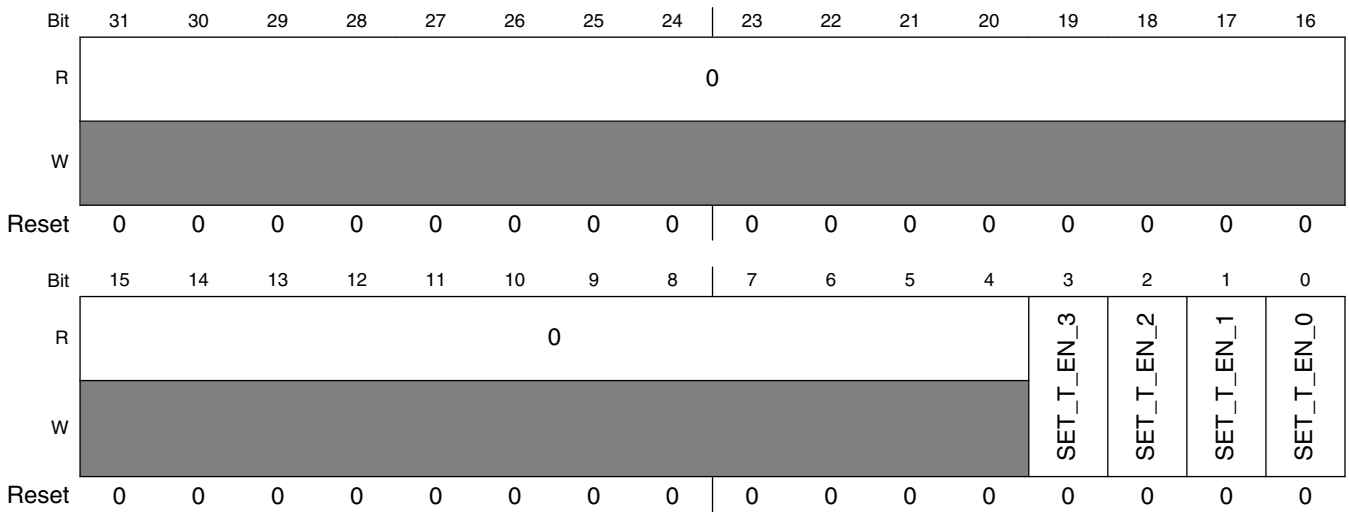
LPITx_MIER field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 TIE3	Channel 3 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled
2 TIE2	Channel 2 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled
1 TIE1	Channel 1 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled
0 TIE0	Channel 0 Timer Interrupt Enable Enables interrupt generation when this bit is set to 1 and if corresponding Timer Interrupt Flag is asserted. 0 Interrupt generation is disabled 1 Interrupt generation is enabled

28.3.6 Set Timer Enable Register (LPITx_SETTEN)

This register allows simulataneous enabling of timer channels. Timer channels can be enabled either by writing '1' to T_EN in respective TCTRLn register or setting the corresponding bit in this register. Writing a '0' to this register has no effect. CLRTEN register should be used to disable timer channels simultaneously.

Address: 4003_0000h base + 14h offset = 4003_0014h



LPITx_SETTEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SET_T_EN_3	Set Timer 3 Enable Writing '1' to this bit will enable the timer channel 3. This bit can be used in addition to T_EN bit in TCTRL3 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL3 is set to '0' or '1' is written to the CLR_T_EN_3 bit in CLRTEN register. 0 No effect 1 Enables the Timer Channel 3
2 SET_T_EN_2	Set Timer 2 Enable Writing '1' to this bit will enable the timer channel 2. This bit can be used in addition to T_EN bit in TCTRL2 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL2 is set to '0' or '1' is written to the CLR_T_EN_2 bit in CLRTEN register. 0 No Effect 1 Enables the Timer Channel 2
1 SET_T_EN_1	Set Timer 1 Enable

Table continues on the next page...

LPITx_SETTEN field descriptions (continued)

Field	Description
	Writing '1' to this bit will enable the timer channel 1. This bit can be used in addition to T_EN bit in TCTRL1 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL1 is set to '0' or '1' is written to the CLR_T_EN_1 bit in CLRTEN register. 0 No Effect 1 Enables the Timer Channel 1
0 SET_T_EN_0	Set Timer 0 Enable Writing '1' to this bit will enable the timer channel 0. This bit can be used in addition to T_EN bit in TCTRL0 register. Writing a 0 will not disable the counter. This bit will be cleared when T_EN bit in TCTRL0 is set to 0 or '1' is written to the CLR_T_EN_0 bit in CLRTEN register. 0 No effect 1 Enables the Timer Channel 0

28.3.7 Clear Timer Enable Register (LPITx_CLRTEN)

Address: 4003_0000h base + 18h offset = 4003_0018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0												0	0	0	0
W													CLR_T_EN_3	CLR_T_EN_2	CLR_T_EN_1	CLR_T_EN_0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPITx_CLRTEN field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 CLR_T_EN_3	Clear Timer 3 Enable Writing a '1' to this bit will disable the timer channel 3. This bit can be used in addition to T_EN bit in TCTRL3 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No Action 1 Clear T_EN bit for Timer Channel 3
2 CLR_T_EN_2	Clear Timer 2 Enable Writing a '1' to this bit will disable the timer channel 2. This bit can be used in addition to T_EN bit in TCTRL2 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No Action 1 Clear T_EN bit for Timer Channel 2
1 CLR_T_EN_1	Clear Timer 1 Enable Writing a '1' to this bit will disable the timer channel 1. This bit can be used in addition to T_EN bit in TCTRL1 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No Action 1 Clear T_EN bit for Timer Channel 1
0 CLR_T_EN_0	Clear Timer 0 Enable Writing a '1' to this bit will disable the timer channel 0. This bit can be used in addition to T_EN bit in TCTRL0 register. Writing a 1 will not enable the counter. This bit is self clearing and will always read 0. 0 No action 1 Clear T_EN bit for Timer Channel 0

28.3.8 Timer Value Register (LPITx_TVAL*n*)

In compare modes, these registers select the timeout period for the timer channels. In capture modes, these registers are loaded with the value of the counter when the trigger asserts.

Address: 4003_0000h base + 20h offset + (16d × i), where i=0d to 3d

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

LPITx_TVAL*n* field descriptions

Field	Description
TMR_VAL	Timer Value

LPITx_TVALn field descriptions (continued)

Field	Description
	In compare modes, sets the timer channel start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer channel; instead the value will be loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer channel must be disabled and enabled again. In capture modes, this register stores the inverse of the counter whenever the trigger asserts. 0 Invalid load value in compare modes >0 Value to be loaded (Compare Mode) or Value of Timer (Capture Mode)

28.3.9 Current Timer Value (LPITx_CVALn)

These registers indicate the current timer counter value.

Address: 4003_0000h base + 24h offset + (16d × i), where i=0d to 3d

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

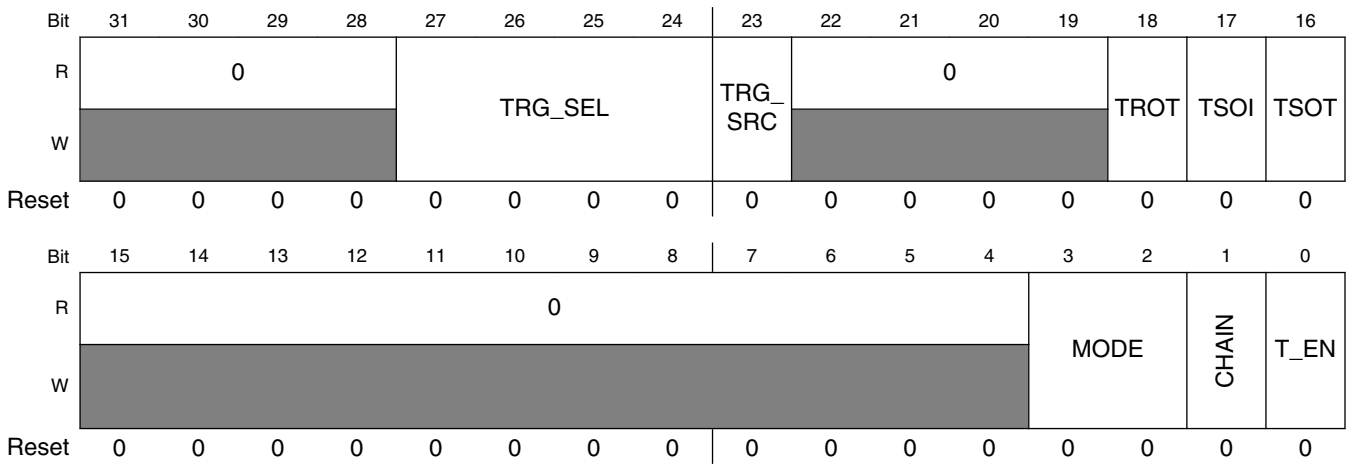
LPITx_CVALn field descriptions

Field	Description
TMR_CUR_VAL	Current Timer Value Represents the current timer value, if the timer is enabled.

28.3.10 Timer Control Register (LPITx_TCTRL*n*)

These registers contain the control bits for each timer channel

Address: 4003_0000h base + 28h offset + (16d × i), where i=0d to 3d



LPITx_TCTRL*n* field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 TRG_SEL	Trigger Select Selects the trigger to use for starting and/or reloading the LPIT timer. This field should only be changed when the LPIT timer channel is disabled. The TRG_SRC bit selects between internal and external trigger signals for each channel. The TRG_SEL bits select one trigger from the set of internal or external triggers selected by TRG_SRC. 0 Timer channel 0 trigger source is selected 1 Timer channel 1 trigger source is selected 2 Timer channel 2 trigger source is selected . . . . . . n Timer channel 'n' trigger source is selected
23 TRG_SRC	Trigger Source Selects between internal or external trigger sources. The final trigger is selected by TRG_SEL depending on which trigger source out of internal triggers or external triggers are selected by TRG_SRC. Refer to the chip configuration section for available external trigger options. If a channel does not have an associated external trigger then this bit for that channel should be set to 1. 0 Trigger source selected in external 1 Trigger source selected is the internal trigger
22–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 TROT	Timer Reload On Trigger

Table continues on the next page...

LPITx_TCTRL*n* field descriptions (continued)

Field	Description
	When set, the LPIT timer will reload when a rising edge is detected on the selected trigger input. The trigger input is ignored if the LPIT is disabled during debug mode or DOZE mode (DOZE_EN or DBGEN = 0) 0 Timer will not reload on selected trigger 1 Timer will reload on selected trigger
17 TSOI	Timer Stop On Interrupt This bit controls whether the channel timer will stop after it times out and when it can restart (when TSOT = 0). If TSOT = 1, then the timer will stop on timeout and will restart after a rising edge on the selected trigger is detected. If TSOT = 0, then this bit controls when the timer restarts. 0 Timer does not stop after timeout 1 Timer will stop after timeout and will restart after rising edge on the T_EN bit is detected (i.e. timer channel is disabled and then enabled)
16 TSOT	Timer Start On Trigger This bit controls when the timer starts decrementing. 0 Timer starts to decrement immediately based on restart condition (controlled by TSOI bit) 1 Timer starts to decrement when rising edge on selected trigger is detected
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 MODE	Timer Operation Mode Configures the Channel Timer Mode of Operation. The mode bits control how the timer decrements. See Functional Description for more details. 00 32-bit Periodic Counter 01 Dual 16-bit Periodic Counter 10 32-bit Trigger Accumulator 11 32-bit Trigger Input Capture
1 CHAIN	Chain Channel When enabled, timer channel will decrement when channel N-1 trigger asserts. Channel 0 cannot be chained. 0 Channel Chaining is disabled. Channel Timer runs independently. 1 Channel Chaining is enabled. Timer decrements on previous channel's timeout
0 T_EN	Timer Enable Enables or disables the Timer Channel 0 Timer Channel is disabled 1 Timer Channel is enabled

28.4 Functional description

28.4.1 Initialization

The following steps can be used to initialize the LPIT module

- Enable the protocol clock by setting the M_CEN bit in the MCR register.

NOTE

Writing to certain registers while M_CEN = 0 will lead to assertion of transfer error for that bus access. These registers are MSR, SETTEN, CLRTEN, TVAL, and TCTRL. Writing to CVAL and Reserved registers will generate a transfer error irrespective of M_CEN bit value. Reads to these registers can happen irrespective of M_CEN bit value.

- Wait for 4 protocol clock cycles to allow time for clock synchronization and reset de-assertion.
- For each timer channel that is to be enabled, configure the timer mode of operation (MODE bits), Trigger source selection (TRG_SEL & TRG_SRC) and Trigger control bits (TROT, TSOT, TSOI bits) in the TCTRLn register.
- Configure the channels that are to be chained by setting the CHAIN bit in the corresponding channel's TCTRLn register.
- For channels configured in Compare Mode, set the timer timeout value by programming the appropriate value in TVAL register for those channels.
- Configure TIEn bits in MIER register for those channels which are required to generate interrupt on timer timeout.
- Configure the low power mode functionality of the module by setting the DBG_EN and DOZE_EN bits in the MCR register. This is common to all timer channels.
- Enable the channel timers by setting the corresponding T_EN bit in the corresponding channel's TCRTLn register.
- For channels configured in Capture Mode, the timer value can be read from TVALn register when channel timeout occurs.
- At any time, the current value of the timer for any channel can be read by reading the corresponding channel's CVALn register.
- The timer interrupt flag bits (TIFn) in MSR register get asserted on timer timeout. These bits can be cleared by writing '1' to them.

28.4.2 Timer Modes

The timer mode is configured by setting an appropriate value in the MODE bits in TCTRLn register. The timer modes supported are:

- *32-bit Periodic Counter:* In this mode the counter will load and then decrement down to zero. It will then set the timer interrupt flag and assert the output pre-trigger.
- *Dual 16-bit Periodic Counter:* In this mode, the counter will load and then the lower 16-bits will decrement down to zero, which will assert the output pre-trigger. The upper 16-bits will then decrement down to zero, which will negate the output pre-trigger and set the timer interrupt flag.
- *32-bit Trigger Accumulator:* In this mode, the counter will load on the first trigger rising edge and then decrement down to zero on each trigger rising edge. It will then set the timer interrupt flag and assert the output pre-trigger.
- *32-bit Trigger Input Capture:* In this mode, the counter will load with 0xFFFF_FFFF and then decrement down to zero. If a trigger rising edge is detected, it will store the inverse of the current counter value in the load value register, set the timer interrupt flag and assert the output pre-trigger.

The timer operation is further controlled by Trigger Control bits (TSOT, TSOI, TROT) which control the timer load, reload, start and restart of the timers.

NOTE

- The trigger output is asserted one Protocol Timer Clock cycle later than pre-trigger output. The trigger output and the pre-trigger output de-assert at the same time.
- The pre-trigger output is asserted for two clock cycles and trigger output is asserted for one clock cycle (except in 16-bit Periodic Counter mode where both pre-trigger and trigger are asserted for many cycles depending on TMR_VAL[31:16]).

28.4.3 Trigger Control for Timers

The TSOT, TROT, TSOI and TRG_SEL, TRG_SRC bits control how the trigger input affects the timer operation. The TRG_SEL selects the input trigger for the channel from all other channel's trigger outputs. The TRG_SRC further selects between the selected internal trigger and the external trigger input to the channel.

The selected trigger affects the timer operation based on TROT, TSOI & TSOT bits. The behavior due to these bits is as follows:

- If TSOI = 1, counter stops on TIF assertion. Requires trigger (if TSOT = 1) or T_EN rising edge (if TSOT = 0), to reload and decrement. If TSOI = 0, counter does not stop after timeout.

- If TROT = 1, counter is loaded on each trigger; else, counter is loaded on every T_EN rising edge or timeout rising edge (timeout not used in Capture modes).
- If TSOT = 1, counter will start to decrement on trigger. Subsequent triggers are ignored till a counter timeout. If TSOT = 0, counter decrements immediately from the next clock edge. TSOT has no effect when channel is Chained or in Capture mode.

These bits affect the timer operation differently in different timer modes:

- In 32-bit Periodic Counter and Dual 16-bit Periodic Counter modes, all bits (TSOT, TSOI & TROT) affect the timer operation as described above.
- In 32-bit Trigger Accumulator mode, only TSOI bit controls the timer function. TROT & TSOT bits have no effect on timer operation.
- In 32-bit Input Trigger Capture mode, TSOI and TROT bits control the timer function. TSOT bit has no effect on timer operation.

28.4.4 Channel Chaining

Individual timer channels can be chained together to achieve a larger value of timeout. Chaining the timer channel causes them to work in a *'nested loop'* manner thereby leading to an effective timeout value of $TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$.

The channels are chained by setting the CHAIN bit in corresponding channel's TCTRLn register. When a channel is chained, that channel's timer decrements on previous channel's timeout pulse, irrespective of the timer mode (MODE bits). The TSOT bit does not have any effect if the channel timer (Channel 'n') is chained to previous channel's timer (Channel 'n-1').

Chapter 29
Low Power Serial Peripheral Interface (LPSPI)

29.1 Introduction

29.1.1 Overview

The LPSPI is a low power Serial Peripheral Interface (SPI) module that supports an efficient interface to an SPI bus as a master and/or a slave. The LPSPI can continue operating in stop modes provided an appropriate clock is available and is designed for low CPU overhead with DMA offloading of FIFO register accesses.

29.1.2 Features

The LPSPI supports the following features:

- Word size = 32 bits
- Command/transmit FIFO of 4 words.
- Receive FIFO of 4 words.
- Host request input can be used to control the start time of an SPI bus transfer.

29.1.3 Block Diagram

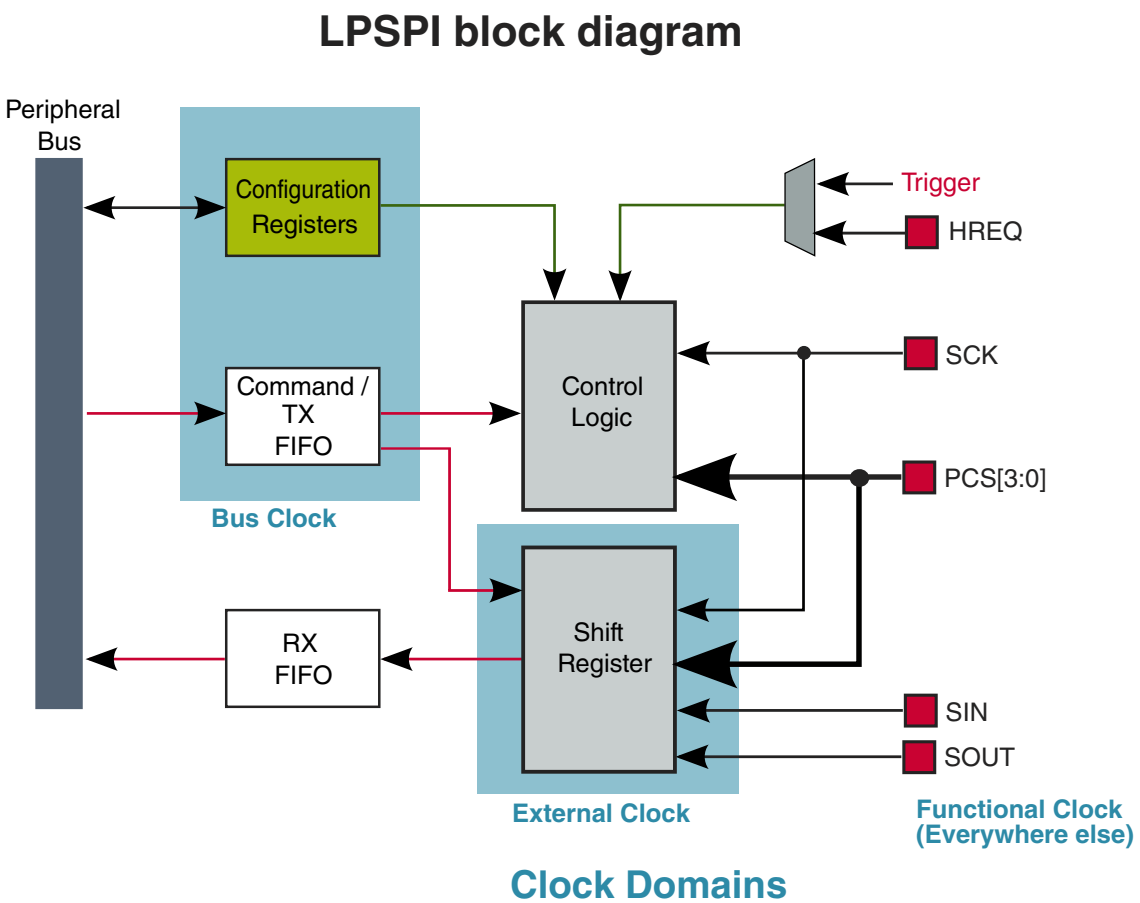


Figure 29-1. Block Diagram

29.1.4 Modes of operation

The LPSPI module supports the chip modes described in the following table.

Table 29-1. Chip modes supported by the LPSPI module

Chip mode	LPSPI Operation
Run	Normal operation
Stop/Wait	Can continue operating if the Doze Enable bit (MCR[DOZEN]) is set and the LPSPI is using an external or internal clock source, which remains operating during stop/wait modes.
Low Leakage Stop (LLS/VLLS modes)	The Doze Enable (MCR[DOZEN]) bit is ignored and the LPSPI will wait for the current transfer to finish any pending operation, before the LPSPI acknowledges entry into low leakage mode.
Debug (the core is in Debug/Halted mode)	Can continue operating if the Debug Enable bit (MCR[DBGE]) is set.

29.1.5 Signal Descriptions

Signal	Description	I/O
SCK	Serial clock. Input in slave mode, output in master mode.	I/O
PCS[0]	Peripheral Chip Select. Input in slave mode, output in master mode.	I/O
PCS[1] / HREQ	Peripheral Chip Select or Host Request. Host Request pin is selected when HREN=1 and HRSEL=0. Input in either slave mode or when used as Host Request, output in master mode.	I/O
PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during quad-data transfers. Input in slave mode, output in master mode, input in quad-data receive transfers, output in quad-data transmit transfers.	I/O
SOUT / DATA[0]	Serial Data Output. Can be configured as serial data input signal. Used as data pin 0 in quad-data and dual-data transfers.	I/O
SIN / DATA[1]	Serial Data Input. Can be configured as serial data output signal. Used as data pin 1 in quad-data and dual-data transfers.	I/O

29.2 Memory Map and Registers

LPSPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Version ID Register (LPSPI_VERID)	32	R	0100_0004h	29.2.1/754
4	Parameter Register (LPSPI_PARAM)	32	R	See section	29.2.2/755
10	Control Register (LPSPI_CR)	32	R/W	0000_0000h	29.2.3/756
14	Status Register (LPSPI_SR)	32	R/W	0000_0001h	29.2.4/757
18	Interrupt Enable Register (LPSPI_IER)	32	R/W	0000_0000h	29.2.5/759
1C	DMA Enable Register (LPSPI_DER)	32	R/W	0000_0000h	29.2.6/760
20	Configuration Register 0 (LPSPI_CFGR0)	32	R/W	0000_0000h	29.2.7/761

Table continues on the next page...

LPSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
24	Configuration Register 1 (LPSPI_CFGR1)	32	R/W	0000_0000h	29.2.8/762
30	Data Match Register 0 (LPSPI_DMR0)	32	R/W	0000_0000h	29.2.9/764
34	Data Match Register 1 (LPSPI_DMR1)	32	R/W	0000_0000h	29.2.10/ 764
40	Clock Configuration Register (LPSPI_CCR)	32	R/W	0000_0000h	29.2.11/ 765
58	FIFO Control Register (LPSPI_FCR)	32	R/W	0000_0000h	29.2.12/ 766
5C	FIFO Status Register (LPSPI_FSR)	32	R	0000_0000h	29.2.13/ 766
60	Transmit Command Register (LPSPI_TCR)	32	R/W	0000_001Fh	29.2.14/ 767
64	Transmit Data Register (LPSPI_TDR)	32	W	0000_0000h	29.2.15/ 770
70	Receive Status Register (LPSPI_RSR)	32	R	0000_0002h	29.2.16/ 771
74	Receive Data Register (LPSPI_RDR)	32	R	0000_0000h	29.2.17/ 772

29.2.1 Version ID Register (LPSPI_VERID)

.

Address: 0h base + 0h offset = 0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

LPSPI_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Module Identification Number This read only field returns the feature set number. 0x0004 Standard feature set supporting 32-bit shift register.

29.2.2 Parameter Register (LPSPI_PARAM)

.

Address: 0h base + 4h offset = 4h

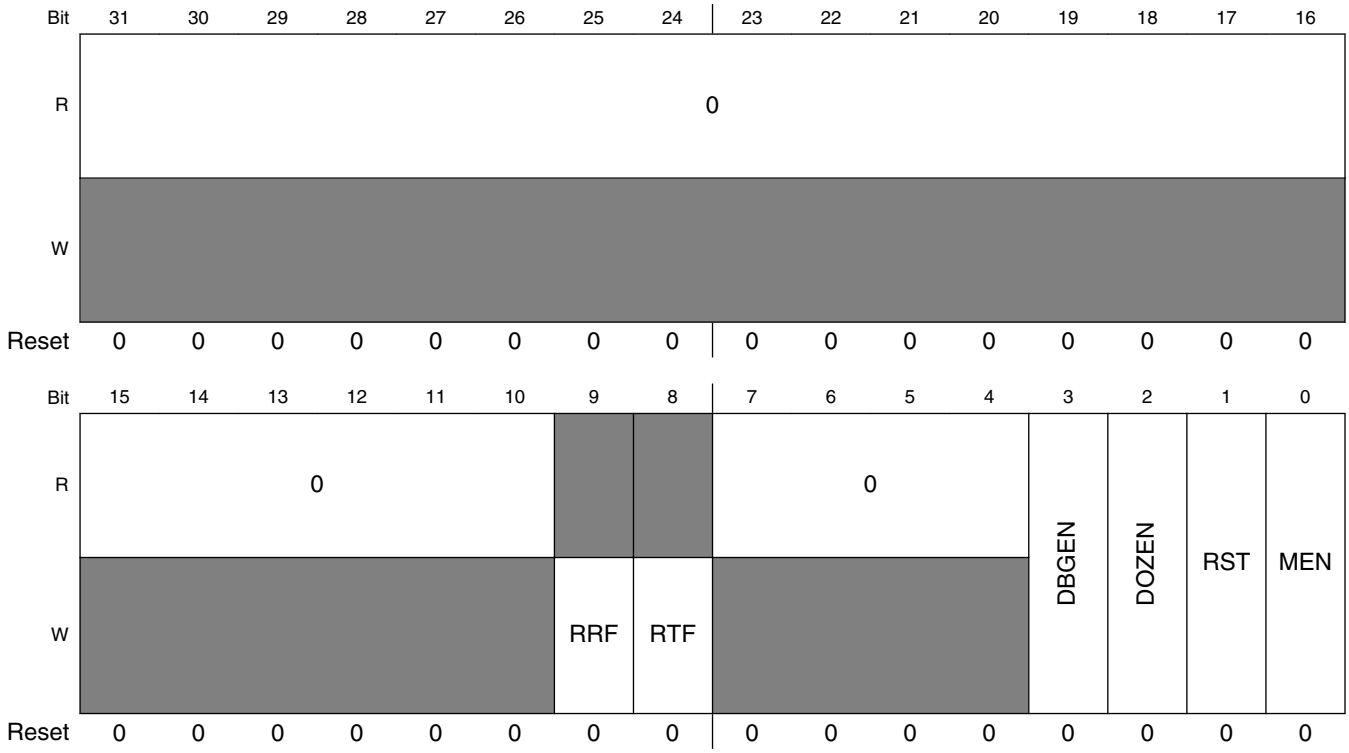
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																RXFIFO								TXFIFO							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

LPSPI_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 RXFIFO	Receive FIFO Size The number of words in the receive FIFO is 2^RXFIFO.
TXFIFO	Transmit FIFO Size The number of words in the transmit FIFO is 2^TXFIFO.

29.2.3 Control Register (LPSPI_CR)

Address: 0h base + 10h offset = 10h



LPSPI_CR field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RRF	Reset Receive FIFO 0 No effect. 1 Receive FIFO is reset.
8 RTF	Reset Transmit FIFO 0 No effect. 1 Transmit FIFO is reset.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 DBGEN	Debug Enable 0 Module is disabled in debug mode. 1 Module is enabled in debug mode.
2 DOZEN	Doze mode enable Enables or disables Doze mode

Table continues on the next page...

LPSPI_CR field descriptions (continued)

Field	Description
	0 Module is enabled in Doze mode. 1 Module is disabled in Doze mode.
1 RST	Software Reset Reset all internal logic and registers, except the Control Register. Remains set until cleared by software. 0 Master logic is not reset. 1 Master logic is reset.
0 MEN	Module Enable 0 Module is disabled. 1 Module is enabled.

29.2.4 Status Register (LPSPI_SR)

Address: 0h base + 14h offset = 14h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MBF	0							
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
R	0		DMF	REF	TEF	TCF	FCF	WCF	0						RDF	TDF
W			w1c	w1c	w1c	w1c	w1c	w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

LPSPI_SR field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 MBF	Module Busy Flag 0 LPSPI is idle. 1 LPSPI is busy.
23–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields as configured by MATCFG. 0 Have not received matching data. 1 Have received matching data.

Table continues on the next page...

LPSPI_SR field descriptions (continued)

Field	Description
12 REF	Receive Error Flag This flag will set when the Receiver FIFO overflows. 0 Receive FIFO has not overflowed. 1 Receive FIFO has overflowed.
11 TEF	Transmit Error Flag This flag will set when the Transmit FIFO underruns. 0 Transmit FIFO underrun has not occurred. 1 Transmit FIFO underrun has occurred
10 TCF	Transfer Complete Flag This flag will set in master mode when the LPSPI returns to idle state with the transmit FIFO empty. 0 All transfers have not completed. 1 All transfers have completed.
9 FCF	Frame Complete Flag This flag will set at the end of each frame transfer, when the PCS negates. 0 Frame transfer has not completed. 1 Frame transfer has completed.
8 WCF	Word Complete Flag This flag will set when the last bit of a received word is sampled. 0 Transfer word not completed. 1 Transfer word completed.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. 0 Receive Data is not ready. 1 Receive data is ready.
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. 0 Transmit data not requested. 1 Transmit data is requested.

29.2.5 Interrupt Enable Register (LPSPI_IER)

Address: 0h base + 18h offset = 18h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0		DMIE	REIE	TEIE	TCIE	FCIE	WCIE	0						RDIE	TDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPI_IER field descriptions

Field	Description
31–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 DMIE	Data Match Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
12 REIE	Receive Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
11 TEIE	Transmit Error Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
10 TCIE	Transfer Complete Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
9 FCIE	Frame Complete Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
8 WCIE	Word Complete Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDIE	Receive Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled.

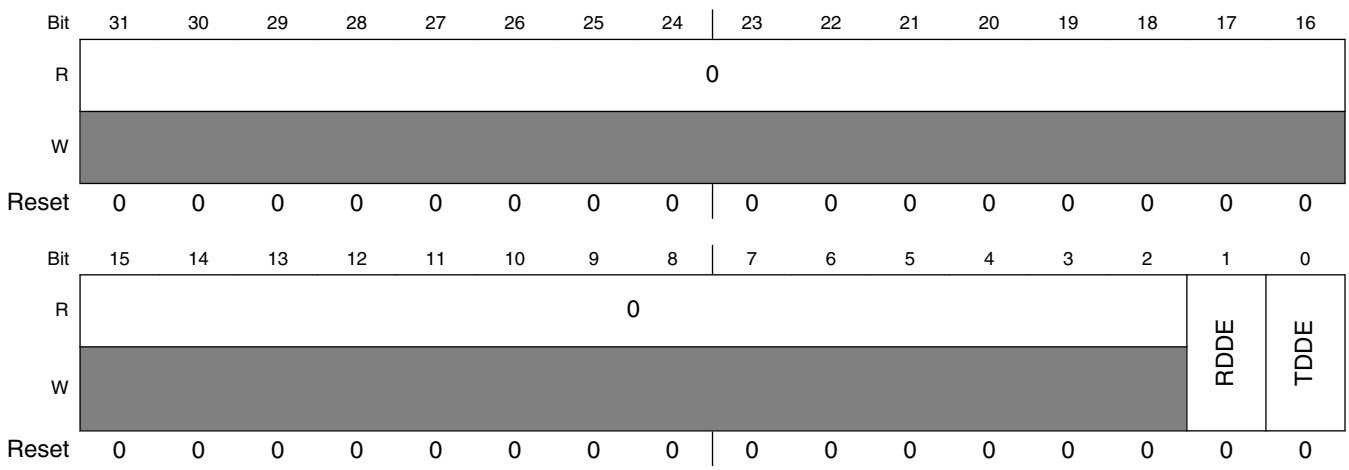
Table continues on the next page...

LPSPI_IER field descriptions (continued)

Field	Description
0 TDIE	Transmit Data Interrupt Enable 0 Interrupt disabled. 1 Interrupt enabled

29.2.6 DMA Enable Register (LPSPI_DER)

Address: 0h base + 1Ch offset = 1Ch

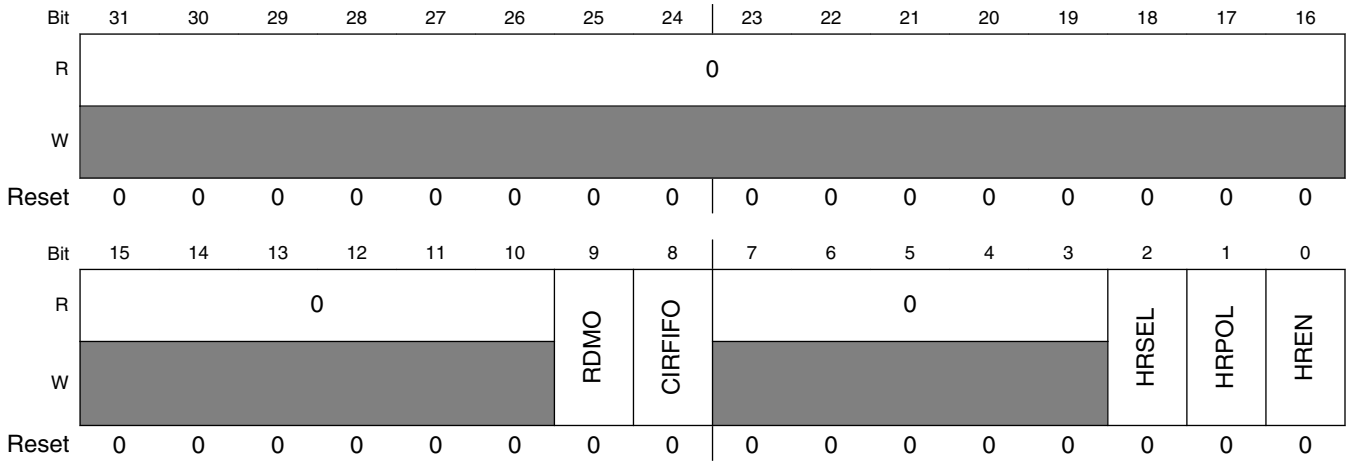


LPSPI_DER field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RDDE	Receive Data DMA Enable 0 DMA request disabled. 1 DMA request enabled.
0 TDDE	Transmit Data DMA Enable 0 DMA request disabled. 1 DMA request enabled

29.2.7 Configuration Register 0 (LPSPI_CFGR0)

Address: 0h base + 20h offset = 20h



LPSPI_CFGR0 field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause DMF to set is discarded. Once DMF is set, the RDMO configuration is ignored. When disabling RDMO, clear RDMO before clearing DMF to ensure no receive data is lost. 0 Received data is stored in the receive FIFO as normal. 1 Received data is discarded unless the DMF is set.
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as normal, but once the LPSPI is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This will cause the contents of the transmit FIFO to be cycled through repeatedly. 0 Circular FIFO is disabled. 1 Circular FIFO is enabled.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 HRSEL	Host Request Select Selects the source of the host request input. When the host request function is enabled with the LPSPI_HREQ pin, the LPSPI_PCS[1] function is disabled. 0 Host request input is pin LPSPI_HREQ. 1 Host request input is input trigger.
1 HRPOL	Host Request Polarity

Table continues on the next page...

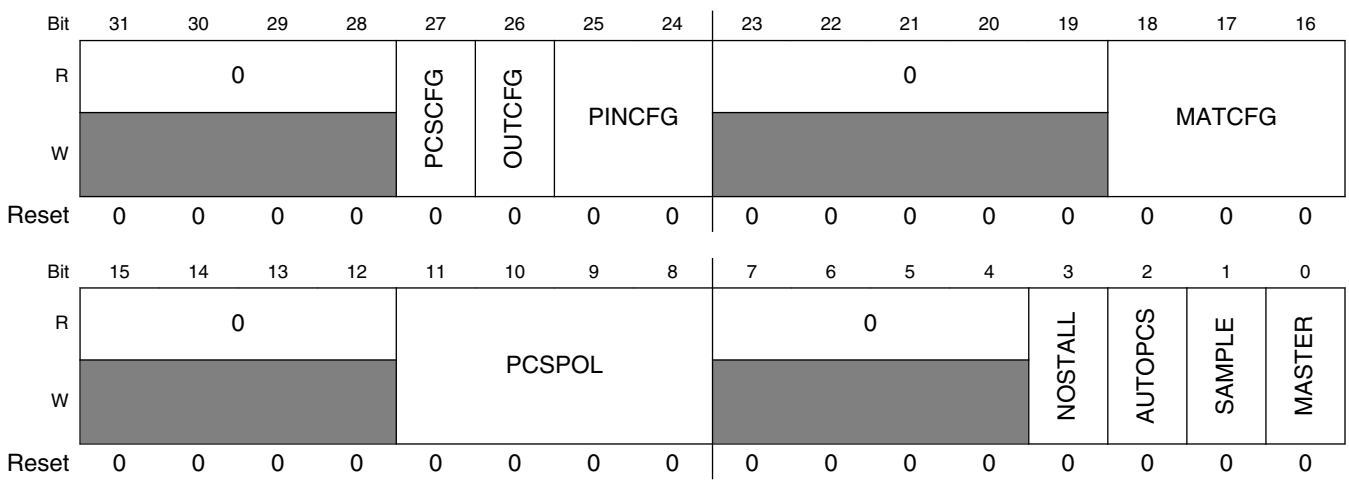
LPSPI_CFGR0 field descriptions (continued)

Field	Description
	Configures the polarity of the host request pin. 0 Active low. 1 Active high.
0 HREN	Host Request Enable When enabled in master mode, the LPSPI will only initiate a SPI bus transfer if the host request input is asserted. 0 Host request is disabled. 1 Host request is enabled.

29.2.8 Configuration Register 1 (LPSPI_CFGR1)

The CFGR1 should only be written when the LPSPI is disabled.

Address: 0h base + 24h offset = 24h



LPSPI_CFGR1 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27 PCSCFG	Peripheral Chip Select Configuration PCSCFG must be set if performing 4-bit transfers. 0 PCS[3:2] are enabled. 1 PCS[3:2] are disabled.
26 OUTCFG	Output Config Configures if the output data is tristated between accesses (LPSPI_PCS is negated).

Table continues on the next page...

LPSPI_CFGR1 field descriptions (continued)

Field	Description
	0 Output data retains last value when chip select is negated. 1 Output data is tristated when chip select is negated.
25–24 PINCFG	Pin Configuration Configures which pins are used for input and output data during single bit transfers. 00 SIN is used for input data and SOUT for output data. 01 SIN is used for both input and output data. 10 SOUT is used for both input and output data. 11 SOUT is used for input data and SIN for output data.
23–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. 000 Match disabled. 001 Reserved 010 Match enabled (1st data word equals MATCH0 OR MATCH1). 011 Match enabled (any data word equals MATCH0 OR MATCH1). 100 Match enabled (1st data word equals MATCH0 AND 2nd data word equals MATCH1). 101 Match enabled (any data word equals MATCH0 AND next data word equals MATCH1) 110 Match enabled (1st data word AND MATCH1 equals MATCH0 AND MATCH1) 111 Match enabled (any data word AND MATCH1 equals MATCH0 AND MATCH1).
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 PCSPOL	Peripheral Chip Select Polarity Configures the polarity of each Peripheral Chip Select pin. 0 The PCSx is active low. 1 The PCSx is active high.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 NOSTALL	No Stall In master mode, the LPSPI will stall transfers when the transmit FIFO is empty or receive FIFO is full ensuring that no transmit FIFO underrun or receive FIFO overrun can occur. Setting this bit will disable this functionality. 0 Transfers will stall when transmit FIFO is empty or receive FIFO is full. 1 Transfers will not stall, allowing transmit FIFO underrun or receive FIFO overrun to occur.
2 AUTOPCS	Automatic PCS The LPSPI slave normally requires the PCS to negate between frames for correct operation. Setting this bit will cause the LPSPI to generate an internal PCS signal at the end of each transfer word when CPHA=1. When this bit is set, the SCK must remain idle for at least 4 LPSPI functional clock cycles (divided by PRESCALE configuration) between each word to ensure correct operation. This bit is ignored in master mode.

Table continues on the next page...

LPSPI_CFGR1 field descriptions (continued)

Field	Description
	0 Automatic PCS generation disabled. 1 Automatic PCS generation enabled.
1 SAMPLE	Sample Point When set, the LPSPI master will sample the input data on a delayed LPSPI_SCK edge. This improves the setup time when sampling data. The input data setup time in master mode with delayed LPSPI_SCK edge is equal to the input data setup time in slave mode. This bit is ignored in slave mode. 0 Input data sampled on SCK edge. 1 Input data sampled on delayed SCK edge.
0 MASTER	Master Mode Configures the LPSPI in master or slave mode. This bit directly controls the direction of the LPSPI_SCK and LPCPI_PCS pins. 0 Slave mode. 1 Master mode.

29.2.9 Data Match Register 0 (LPSPI_DMR0)

Address: 0h base + 30h offset = 30h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPI_DMR0 field descriptions

Field	Description
MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

29.2.10 Data Match Register 1 (LPSPI_DMR1)

Address: 0h base + 34h offset = 34h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH1																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPI_DMR1 field descriptions

Field	Description
MATCH1	Match 1 Value

LPSPI_DMR1 field descriptions (continued)

Field	Description
	Compared against the received data when receive data match is enabled.

29.2.11 Clock Configuration Register (LPSPI_CCR)

The CCR is only used in master mode and cannot be changed when the LPSPI is enabled.

Address: 0h base + 40h offset = 40h

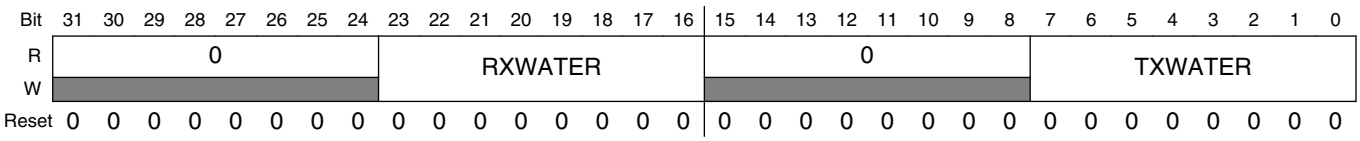
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SCKPCS								PCSSCK								DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPSPI_CCR field descriptions

Field	Description
31–24 SCKPCS	SCK to PCS Delay Configures the delay in master mode from the last SCK edge to the PCS negation. The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 1 cycle.
23–16 PCSSCK	PCS to SCK Delay Configures the delay in master mode from the PCS assertion to the first SCK edge. The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 1 cycle.
15–8 DBT	Delay Between Transfers Configures the delay in master mode from the PCS negation to the next PCS assertion. The delay is equal to (DBT + 2) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 2 cycles. Note that half the delay occurs before PCS assertion and the other half of the delay occurs after PCS negation; the full command word can only update in the middle. Also configures the delay in master mode from the last SCK edge of a transfer word and the first SCK edge of the next transfer word in a continuous transfer. The delay is equal to (DBT + 1) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum delay is 1 cycle.
SCKDIV	SCK Divider Configures the divide ratio of the SCK pin in master mode. The SCK period is equal to (SCKDIV+2) cycles of the LPSPI functional clock divided by the PRESCALE configuration, and the minimum period is 2 cycles. If the period is an odd number of cycles, then the first half of the period will be one cycle longer than the second half of the period.

29.2.12 FIFO Control Register (LPSPI_FCR)

Address: 0h base + 58h offset = 58h

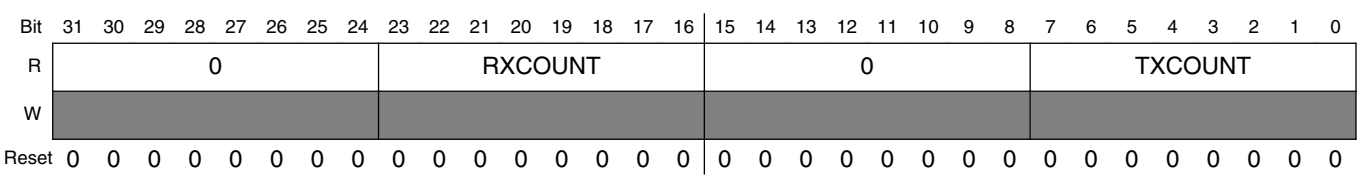


LPSPI_FCR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size will be truncated.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size will be truncated.

29.2.13 FIFO Status Register (LPSPI_FSR)

Address: 0h base + 5Ch offset = 5Ch



LPSPI_FSR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

29.2.14 Transmit Command Register (LPSPI_TCR)

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO in the order they are written. Command Register writes will be tagged and cause the command register to update once that entry reaches the top of the FIFO. This allows changes to the command word and the transmit data itself to be interleaved. Changing the command word will cause all subsequent SPI bus transfer to be performed using the new command word.

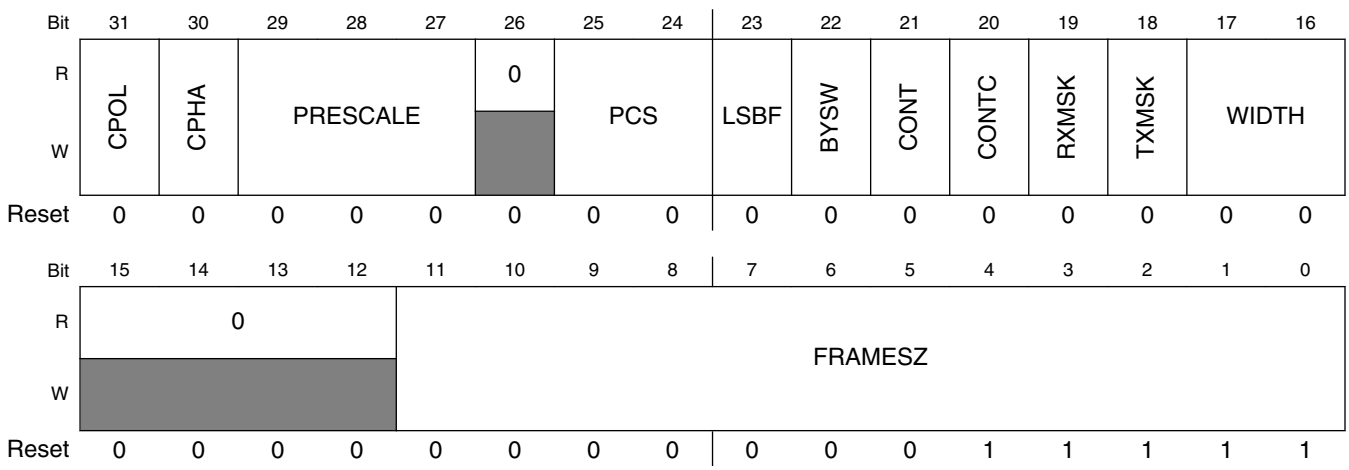
In master mode, writing a new command word does not initiate a new transfer, unless TXMSK is set. Transfers are initiated by transmit data in the transmit FIFO, or a new command word with TXMSK set. Hardware will clear TXMSK when the LPSPI_PCS negates.

In master mode if the command word is changed before an existing frame has completed, then the existing frame will terminate and the command word will then update. The command word can be changed during a continuous transfer, provided CONTC of the new command word is set and the command word is written on a frame size boundary.

In slave mode, the command word should be changed only when the LPSPI is idle and there is no SPI bus transfer.

Reading the Transmit Command Register will return the current state of the command register.

Address: 0h base + 60h offset = 60h



LPSPI_TCR field descriptions

Field	Description
31 CPOL	Clock Polarity This field is only updated between frames.

Table continues on the next page...

LPSPI_TCR field descriptions (continued)

Field	Description
	0 The inactive state value of SCK is low. 1 The inactive state value of SCK is high.
30 CPHA	Clock Phase This field is only updated between frames. 0 Data is captured on the leading edge of SCK and changed on the following edge. 1 Data is changed on the leading edge of SCK and captured on the following edge.
29–27 PRESCALE	Prescaler Value Prescaler applied to the clock configuration register for all SPI bus transfers. This field is only updated between frames. 000 Divide by 1. 001 Divide by 2. 010 Divide by 4. 011 Divide by 8. 100 Divide by 16. 101 Divide by 32. 110 Divide by 64. 111 Divide by 128.
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. This field is only updated between frames. 00 Transfer using LPSPI_PCS[0] 01 Transfer using LPSPI_PCS[1] 10 Transfer using LPSPI_PCS[2] 11 Transfer using LPSPI_PCS[3]
23 LSBF	LSB First 0 Data is transferred MSB first. 1 Data is transferred LSB first.
22 BYSW	Byte Swap Byte swap will swap the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and each received data word stored to the FIFO (or compared with match registers). 0 Byte swap disabled. 1 Byte swap enabled.
21 CONT	Continuous Transfer In master mode, continuous transfer will keep the PCS asserted at the end of the frame size, until a command word is received that starts a new frame. In slave mode, when continuous transfer is enabled the LPSPI will only transmit the first FRAMESZ bits, after which it will transmit received data assuming a 32-bit shift register.

Table continues on the next page...

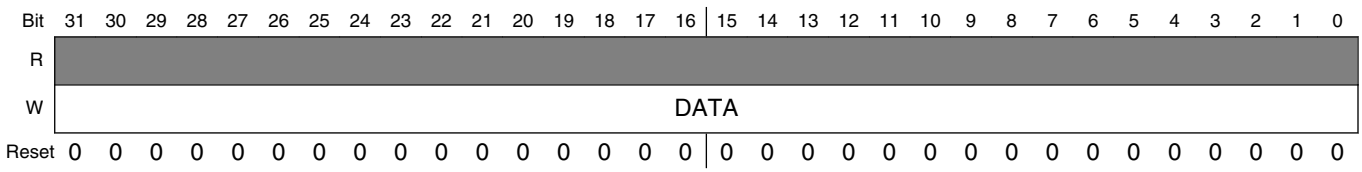
LPSPI_TCR field descriptions (continued)

Field	Description
	0 Continuous transfer disabled. 1 Continuous transfer enabled.
20 CONTC	Continuing Command In master mode, this bit allows the command word to be changed within a continuous transfer. The initial command word must enable continuous transfer (CONT=1), the continuing command must set this bit (CONTC=1) and the continuing command word must be loaded on a frame size boundary. For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary. 0 Command word for start of new transfer. 1 Command word for continuing transfer.
19 RXMSK	Receive Data Mask When set, receive data is masked (receive data is not stored in receive FIFO). 0 Normal transfer. 1 Receive data is masked.
18 TXMSK	Transmit Data Mask When set, transmit data is masked (no data is loaded from transmit FIFO and output pin is tristated). In master mode, this bit will initiate a new transfer which cannot be aborted by another command word and the bit will be cleared by hardware at the end of the transfer. 00 Normal transfer. 01 Mask transmit data.
17–16 WIDTH	Transfer Width Either RXMSK or TXMSK must be set for 2-bit or 4-bit transfers. 00 Single bit transfer. 01 Two bit transfer. 10 Four bit transfer. 11 Reserved.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1). The minimum frame size is 8 bits. If the frame size is larger than 32 bits, data will be loaded from the transmit FIFO and stored to the receive FIFO every 32 bits. If the size of the transfer word is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits (e.g.: a 72-bit transfer will load/store 32-bits from the FIFO and then another 32-bits from the FIFO and then the final 8-bits from the FIFO).

29.2.15 Transmit Data Register (LPSPI_TDR)

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO in the order it was written.

Address: 0h base + 64h offset = 64h

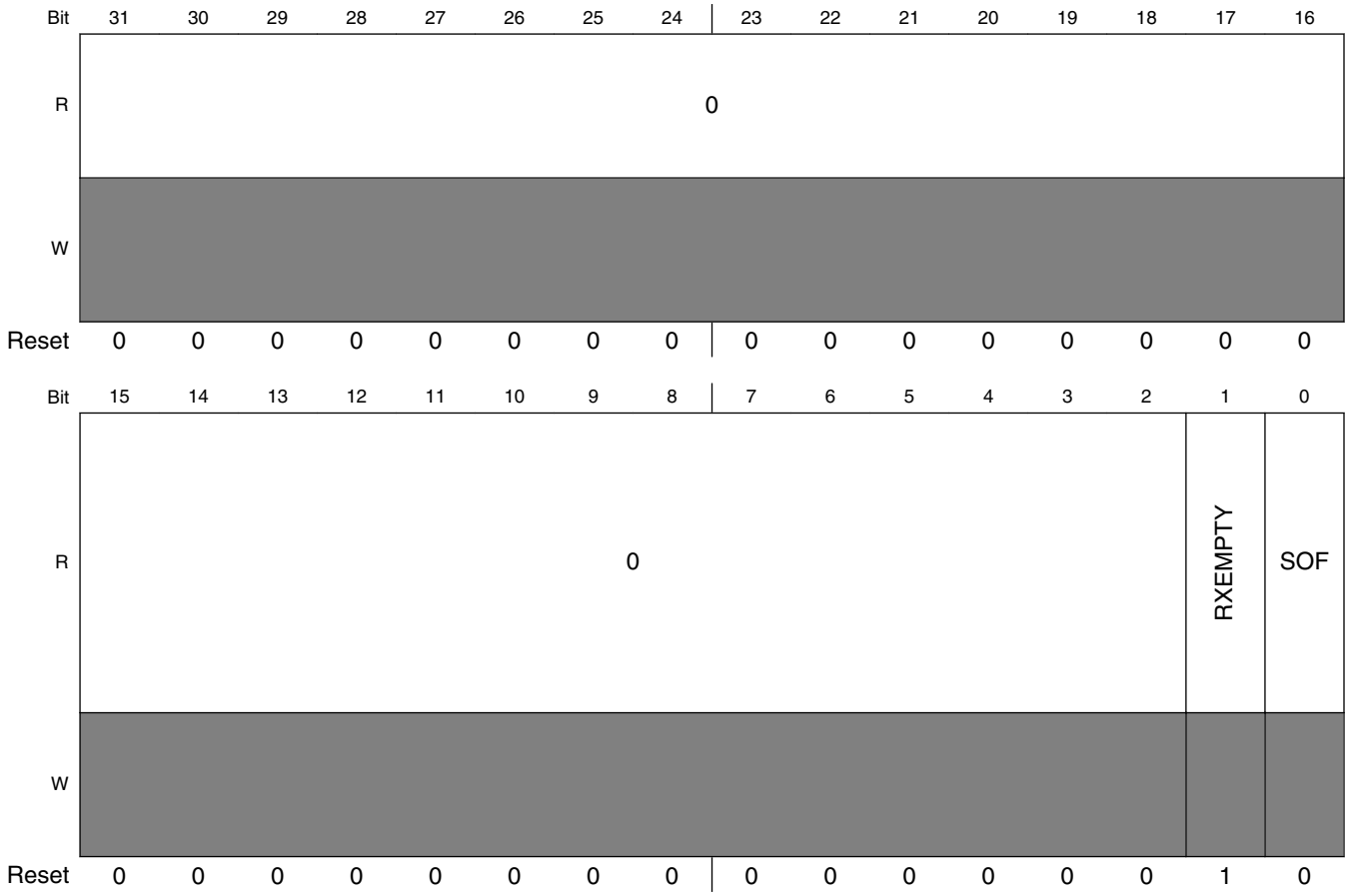


LPSPI_TDR field descriptions

Field	Description
DATA	Transmit Data Both 8-bit and 16-bit writes of transmit data will zero extend the data written and push the data into the transmit FIFO.

29.2.16 Receive Status Register (LPSPI_RSR)

Address: 0h base + 70h offset = 70h



LPSPI_RSR field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RXEMPTY	RX FIFO Empty 0 RX FIFO is not empty. 1 RX FIFO is empty.
0 SOF	Start Of Frame Indicates that this is the first data word received after LPSPI_PCS assertion. 0 Subsequent data word received after LPSPI_PCS assertion. 1 First data word received after LPSPI_PCS assertion.

29.2.17 Receive Data Register (LPSPI_RDR)

Address: 0h base + 74h offset = 74h

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

LPSPI_RDR field descriptions

Field	Description
DATA	Receive Data

29.3 Functional description

29.3.1 Clocking and Resets

29.3.1.1 Functional clock

The LPSPI functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support SPI bus transfers in both master and slave modes. If the functional clock is disabled in slave mode, the LPSPI can transfer a single word before the functional clock needs to be enabled. The LPSPI divides the functional clock by a prescaler and the resulting frequency must be at least two times faster than the SPI clock frequency.

29.3.1.2 External clock

The LPSPI shift register is clocked directly by the external pins. This allows the LPISPI slave to remain operational in low power modes, even when the LPSPI functional clock is disabled, although this is limited to a single word transfer.

29.3.1.3 Bus clock

The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPSPI registers, including FIFOs.

29.3.1.4 Chip reset

The logic and registers for the LPSPI are reset to their default state on a chip reset.

29.3.1.5 Software reset

The LPSPI implements a software reset bit in the Control Register. The MCR[RST] will reset all logic and registers to their default state, except for the MCR itself.

29.3.1.6 FIFO reset

The LPSPI implements write-only control bits that resets the transmit/command FIFO (MCR[RTF] and receive FIFO (MCR[RRF]). A FIFO is empty after being reset.

29.3.2 Master Mode

29.3.2.1 Transmit and Command FIFO

The transmit and command FIFO is a combined FIFO that includes both transmit data and command words. Command words are stored to the transmit/command FIFO by writing the transmit command register. Transmit data words are stored to the transmit/command FIFO by writing the transmit data register.

When a command word is at the top of the transmit/command FIFO, the following actions can occur:

- If the LPSPI is between frames, the command word is pulled from the FIFO and controls all subsequent transfers.
- If the LPSPI is busy and either the existing CONT bit is clear or the new CONTC value is clear, the SPI frame will complete at the end of the existing word, ignoring

the FRAMESZ configuration. The command word is then pulled from the FIFO and controls all subsequent transfers (or until the next update to the command word).

- If the LPSPI is busy and the existing CONT bit is set and the new CONTC value is set, the command word is pulled from the FIFO during the last LPSPI_SCK pulse of the existing frame (based on FRAMESZ configuration) and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When CONTC is set, only the lower 24-bits of the command word are updated.

The current state of the existing command word can be read by reading the transmit command register. It requires at least three LPSPI functional clock cycles for the transmit command register to update after it is written (assuming an empty FIFO) and the LPSPI must be enabled (MCR[MEN] is set).

Writing the transmit command register does not initiate a SPI bus transfer, unless the TXMSK bit is set. When TXMSK is set, a new command word will not be loaded until the end of the existing frame (based on FRAMESZ configuration) and the TXMSK bit will be cleared at the end of the transfer.

The following table describes the attributes that are controlled by the command word.

Table 29-2. LPSPI Command Word

Field	Description	Modify During Transfer
CPOL	Configures polarity of the LPSPI_SCK pin. Any change of CPOL value will cause a transition on the LPSPI_SCK pin.	N
CPHA	Configures clock phase of transfer.	N
PRESCALE	Configures prescaler used to divide the LPSPI functional clock to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS allows the LPSPI to connect to different slave devices at different frequencies.	N
PCS	Configures which LPSPI_PCS asserts for the transfer, the polarity of LPSPI_PCS is static and configured by PCSPOL. If PCSCFG is set, then PCS[3:2] should not be selected.	N
LSBF	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted/received first.	Y
BYSW	Enables byte swap on each 32-bit word when transmitting and receiving data. Can be useful when interfacing to devices that organize data as big endean.	Y

Table continues on the next page...

Table 29-2. LPSPI Command Word (continued)

Field	Description	Modify During Transfer
CONT	Configures for a continuous transfer that keeps PCS asserted between frames (as configured by FRAMESZ). A new command word is required to cause PCS to negate. Also supports changing the command word at frame size boundaries.	Y
CONTC	Indicates this is a new command word for the existing continuous transfer. This bit is ignored when not written to the transmit/command FIFO on a frame boundary.	Y
RXMSK	Masks the receive data and does not store to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.	Y
TXMSK	Masks the transmit data, so that data is not pulled from transmit FIFO and the output data pin is tristated (unless configured by OUTCFG). Useful for half-duplex transfers.	Y
WIDTH	Configures the number of bits shifted on each LPSPI_SCK pulse. Single bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. Two and four bit transfers are useful for interfacing to QuadSPI memory devices and only support half-duplex data formats (at least one of TXMSK or RXMSK must also be set).	Y
FRAMESZ	Configures the number of bits in each frame to FRAMESZ+1. The minimum frame size is 8-bits and the maximum frame size is 4096-bits. If the frame size is less than or equal to 32-bits, the word size and frame size are identical. If the frame size is greater than 32-bits, then the word size is 32-bits for each word except the last (the last word contains the remainder bits if the frame size is not divisible by 32). The minimum word size is 2-bits, a frame size of 33-bits (or similar) is not supported.	Y

The LPSPI initiates a SPI bus transfer when data is written to the transmit FIFO, the HREQ pin is asserted (or disabled) and the LPSPI is enabled. The SPI bus transfer uses the attributes configured in the transmit command register and timing parameters from the clock configuration register to perform the transfer. The SPI bus transfer ends once

the FRAMESZ configuration is reached, or at the end of a word when a new transmit command word is at the top of the transmit/command FIFO. The HREQ input is only checked the next time the LPSPI goes idle (completes the current transfer and transmit/command register is empty).

The transmit/command FIFO also supports a Circular FIFO feature. This allows the LPSPI master to (periodically) repeat a short data transfer that can fit within the transmit/command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled, the current state of the FIFO read pointer is saved and the status flags do not update. Once the transmit/command FIFO is considered empty and the LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit/command FIFO are not permanently pulled from the FIFO while circular FIFO mode is enabled.

29.3.2.2 Receive FIFO and Data Match

The receive FIFO is used to store receive data during SPI bus transfers. When RXMSK is set, receive data is discarded instead of storing in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame. Receive data that is already discarded due to RXMSK bit cannot cause the data match to set and will delay the match on first received data word until after all discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF] to allow all subsequent data to be received.

29.3.2.3 Timing Parameters

The following table lists the timing parameters that are used for all SPI bus transfers, these timing parameters are relative to the LPSPI functional clock divided by the PRESCALE configuration. Although the Clock Configuration Register cannot be changed when the LPSPI is busy, the PRESCALE configuration can be altered between transfers using the command register, to support interfacing to different slave devices at different frequencies.

Table 29-3. LPSPI Timing Parameters

Field	Description	Min	Max
SCKDIV	Configures the LPSPI_SCK clock period to (SCKDIV+2) cycles. When configured to an odd number of cycles, the first half of the LPSPI_SCK cycle is one cycle longer than the second half.	0 (2 cycles)	255 (257 cycles)
DBT	Configures the minimum delay between PCS negation and the next PCS assertion to (DBT + 2) cycles. When the command word is updated between transfers, there is a minimum of (DBT/2)+1 cycles between the command word update and any change on LPSPI_PCS pins.	0 (2 cycles)	255 (257 cycles)
DBT	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (DBT + 1) cycles. This is useful where the external slave requires a large delay between different words of a SPI bus transfer.	0 (1 cycle)	255 (256 cycles)
PCSSCK	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	Configures the minimum delay between the last SCK edge and the PCS assertion to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

29.3.2.4 Pin Configuration

The LPSPI_SIN and LPSPI_SOUT pins can be configured via the PINCFG configuration to swap directions or even support half-duplex transfers on the same pin.

The OUTCFG configuration can be used to determine if output data pin (eg: LPSPI_SOUT) will tristate when the LPSPI_PCS is negated, or if it will simply retain the last value. When configuring for half-duplex transfers using the same data pin in single bit transfer mode, or any transfer in 2-bit and 4-bit transfer modes, then the output data pins must be configured to tristate when LPSPI_PCS is negated.

The PCSCFG configuration is used to disable LPSPI_PCS[3:2] functions and to use them for quad-data transfers. This option must be enabled when performing quad-data transfers.

29.3.3 Slave Mode

LPSPI slave mode uses the same shift register and logic as the master mode, but does not use the clock configuration register and the transmit command register must remain static during SPI bus transfers.

29.3.3.1 Transmit and Command FIFO

The transmit command register should be initialized before enabling the LPSPI in slave mode, although the command register will not update until after the LPSPI is enabled. Once enabled, the transmit command register should only be changed if the LPSPI is idle. The following table lists how the command register functions in slave mode.

Table 29-4. LPSPI Command Word in Slave Mode

Field	Description
CPOL	Configures polarity of the external LPSPI_SCK input.
CPHA	Configures clock phase of transfer.
PRESCALE	Configures LPSPI functional clock prescaler.
PCS	Configures which LPSPI_PCS is used, the polarity of LPSPI_PCS is static and configured by PCSPOL. If PCSCFG is set, then PCS[3:2] should not be selected.
LSBF	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted/received first.
BYSW	Enables byte swap on each 32-bit word when transmitting and receiving data. Can be useful when interfacing to devices that organize data as big endean.
CONT	When set, only the first FRAMSZ bits will be transmitted/ received by the LPSPI.
CONTC	This bit is reserved in slave mode.
RXMSK	Masks the receive data and does not store to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.
TXMSK	Masks the transmit data, so that data is not pulled from transmit FIFO and the output data pin is tristated (unless configured by OUTCFG). Useful for half-duplex transfers.
WIDTH	Configures the number of bits shifted on each LPSPI_SCK pulse. Single bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. Two and four bit transfers are useful for interfacing to QuadSPI memory

Table continues on the next page...

Table 29-4. LPSPI Command Word in Slave Mode (continued)

Field	Description
	devices and only support half-duplex data formats (at least one of TXMSK or RXMSK must also be set).
FRAMESZ	Configures the number of bits in each frame to FRAMESZ+1. The minimum frame size is 8-bits and the maximum frame size is 4096-bits. If the frame size is less than or equal to 32-bits, the word size and frame size are identical. If the frame size is greater than 32-bits, then the word size is 32-bits for each word except the last (the last word contains the remainder bits if the frame size is not divisible by 32). The minimum word size is 2-bits, a frame size of 33-bits (or similar) is not supported.

The transmit FIFO must be filled with transmit data before the LPSPI_PCS input asserts, otherwise the transmit error flag will set.

29.3.3.2 Receive FIFO and Data Match

The receive FIFO is used to store receive data during SPI bus transfers. When RXMSK is set, receive data is discarded instead of storing in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of two words or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame. Receive data that is already discarded due to RXMSK bit cannot cause the data match to set and will delay the match on first received data word until after all discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF] to allow all subsequent data to be received.

29.3.3.3 Clocked Interface

The LPSPI supports interfacing to external masters that provide only clock and data pins (LPSPI_PCS is not required). This requires using CPHA=1, configuring the LPSPI_PCS input to be always asserted (configure PCSPOL) and setting the AUTOPCS bit. When AUTOPCS is set, a minimum of four LPSPI functional clock cycles (divided by PRESCALE configuration) is required between the last LPSPI_SCK edge of one word and the first LPSPI_SCK edge of the next word.

29.3.4 Interrupts and DMA Requests

The following table illustrates the status flags that can generate the LPSPI interrupt and LPSPI transmit/receive DMA requests.

Table 29-5. LPSPI Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
TDF	Data can be written to transmit FIFO, as configured by TXWATER.	Y	TX	Y
RDF	Data can be read from the receive FIFO, as configured by RXWATER.	Y	RX	Y
WCF	Word complete, last bit of word has been sampled.	Y	N	Y
FCF	Frame complete, PCS has negated .	Y	N	Y
TCF	Transfer complete, PCS has negated and transmit/command FIFO is empty.	Y	N	Y
TEF	Transmit error flag, indicates transmit/ command FIFO underrun. This bit cannot set in master mode when NOSTALL is clear.	Y	N	Y
REF	Receive error flag, indicates receive FIFO overflow. This bit cannot set in master mode when NOSTALL is clear.	Y	N	Y
DMF	Data match flag, received data has matched the configured data match value.	Y	N	Y
MBF	LPSPI is busy performing a SPI bus transfer.	N	N	N

29.3.5 Peripheral Triggers

The connection of the LPSPI peripheral triggers with other peripherals are device specific.

29.3.5.1 Output Triggers

The LPSPI generates two output triggers that can be connected to other peripherals on the device. The frame output trigger asserts at the end of each frame (when PCS negates) and remains asserted until PCS next asserts. The word output trigger asserts at the end of each received word and remains asserted for one LPSPI_SCK period.

29.3.5.2 Input Trigger

The LPSPI input trigger can be selected in place of the LPSPI_HREQ input to control the start of a LPSPI bus transfer. The input trigger must assert for longer than one LPSPI functional clock cycle to be detected.

29.4 Application Information

Chapter 30
Low-Power Timer (LPTMR)

30.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The low-power timer (LPTMR) can be configured to operate as a time counter with optional prescaler, or as a pulse counter with optional glitch filter, across all power modes, including the low-leakage modes. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

30.1.1 Features

The features of the LPTMR module include:

- 16-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wakeup from any low-power mode
  - Hardware trigger output
  - Counter supports free-running mode or reset on compare
- Configurable clock source for prescaler/glitch filter
- Configurable input source for pulse counter
  - Rising-edge or falling-edge

30.1.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

Table 30-1. Modes of operation

Modes	Description
Run	The LPTMR operates normally.
Wait	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Stop	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Low-Leakage	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Debug	The LPTMR operates normally in Pulse Counter mode, but the counter does not increment in Time Counter mode.

30.2 LPTMR signal descriptions

Table 30-2. LPTMR signal descriptions

Signal	I/O	Description
LPTMR_ALT*n*	I	Pulse Counter Input pin

30.2.1 Detailed signal descriptions

Table 30-3. LPTMR interface—detailed signal descriptions

Signal	I/O	Description	
LPTMR_ALT*n*	I	Pulse Counter Input The LPTMR can select one of the input pins to be used in Pulse Counter mode.	
		State meaning	Assertion—If configured for pulse counter mode with active-high input, then assertion causes the CNR to increment. Deassertion—If configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment.
		Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.

30.3 Memory map and register definition

NOTE

The LPTMR registers are reset only on a POR or LVD event.
See LPTMR power and reset for more details.

LPTMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_4000	Low Power Timer Control Status Register (LPTMR0_CSR)	32	R/W	0000_0000h	30.3.1/785
4003_4004	Low Power Timer Prescale Register (LPTMR0_PSR)	32	R/W	0000_0000h	30.3.2/787
4003_4008	Low Power Timer Compare Register (LPTMR0_CMR)	32	R/W	0000_0000h	30.3.3/788
4003_400C	Low Power Timer Counter Register (LPTMR0_CNR)	32	R/W	0000_0000h	30.3.4/789
400B_5000	Low Power Timer Control Status Register (LPTMR1_CSR)	32	R/W	0000_0000h	30.3.1/785
400B_5004	Low Power Timer Prescale Register (LPTMR1_PSR)	32	R/W	0000_0000h	30.3.2/787
400B_5008	Low Power Timer Compare Register (LPTMR1_CMR)	32	R/W	0000_0000h	30.3.3/788
400B_500C	Low Power Timer Counter Register (LPTMR1_CNR)	32	R/W	0000_0000h	30.3.4/789

30.3.1 Low Power Timer Control Status Register (LPTMRx_CSR)

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0							TDRE	TCF	TIE	TPS		TPP	TFC	TMS	TEN
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPTMRx_CSR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 TDRE	Timer DMA Request Enable When TDRE is set, the LPTMR DMA Request is generated whenever TCF is also set and the TCF is cleared when the DMA Controller is done.

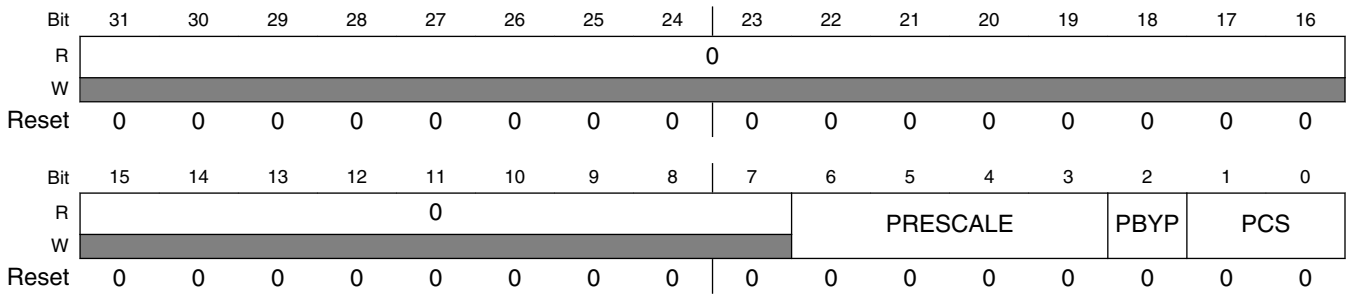
Table continues on the next page...

LPTMRx_CSR field descriptions (continued)

Field	Description
	0 Timer DMA Request disabled. 1 Timer DMA Request enabled.
7 TCF	Timer Compare Flag TCF is set when the LPTMR is enabled and the CNR equals the CMR and increments. TCF is cleared when the LPTMR is disabled or a logic 1 is written to it. 0 The value of CNR is not equal to CMR and increments. 1 The value of CNR is equal to CMR and increments.
6 TIE	Timer Interrupt Enable When TIE is set, the LPTMR Interrupt is generated whenever TCF is also set. 0 Timer interrupt disabled. 1 Timer interrupt enabled.
5–4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. TPS must be altered only when the LPTMR is disabled. The input connections vary by device. See the chip configuration information about connections to these inputs. 00 Pulse counter input 0 is selected. 01 Pulse counter input 1 is selected. 10 Pulse counter input 2 is selected. 11 Pulse counter input 3 is selected.
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. TPP must be changed only when the LPTMR is disabled. 0 Pulse Counter input source is active-high, and the CNR will increment on the rising-edge. 1 Pulse Counter input source is active-low, and the CNR will increment on the falling-edge.
2 TFC	Timer Free-Running Counter When clear, TFC configures the CNR to reset whenever TCF is set. When set, TFC configures the CNR to reset on overflow. TFC must be altered only when the LPTMR is disabled. 0 CNR is reset whenever TCF is set. 1 CNR is reset on overflow.
1 TMS	Timer Mode Select Configures the mode of the LPTMR. TMS must be altered only when the LPTMR is disabled. 0 Time Counter mode. 1 Pulse Counter mode.
0 TEN	Timer Enable When TEN is clear, it resets the LPTMR internal logic, including the CNR and TCF. When TEN is set, the LPTMR is enabled. While writing 1 to this field, CSR[5:1] must not be altered. 0 LPTMR is disabled and internal logic is reset. 1 LPTMR is enabled.

30.3.2 Low Power Timer Prescale Register (LPTMRx_PSR)

Address: Base address + 4h offset



LPTMRx_PSR field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–3 PRESCALE	Prescale Value Configures the size of the Prescaler in Time Counter mode or width of the glitch filter in Pulse Counter mode. PRESCALE must be altered only when the LPTMR is disabled. 0000 Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration. 0001 Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges. 0010 Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges. 0011 Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges. 0100 Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges. 0101 Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges. 0110 Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges. 0111 Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges. 1000 Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges. 1001 Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges. 1010 Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges. 1011 Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges. 1100 Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges. 1101 Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges.

Table continues on the next page...

LPTMRx_PSR field descriptions (continued)

Field	Description
	1110 Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges. 1111 Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges.
2 PBYP	Prescaler Bypass When PBYP is set, the selected prescaler clock in Time Counter mode or selected input source in Pulse Counter mode directly clocks the CNR. When PBYP is clear, the CNR is clocked by the output of the prescaler/glitch filter. PBYP must be altered only when the LPTMR is disabled. 0 Prescaler/glitch filter is enabled. 1 Prescaler/glitch filter is bypassed.
PCS	Prescaler Clock Select Selects the clock to be used by the LPTMR prescaler/glitch filter. PCS must be altered only when the LPTMR is disabled. The clock connections vary by device. **NOTE:** See the chip configuration details for information on the connections to these inputs. 00 Prescaler/glitch filter clock 0 selected. 01 Prescaler/glitch filter clock 1 selected. 10 Prescaler/glitch filter clock 2 selected. 11 Prescaler/glitch filter clock 3 selected.

30.3.3 Low Power Timer Compare Register (LPTMRx_CMR)

Address: Base address + 8h offset

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

LPTMRx_CMR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COMPARE	Compare Value When the LPTMR is enabled and the CNR equals the value in the CMR and increments, TCF is set and the hardware trigger asserts until the next time the CNR increments. If the CMR is 0, the hardware trigger will remain asserted until the LPTMR is disabled. If the LPTMR is enabled, the CMR must be altered only when TCF is set.

30.3.4 Low Power Timer Counter Register (LPTMRx_CNR)

Address: Base address + Ch offset

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

LPTMRx_CNR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNTER	Counter Value The CNR returns the current value of the LPTMR counter at the time this register was last written.

30.4 Functional description

30.4.1 LPTMR power and reset

The LPTMR remains powered in all power modes, including low-leakage modes. If the LPTMR is not required to remain operating during a low-power mode, then it must be disabled before entering the mode.

The LPTMR is reset only on global Power On Reset (POR) or Low Voltage Detect (LVD). When configuring the LPTMR registers, the CSR must be initially written with the timer disabled, before configuring the PSR and CMR. Then, CSR[TIE] must be set as the last step in the initialization. This ensures the LPTMR is configured correctly and the LPTMR counter is reset to zero following a warm reset.

30.4.2 LPTMR clocking

The LPTMR prescaler/glitch filter can be clocked by one of the four clocks. The clock source must be enabled before the LPTMR is enabled.

NOTE

The clock source selected may need to be configured to remain enabled in low-power modes, otherwise the LPTMR will not operate during low-power modes.

In Pulse Counter mode with the prescaler/glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the prescaler clock source for a clock that is not toggling.

NOTE

The clock source or pulse input source selected for the LPTMR should not exceed the frequency f_{LPTMR} defined in the device datasheet.

30.4.3 LPTMR prescaler/glitch filter

The LPTMR prescaler and glitch filter share the same logic which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

NOTE

The prescaler/glitch filter configuration must not be altered when the LPTMR is enabled.

30.4.3.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks the CNR. When the LPTMR is enabled, the CNR will increment every 2^2 to 2^{16} prescaler clock cycles. After the LPTMR is enabled, the first increment of the CNR will take an additional one or two prescaler clock cycles due to synchronization logic.

30.4.3.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments the CNR on every clock cycle. When the LPTMR is enabled, the first increment will take an additional one or two prescaler clock cycles due to synchronization logic.

30.4.3.3 Glitch filter

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks the CNR. When the LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

If	Then
The selected input source remains deasserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output will also deassert.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising-edges	The glitch filter output will also assert.

NOTE

The input is only sampled on the rising clock edge.

The CNR will increment each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which the CNR can increment is once every 2^2 to 2^{16} prescaler clock edges. When first enabled, the glitch filter will wait an additional one or two prescaler clock edges due to synchronization logic.

30.4.3.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the CNR every time it asserts. Before the LPTMR is first enabled, the selected input source is forced to be asserted. This prevents the CNR from incrementing if the selected input source is already asserted when the LPTMR is first enabled.

30.4.4 LPTMR compare

When the CNR equals the value of the CMR and increments, the following events occur:

- CSR[TCF] is set.
- LPTMR interrupt is generated if CSR[TIE] is also set.
- LPTMR hardware trigger is generated.
- CNR is reset if CSR[TFC] is clear.

When the LPTMR is enabled, the CMR can be altered only when CSR[TCF] is set. When updating the CMR, the CMR must be written and CSR[TCF] must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

30.4.5 LPTMR counter

The CNR increments by one on every:

- Prescaler clock in Time Counter mode with prescaler bypassed
- Prescaler output in Time Counter mode with prescaler enabled
- Input source assertion in Pulse Counter mode with glitch filter bypassed
- Glitch filter output in Pulse Counter mode with glitch filter enabled

The CNR is reset when the LPTMR is disabled or if the counter register overflows. If CSR[TFC] is cleared, then the CNR is also reset whenever CSR[TCF] is set.

When the core is halted in Debug mode:

- If configured for Pulse Counter mode, the CNR continues incrementing.
- If configured for Time Counter mode, the CNR stops incrementing.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This will synchronize and register the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

30.4.6 LPTMR hardware trigger

The LPTMR hardware trigger asserts at the same time the CSR[TCF] is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

When	Then
The CMR is set to 0 with CSR[TFC] clear	The LPTMR hardware trigger will assert on the first compare and does not deassert.
The CMR is set to a nonzero value, or, if CSR[TFC] is set	The LPTMR hardware trigger will assert on each compare and deassert on the following increment of the CNR.

30.4.7 LPTMR interrupt

The LPTMR interrupt is generated whenever CSR[TIE] and CSR[TCF] are set. CSR[TCF] is cleared by disabling the LPTMR or by writing a logic 1 to it.

CSR[TIE] can be altered and CSR[TCF] can be cleared while the LPTMR is enabled.

The LPTMR interrupt is generated asynchronously to the system clock and can be used to generate a wakeup from any low-power mode, including the low-leakage modes, provided the LPTMR is enabled as a wakeup source.

Chapter 31
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

31.1 Introduction

31.1.1 Features

Features of the LPUART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Transmit and receive baud rate can operate asynchronous to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in Stop modes
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Three receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
  - Receive data match
- Automatic address matching to reduce ISR overhead:

    - Address mark matching
    - Idle line address matching
    - Address match start, address match end
- Optional 13-bit break character generation / 11-bit break character detection
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64 or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width
- Independent FIFO structure for transmit and receive
    - Separate configurable watermark for receive and transmit requests
    - Option for receiver to assert request after a configurable number of idle characters if receive FIFO is not empty

31.1.2 Modes of operation

31.1.2.1 Stop mode

The LPUART will remain functional during Stop mode, provided the asynchronous transmit and receive clock remains enabled. The LPUART can generate an interrupt or DMA request to cause a wakeup from Stop mode.

31.1.2.2 Wait mode

The LPUART can be configured to Stop in Wait modes, when the DOZEEN bit is set. The transmitter and receiver will finish transmitting/receiving the current word.

31.1.2.3 Debug mode

The LPUART remains functional in debug mode.

31.1.3 Signal Descriptions

Signal	Description	I/O
LPUART_TX	Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is disabled or transmit direction is configured for receive data.	I/O
LPUART_RX	Receive data.	I
LPUART_CTS	Clear to send.	I
LPUART_RTS	Request to send.	O

31.1.4 Block diagram

The following figure shows the transmitter portion of the LPUART.

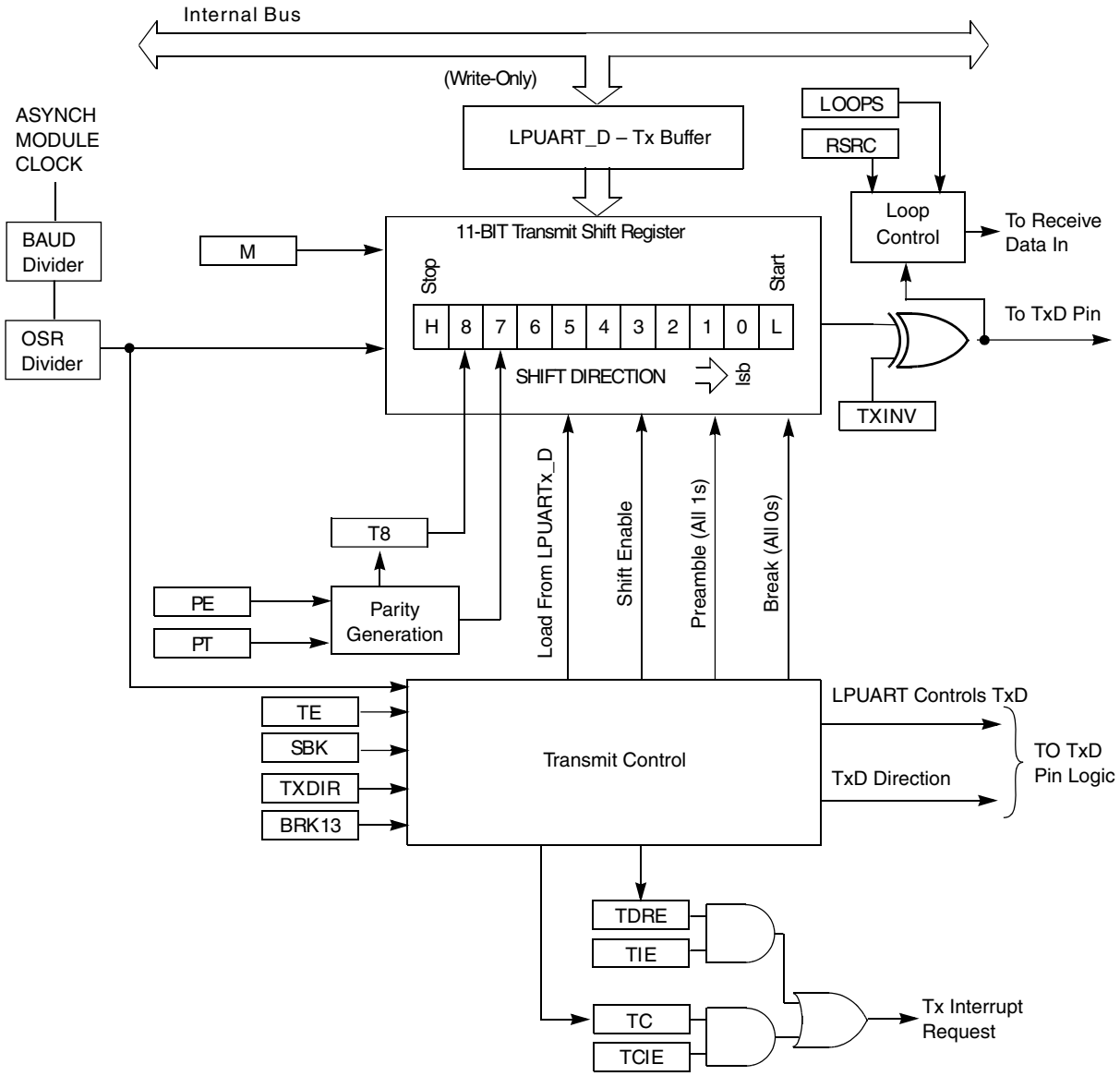


Figure 31-1. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

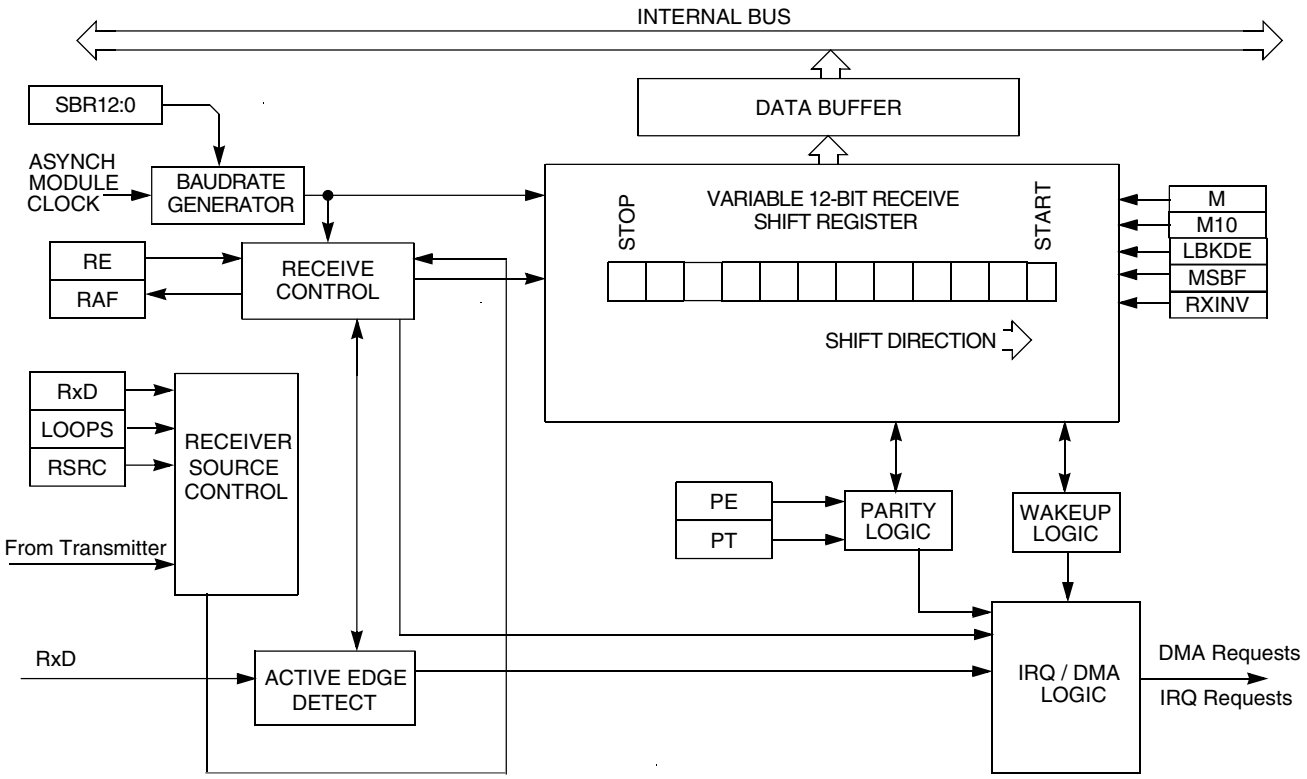


Figure 31-2. LPUART receiver block diagram

31.2 Register definition

The LPUART includes registers to control baud rate, select LPUART options, report LPUART status, and for transmit/receive data. Access to an address outside the valid memory map will generate a bus error.

LPUART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Version ID Register (LPUART_VERID)	32	R	0400_0003h	31.2.1/800
4	Parameter Register (LPUART_PARAM)	32	R	See section	31.2.2/800
8	LPUART Global Register (LPUART_GLOBAL)	32	R/W	0000_0000h	31.2.3/801
C	LPUART Pin Configuration Register (LPUART_PINCFG)	32	R/W	0000_0000h	31.2.4/802
10	LPUART Baud Rate Register (LPUART_BAUD)	32	R/W	0F00_0004h	31.2.5/802
14	LPUART Status Register (LPUART_STAT)	32	R/W	00C0_0000h	31.2.6/805
18	LPUART Control Register (LPUART_CTRL)	32	R/W	0000_0000h	31.2.7/809
1C	LPUART Data Register (LPUART_DATA)	32	R/W	0000_1000h	31.2.8/814
20	LPUART Match Address Register (LPUART_MATCH)	32	R/W	0000_0000h	31.2.9/816

Table continues on the next page...

LPUART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
24	LPUART Modem IrDA Register (LPUART_MODIR)	32	R/W	0000_0000h	31.2.10/ 816
28	LPUART FIFO Register (LPUART_FIFO)	32	R/W	See section	31.2.11/ 819
2C	LPUART Watermark Register (LPUART_WATER)	32	R/W	0000_0000h	31.2.12/ 822

31.2.1 Version ID Register (LPUART_VERID)

Address: 0h base + 0h offset = 0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

LPUART_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Identification Number This read only field returns the feature set number. 0x0001 Standard feature set. 0x0003 Standard feature set with MODEM/IrDA support.

31.2.2 Parameter Register (LPUART_PARAM)

Address: 0h base + 4h offset = 4h

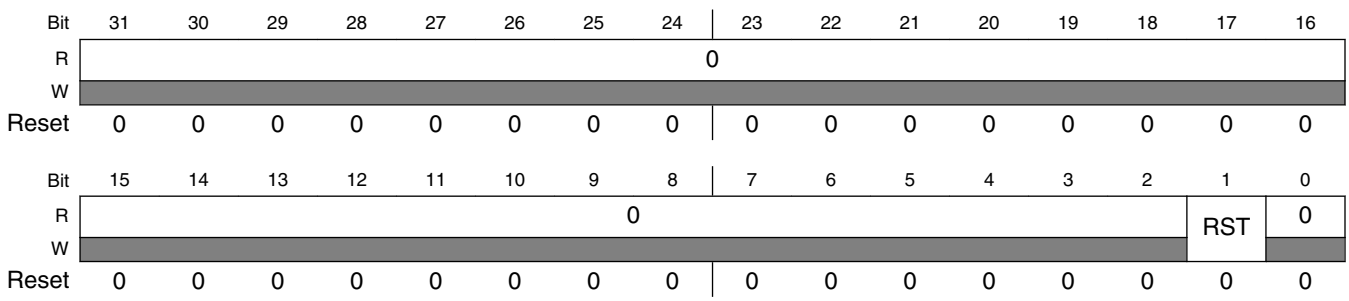
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																RXFIFO								TXFIFO							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

LPUART_PARAM field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 RXFIFO	Receive FIFO Size The number of words in the receive FIFO is 2^RXFIFO.
TXFIFO	Transmit FIFO Size The number of words in the transmit FIFO is 2^TXFIFO.

31.2.3 LPUART Global Register (LPUART_GLOBAL)

Address: 0h base + 8h offset = 8h



LPUART_GLOBAL field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RST	Software Reset Reset all internal logic and registers, except the Global Register. Remains set until cleared by software. 0 Module is not reset. 1 Module is reset.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.2.4 LPUART Pin Configuration Register (LPUART_PINCFG)

Address: 0h base + Ch offset = Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0														TRGSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPUART_PINCFG field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRGSEL	Trigger Select Configures the input trigger usage. 00 Input trigger is disabled. 01 Input trigger is used instead of RXD pin input. 10 Input trigger is used instead of CTS pin input. 11 Input trigger is used to modulate the TXD pin output.

31.2.5 LPUART Baud Rate Register (LPUART_BAUD)

Address: 0h base + 10h offset = 10h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAEN1	MAEN2	M10	OSR					TDMAE	0	RDMAE	0	MATCFG		BOTHEDGE	RESYNCDIS
W																
Reset	0	0	0	0	1	1	1	1	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
R	LBKDIE	RXEDGIE	SBNS	SBR												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

LPUART_BAUD field descriptions

Field	Description
31 MAEN1	Match Address Mode Enable 1 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA1].
30 MAEN2	Match Address Mode Enable 2 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA2].
29 M10	10-bit Mode select The M10 bit causes a tenth bit to be part of the serial transmission. This bit should only be changed when the transmitter and receiver are both disabled. 0 Receiver and transmitter use 8-bit or 9-bit data characters. 1 Receiver and transmitter use 10-bit data characters.
28–24 OSR	Oversampling Ratio This field configures the oversampling ratio for the receiver between 4x (00011) and 32x (11111). Writing an invalid oversampling ratio (for example, a value not between 4x and 32x) will default to an oversampling ratio of 16 (01111). The OSR field should only be changed when the transmitter and receiver are both disabled. Note that the oversampling ratio = OSR + 1.
23 TDMAE	Transmitter DMA Enable TDMAE configures the transmit data register empty flag, LPUART_STAT[TDRE], to generate a DMA request. 0 DMA request disabled. 1 DMA request enabled.
22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 RDMAE	Receiver Full DMA Enable RDMAE configures the receiver data register full flag, LPUART_STAT[RDRF], to generate a DMA request. 0 DMA request disabled. 1 DMA request enabled.
20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 MATCFG	Match Configuration Configures the match addressing mode used. 00 Address Match Wakeup 01 Idle Match Wakeup 10 Match On and Match Off 11 Enables RWU on Data Match and Match On/Off for transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set for

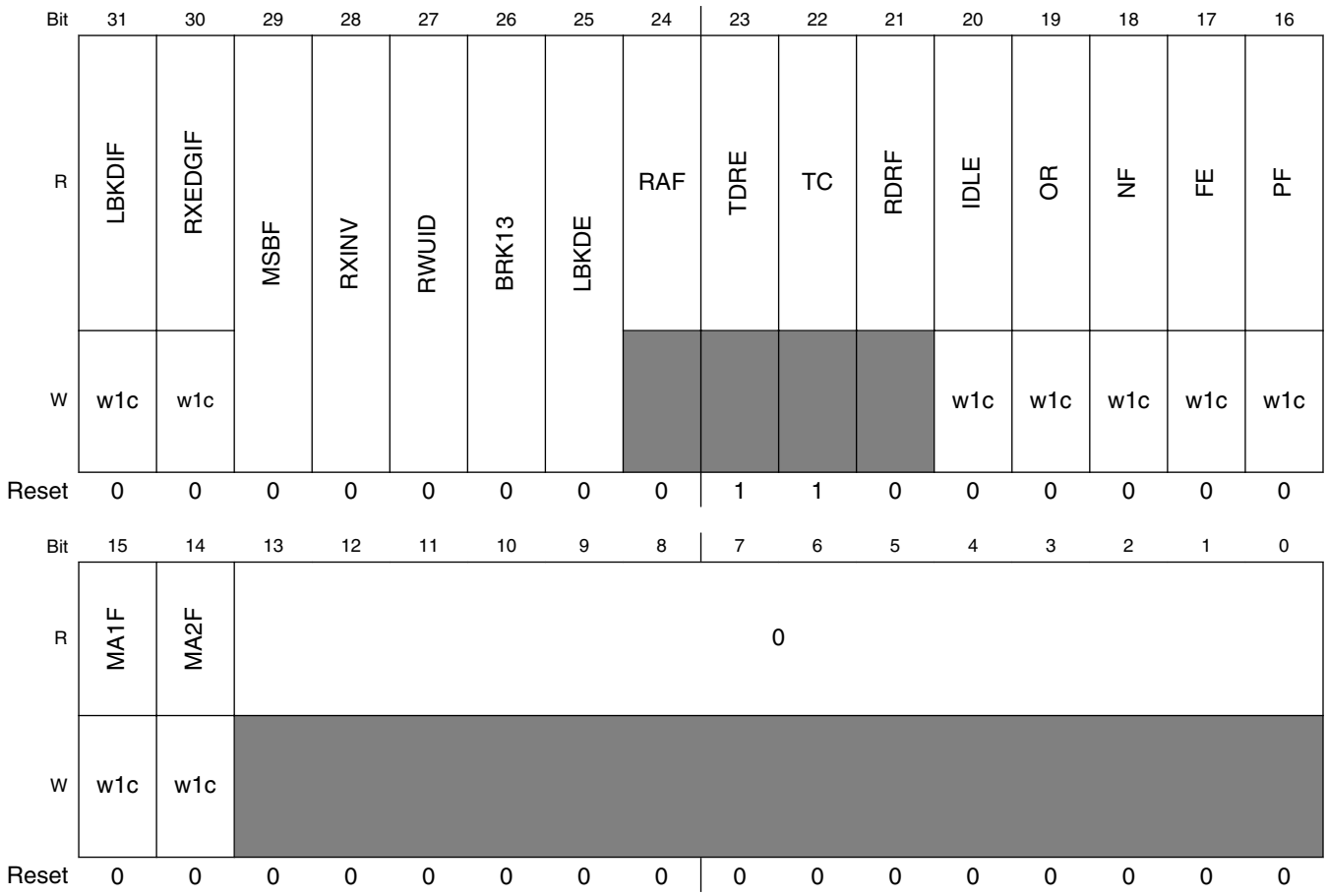
Table continues on the next page...

LPUART_BAUD field descriptions (continued)

Field	Description
	oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should only be changed when the receiver is disabled. 0 Receiver samples input data using the rising edge of the baud rate clock. 1 Receiver samples input data using the rising and falling edge of the baud rate clock.
16 RESYNCDIS	Resynchronization Disable When set, disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should only be changed when the receiver is disabled. 0 Resynchronization during received data word is supported 1 Resynchronization during received data word is disabled
15 LBKDIE	LIN Break Detect Interrupt Enable LBKDIE enables the LIN break detect flag, LBKDIF, to generate interrupt requests. 0 Hardware interrupts from LPUART_STAT[LBKDIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[LBKDIF] flag is 1.
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables the receive input active edge, RXEDGIF, to generate interrupt requests. Changing CTRL[LOOP] or CTRL[RSRC] when RXEDGIE is set can cause the RXEDGIF to set. 0 Hardware interrupts from LPUART_STAT[RXEDGIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[RXEDGIF] flag is 1.
13 SBNS	Stop Bit Number Select SBNS determines whether data characters are one or two stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0 One stop bit. 1 Two stop bits.
SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) × SBR)". The 13-bit baud rate setting [SBR12:SBR0] must only be updated when the transmitter and receiver are both disabled (LPUART_CTRL[RE] and LPUART_CTRL[TE] are both 0).

31.2.6 LPUART Status Register (LPUART_STAT)

Address: 0h base + 14h offset = 14h



LPUART_STAT field descriptions

Field	Description
31 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it. 0 No LIN break character has been detected. 1 LIN break character has been detected.
30 RXEDGIF	LPUART_RX Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge, falling if RXINV = 0, rising if RXINV=1, on the LPUART_RX pin occurs. RXEDGIF is cleared by writing a 1 to it. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
29 MSBF	MSB First

Table continues on the next page...

LPUART_STAT field descriptions (continued)

Field	Description
	Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit or the location of the start or stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0 LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1 MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of CTRL[M] and CTRL[PE].
28 RXINV	Receive Data Inversion Setting this bit reverses the polarity of the received data input. **NOTE:** Setting RXINV inverts the LPUART_RX input for all cases: data bits, start and stop bits, break, and idle. 0 Receive data not inverted. 1 Receive data inverted.
27 RWUID	Receive Wake Up Idle Detect For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should only be changed when the receiver is disabled. 0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not get set when an address does not match. 1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does get set when an address does not match.
26 BRK13	Break Character Generation Length BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. This bit should only be changed when the transmitter is disabled. 0 Break character is transmitted with length of 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1). 1 Break character is transmitted with length of 13 bit times (if M = 0, SBNS = 0) or 14 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 15 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 16 (if M10 = 1, SNBS = 1).
25 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer. 0 Break character is detected at length 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1). 1 Break character is detected at length of 11 bit times (if M = 0, SBNS = 0) or 12 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 14 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 15 (if M10 = 1, SNBS = 1).
24 RAF	Receiver Active Flag RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line.

Table continues on the next page...

LPUART_STAT field descriptions (continued)

Field	Description
	0 LPUART receiver idle waiting for a start bit. 1 LPUART receiver active (LPUART_RX input not idle).
23 TDRE	Transmit Data Register Empty Flag When the transmit FIFO is enabled, TDRE will set when the number of datawords in the transmit FIFO (LPUART_DATA) is equal to or less than the number indicated by LPUART_WATER[TXWATER]). To clear TDRE, write to the LPUART data register (LPUART_DATA) until the number of words in the transmit FIFO is greater than the number indicated by LPUART_WATER[TXWATER]. When the transmit FIFO is disabled,TDRE will set when the transmit data register (LPUART_DATA) is empty. To clear TDRE, write to the LPUART data register (LPUART_DATA). TDRE is not affected by a character that is in the process of being transmitted, it is updated at the start of each transmitted character. 0 Transmit data buffer full. 1 Transmit data buffer empty.
22 TC	Transmission Complete Flag TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to LPUART_DATA to transmit new data, queuing a preamble by clearing and then setting LPUART_CTRL[TE], queuing a break character by writing 1 to LPUART_CTRL[SBK]. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
21 RDRF	Receive Data Register Full Flag When the receive FIFO is enabled, RDRF is set when the number of datawords in the receive buffer is greater than the number indicated by LPUART_WATER[RXWATER]. To clear RDRF, read LPUART_DATA until the number of datawords in the receive data buffer is equal to or less than the number indicated by LPUART_WATER[RXWATER]. When the receive FIFO is disabled, RDRF is set when the receive buffer (LPUART_DATA) is full. To clear RDRF, read the LPUART_DATA register. A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character will continue to be received until an overrun condition occurs once the entire character is received. 0 Receive data buffer empty. 1 Receive data buffer full.
20 IDLE	Idle Line Flag IDLE is set when the LPUART receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot become set again until after a new character has been stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period. 0 No idle line detected. 1 Idle line was detected.

Table continues on the next page...

LPUART_STAT field descriptions (continued)

Field	Description
19 OR	Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received. While the OR flag is set, no additional data is stored in the data buffer even if sufficient room exists. To clear OR, write logic 1 to the OR flag. 0 No overrun. 1 Receive overrun (new LPUART data lost).
18 NF	Noise Flag The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame then noise is detected for that character. NF is set whenever the next character to be read from LPUART_DATA was received with noise detected within the character. To clear NF, write logic one to the NF. 0 No noise detected. 1 Noise detected in the received character in LPUART_DATA.
17 FE	Framing Error Flag FE is set whenever the next character to be read from LPUART_DATA was received with logic 0 detected where a stop bit was expected. To clear FE, write logic one to the FE. 0 No framing error detected. This does not guarantee the framing is correct. 1 Framing error.
16 PF	Parity Error Flag PF is set whenever the next character to be read from LPUART_DATA was received when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic one to the PF. 0 No parity error. 1 Parity error.
15 MA1F	Match 1 Flag MA1F is set whenever the next character to be read from LPUART_DATA matches MA1. To clear MA1F, write a logic one to the MA1F. 0 Received data is not equal to MA1 1 Received data is equal to MA1
14 MA2F	Match 2 Flag MA2F is set whenever the next character to be read from LPUART_DATA matches MA2. To clear MA2F, write a logic one to the MA2F. 0 Received data is not equal to MA2 1 Received data is equal to MA2
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.2.7 LPUART Control Register (LPUART_CTRL)

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

Address: 0h base + 18h offset = 18h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
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
R W	MA1IE	MA2IE	0			IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPUART_CTRL field descriptions

Field	Description
31 R8T9	Receive Bit 8 / Transmit Bit 9 R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading LPUART_DATA. T9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing LPUART_DATA. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.
30 R9T8	Receive Bit 9 / Transmit Bit 8 R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading LPUART_DATA T8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing LPUART_DATA. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.
29 TXDIR	LPUART_TX Pin Direction in Single-Wire Mode When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the LPUART_TX pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the LPUART_TX pin. 0 LPUART_TX pin is an input in single-wire mode. 1 LPUART_TX pin is an output in single-wire mode.

Table continues on the next page...

LPUART_CTRL field descriptions (continued)

Field	Description
28 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE:** Setting TXINV inverts the LPUART_TX output for all cases: data bits, start and stop bits, break, and idle. 0 Transmit data not inverted. 1 Transmit data inverted.
27 ORIE	Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0 OR interrupts disabled; use polling. 1 Hardware interrupt requested when OR is set.
26 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests. 0 NF interrupts disabled; use polling. 1 Hardware interrupt requested when NF is set.
25 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0 FE interrupts disabled; use polling. 1 Hardware interrupt requested when FE is set.
24 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0 PF interrupts disabled; use polling). 1 Hardware interrupt requested when PF is set.
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests. 0 Hardware interrupts from TDRE disabled; use polling. 1 Hardware interrupt requested when TDRE flag is 1.
22 TCIE	Transmission Complete Interrupt Enable for TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0 Hardware interrupts from TC disabled; use polling. 1 Hardware interrupt requested when TC flag is 1.
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate interrupt requests. 0 Hardware interrupts from RDRF disabled; use polling. 1 Hardware interrupt requested when RDRF flag is 1.

Table continues on the next page...

LPUART_CTRL field descriptions (continued)

Field	Description
20 ILIE	Idle Line Interrupt Enable ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests. 0 Hardware interrupts from IDLE disabled; use polling. 1 Hardware interrupt requested when IDLE flag is 1.
19 TE	Transmitter Enable Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit will read as 1 until the transmitter has completed the current character and the LPUART_TX pin is tristated. 0 Transmitter disabled. 1 Transmitter enabled.
18 RE	Receiver Enable Enables the LPUART receiver. When RE is written to 0, this register bit will read as 1 until the receiver finishes receiving the current character (if any). 0 Receiver disabled. 1 Receiver enabled.
17 RWU	Receiver Wakeup Control This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set with STAT[RWUID] is clear. **NOTE:** RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to be reasserted. 0 Normal receiver operation. 1 LPUART receiver in standby waiting for wakeup condition.
16 SBK	Send Break Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 to 13, or 13 to 16 if LPUART_STATBRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.
15 MA1IE	Match 1 Interrupt Enable 0 MA1F interrupt disabled 1 MA1F interrupt enabled
14 MA2IE	Match 2 Interrupt Enable 0 MA2F interrupt disabled 1 MA2F interrupt enabled

Table continues on the next page...

LPUART_CTRL field descriptions (continued)

Field	Description
13–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 000 1 idle character 001 2 idle characters 010 4 idle characters 011 8 idle characters 100 16 idle characters 101 32 idle characters 110 64 idle characters 111 128 idle characters
7 LOOPS	Loop Mode Select When LOOPS is set, the LPUART_RX pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0 Normal operation - LPUART_RX and LPUART_TX use separate pins. 1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit).
6 DOZEEN	Doze Enable 0 LPUART is enabled in Doze mode. 1 LPUART is disabled in Doze mode.
5 RSRC	Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the LPUART_RX pin. 1 Single-wire LPUART mode where the LPUART_TX pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0 Receiver and transmitter use 8-bit data characters. 1 Receiver and transmitter use 9-bit data characters.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the LPUART when RWU=1: • Address mark in the most significant bit position of a received data character, or • An idle condition on the receive pin input signal. 0 Configures RWU for idle-line wakeup. 1 Configures RWU with address-mark wakeup.
2 ILT	Idle Line Type Select

Table continues on the next page...

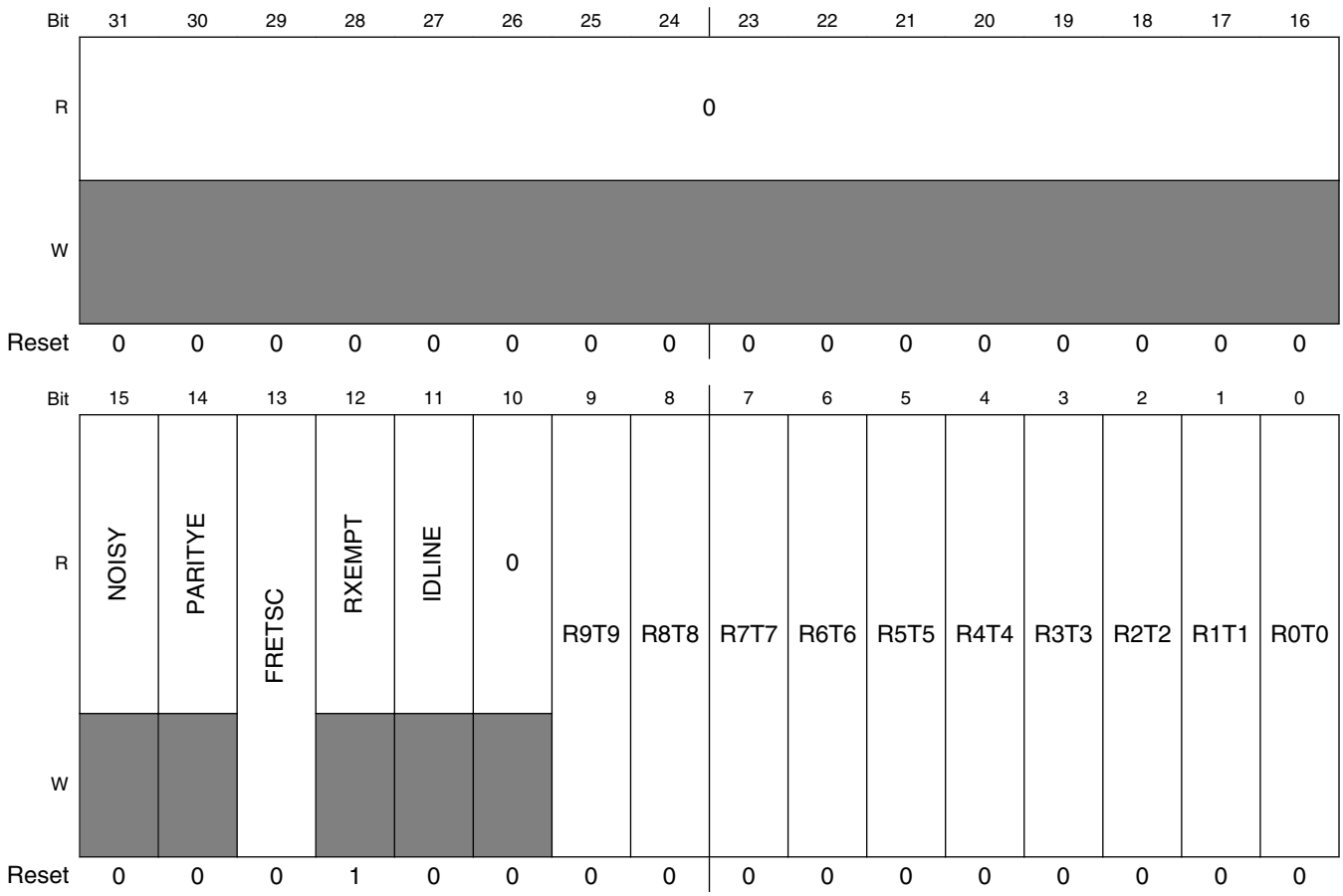
LPUART_CTRL field descriptions (continued)

Field	Description
	Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE:** In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0 No hardware parity generation or checking. 1 Parity enabled.
0 PT	Parity Type Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even. 0 Even parity. 1 Odd parity.

31.2.8 LPUART Data Register (LPUART_DATA)

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status flags.

Address: 0h base + 1Ch offset = 1Ch



LPUART_DATA field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 NOISY	The current received dataword contained in DATA[R9:R0] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise.

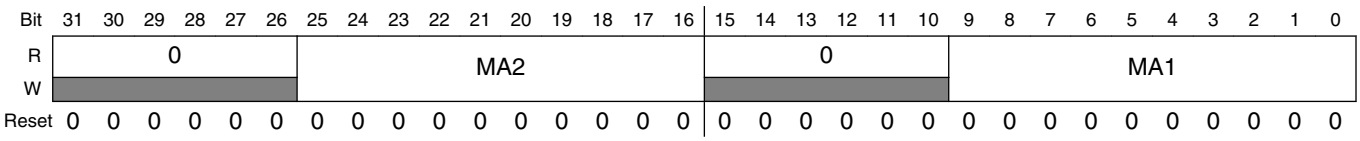
Table continues on the next page...

LPUART_DATA field descriptions (continued)

Field	Description
14 PARITYE	The current received dataword contained in DATA[R9:R0] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error.
13 FRETSC	Frame Error / Transmit Special Character For reads, indicates the current received dataword contained in DATA[R9:R0] was received with a frame error. For writes, indicates a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 is used to indicate a break character when 0 and a idle character when 1, he contents of DATA[T8:T0] should be zero. 0 The dataword was received without a frame error on read, transmit a normal character on write. 1 The dataword was received with a frame error, transmit an idle or break character on transmit.
12 RXEMPT	Receive Buffer Empty Asserts when there is no data in the receive buffer. This field does not take into account data that is in the receive shift register. 0 Receive buffer contains valid data. 1 Receive buffer is empty, data returned on read is not valid.
11 IDLINE	Idle Line Indicates the receiver line was idle before receiving the character in DATA[9:0]. Unlike the IDLE flag, this bit can set for the first character received when the receiver is first enabled. 0 Receiver was not idle before receiving this character. 1 Receiver was idle before receiving this character.
10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 R9T9	Read receive data buffer 9 or write transmit data buffer 9.
8 R8T8	Read receive data buffer 8 or write transmit data buffer 8.
7 R7T7	Read receive data buffer 7 or write transmit data buffer 7.
6 R6T6	Read receive data buffer 6 or write transmit data buffer 6.
5 R5T5	Read receive data buffer 5 or write transmit data buffer 5.
4 R4T4	Read receive data buffer 4 or write transmit data buffer 4.
3 R3T3	Read receive data buffer 3 or write transmit data buffer 3.
2 R2T2	Read receive data buffer 2 or write transmit data buffer 2.
1 R1T1	Read receive data buffer 1 or write transmit data buffer 1.
0 R0T0	Read receive data buffer 0 or write transmit data buffer 0.

31.2.9 LPUART Match Address Register (LPUART_MATCH)

Address: 0h base + 20h offset = 20h



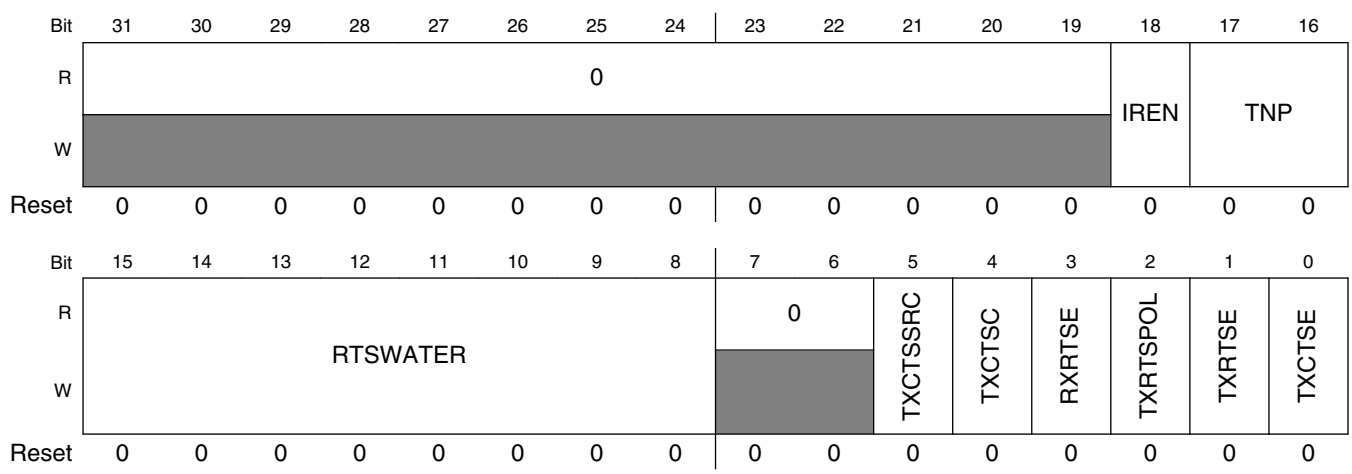
LPUART_MATCH field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–16 MA2	Match Address 2 The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated BAUD[MAEN] bit is clear.
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MA1	Match Address 1 The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated BAUD[MAEN] bit is clear.

31.2.10 LPUART Modem IrDA Register (LPUART_MODIR)

The MODEM register controls options for setting the modem configuration.

Address: 0h base + 24h offset = 24h



LPUART_MODIR field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 IREN	Infrared enable Enables/disables the infrared modulation/demodulation. 0 IR disabled. 1 IR enabled.
17–16 TNP	Transmitter narrow pulse Enables whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse. 00 1/OSR. 01 2/OSR. 10 3/OSR. 11 4/OSR.
15–8 RTSWATER	Receive RTS Configuration Configures the point at which the RX RTS output negates based on the number of additional characters that can be stored in the Receive FIFO. When configured to 0, RTS negates when the the start bit is detetected for the character that will cause the FIFO to become full. 0 RTS asserts when the receiver FIFO is full or receiving a character that causes the FIFO to become full. 1 RTS asserts when the receive FIFO is less than or equal to the RXWATER configuration and negates when the receive FIFO is greater than the RXWATER configuration.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0 CTS input is the LPUART_CTS pin. 1 CTS input is the inverted Receiver Match result.
4 TXCTSC	Transmit CTS Configuration Configures if the CTS state is checked at the start of each character or only when the transmitter is idle. 0 CTS input is sampled at the start of each character. 1 CTS input is sampled when the transmitter is idle.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. **NOTE:** Do not set both RXRTSE and TXRTSE. 0 The receiver has no effect on RTS. 1 RTS assertion is configured by the RTSWATER field
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set.

Table continues on the next page...

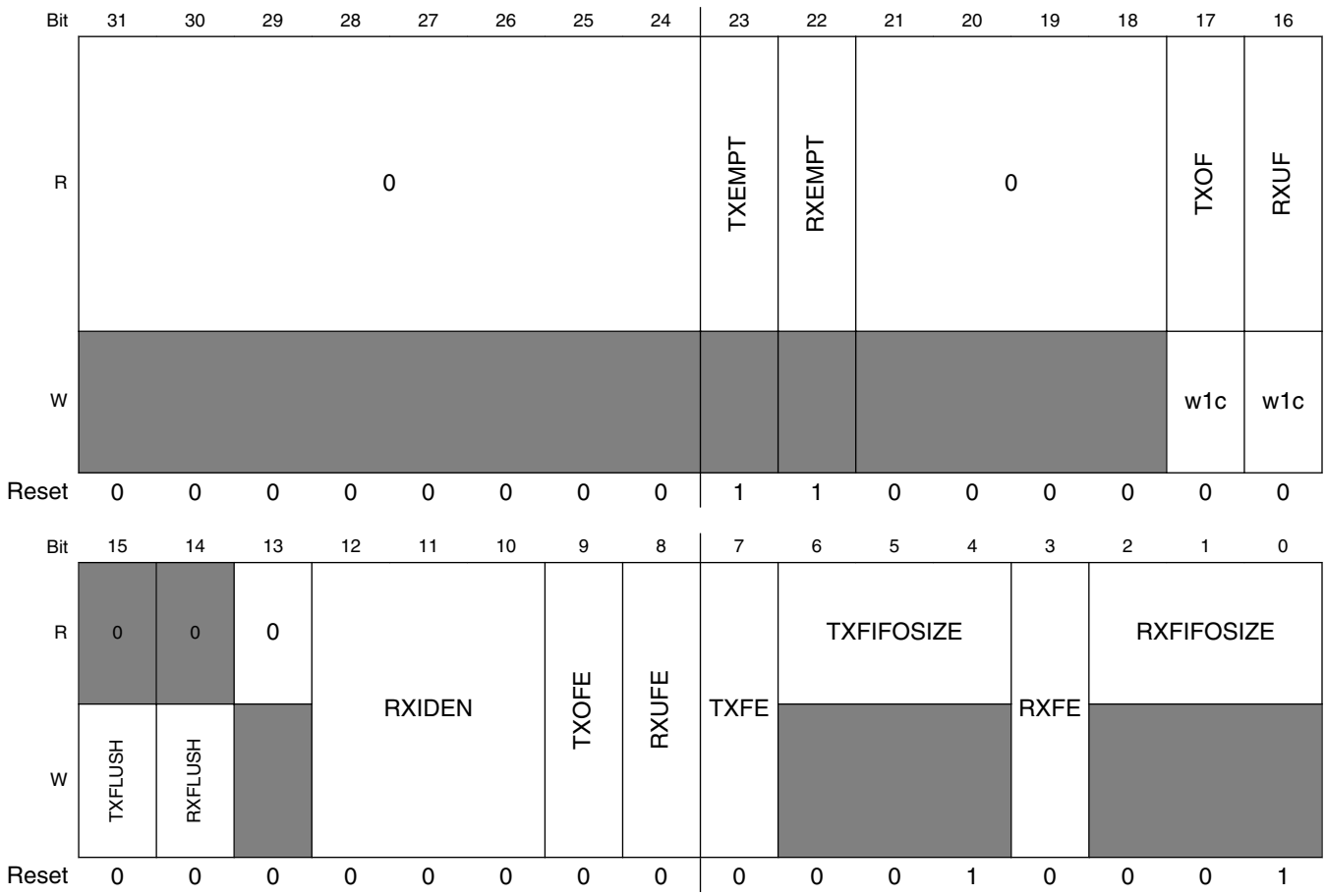
LPUART_MODIR field descriptions (continued)

Field	Description
	0 Transmitter RTS is active low. 1 Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. 0 The transmitter has no effect on RTS. 1 When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0 CTS has no effect on the transmitter. 1 Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

31.2.11 LPUART FIFO Register (LPUART_FIFO)

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when CTRL[RE] and CTRL[TE] are cleared/not set and when the data buffer/FIFO is empty.

Address: 0h base + 28h offset = 28h



LPUART_FIFO field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 TXEMPT	Transmit Buffer/FIFO Empty Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register. 0 Transmit buffer is not empty. 1 Transmit buffer is empty.

Table continues on the next page...

LPUART_FIFO field descriptions (continued)

Field	Description
22 RXEMPT	Receive Buffer/FIFO Empty Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0 Receive buffer is not empty. 1 Receive buffer is empty.
21–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 TXOF	Transmitter Buffer Overflow Flag Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of TXOFE. However, an interrupt will be issued to the host only if TXOFE is set. This flag is cleared by writing a 1. 0 No transmit buffer overflow has occurred since the last time the flag was cleared. 1 At least one transmit buffer overflow has occurred since the last time the flag was cleared.
16 RXUF	Receiver Buffer Underflow Flag Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of RXUFE. However, an interrupt will be issued to the host only if RXUFE is set. This flag is cleared by writing a 1. 0 No receive buffer underflow has occurred since the last time the flag was cleared. 1 At least one receive buffer underflow has occurred since the last time the flag was cleared.
15 TXFLUSH	Transmit FIFO/Buffer Flush Writing to this field causes all data that is stored in the transmit FIFO/buffer to be flushed. This does not affect data that is in the transmit shift register. 0 No flush operation occurs. 1 All data in the transmit FIFO/Buffer is cleared out.
14 RXFLUSH	Receive FIFO/Buffer Flush Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register. 0 No flush operation occurs. 1 All data in the receive FIFO/buffer is cleared out.
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–10 RXIDEN	Receiver Idle Empty Enable When set, enables the assertion of RDRF when the receiver is idle for a number of idle characters and the FIFO is not empty. 000 Disable RDRF assertion due to partially filled FIFO when receiver is idle. 001 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 1 character. 010 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 2 characters. 011 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 4 characters. 100 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 8 characters. 101 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 16 characters.

Table continues on the next page...

LPUART_FIFO field descriptions (continued)

Field	Description
	110 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 32 characters. 111 Enable RDRF assertion due to partially filled FIFO when receiver is idle for 64 characters.
9 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0 TXOF flag does not generate an interrupt to the host. 1 TXOF flag generates an interrupt to the host.
8 RXUFE	Receive FIFO Underflow Interrupt Enable When this field is set, the RXUF flag generates an interrupt to the host. 0 RXUF flag does not generate an interrupt to the host. 1 RXUF flag generates an interrupt to the host.
7 TXFE	Transmit FIFO Enable When this field is set, the built in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0 Transmit FIFO is not enabled. Buffer is depth 1. (Legacy support). 1 Transmit FIFO is enabled. Buffer is depth indicated by TXFIFOSIZE.
6–4 TXFIFOSIZE	Transmit FIFO. Buffer Depth The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read only. 000 Transmit FIFO/Buffer depth = 1 dataword. 001 Transmit FIFO/Buffer depth = 4 datawords. 010 Transmit FIFO/Buffer depth = 8 datawords. 011 Transmit FIFO/Buffer depth = 16 datawords. 100 Transmit FIFO/Buffer depth = 32 datawords. 101 Transmit FIFO/Buffer depth = 64 datawords. 110 Transmit FIFO/Buffer depth = 128 datawords. 111 Transmit FIFO/Buffer depth = 256 datawords
3 RXFE	Receive FIFO Enable When this field is set, the built in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0 Receive FIFO is not enabled. Buffer is depth 1. (Legacy support) 1 Receive FIFO is enabled. Buffer is depth indicted by RXFIFOSIZE.
RXFIFOSIZE	Receive FIFO. Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only. 000 Receive FIFO/Buffer depth = 1 dataword.

Table continues on the next page...

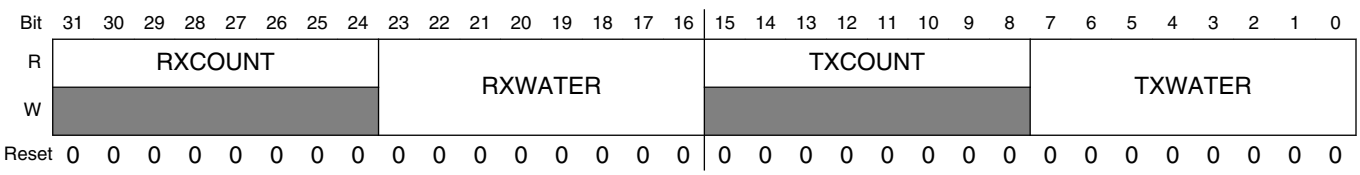
LPUART_FIFO field descriptions (continued)

Field	Description
	001 Receive FIFO/Buffer depth = 4 datawords. 010 Receive FIFO/Buffer depth = 8 datawords. 011 Receive FIFO/Buffer depth = 16 datawords. 100 Receive FIFO/Buffer depth = 32 datawords. 101 Receive FIFO/Buffer depth = 64 datawords. 110 Receive FIFO/Buffer depth = 128 datawords. 111 Receive FIFO/Buffer depth = 256 datawords.

31.2.12 LPUART Watermark Register (LPUART_WATER)

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when CTRL[TE] is not set.

Address: 0h base + 2Ch offset = 2Ch



LPUART_WATER field descriptions

Field	Description
31–24 RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.
23–16 RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is greater than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE] and must be greater than 0.
15–8 TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with FIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.
TXWATER	Transmit Watermark When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

LPUART_WATER field descriptions (continued)

Field	Description

31.3 Functional description

The LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following describes each of the blocks of the LPUART.

31.3.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to SBR[12:0] determines the baud clock divisor for the asynchronous LPUART baud clock. The SBR bits are in the LPUART baud rate registers, BDH and BDL. The baud rate clock drives the receiver, while the transmitter is driven by the baud rate clock divided by the over sampling ratio. Depending on the over sampling ratio, the receiver has an acquisition rate of 4 to 32 samples per bit time.

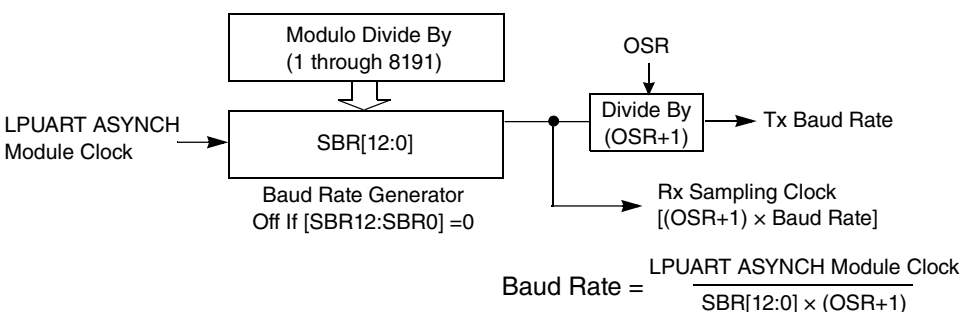


Figure 31-3. LPUART baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

31.3.2 Transmitter functional description

This section describes the overall block diagram for the LPUART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (LPUART_TX) idle state defaults to logic high, CTRL[TXINV] is cleared following reset. The transmitter output is inverted by setting CTRL[TXINV]. The transmitter is enabled by setting the CTRL[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the LPUART data register.

The central element of the LPUART transmitter is the transmit shift register that is 10-bit to 13 bits long depending on the setting in the CTRL[M], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], BAUD[M10] and BAUD[SBNS] are cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at LPUART_DATA.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the LPUART_TX pin, the transmitter sets the transmit complete flag and enters an idle mode, with LPUART_TX high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter will not start transmitting another character.

31.3.2.1 Send break and queued idle

The LPUART_CTRL[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 10-bit to 12-bit times including the start and stop bits. A longer break of 13-bit times can be enabled by setting LPUART_STAT[BRK13]. Normally, a program would wait for LPUART_STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the LPUART_CTRL[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If LPUART_CTRL[SBK] remains 1 when the queued break moves into the shifter,

synchronized to the baud rate clock, an additional break character is queued. If the receiving device is another LPUART, the break characters are received as 0s in all data bits and a framing error (LPUART_STAT[FE] = 1) occurs.

A break character can also be transmitted by writing to the LPUART_DATA register with bit 13 set and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows the DMA to transmit a break character.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for LPUART_STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the LPUART_CTRL[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while LPUART_CTRL[TE] is cleared, the LPUART transmitter never actually releases control of the LPUART_TX pin.

An idle character can also be transmitted by writing to the LPUART_DATA register with bit 13 set and the data bits also set. This supports transmitting the idle character as part of the normal data stream and also allows the DMA to transmit a break character.

The length of the break character is affected by the LPUART_STAT[BRK13], LPUART_CTRL[M], LPUART_BAUD[M10] and LPUART_BAUD[SNBS] bits as shown below.

Table 31-1. Break character length

BRK13	M	M10	SBNS	Break character length
0	0	0	0	10 bit times
0	0	0	1	11 bit times
0	1	0	0	11 bit times
0	1	0	1	12 bit times
0	X	1	0	12 bit times
0	X	1	1	13 bit times
1	0	0	0	13 bit times
1	0	0	1	13 bit times
1	1	0	0	14 bit times
1	1	0	1	14 bit times
1	X	1	0	15 bit times
1	X	1	1	15 bit times

31.3.2.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS. If the clear-to-send operation is enabled, the character is transmitted when CTS is asserted. If CTS is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and LPUART_TX remains in the mark state until CTS is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS.

The transmitter's CTS signal can also be enabled even if the same LPUART receiver's RTS signal is disabled.

31.3.2.3 Transceiver driver enable

The transmitter can use LPUART_RTS as an enable signal for the driver of an external transceiver. See Transceiver driver enable using LPUART_RTS for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, LPUART_RTS asserts one bit time before the start bit is transmitted. LPUART_RTS remains asserted for the whole time that the transmitter data buffer has any characters. LPUART_RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts LPUART_RTS, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's LPUART_RTS signal asserts only when the transmitter is enabled. However, the transmitter's LPUART_RTS signal is unaffected by its LPUART_CTS signal. LPUART_RTS will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

31.3.3 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, different variations of the receiver wakeup function are explained.

The receiver input is inverted by setting LPUART_STAT[RXINV]. The receiver is enabled by setting the LPUART_CTRL[RE] bit. Character frames consist of a start bit of logic 0, eight to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For

information about 9-bit or 10-bit data mode, refer to 8-bit, 9-bit and 10-bit data modes. For the remainder of this discussion, assume the LPUART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (LPUART_STAT[RDRF]) status flag is set. If LPUART_STAT[RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the LPUART receiver is double-buffered, the program has one full character time after LPUART_STAT[RDRF] is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (LPUART_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART_DATA. Refer to Interrupts and status flags for details about flag clearing.

31.3.3.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by taking logic level samples at the oversampling rate times the baud rate to search for a falling edge on the LPUART_RX serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (LPUART_STAT[NF]) is set when the received character is transferred to the receive data buffer.

When the LPUART receiver is configured to sample on both edges of the baud rate clock, the number of segments in each received bit is effectively doubled (from 1 to OSR×2). The start and data bits are then sampled at OSR, OSR+1 and OSR+2. Sampling on both edges of the clock must be enabled for oversampling rates of 4× to 7× and is optional for higher oversampling rates.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times (unless resynchronization has been disabled). This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

In the case of a framing error, provided the received character was not a break character, the sampling logic that searches for a falling edge is filled with three logic 1 samples so that a new start bit can be detected almost immediately.

31.3.3.2 Receiver wakeup operation

Receiver wakeup and receiver address matching is a hardware mechanism that allows an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit (LPUART_CTRL[RWU]). When RWU bit and LPUART_S2[RWUID] bit are set, the status flags associated with the receiver, with the exception of the idle bit, IDLE, are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force LPUART_CTRL[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver will ignore all characters that do not meet the address match requirements.

Table 31-2. Receiver Wakeup Options

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
0	0	X	X	Normal operation
1	0	00	00	Receiver wakeup on idle line, IDLE flag not set
1	0	00	01	Receiver wakeup on idle line, IDLE flag set

Table continues on the next page...

Table 31-2. Receiver Wakeup Options (continued)

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
1	0	00	10	Receiver wakeup on address mark
1	1	11	X0	Receiver wakeup on data match
0	1	00	X0	Address mark address match, IDLE flag not set for discarded characters
0	1	00	X1	Address mark address match, IDLE flag set for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Address match on and address match off, IDLE flag not set for discarded characters
0	1	10	X1	Address match on and address match off, IDLE flag set for discarded characters

31.3.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The LPUART_CTRL[M] and LPUART_BAUD[M10] control bit selects 8-bit to 10-bit data mode and the LPUART_BAUD[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 to 13 bit times because of the start and stop bits.

When LPUART_CTRL[RWU] is one and LPUART_STAT[RWUID] is zero, the idle condition that wakes up the receiver does not set the LPUART_STAT[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the LPUART_STAT[RDRF] flag and generates an interrupt if enabled. When LPUART_STAT[RWUID] is one, any idle condition sets the LPUART_STAT[IDLE] flag and generates an interrupt if enabled, regardless of whether LPUART_CTRL[RWU] is zero or one.

The idle-line type (LPUART_CTRL[ILT]) control bit selects one of two ways to detect an idle line. When LPUART_CTRL[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full

character time of idle. When LPUART_CTRL[ILT] is set, the idle bit counter does not start until after the stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

31.3.3.2.2 Address-mark wakeup

When LPUART_CTRL[WAKE] is set, the receiver is configured for address-mark wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a logic 1 in the most significant bit of a received character.

Address-mark wakeup allows messages to contain idle characters, but requires the MSB be reserved for use in address frames. The logic 1 in the MSB of an address frame clears the LPUART_CTRL[RWU] bit before the stop bits are received and sets the LPUART_STAT[RDRF] flag. In this case, the character with the MSB set is received even though the receiver was sleeping during most of this character time.

31.3.3.2.3 Data match wakeup

When LPUART_CTRL[RWU] is set and LPUART_BAUD[MATCFG] equals 11, the receiver is configured for data match wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a character that matches MATCH[MA1] field when BAUD[MAEN1] is set, or that matches MATCH[MA2] when BAUD[MAEN2] is set.

31.3.3.2.4 Address Match operation

Address match operation is enabled when the LPUART_BAUD[MAEN1] or LPUART_BAUD[MAEN2] bit is set and LPUART_BAUD[MATCFG] is equal to 00. In this function, a character received by the LPUART_RX pin with a logic 1 in the bit position immediately preceding the stop bit is considered an address and is compared with the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and LPUART_STAT[RDRF] is set, if the comparison matches. All subsequent characters received with a logic 0 in the bit position immediately preceding the stop bit are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs then no transfer is made to the receive data buffer, and all following characters with logic zero in the bit position immediately preceding the stop bit are also discarded. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] fields.

- If only one of LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

31.3.3.2.5 Idle Match operation

Idle match operation is enabled when the LPUART_BAUD[MAEN1] or LPUART_BAUD[MAEN2] bit is set and LPUART_BAUD[MATCFG] is equal to 01. In this function, the first character received by the LPUART_RX pin after an idle line condition is considered an address and is compared with the associated MA1 or MA2 register. The character is only transferred to the receive buffer, and LPUART_STAT[RDRF] is set, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive data buffer until the next idle line condition is detected. If no address match occurs then no transfer is made to the receive data buffer, and all following frames until the next idle condition are also discarded. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Idle match operation functions in the same way for both MA1 and MA2 registers.

- If only one of LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] is asserted, the first character after an idle line is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are asserted, the first character after an idle line is compared with both match registers and data is transferred only on a match with either register.

31.3.3.2.6 Match On Match Off operation

Match on, match off operation is enabled when both LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are set and LPUART_BAUD[MATCFG] is equal to 10. In this function, a character received by the LPUART_RX pin that matches MATCH[MA1] is received and transferred to the receive buffer, and LPUART_STAT[RDRF] is set. All subsequent characters are considered to be data and are also transferred to the receive data buffer, until a character is received that matches MATCH[MA2] register. The character that matches MATCH[MA2] and all following characters are discarded, this

continues until another character that matches MATCH[MA1] is received. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

NOTE

Match on, match off operation requires both LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] to be asserted.

31.3.3.3 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert LPUART_RTS.

- LPUART_RTS remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using LPUART_RTS for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts LPUART_RTS if the number of characters in the receiver data register is full or a start bit is detected that will cause the receiver data register to be full.
- The receiver asserts LPUART_RTS when the number of characters in the receiver data register is not full and has not detected a start bit that will cause the receiver data register to be full. It is not affected if STAT[RDRF] is asserted.
- Even if LPUART_RTS is deasserted, the receiver continues to receive characters until the receiver data buffer is overrun.
- If the receiver request-to-send functionality is disabled, the receiver LPUART_RTS remains deasserted.

31.3.3.4 Infrared decoder

The infrared decoder converts the received character from the IrDA format to the NRZ format used by the receiver. It also has a OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received.

31.3.3.4.1 Start bit detection

When STAT[RXINV] is cleared, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the

receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently from each other.

31.3.3.4.2 Noise filtering

Any further rising edges detected during the first half of the infrared decoder counter are ignored by the decoder. Any pulses less than one oversampling baud clock can be undetected by it regardless of whether it is seen in the first or second half of the count.

31.3.3.4.3 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as a 0, which is sent to the receiver. The decoder counter is also reset.

31.3.3.4.4 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, then the decoder sends a 1 to the receiver.

If the next bit is a 0, which arrives late, then a low-bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to a 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, then the delay of a 0 is not recorded as noise.

31.3.4 Additional LPUART functions

The following sections describe additional LPUART functions.

31.3.4.1 8-bit, 9-bit and 10-bit data modes

The LPUART transmitter and receiver can be configured to operate in 9-bit data mode by setting the LPUART_CTRL[M] or 10-bit data mode by setting LPUART_CTRL[M10]. In 9-bit mode, there is a ninth data bit in 10-bit mode there is a tenth data bit. For the transmit data buffer, these bits are stored in LPUART_CTRL[T8] and LPUART_CTRL[T9]. For the receiver, these bits are held in LPUART_CTRL[R8] and LPUART_CTRL[R9]. They are also accessible via 16-bit or 32-bit accesses to the LPUART_DATA register.

For coherent 8-bit writes to the transmit data buffer, write to LPUART_CTRL[T8] and LPUART_CTRL[T9] before writing to LPUART_DATA[7:0]. For 16-bit and 32-bit writes to the LPUART_DATA register all 10 transmit bits are written to the transmit data buffer at the same time.

If the bit values to be transmitted as the ninth and tenth bit of a new character are the same as for the previous character, it is not necessary to write to LPUART_CTRL[T8] and LPUART_CTRL[T9] again. When data is transferred from the transmit data buffer to the transmit shifter, the value in LPUART_CTRL[T8] and LPUART_CTRL[T9] is copied at the same time data is transferred from LPUART_DATA[7:0] to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. The 10-bit data mode is typically used with parity and address-mark wakeup so the ninth data bit can serve as the wakeup bit and the tenth bit as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

31.3.4.2 Idle length

An idle character is a character where the start bit, all data bits and stop bits are in the mark postion. The CTRL[ILT] register can be configured to start detecting an idle character from the previous start bit (any data bits and stop bits count towards the idle character detection) or from the previous stop bit.

The number of idle characters that must be received before an idle line condition is detected can also be configured using the CTRL[IDLECFG] field. This field configures the number of idle characters that must be received before the STAT[IDLE] flag is set, the STAT[RAF] flag is cleared and the DATA[IDLINE] flag is set with the next received character.

Idle-line wakeup and idle match operation are also affected by the CTRL[IDLECFG] field. When address match or match on/off operation is enabled, setting the STAT[RWUID] bit will cause any discarded characters to be treated as if they were idle characters.

31.3.4.3 Loop mode

When LPUART_CTRL[LOOPS] is set, the LPUART_CTRL[RSRC] bit in the same register chooses between loop mode (LPUART_CTRL[RSRC] = 0) or single-wire mode (LPUART_CTRL[RSRC] = 1). Loop mode is sometimes used to check software,

independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the LPUART_RX pin is not used by the LPUART.

31.3.4.4 Single-wire operation

When LPUART_CTRL[LOOPS] is set, the RSRC bit in the same register chooses between loop mode (LPUART_CTRL[RSRC] = 0) or single-wire mode (LPUART_CTRL[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the LPUART_TX pin (the LPUART_RX pin is not used).

In single-wire mode, the LPUART_CTRL[TXDIR] bit controls the direction of serial data on the LPUART_TX pin. When LPUART_CTRL[TXDIR] is cleared, the LPUART_TX pin is an input to the receiver and the transmitter is temporarily disconnected from the LPUART_TX pin so an external device can send serial data to the receiver. When LPUART_CTRL[TXDIR] is set, the LPUART_TX pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

31.3.5 Infrared interface

The LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the LPUART. The IrDA physical layer specification defines a half-duplex infrared communication link for exchanging data. The full standard includes data rates up to 16 Mbits/s. This design covers data rates only between 2.4 kbits/s and 115.2 kbits/s.

The LPUART has an infrared transmit encoder and receive decoder. The LPUART transmits serial bits of data that are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses are detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder, external from the LPUART. The narrow pulses are then stretched by the infrared receive decoder to get back to a serial bit stream to be received by the LPUART. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active high pulses.

The infrared submodule receives its clock sources from the LPUART. One of these two clocks are selected in the infrared submodule to generate either 1/OSR, 2/OSR, 3/OSR, or 4/OSR narrow pulses during transmission.

31.3.5.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the LPUART_TX signal. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent at the start of the bit with a duration of 1/OSR, 2/OSR, 3/OSR, or 4/OSR of a bit time. A narrow low pulse is transmitted for a zero bit when LPUART_CTRL[TXINV] is cleared, while a narrow high pulse is transmitted for a zero bit when LPUART_CTRL[TXINV] is set.

31.3.5.2 Infrared receive decoder

The infrared receive block converts data from the LPUART_RX signal to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow low pulse is expected for a zero bit when LPUART_STAT[RXINV] is cleared, while a narrow high pulse is expected for a zero bit when LPUART_STAT[RXINV] is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

31.3.6 Interrupts and status flags

The LPUART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty LPUART_STAT[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to LPUART_DATA. If the transmit interrupt enable LPUART_CTRL[TIE]) bit is set, a hardware interrupt is requested when LPUART_STAT[TDRE] is set. Transmit complete (LPUART_STAT[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with LPUART_TX at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (LPUART_CTRL[TCIE]) bit is set, a hardware interrupt is requested when LPUART_STAT[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the LPUART_STAT[TDRE] and LPUART_STAT[TC] status flags if the corresponding LPUART_CTRL[TIE] or LPUART_CTRL[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (LPUART_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART_DATA. The LPUART_STAT[RDRF] flag is cleared by reading LPUART_DATA.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the LPUART_RX line remains idle for an extended period of time. IDLE is cleared by writing 1 to the LPUART_STAT[IDLE] flag. After LPUART_STAT[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set LPUART_STAT[RDRF].

If the associated error was detected in the received character that caused LPUART_STAT[RDRF] to be set, the error flags - noise flag (LPUART_STAT[NF]), framing error (LPUART_STAT[FE]), and parity error flag (LPUART_STAT[PF]) - are set at the same time as LPUART_STAT[RDRF]. These flags are not set in overrun cases.

If LPUART_STAT[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (LPUART_STAT[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2] then the LPUART_STAT[MA1F] and/or LPUART_STAT[MA2F] flags are set at the same time that LPUART_STAT[RDRF] is set.

At any time, an active edge on the LPUART_RX serial data input pin causes the LPUART_STAT[RXEDGIF] flag to set. The LPUART_STAT[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (LPUART_CTRL[RE] = 1).

Chapter 32
Memory-Mapped Divide and Square Root (MMDVSQ)

32.1 Introduction

ARM processor cores in the Cortex-M family implementing the ARMv6-M instruction set architecture do not include hardware support for integer divide operations. The affected processors include the Cortex-M0+ core. However, in certain deeply embedded application spaces, hardware support for this class of arithmetic operation (along with an unsigned square root function) is important to maximize system performance and minimize device power dissipation. Accordingly, the MMDVSQ module is included in select microcontrollers, to serve as a memory-mapped co-processor located in a special address space (within the system memory map) that is accessible only to the processor core.

The MMDVSQ module supports execution of the integer divide operations defined in the ARMv7-M instruction set architecture, plus an unsigned integer square root operation. The supported integer divide operations include 32/32 signed (SDIV) and unsigned (UDIV) calculations.

32.1.1 Features

The key features of the MMDVSQ include:

- Lightweight implementation of 32-bit integer divide and square root arithmetic operations
  - Supports 32/32 signed and unsigned divide (or remainder) calculations
  - Supports 32-bit unsigned square root calculations
- Simple programming model includes input data and result registers plus a control/status register
- Programming model interface optimized for activation from inline code or software library call

- "Fast Start" configuration minimizes the memory-mapped register write overhead
- Supports two methods to determine when result is valid, including software polling
- Configurable divide-by-zero response

• Pipelined design processes 2 bits per cycle with early termination exit for minimum execution time

32.1.2 Block diagram

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown in Figure 32-1. The MMDVSQ module's location as a memory-mapped co-processor is highlighted.

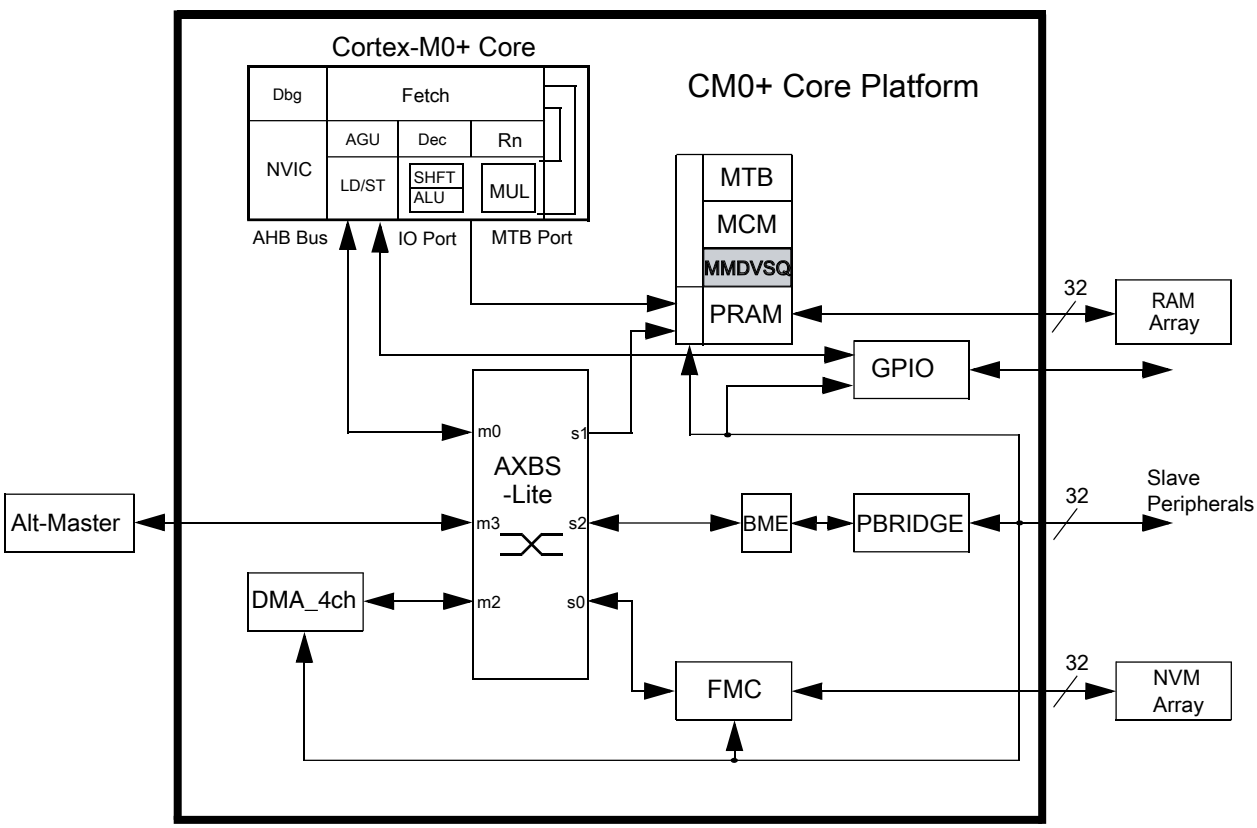


Figure 32-1. Generic Cortex-M0+ Core Platform Block Diagram

Next, a block diagram of the internal structure of the MMDVSQ module is presented. See Figure 32-2.

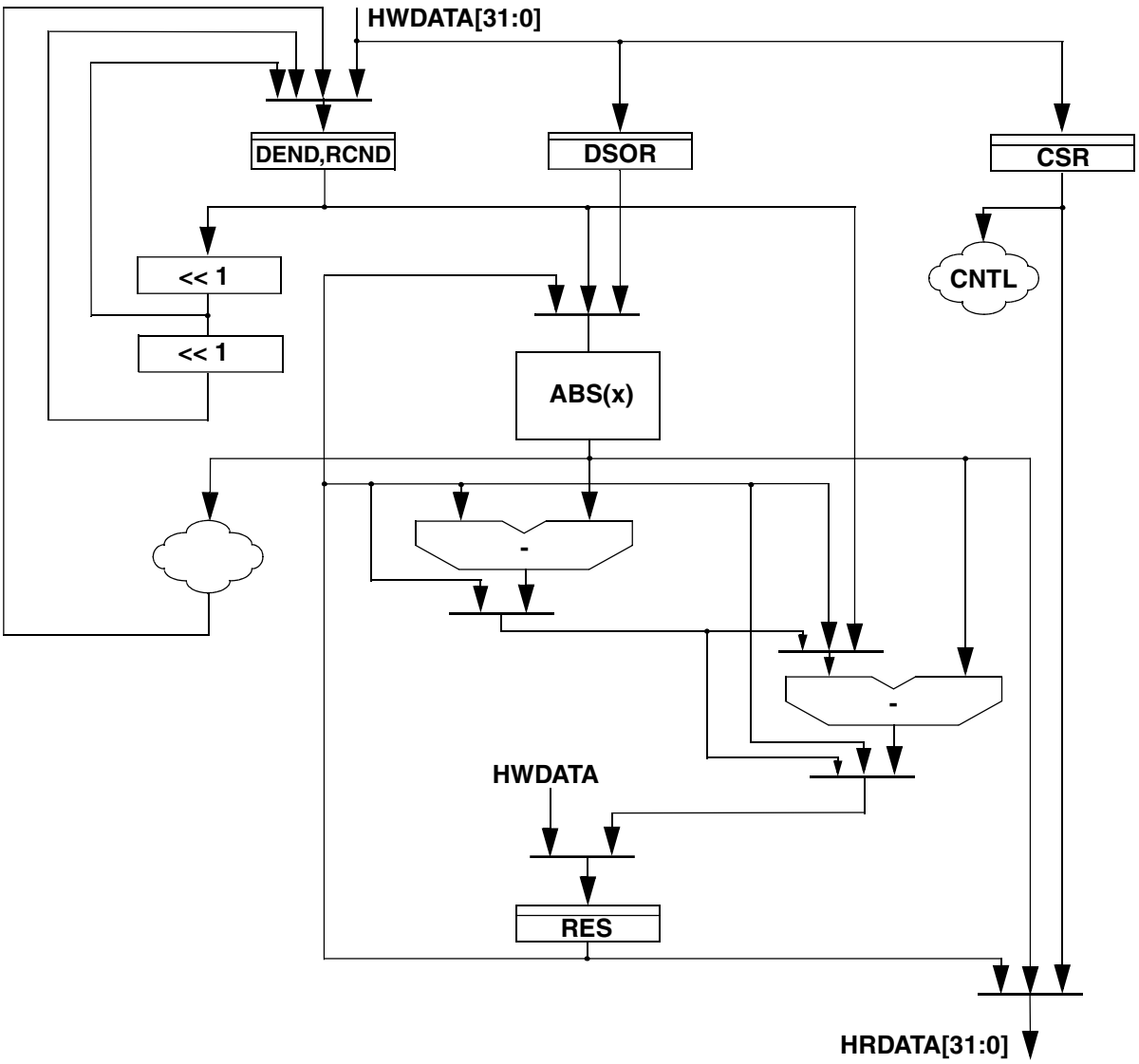


Figure 32-2. MMDVSQ Block Diagram

32.1.3 Modes of operation

The MMDVSQ module does not support any special modes of operation. As a memory-mapped device located on a crossbar slave AHB system bus port, MMDVSQ responds based strictly on memory addresses to its programming model.

All functionality associated with the MMDVSQ module resides in the core platform's clock domain; this includes its connections with the crossbar slave port. To minimize power dissipation, the design supports an architectural clock gate for the entire module, that is, the MMDVSQ is only clocked when responding to bus requests to its programming model or is busy performing a calculation.

32.2 External signal description

The MMDVSQ module does not directly support any external interfaces.

The internal interface includes a standard 32-bit AHB bus as shown in Figure 32-1.

32.3 Memory map and register definition

The MMDVSQ module supports a small number of program-visible registers used for passing input operands and retrieving the output result plus a configuration/status register.

The programming model occupies the first 20 bytes of a standard 4 Kb address slot. It can only be accessed via word-sized (32 bit) accesses. Attempted accesses using smaller data sizes, reading the write-only location or to reserved space are terminated with an error.

At any instant in time, the MMDVSQ can perform either a divide or square root calculation. The basic integer operations supported by the MMDVSQ are:

For divide:

```
MMDVSQ_RES = quotient  (MMDVSQ_DEND / MMDVSQ_DSOR)

MMDVSQ_RES = remainder (MMDVSQ_DEND % MMDVSQ_DSOR)
```

For square root:

```
MMDVSQ_RES = integer   (√MMDVSQ_RCND)
```

The register usage, based on the operation (divide, square root), is detailed in Table 32-1.

Table 32-1. Register Usage = f(Divide, Square Root)

Register	Divide	Square Root	Description
Dividend (MMDVSQ_DEND)	Yes	No	Input dividend (numerator) for the divide
Divisor (MMDVSQ_DSOR)	Yes	No	Input divisor (denominator) for the divide
Control/Status (MMDVSQ_CSR)	Yes	Yes	Control for divide, status for divide and square root
Result (MMDVSQ_RES)	Yes	Yes	Output result
Radicand (MMDVSQ_RCND)	No	Yes	Input "square" data

MMDVSQ memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_4000	Dividend Register (MMDVSQ0_DEND)	32	R/W	Undefined	32.3.1/843
F000_4004	Divisor Register (MMDVSQ0_DSOR)	32	R/W	Undefined	32.3.2/843
F000_4008	Control/Status Register (MMDVSQ0_CSR)	32	R/W	See section	32.3.3/845
F000_400C	Result Register (MMDVSQ0_RES)	32	R/W	Undefined	32.3.4/848
F000_4010	Radicand Register (MMDVSQ0_RCND)	32	W	Undefined	32.3.5/848

32.3.1 Dividend Register (MMDVSQx_DEND)

This register is loaded with the input dividend operand before a divide operation is initiated. The register is updated by the MMDVSQ hardware during the execution of a divide or square root calculation. Any memory access (read or write) of the DEND register while the module is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes.

Address: F000_4000h base + 0h offset = F000_4000h

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

* Notes:
- x = Undefined at reset.

MMDVSQx_DEND field descriptions

Field	Description
DIVIDEND	Dividend This is the input dividend operand for divide calculations.

32.3.2 Divisor Register (MMDVSQx_DSOR)

This register is loaded with the input divisor operand before a divide operation is initiated. If CSR[DFS] = 0, a write to this register inititates a divide operation. Any memory access (read or write) of the DSOR register while the module is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes.

If a divide operation is inititated with DSOR = 0, the hardware signals a divide-by-zero condition and sets RES = 0 and CSR[DZ] = 1. If CSR[DZE] = 1, an attempted read of the RES result is error terminated.

Address: F000_4000h base + 4h offset = F000_4004h

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

* Notes:
- x = Undefined at reset.

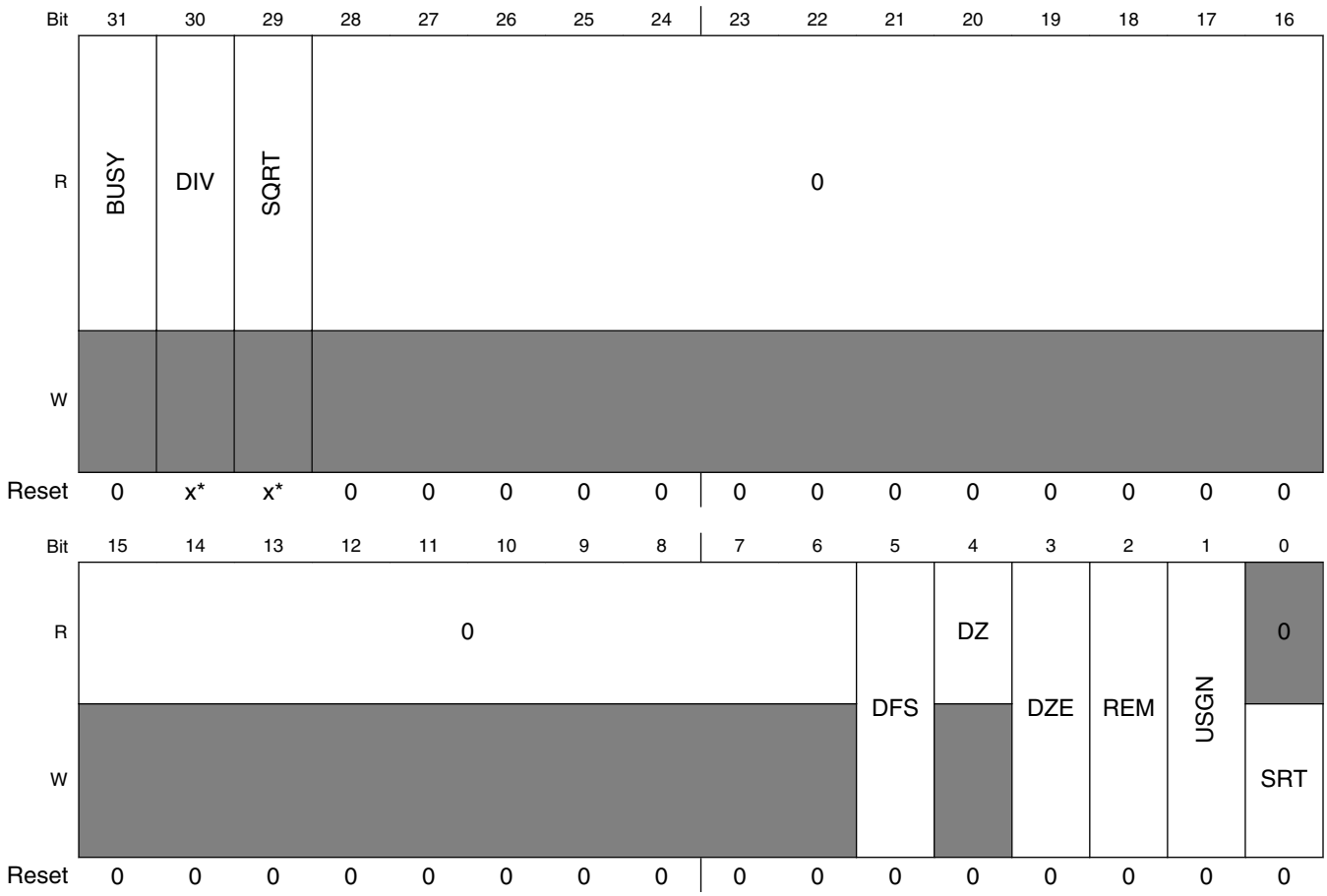
MMDVSQx_DSOR field descriptions

Field	Description
DIVISOR	Divisor This is the input divisor operand for divide calculations.

32.3.3 Control/Status Register (MMDVSQx_CSR)

This register defines the operating configuration of divide operations and provides status information. The upper 3 bits provide busy status indicators, while the low-order byte defines the configuration for divide operations. The read-only status bits in CSR[31:29] are valid for both divide and square root operations; the configuration and status bit in CSR[5:0] are only valid for divides. A memory write access of the CSR register while the module is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes.

Address: F000_4000h base + 8h offset = F000_4008h



* Notes:
- x = Undefined at reset.

MMDVSQx_CSR field descriptions

Field	Description
31 BUSY	BUSY This read-only bit is asserted when the MMDVSQ is performing a divide or square root. When an operation is initiated, the hardware sets this flag. It remains asserted until the operation completes and the

Table continues on the next page...

MMDVSQx_CSR field descriptions (continued)

Field	Description
	hardware automatically clears the indicator. This bit can be used to poll the DVSQ's execution status. The combined CSR[BUSY, DIV, SQRT] indicators provide an encoded module status: • If 0b001, then MMDVSQ is idle and the last calculation was a square root • If 0b010, then MMDVSQ is idle and the last calculation was a divide • If 0b101, then MMDVSQ is busy processing a square root calculation • If 0b110, then MMDVSQ is busy processing a divide calculation The remaining encodings of CSR[BUSY, DIV, SQRT] are reserved. 0 MMDVSQ is idle 1 MMDVSQ is busy performing a divide or square root calculation
30 DIV	DIVIDE Current or last operation was a divide. This read-only indicator bit signals if the current or last operation performed by the MMDVSQ was a divide. 0 Current or last MMDVSQ operation was not a divide 1 Current or last MMDVSQ operation was a divide
29 SQRT	SQUARE ROOT Current or last operation was a square root. This read-only indicator bit signals if the current or last operation performed by the MMDVSQ was a square root. 0 Current or last MMDVSQ operation was not a square root 1 Current or last MMDVSQ operation was a square root
28–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 DFS	Disable Fast Start The MMDVSQ supports 2 mechanisms for initiating a divide operation. The default mechanism is a "fast start" where a write to the DSOR register begins the divide. Alternatively, the start mechanism can begin after a write to the CSR register with CSR[SRT] set. The CSR[DFS] indicator selects the divide start mechanism. 0 A divide operation is initiated by a write to the DSOR register 1 A divide operation is initiated by a write to the CSR register with CSR[SRT] = 1
4 DZ	Divide-by-Zero This read-only status indicator signals the last divide operation had a zero divisor, that is, DSOR = 0x0000_0000. For this case, RES is set to 0x0000_0000 and this indicator bit set. After a divide-by-zero operation, a read of the RES register returns either the zero result, or, if CSR[DZE] = 1, terminates the read with an error. The CSR[DZ] indicator is cleared by the hardware at the beginning of each operation. 0 The last divide operation had a non-zero divisor, that is, DSOR != 0 1 The last divide operation had a zero divisor, that is, DSOR = 0
3 DZE	Divide-by-Zero-Enable This indicator configures the MMDVSQ's response to divide-by-zero calculations. If both CSR[DZ] and CSR[DZE] are set, then a subsequent read of the RES register is error terminated to signal the processor of the attempted divide-by-zero.

Table continues on the next page...

MMDVSQx_CSR field descriptions (continued)

Field	Description
	0 Reads of the RES register return the register contents 1 If CSR[DZ] = 1, an attempted read of RES register is error terminated to signal a divide-by-zero, else the register contents are returned
2 REM	REMainder calculation This indicator selects whether the quotient or the remainder is returned in the RES register. The combined CSR[REM] and CSR[USGN] bits define four possible divide operations: • If CSR[REM, USGN] = 0b00, perform a signed divide, returning the quotient • If CSR[REM, USGN] = 0b01, perform an unsigned divide, returning the quotient • If CSR[REM, USGN] = 0b10, perform a signed divide, returning the remainder • If CSR[REM, USGN] = 0b11, perform an unsigned divide, returning the remainder 0 Return the quotient in the RES for the divide calculation 1 Return the remainder in the RES for the divide calculation
1 USGN	Unsigned calculation This indicator selects whether a signed (default) or unsigned divide is performed. See the CSR[REM] description for the encoding of the four possible divide operations. 0 Perform a signed divide 1 Perform an unsigned divide
0 SRT	Start When written with a logical one and CSR[DFS] = 1, this flag initiates a divide operation. If written as a logical one with CSR[DFS] = 0, it is ignored. This bit always reads as a zero. The state of the register write data defines this bit's function. 0 No operation initiated 1 If CSR[DFS] = 1, then initiate a divide calculation, else ignore

32.3.4 Result Register (MMDVSQx_RES)

This register is loaded with the result of the divide or square root calculation. It is updated by the MMDVSQ hardware at the completion of the calculation. When a square root operation is performed (on an unsigned 32-bit number), the result is limited to a 16-bit value with RES[31:16] = 0x0000. Any memory access (read or write) of the RES register while the module is busy during a calculation causes the access to be stalled (using wait states) until the calculation completes and the new result written into the register.

Address: F000_4000h base + Ch offset = F000_400Ch

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

* Notes:
- x = Undefined at reset.

MMDVSQx_RES field descriptions

Field	Description
RESULT	Result This is the output result for a divide or square root calculation.

32.3.5 Radicand Register (MMDVSQx_RCND)

The write-only radicand register is loaded with the input “square” number. A memory write to the radicand register initiates a square root calculation. While the MMDVSQ module is busy performing a square root calculation, any memory write access to the RCND register causes the write access to be stalled (using wait states) until the square root calculation finishes. Any attempted read of the radicand register terminates with an error.

Address: F000_4000h base + 10h offset = F000_4010h

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

* Notes:
- x = Undefined at reset.

MMDVSQx_RCND field descriptions

Field	Description
RADICAND	Radicand This is the input radicand for a square root calculation, that is, the input "square" number.

32.4 Functional description

This section details the algorithms, execution times of the MMDVSQ, and the software interface to the module.

32.4.1 Algorithms

This section provides more details on the integer divide and square root algorithms.

32.4.1.1 Integer divide including special cases

32.4.1.1.1 Overview

The MMDVSQ module implements a "shift, test, and restore" radix-2 algorithm for unsigned integer divide operations. When performing a signed divide calculation, negative input operands are converted into 2's complement positive numbers first, an unsigned divide performed, and the sign of the results based on the input operand signs, namely:

- The sign of the remainder is the same as the sign of the dividend
- The quotient is negated if the signs of the dividend and divisor are different

The hardware implementation processes two bits per machine cycle and includes "early termination" logic where the execution time is data dependent, based on the magnitude of the positive dividend. See Table 32-4 for more execution time details.

32.4.1.1.2 Special case: Overflow

There is a single "special overflow case" affecting signed integer divides. If the dividend = 0x8000_0000 and the divisor = 0xFFFF_FFFF, the result of this ($-2^{31}/-1$) operation cannot be expressed as a 32-bit 2's complement number. For this case, the MMDVSQ exactly follows the ARM Cortex-Mx definition and returns 0x8000_0000 (the lower 32 bits of the $+2^{31}$ result) as the quotient with no indication of the overflow condition. If the remainder is selected as the output of this calculation, it returns 0x0000_0000.

32.4.1.1.3 Special case: Divide-by-Zero

For both signed and unsigned divides, if the divisor is zero, the MMDVSQ hardware detects this condition and the CSR[DZ] indicator set. The quotient result is forced to 0x0000_0000. If the remainder is selected as the output of this calculation, it also returns 0x0000_0000. Additionally, if CSR[DZE] = 1, then an attempted read of the Result register (RES) is error terminated to provide a simple mechanism to signal software of the divide-by-zero condition.

32.4.1.2 Integer square root

32.4.1.2.1 Overview

The unsigned square root algorithm begins by creating a 32-bit “one-hot” bit vector signaling the highest power of four of the contents of the Radicand register (RCND). It then iterates through an algorithm involving magnitude comparisons of the RCND register versus the working result plus bit vector summation, conditional decrementing of the radicand, a 1-bit right shift of the result, and a 2-bit right shift of the one-hot bit vector.

Processing two bits of the radicand per cycle, the result register finishes with the integer portion of the square root calculation. The module includes early termination logic so that the execution time is data dependent, based on the magnitude of the input radicand. See Table 32-5 for more execution time details. Since both algorithms share common hardware structures, the incremental cost of the square root logic is an extremely small delta to the basic divide hardware.

The square root algorithm was exhaustively compared (that is, all 2^{32} possible input values) against the standard GNU C library implementation, which converts the unsigned integer input into a double-precision floating-point number, calculates the double-precision square root and then converts it back into an unsigned integer. Each input value calculated identical square root results.

32.4.1.2.2 Square root using Q notation

Consider the use of Q notation for square root calculations returning fractional values. The following description is taken from http://en.wikipedia.org/wiki/Q_(number_format).

Q is a fixed point number format where the number of fractional bits (and optionally the number of integer bits) is specified. For example, a Q15 number has 15 fractional bits; a Q1.14 number has 1 integer bit and 14 fractional bits. Q format is often used in hardware that does not have a floating-point unit and in applications that require constant resolution.

Q format numbers are (notionally) fixed point numbers (but not actually a number itself); that is, they are stored and operated upon as regular binary numbers (i.e. signed integers), thus allowing standard integer hardware/ALU to perform rational number calculations. The number of integer bits, fractional bits and the underlying word size are to be chosen by the programmer on an application-specific basis - the programmer's choices of the foregoing will depend on the range and resolution needed for the numbers. The machine itself remains oblivious to the notional fixed point representation being employed - it merely performs integer arithmetic the way it knows how. Ensuring that the computational results are valid in the Q format representation is the responsibility of the programmer.

The Q notation is written as Qm.n, where:

- *Q designates that the number is in the Q format notation - the Texas Instruments representation for signed fixed-point numbers (the "Q" being reminiscent of the standard symbol for the set of rational numbers).*
- *m is the number of bits set aside to designate the two's complement integer portion of the number, exclusive of the sign bit (therefore if m is not specified it is taken as zero).*
- *n is the number of bits used to designate the fractional portion of the number, i.e. the number of bits to the right of the binary point. (If n = 0, the Q numbers are integers - the degenerate case).*

Note that the most significant bit is always designated as the sign bit (the number is stored as a two's complement number) in order to allow standard arithmetic-logic hardware to manipulate Q numbers. Representing a signed fixed-point data type in Q format therefore always requires m+n+1 bits to account for the sign bit. Hence the smallest machine word size required to accommodate a Qm.n number is m+n+1, with the Q number left justified in the machine word.

For a given Qm.n format, using an m+n+1 bit signed integer container with n fractional bits:

- *its range is $[-2^m, 2^m - 2^{-n}]$*
- *its resolution is 2^{-n}*

For the unsigned integer format used in the MMDVSQ's square root calculation, an u(nsigned)Qm.n notation requires m+n bits (m+n = 32) for the input radicand. An uQm.n format produces an uQ(m/2).(n/2) square root. As examples, consider the following tables involving the square root of 2 and square root of "pi" calculations. As expected, as the number of fractional bits (n) increases, the error between the calculated square root and the "actual" result decreases.

Table 32-2. Square Root of 2 Calculations (√2 = 1.4142135623)

RCND [Hex]	RCND Q format	Results [Hex]	RES Q Format	Decimal	% Error
0x0000_0002	uQ32.00	0x0000_0001	uQ16.00	1.0	-29.289%
0x0002_0000	uQ16.16	0x0000_016A	uQ08.08	1.4140625	-0.011%
0x0200_0000	uQ08.24	0x0000_16A0	uQ04.12	1.4140625	-0.011%
0x2000_0000	uQ04.28	0x0000_5A82	uQ02.14	1.4141845703	-0.002%
0x8000_0000	uQ02.30	0x0000_B504	uQ01.15	1.4141845703	-0.002%

Table 32-3. Square Root of Pi Calculations (√Pi = 1.7724538509)

RCND [Hex]	RCND Q format	Results [Hex]	RES Q Format	Decimal	% Error
0x0000_0003	uQ32.0	0x0000_0001	uQ16.00	1.0	-43.581%
0x0003_243F	uQ16.16	0x0000_01C5	uQ08.08	1.76953125	-0.165%
0x0324_3F6A	uQ08.24	0x0000_1C5B	uQ04.12	1.772216769	-0.013%
0x3243_F6A8	uQ04.28	0x0000_716F	uQ02.14	1.7723999023	-0.003%
0xC90F_DAA0	uQ02.30	0x0000_E2DF	uQ01.15	1.7724304199	-0.001%

The application of the Q notation for square root calculations provides a powerful extension for these types of fractional numeric computations using fixed-point integer processing hardware.

32.4.2 Execution times

The MMDVSQ module includes early termination logic to finish both divide and square root calculations as quickly as possible, based on the magnitude of the input operand. Accordingly, the execution time for the calculations is data dependent as defined in Table 32-4 and Table 32-5. In this context, the execution time is defined from the register write to initiate the calculation until the result register has been updated and available to read. Stated differently, it represents the time CSR[BUSY] is asserted for a given calculation. In the following two tables, "x" signals a bit with a don't care value.

Table 32-4. Divide Execution Times

CSR[USGN] ? DEND[31:0] // unsigned divide : abs(DEND[31:0]) // signed divide	Execution Time with CSR[BUSY] = 1 [cycles]
(01,1x)xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	17
00(01,1x)_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	16
0000_(01,1x)xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	15
0000_00(01,1x)_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	14
0000_0000_(01,1x)xx_xxxx__xxxx_xxxx_xxxx_xxxx	13
0000_0000_00(01,1x)_xxxx__xxxx_xxxx_xxxx_xxxx	12
0000_0000_0000_(01,1x)xx__xxxx_xxxx_xxxx_xxxx	11
0000_0000_0000_00(01,1x)__xxxx_xxxx_xxxx_xxxx	10
0000_0000_0000_0000__(01,1x)xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000__00(01,1x)_xxxx_xxxx_xxxx	8
0000_0000_0000_0000__0000_(01,1x)xx_xxxx_xxxx	7
0000_0000_0000_0000__0000_00(01,1x)_xxxx_xxxx	6
0000_0000_0000_0000__0000_0000_(01,1x)xx_xxxx	5
0000_0000_0000_0000__0000_0000_00(01,1x)_xxxx	4
0000_0000_0000_0000__0000_0000_0000_(01,1x)xx	3
0000_0000_0000_0000__0000_0000_0000_00(01,1x)	2
0000_0000_0000_0000__0000_0000_0000_0000	1

Table 32-5. Square Root Execution Times

RCND[31:0]	Execution Time with CSR[BUSY] = 1 [cycles]
(01,1x)xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	17
00(01,1x)_xxxx_xxxx_xxxx_x_xxx_xxxx_xxxx_xxxx	16
0000_(01,1x)xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	15
0000_00(01,1x)_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	14
0000_0000_(01,1x)xx_xxxx__xxxx_xxxx_xxxx_xxxx	13
0000_0000_00(01,1x)_xxxx__xxxx_xxxx_xxxx_xxxx	12
0000_0000_0000_(01,1x)xx__xxxx_xxxx_xxxx_xxxx	11
0000_0000_0000_00(01,1x)__xxxx_xxxx_xxxx_xxxx	10
0000_0000_0000_0000__(01,1x)xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000__00(01,1x)_xxxx_xxxx_xxxx	8
0000_0000_0000_0000__0000_(01,1x)xx_xxxx_xxxx	7
0000_0000_0000_0000__0000_00(01,1x)_xxxx_xxxx	6
0000_0000_0000_0000__0000_0000_(01,1x)xx_xxxx	5
0000_0000_0000_0000__0000_0000_00(01,1x)_xxxx	4
0000_0000_0000_0000__0000_0000_0000_(01,1x)xx	3
0000_0000_0000_0000__0000_0000_0000_00(01,1x)	2
0000_0000_0000_0000__0000_0000_0000_0000	2

32.4.3 Software interface

The programming model of the MMDVSQ is organized to be similar to the input arguments passed to software libraries for integer divide and square root functions.

32.4.3.1 Operation activation and result retrieval

The MMDVSQ supports 2 mechanisms for initiating a divide operation:

- The default mechanism is a "fast start" where a write to the DSOR register begins the divide.
- Alternatively, the start mechanism can begin after a write to the CSR register with the CSR[SRT] set.

The CSR[DFS] indicator selects the divide start mechanism.

```
if CSR[DFS] = 0
   then a divide is initiated by a write to the DSOR register
   else a divide is initiated by a write to the CSR register with CSR[SRT] = 1
```

A square root calculation is initiated by a write to the RCND register.

For both divide and square root calculations, the result of the operation is retrieved by reading the RES register. A memory read of this register while the calculation is still being performed causes the access to be stalled via the insertion of bus wait states until the new result is loaded into the register. Note a stalled bus cycle cannot be interrupted, so if system interrupt latency is a concern, the processor should execute a simple wait loop, for example, polling CSR[BUSY], before reading the RES register. This code construct is fully interruptible, so interrupt latency is minimized.

32.4.3.2 Context save and restore

Given that multiple memory-mapped register accesses are needed for each divide and square root calculation, interrupts may occur during the required sequence of operations. As a result, the MMDVSQ's programming model can be saved at entry to an interrupt service routine (ISR) and then restored when redispatching to the interrupted task.

The module's context can be saved by reading the DEND, DSOR, CSR, and RES registers and storing them as part of the task state. There is one special consideration for the task state save. If the last calculation was a zero divide and the divide-by-zero enable is set (CSR[DZE] = 1), then a read of the RES register is error terminated. To avoid a zero-divide error termination during a context save, the following sequence can be used:

1. Read DEND, DSOR, and CSR registers and save the values as part of the task state.

2. Clear CSR[DZE].
3. Read the RES register and save its value as part of the task state.

When restoring the context, special care must be taken to not initiate another divide calculation. Specifically, CSR[DFS] must be set first before reloading the DEND and DSOR registers. For example, the following sequence can be used for the context reload:

1. Write 0x0000_0020 to the CSR to disable the fast start mechanism.
2. Reload DEND, DSOR, CSR, and RES registers from the saved state.

Since the original context save of the control/status register is guaranteed to have CSR[SRT] = 0, there is no divide operation initiated when this register is reloaded in step 2.

Chapter 33
Miscellaneous Control Module (MCM)

33.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

33.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration
- Crossbar master arbitration policy selection
- Flash controller speculation buffer and cache configurations

33.2 Memory map/register descriptions

The memory map and register descriptions found here describe the registers using byte addresses. The registers can be written only when in supervisor mode.

MCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_3008	Crossbar Switch (AXBS) Slave Configuration (MCM0_PLASC)	16	R	000Fh	33.2.1/858

Table continues on the next page...

MCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_300A	Crossbar Switch (AXBS) Master Configuration (MCM0_PLAMC)	16	R	See section	33.2.2/859
F000_300C	Platform Control Register (MCM0_PLACR)	32	R/W	See section	33.2.3/859
F000_3040	Compute Operation Control Register (MCM0_CPO)	32	R/W	0000_0000h	33.2.4/862

33.2.1 Crossbar Switch (AXBS) Slave Configuration (MCMx_PLASC)

PLASC is a 16-bit read-only register identifying the presence/absence of bus slave connections to the device's crossbar switch.

Address: F000_3000h base + 8h offset = F000_3008h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								ASC							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

MCMx_PLASC field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ASC	Each bit in the ASC field indicates whether there is a corresponding connection to the crossbar switch's slave input port. 0 A bus slave connection to AXBS input port *n* is absent. 1 A bus slave connection to AXBS input port *n* is present.

33.2.2 Crossbar Switch (AXBS) Master Configuration (MCMx_PLAMC)

PLAMC is a 16-bit read-only register identifying the presence/absence of bus master connections to the device's crossbar switch. For MCM0_PLAMC, the reset value is 0x000F. For MCM1_PLAMC, the reset value is 0x0007.

Address: F000_3000h base + Ah offset = F000_300Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								AMC							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

MCMx_PLAMC field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
AMC	Each bit in the AMC field indicates whether there is a corresponding connection to the AXBS master input port. 0 A bus master connection to AXBS input port *n* is absent 1 A bus master connection to AXBS input port *n* is present

33.2.3 Platform Control Register (MCMx_PLACR)

The PLACR register selects the arbitration policy for the crossbar masters and configures the flash memory controller.

The speculation buffer and cache in the flash memory controller is configurable via PLACR[15:10].

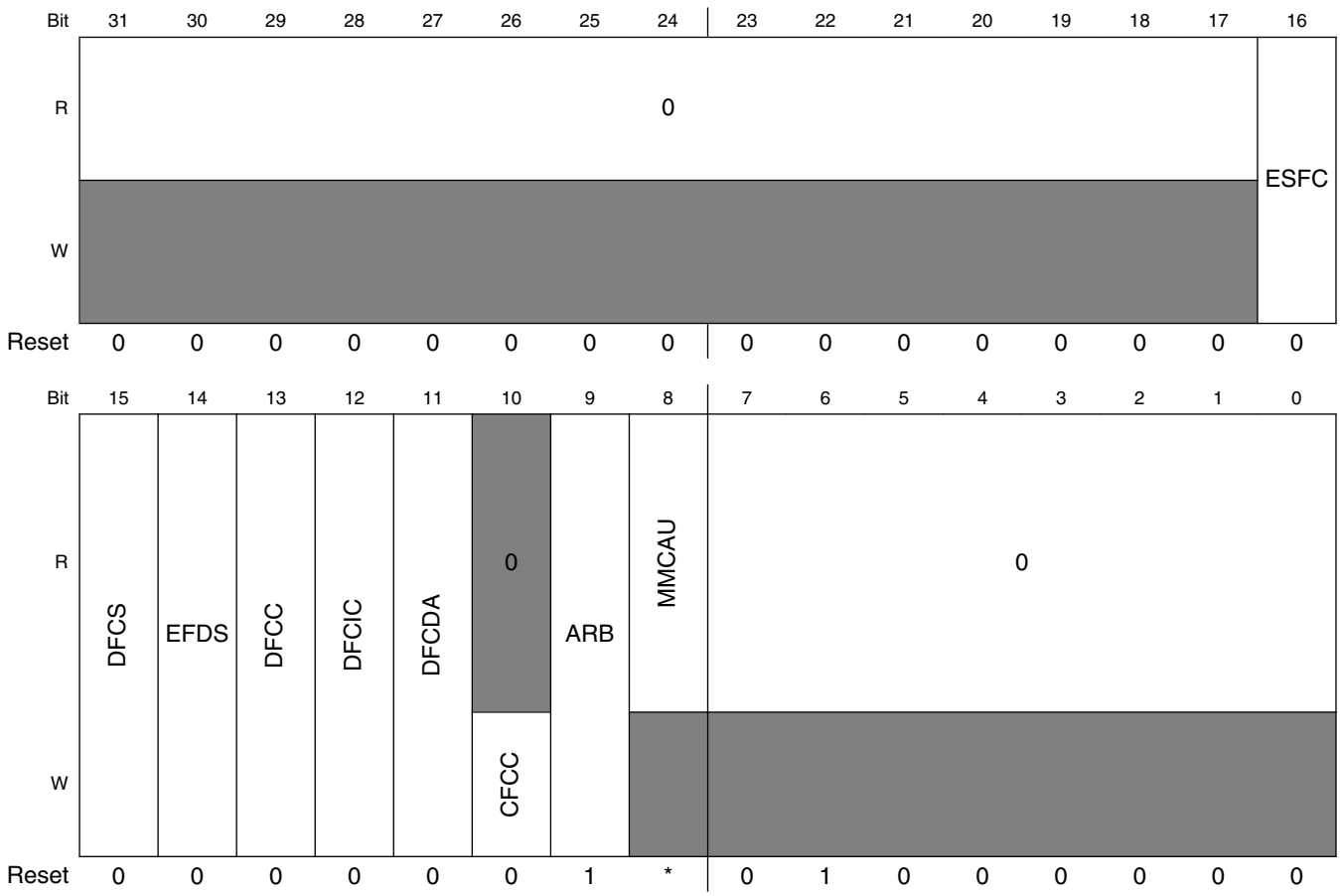
The speculation buffer is enabled only for instructions after reset. It is possible to have these states for the speculation buffer:

DFCS	EFDS	Description
0	0	Speculation buffer is on for instruction and off for data.
0	1	Speculation buffer is on for instruction and on for data.
1	X	Speculation buffer is off.

The cache in flash controller is enabled and caching both instruction and data type fetches after reset. It is possible to have these states for the cache:

DFCC	DFCIC	DFCDA	Description
0	0	0	Cache is on for both instruction and data.
0	0	1	Cache is on for instruction and off for data.
0	1	0	Cache is off for instruction and on for data.
0	1	1	Cache is off for both instruction and data.
1	X	X	Cache is off.

Address: F000_3000h base + Ch offset = F000_300Ch



* Notes:

- MMCAU field: Reset value loaded during reset from Flash IFR.

MCMx_PLACR field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ESFC	Enable Stalling Flash Controller Enables stalling flash controller when flash is busy. When software needs to access the flash memory while a flash memory resource is being manipulated by a flash command, software can enable a stall mechanism to avoid a read collision. The stall mechanism allows software to execute code from the same block on which flash operations are being performed. However, software must ensure the sector the flash operations are being performed on is not the same sector from which the code is executing. ESFC enables the stall mechanism. This bit must be set only just before the flash operation is executed and must be cleared when the operation completes. 0 Disable stalling flash controller when flash is busy. 1 Enable stalling flash controller when flash is busy.
15 DFCS	Disable Flash Controller Speculation Disables flash controller speculation. 0 Enable flash controller speculation. 1 Disable flash controller speculation.
14 EFDS	Enable Flash Data Speculation Enables flash data speculation. 0 Disable flash data speculation. 1 Enable flash data speculation.
13 DFCC	Disable Flash Controller Cache Disables flash controller cache. 0 Enable flash controller cache. 1 Disable flash controller cache.
12 DFCIC	Disable Flash Controller Instruction Caching Disables flash controller instruction caching. 0 Enable flash controller instruction caching. 1 Disable flash controller instruction caching.
11 DFCDA	Disable Flash Controller Data Caching Disables flash controller data caching. 0 Enable flash controller data caching 1 Disable flash controller data caching.
10 CFCC	Clear Flash Controller Cache Writing a 1 to this field clears the cache. Writing a 0 to this field is ignored. This field always reads as 0.
9 ARB	Arbitration select

Table continues on the next page...

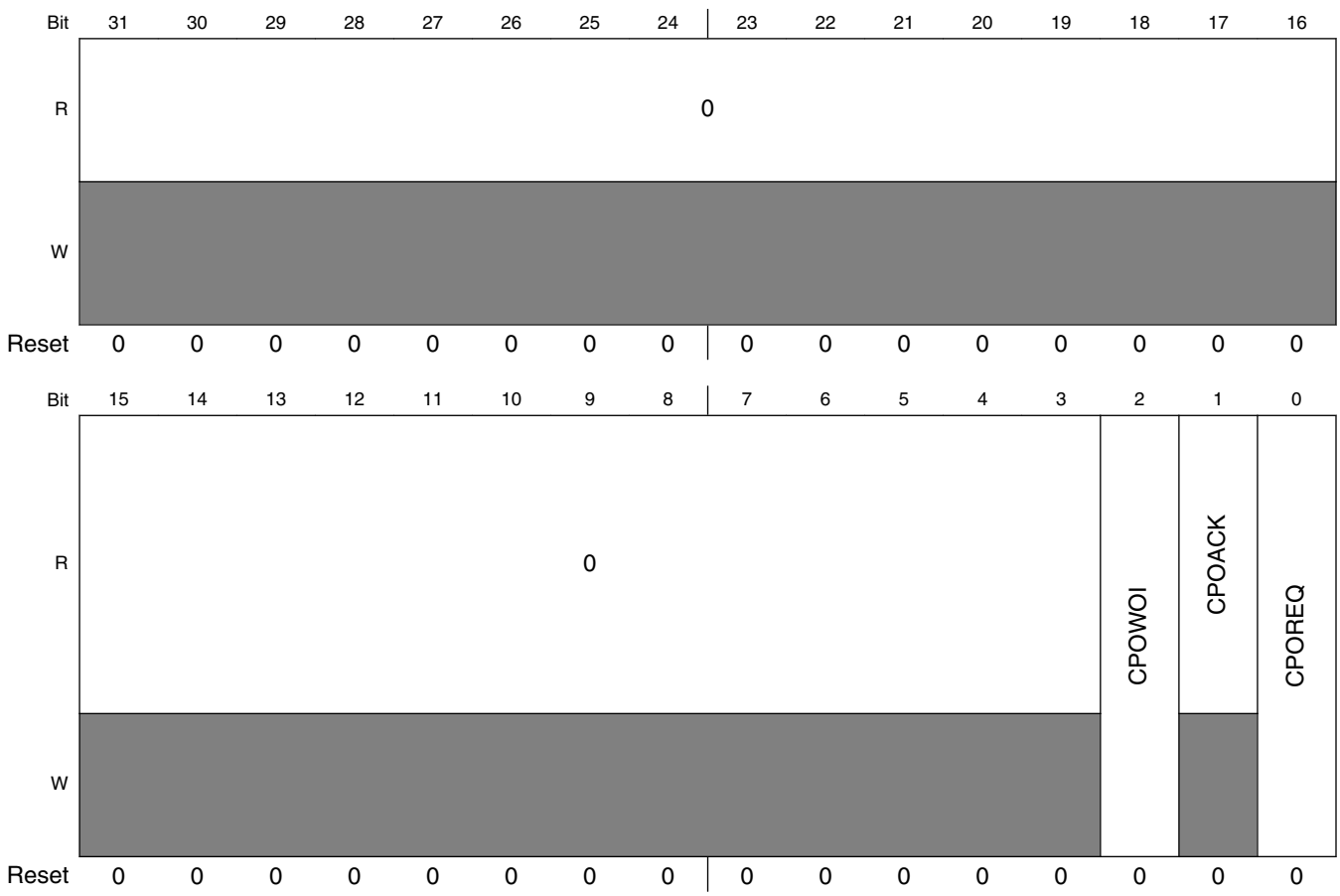
MCMx_PLACR field descriptions (continued)

Field	Description
	0 Fixed-priority arbitration for the crossbar masters 1 Round-robin arbitration for the crossbar masters
8 MMCAU	MMCAU Present 0 MMCAU is disabled 1 MMCAU is enabled
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

33.2.4 Compute Operation Control Register (MCMx_CPO)

This register controls the Compute Operation.

Address: F000_3000h base + 40h offset = F000_3040h



MCMx_CPO field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CPOWOI	Compute Operation Wake-up on Interrupt 0 No effect. 1 When set, the CPOREQ is cleared on any interrupt or exception vector fetch.
1 CPOACK	Compute Operation Acknowledge 0 Compute operation entry has not completed or compute operation exit has completed. 1 Compute operation entry has completed or compute operation exit has not completed.
0 CPOREQ	Compute Operation Request This bit is auto-cleared by vector fetching if CPOWOI = 1. 0 Request is cleared. 1 Request Compute Operation.

33.3 Functional description

This section describes the functional description of MCM module.

33.3.1 Interrupts

The MCM generates two interrupt requests:

- Non-maskable interrupt
- Normal interrupt

33.3.1.1 Non-maskable interrupt

The MCM's NMI is generated if:

- ISCR[ETBN] is set, when
  - The ETB counter is enabled, ETBCC[CNTEN] = 1
  - The ETB count expires
  - The response to counter expiration is an NMI, MCM_ETBCC[RSPT] = 10

33.3.1.2 Normal interrupt

The MCM's normal interrupt is generated if any of the following is true:

- ISCR[ETBI] is set, when
  - The ETB counter is enabled, ETBCC[CNTEN] = 1
  - The ETB count expires
  - The response to counter expiration is a normal interrupt, ETBCC[RSPT] = 01

Chapter 34
Miscellaneous System Control Module (MSCM)

34.1 Chip-Specific Information

This device has a single core (Core 0).

34.2 Overview

The Miscellaneous System Control Module (MSCM) contains CPU configuration registers for Core 0 and on-chip memory controller registers.

34.3 Chip Configuration and Boot

The device configuration is defined by e-fuse bits, supported memory sizes and packing options. Collectively, these configuration bits define a reset configuration value (RCON).

After the core has fetched the needed reset vector(s), it is expected that the core reads core and system configuration information from a globally-accessible slave peripheral that properly converts the information into more appropriate values. More specifically, the core accesses configuration information from a common set of peripheral addresses, and the chip configuration logic properly evaluates configuration info based on the requesting processor, and returns the appropriate value for the given processor, including core identification.

For example, there is a single 32-bit read-only location for the core identification. A 32-bit read from this location returns a 4-character ASCII string: 0x43_4D_30_01 ("CM0").

The programming model associated with the core configuration information is included as part of the Miscellaneous System Control Module (MSCM). It specifically includes multiple views of the processor configuration; one view that is available generically to the core, and other views that are available to any bus masters in the system.

34.4 MSCM Memory Map/Register Definition

34.4.1 CPU Configuration Memory Map and Registers

The CPU configuration portion of the MSCM module provides a set of memory-mapped read-only addresses defining the processor set-up. This portion of the MSCM programming model can only be accessed with privileged mode 32-bit read references; any other access type or size are terminated with an error. If the processor is logically not included in the chip configuration, then reads of its configuration registers return zeroes.

The CPU Configuration registers are organized based on the logical processor number (not any type of physical port number) and partitioned into 3 equal sections:

Table 34-1. CPU Configuration Register Sections

Offset addresses	Function
0x000 - 0x01F	Defines the generic processor "x" configuration. This region is only accessible to the processor core; reads by non-core bus masters are treated as read-as-zero (RAZ) accesses.
0x020 - 0x03F	Defines the configuration information for processor 0 (CP0). This region is accessible to any bus master.
0x040 - 0x05F	Reserved; reads of this section are read as zero (RAZ).

Reads from any other bus master return all zeroes. Attempted user mode or write accesses are terminated with an error.

MSCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_1000	Processor X Type Register (MSCM_CPxTYPE)	32	R	See section	34.4.2/867
4000_1004	Processor X Number Register (MSCM_CPxNUM)	32	R	See section	34.4.3/868
4000_1008	Processor X Master Register (MSCM_CPxMASTER)	32	R	See section	34.4.4/869
4000_100C	Processor X Count Register (MSCM_CPxCOUNT)	32	R	See section	34.4.5/870
4000_1010	Processor X Configuration Register (MSCM_CPxCFG0)	32	R	See section	34.4.6/870
4000_1014	Processor X Configuration Register (MSCM_CPxCFG1)	32	R	See section	34.4.6/870
4000_1018	Processor X Configuration Register (MSCM_CPxCFG2)	32	R	See section	34.4.6/870
4000_101C	Processor X Configuration Register (MSCM_CPxCFG3)	32	R	See section	34.4.6/870
4000_1020	Processor 0 Type Register (MSCM_CP0TYPE)	32	R	See section	34.4.7/871
4000_1024	Processor 0 Number Register (MSCM_CP0NUM)	32	R	See section	34.4.8/872
4000_1028	Processor 0 Master Register (MSCM_CP0MASTER)	32	R	See section	34.4.9/873

Table continues on the next page...

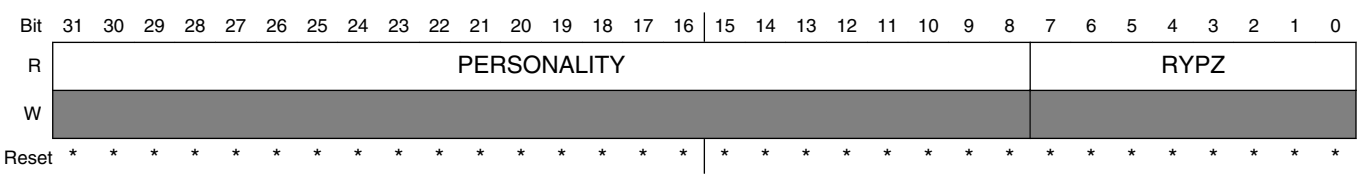
MSCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_102C	Processor 0 Count Register (MSCM_CP0COUNT)	32	R	See section	34.4.10/ 874
4000_1030	Processor 0 Configuration Register (MSCM_CP0CFG0)	32	R	0000_0000h	34.4.11/ 874
4000_1034	Processor 0 Configuration Register (MSCM_CP0CFG1)	32	R	0000_0000h	34.4.11/ 874
4000_1038	Processor 0 Configuration Register (MSCM_CP0CFG2)	32	R	0000_0000h	34.4.11/ 874
4000_103C	Processor 0 Configuration Register (MSCM_CP0CFG3)	32	R	0000_0000h	34.4.11/ 874
4000_1400	On-Chip Memory Descriptor Register (MSCM_OCMDR0)	32	R/W	See section	34.4.12/ 875
4000_1404	On-Chip Memory Descriptor Register (MSCM_OCMDR1)	32	R/W	See section	34.4.12/ 875
4000_1408	On-Chip Memory Descriptor Register (MSCM_OCMDR2)	32	R/W	See section	34.4.12/ 875

34.4.2 Processor X Type Register (MSCM_CPxTYPE)

The register provides a CPU-specific response indicating the personality of the core making the access. The 32-bit response includes 3 ASCII characters that define the CPU type, along with a byte that defines the logical revision number. The logical revision number follows ARM's rYpZ nomenclature.

Address: 4000_1000h base + 0h offset = 4000_1000h



* Notes:
- PERSONALITY field: See bit field description
- RYPZ field: See bit field description

MSCM_CPxTYPE field descriptions

Field	Description
31–8 PERSONALITY	Processor x Personality This read-only field defines the processor personality for CPx

Table continues on the next page...

MSCM_CPxTYPE field descriptions (continued)

Field	Description
	if CPx = Cortex-M0, then PERSONALITY = 0x43_4D_30 (“CM0”).
RYPZ	Processor x Revision This read-only field defines the processor revision for CPx: 0x00 corresponds to the r0p0 core release. 0x01 corresponds to the r0p1 core release. ...

34.4.3 Processor X Number Register (MSCM_CPxNUM)

The register provides a CPU-specific response indicating the logical processor number of the core making the access. The logical processor number is always 0.

Address: 4000_1000h base + 4h offset = 4000_1004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0															CPN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	*

* Notes:
- CPN field: See bit field description

MSCM_CPxNUM field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CPN	Processor x Number This zero-filled word defines the logical processor number for CPx CPN = 0

34.4.4 Processor X Master Register (MSCM_CPxMASTER)

The register provides a CPU-specific response indicating the physical bus master number of the core that is making the access. The 32-bit response defines the physical master number for processor x.

- A privileged read from the CM0+ returns the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Attempted user mode or write accesses are terminated with an error.

Address: 4000_1000h base + 8h offset = 4000_1008h

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

* Notes:
- PPN field: See the bit field description.

MSCM_CPxMASTER field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PPN	Processor x Physical Port Number This read-only field defines the physical port number for the core. For the core, PPN = 0x00

34.4.5 Processor X Count Register (MSCM_CPxCOUNT)

The register provides a CPU-specific response indicating the total number of processor cores in the chip configuration.

Address: 4000_1000h base + Ch offset = 4000_100Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0														PCNT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	*	*

* Notes:
- PCNT field: See bit field description

MSCM_CPxCOUNT field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PCNT	Processor Count This read-only field defines the processor count for the chip configuration: PCNT = 00

34.4.6 Processor X Configuration Register (MSCM_CPxCFG*n*)

The register provides information on the Level 1 caches (if present).

NOTE

Reset values for the 4 Processor X Configuration Registers:
- MSCM_CPxCFG0 = 0x0000_0000
- MSCM_CPxCFG1 = 0x0000_0000
- MSCM_CPxCFG2 = 0x0001_0001
- MSCM_CPxCFG3 = 0x0000_0120

Address: 4000_1000h base + 10h offset + (4d × i), where i=0d to 3d

Bit	31–24	23–16	15–8	7–0
R	ICSZ	ICWY	DCSZ	DCWY
W				
Reset	*	*	*	*

* Notes:
- ICSZ field: See register description for reset values.
- ICWY field: See register description for reset values.
- DCSZ field: See register description for reset values.
- DCWY field: See register description for reset values.

MSCM_CPxCFG*n* field descriptions

Field	Description
31–24 ICSZ	Level 1 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(8+SZ)}$, where SZ is non-zero; a SZ = 0 indicates the memory is not present.
23–16 ICWY	Level 1 Instruction Cache Ways This read-only field provides the number of cache ways for the Instruction Cache.
15–8 DCSZ	Level 1 Data Cache Size This read-only field provides an encoded value of the Data Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(8+SZ)}$,where SZ is non-zero; a SZ = 0 indicates the memory is not present.
DCWY	Level 1 Data Cache Ways This read-only field provides the number of cache ways for the Data Cache.

34.4.7 Processor 0 Type Register (MSCM_CP0TYPE)

The register provides a CPU-specific response indicating the personality of the core making the access. The 32-bit response includes 3 ASCII characters defining the CPU type, along with a byte defining the logical revision number. The logical revision number follows ARM's rYpZ nomenclature.

Address: 4000_1000h base + 20h offset = 4000_1020h

Bit	31–8	7–0
R	PERSONALITY	RYPZ
W		
Reset	*	*

* Notes:
- PERSONALITY field: See bit field description
- RYPZ field: See bit field description

MSCM_CP0TYPE field descriptions

Field	Description
31–8 PERSONALITY	Processor x Personality This read-only field defines the processor personality for CPx if CPx = Cortex-M0, then PERSONALITY = 0x43_4D_30 (“CM0”).
RYPZ	Processor x Revision This read-only field defines the processor revision for CPx: 0x00 corresponds to the r0p0 core release. 0x01 corresponds to the r0p1 core release. ...

34.4.8 Processor 0 Number Register (MSCM_CP0NUM)

The register provides a CPU-specific response indicating the logical processor number of the core making the access. The logical processor number is always 0.

Address: 4000_1000h base + 24h offset = 4000_1024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0															CPN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	*

* Notes:
- CPN field: See bit field description

MSCM_CP0NUM field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CPN	Processor x Number This zero-filled word defines the logical processor number for CPx CPN = 0

34.4.9 Processor 0 Master Register (MSCM_CP0MASTER)

The register provides a CPU-specific response, that indicates the physical bus master number of the core making the access. The 32-bit response defines the physical master number for processor x.

- A privileged read from the CM0+ returns the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Attempted user mode or write accesses are terminated with an error.

Address: 4000_1000h base + 28h offset = 4000_1028h

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

* Notes:
- PPN field: See the bit field description.

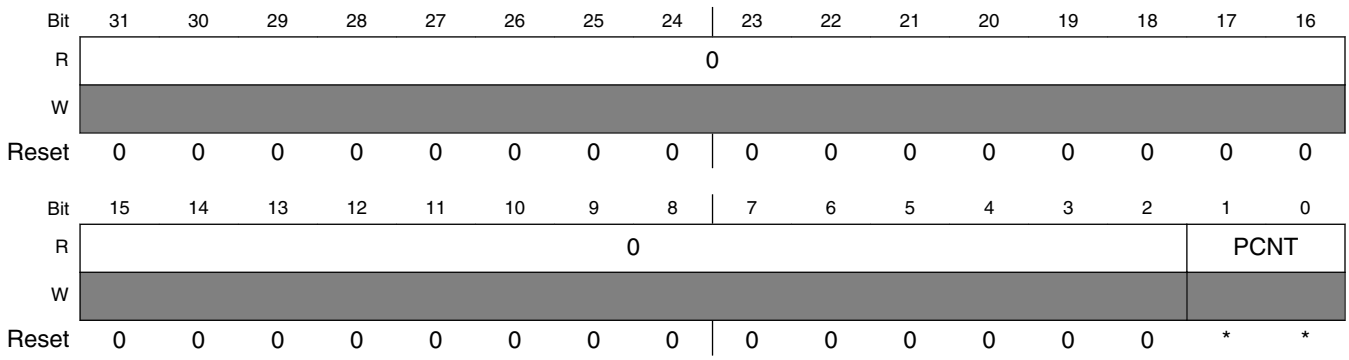
MSCM_CP0MASTER field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PPN	Processor x Physical Port Number This read-only field defines the physical port number for CPUx. For CPU0, PPN = 0x00

34.4.10 Processor 0 Count Register (MSCM_CP0COUNT)

The register provides a CPU-specific response indicating the total number of processor cores in the chip configuration.

Address: 4000_1000h base + 2Ch offset = 4000_102Ch



* Notes:
- PCNT field: See bit field description

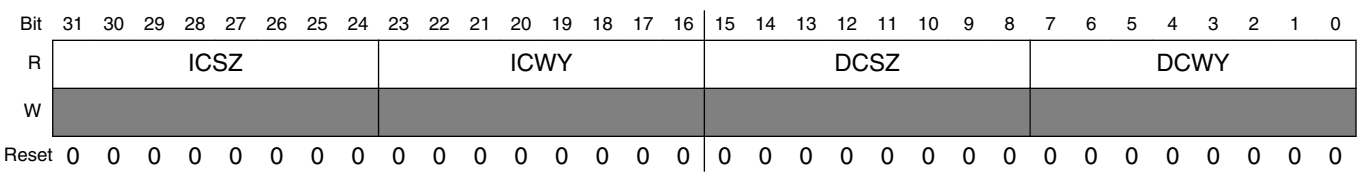
MSCM_CP0COUNT field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PCNT	Processor Count This read-only field defines the processor count for the chip configuration: PCNT = 00

34.4.11 Processor 0 Configuration Register (MSCM_CP0CFG*n*)

The register provides information on the Level 1 caches (if present).

Address: 4000_1000h base + 30h offset + (4d × i), where i=0d to 3d



MSCM_CP0CFG*n* field descriptions

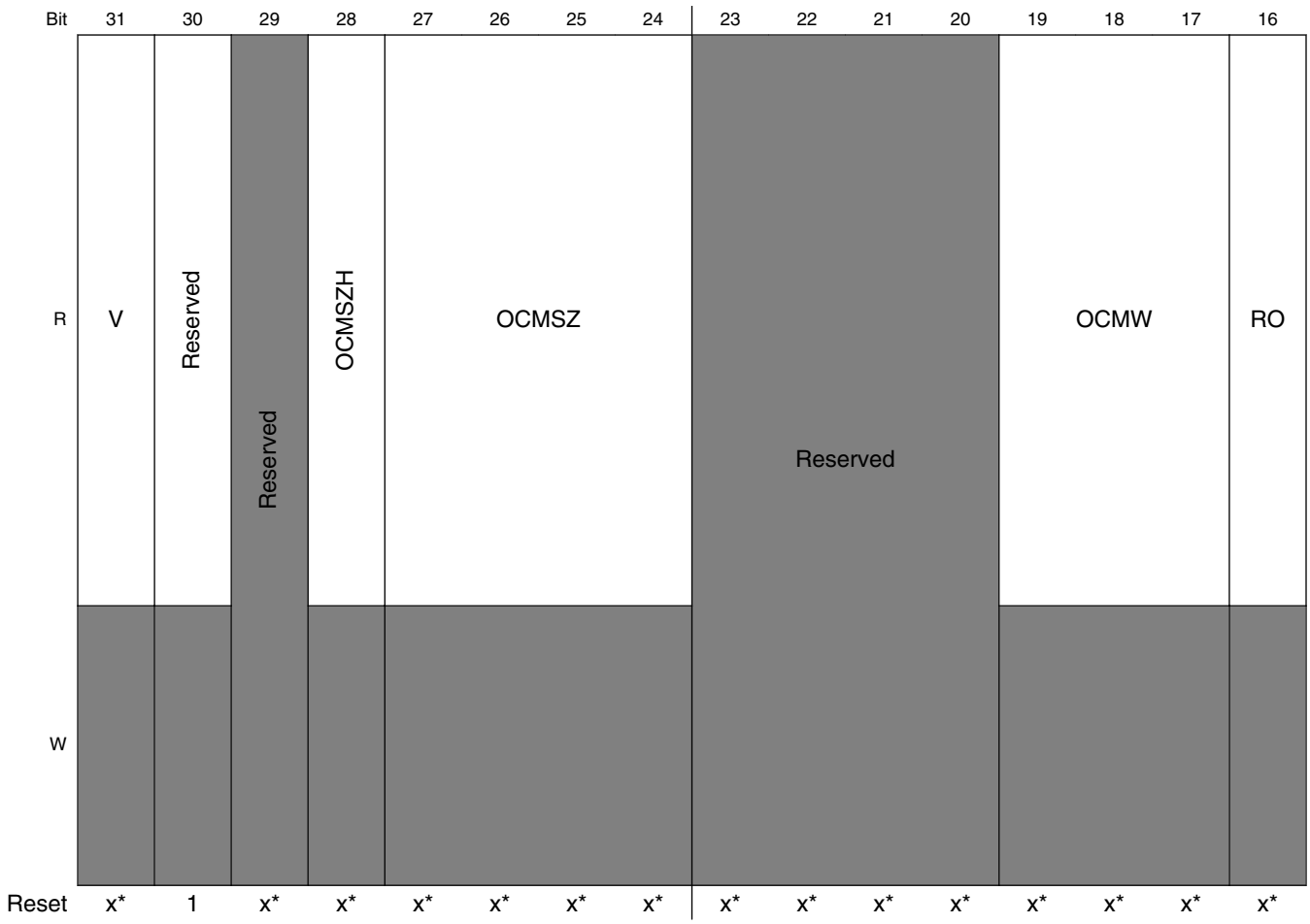
Field	Description
31–24 ICSZ	Level 1 Instruction Cache Size This read-only field provides an encoded value of the Instruction Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(8+SZ)}$, where SZ is non-zero; a SZ = 0 indicates the memory is not present.
23–16 ICWY	Level 1 Instruction Cache Ways This read-only field provides the number of cache ways for the Instruction Cache.
15–8 DCSZ	Level 1 Data Cache Size This read-only field provides an encoded value of the Data Cache size. The capacity of the memory is expressed as Size [bytes] = $2^{(8+SZ)}$,where SZ is non-zero; a SZ = 0 indicates the memory is not present.
DCWY	Level 1 Data Cache Ways This read-only field provides the number of cache ways for the Data Cache.

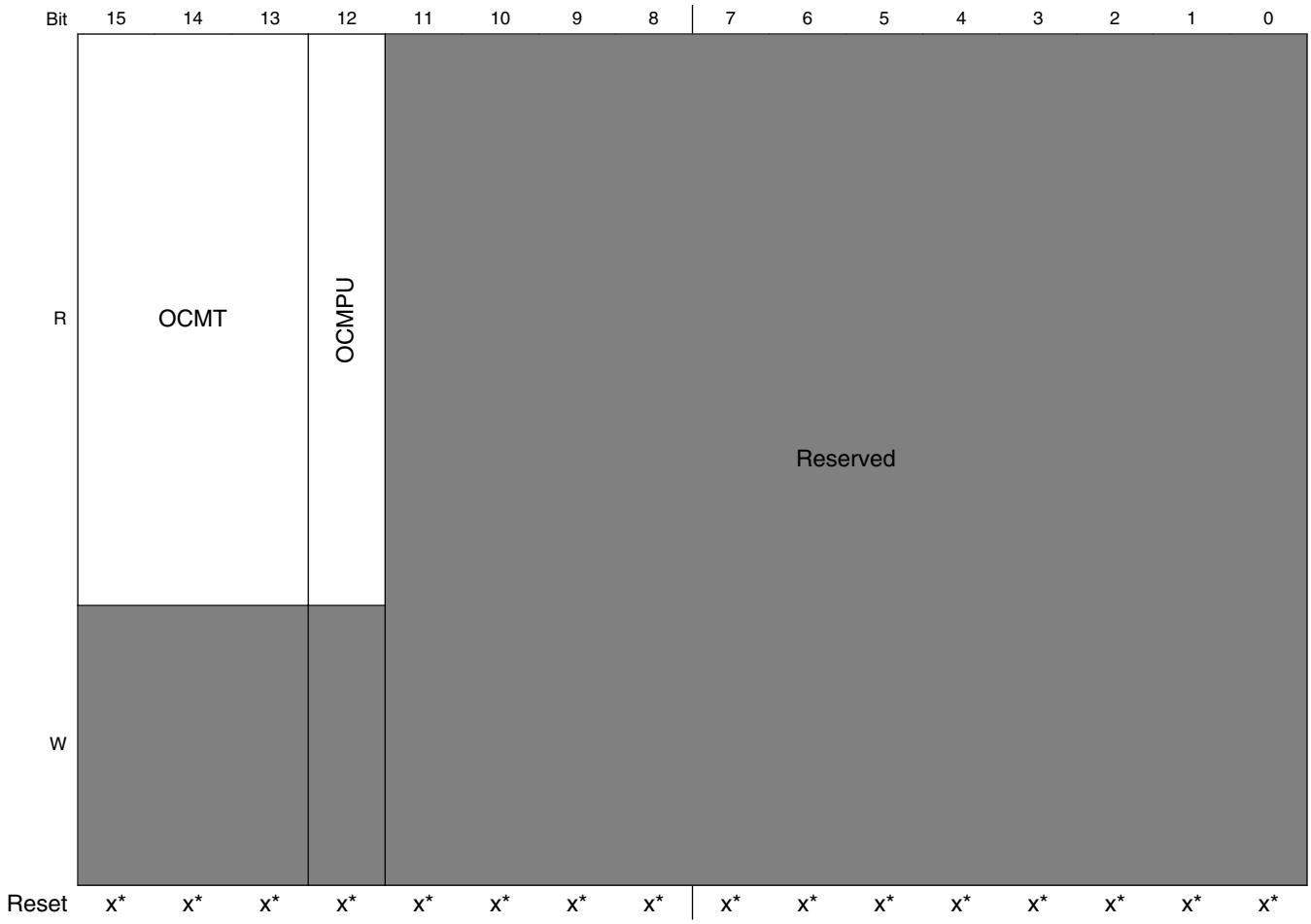
34.4.12 On-Chip Memory Descriptor Register (MSCM_OCMDR*n*)

This section of the programming model is an array of 32-bit generic on-chip memory descriptor registers that provide static information about the attached memories, as well as configurable controls (where appropriate).

- Privileged 32-bit reads from a processor core or the debugger return the appropriate processor information.
- Reads from any other bus master return all zeroes.
- Privileged writes from a processor core or the debugger to writeable registers update the appropriate fields.
- Privileged writes from other bus masters are ignored.
- Attempted user mode accesses or any access with a size other than 32 bits are terminated with an error.

Address: 4000_1000h base + 400h offset + (4d × i), where i=0d to 2d





* Notes:
- The reset values are different for the individual OCMDR registers. OCMDR0: 0xD904_0000; OCMDR1: 0xC704_0000; OCMDR2: 0xC704_6000x = Undefined at reset.

MSCM_OCMDR*n* field descriptions

Field	Description
31 V	OCMEM Valid bit. This read-only field defines the validity (presence) of the on-chip memory 0 OCMEMn is not present. 1 OCMEMn is present.
30 Reserved	This field is reserved. This Reserved field always has the value of 1.
29 Reserved	This field is reserved.
28 OCMSZH	OCMEM Size “Hole”. For on-chip memories that are not fully populated, that is, include a memory “hole” in the upper 25% of the address range, this bit is used. 0 OCMEMn is a power-of-2 capacity. 1 OCMEMn is not a power-of-2, with a capacity is 0.75 * OCMSZ.

Table continues on the next page...

MSCM_OCMDR*n* field descriptions (continued)

Field	Description
27–24 OCMSZ	OCMEM Size. This read-only field provides an encoded value of the on-chip memory size. The capacity of the memory is expressed as Size [bytes] = 2(8+SZ) where SZ is non-zero; a SZ = 0 indicates the memory is not present. 0000 no OCMEMn 0100 4KB OCMEMn 0101 8KB OCMEMn 0110 16KB OCMEMn 0111 32KB OCMEMn : 1111 8192KB OCMEMn
23–20 Reserved	This field is reserved.
19–17 OCMW	OCMEM datapath Width. This read-only field defines the width of the on-chip memory: 000-001 Reserved 010 OCMEMn 32-bits wide 011 OCMEMn 64-bits wide 100-111 Reserved
16 RO	Read-Only. This register bit provides a mechanism to “lock” the configuration state defined by OCMDRn[11:0]. Once asserted, attempted writes to the OCMDRn[11:0] register are ignored until the next reset clears the flag. 0 Writes to the OCMDRn[11:0] are allowed 1 Writes to the OCMDRn[11:0] are ignored
15–13 OCMT	OCMEM Type. This field defines the type of the on-chip memory: 000 OCMEMn is a system RAM. 001 OCMEMn is a graphics RAM. 010 Reserved 011 OCMEMn is a ROM. 100 Reserved 101 Reserved 110 Reserved 111 Reserved
12 OCMPU	OCMEM Memory Protection Unit. This field is reserved for this device.
Reserved	This field is reserved.

Chapter 35
Micro Trace Buffer (MTB)

35.1 Introduction

Microcontrollers using the Cortex-M0+ processor core include support for a CoreSight Micro Trace Buffer to provide program trace capabilities.

The proper name for this function is the CoreSight Micro Trace Buffer for the Cortex-M0+ Processor; in this document, it is simply abbreviated as the MTB.

The simple program trace function creates instruction address change-of-flow data packets in a user-defined region of the system RAM. Accordingly, the system RAM controller manages requests from two sources:

- AMBA-AHB reads and writes from the system bus
- program trace packet writes from the processor

As part of the MTB functionality, there is a DWT (Data Watchpoint and Trace) module that allows the user to define watchpoint addresses, or optionally, an address and data value, that when triggered, can be used to start or stop the program trace recording.

This document details the functionality of both the MTB_RAM and MTB_DWT capabilities.

35.1.1 Overview

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown as follows:

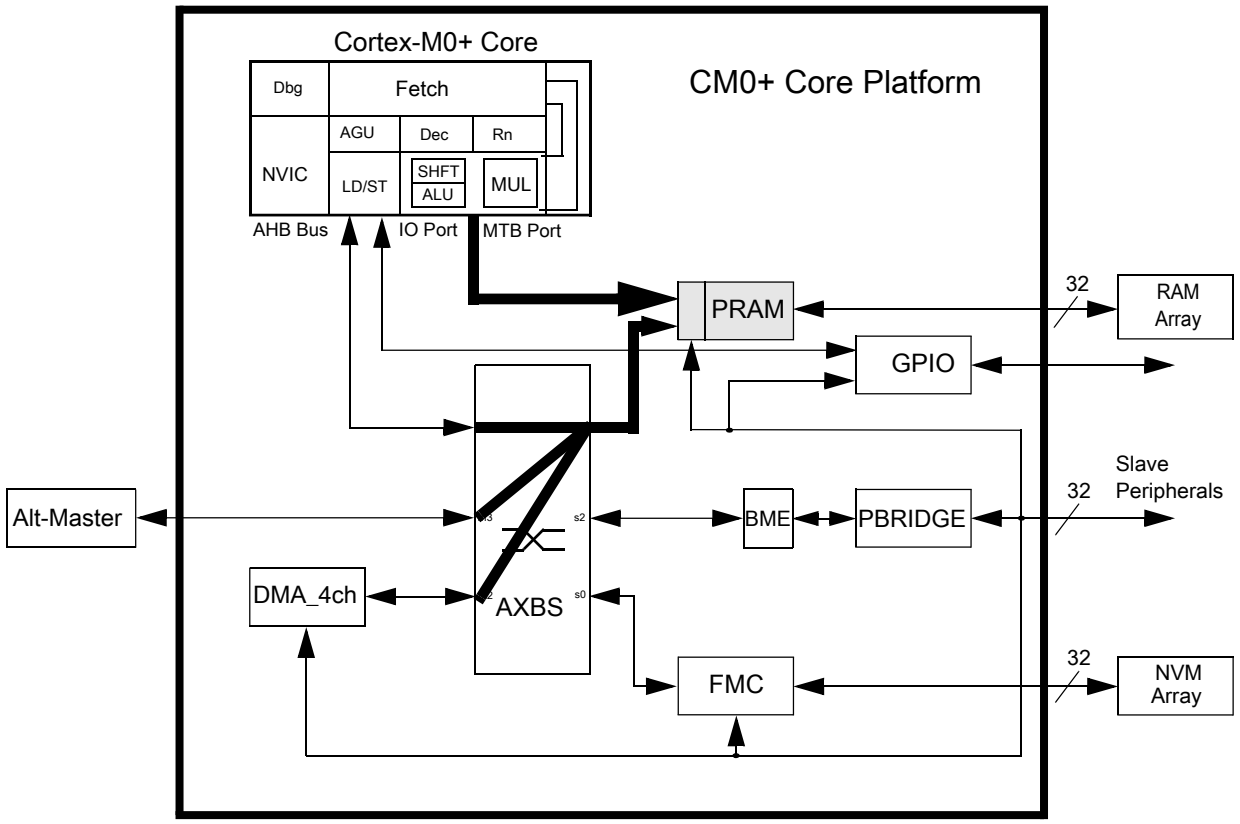


Figure 35-1. Generic Cortex-M0+ core platform block diagram

As shown in the block diagram, the platform RAM (PRAM) controller connects to two input buses:

- the crossbar slave port for system bus accesses
- a "private execution MTB port" from the core

The logical paths from the crossbar master input ports to the PRAM controller are highlighted along with the private execution trace port from the processor core. The private MTB port signals the instruction address information needed for the 64-bit program trace packets written into the system RAM. The PRAM controller output interfaces to the attached RAM array. In this document, the PRAM controller is the MTB_RAM controller.

The following information is taken from the ARM CoreSight Micro Trace Buffer documentation.

"The execution trace packet consists of a pair of 32-bit words that the MTB generates when it detects the processor PC value changes non-sequentially. A non-sequential PC change can occur during branch instructions or during exception entry.

The processor can cause a trace packet to be generated for any instruction.

The following figure shows how the execution trace information is stored in memory as a sequence of packets.

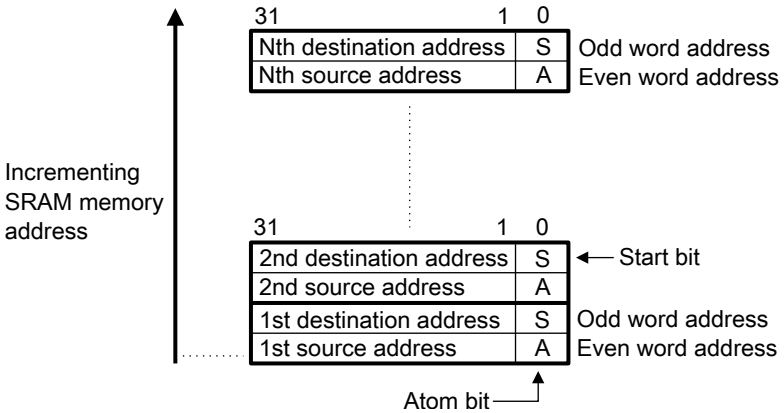


Figure 35-2. MTB execution trace storage format

The first, lower addressed, word contains the source of the branch, the address it branched from. The value stored only records bits[31:1] of the source address, because Thumb instructions are at least halfword aligned. The least significant bit of the value is the A-bit. The A-bit indicates the atomic state of the processor at the time of the branch, and can differentiate whether the branch originated from an instruction in a program, an exception, or a PC update in Debug state. When it is zero the branch originated from an instruction, when it is one the branch originated from an exception or PC update in Debug state. This word is always stored at an even word location.

The second, higher addressed word contains the destination of the branch, the address it branched to. The value stored only records bits[31:1] of the branch address. The least significant bit of the value is the S-bit. The S-bit indicates where the trace started. An S-bit value of 1 indicates where the first packet after the trace started and a value of 0 is used for other packets. Because it is possible to start and stop tracing multiple times in a trace session, the memory might contain several packets with the S-bit set to 1. This word is always stored in the next higher word in memory, an odd word address.

When the A-bit is set to 1, the source address field contains the architecturally-preferred return address for the exception. For example, if an exception was caused by an SVC instruction, then the source address field contains the address of the following instruction. This is different from the case where the A-bit is set to 0. In this case, the source address contains the address of the branch instruction.

For an exception return operation, two packets are generated:

- The first packet has the:
  - Source address field set to the address of the instruction that causes the exception return, BX or POP.

    - Destination address field set to bits[31:1] of the EXC_RETURN value. See the ARM v6-M Architecture Reference Manual.
    - The A-bit set to 0.

- The second packet has the:
    - Source address field set to bits[31:1] of the EXC_RETURN value.
    - Destination address field set to the address of the instruction where execution commences.
    - A-bit set to 1."

Given the recorded change-of-flow trace packets in system RAM and the memory image of the application, a debugger can read out the data and create an instruction-by-instruction program trace. In keeping with the low area and power implementation cost design targets, the MTB trace format is less efficient than other CoreSight trace modules, for example, the ETM (Embedded Trace Macrocell). Since each branch packet is 8 bytes in size, a 1 KB block of system RAM can contain 128 branches. Using the Dhrystone 2.1 benchmark's dynamic runtime as an example, this corresponds to about 875 instructions per KB of trace RAM, or with a zero wait state memory, this corresponds to approximately 1600 processor cycles per KB. This metric is obviously very sensitive to the runtime characteristics of the user code.

The MTB_DWT function (not shown in the core platform block diagram) monitors the processor address and data buses so that configurable watchpoints can be detected to trigger the appropriate response in the MTB recording.

35.1.2 Features

The key features of the MTB_RAM and MTB_DWT include:

- Memory controller for system RAM and Micro Trace Buffer for program trace packets
- Read/write capabilities for system RAM accesses, write-only for program trace packets
- Supports zero wait state response to system bus accesses when no trace data is being written
- Can buffer two AHB address phases and one data write for system RAM accesses
- Supports 64-bit program trace packets including source and destination instruction addresses
- Program trace information in RAM available to MCU's application code or external debugger
- Program trace watchpoint configuration accessible by MCU's application code or debugger
- Location and size of RAM trace buffer is configured by software

- Two DWT comparators (addresses or address + data) provide programmable start/ stop recording
- CoreSight compliant debug functionality

35.1.3 Modes of operation

The MTB_RAM and MTB_DWT functions do not support any special modes of operation. The MTB_RAM controller, as a memory-mapped device located on the platform's slave AHB system bus, responds strictly on the basis of memory addresses for accesses to its attached RAM array. The MTB private execution bus provides program trace packet write information to the RAM controller. Both the MTB_RAM and MTB_DWT modules are memory-mapped, so their programming models can be accessed.

All functionality associated with the MTB_RAM and MTB_DWT modules resides in the core platform's clock domain; this includes its connections with the RAM array.

35.2 External signal description

The MTB_RAM and MTB_DWT modules do not directly support any external interfaces.

The internal interface includes a standard AHB bus with a 32-bit datapath width from the appropriate crossbar slave port plus the private execution trace bus from the processor core. The signals in the private execution trace bus are detailed in the following table taken from the ARM CoreSight Micro Trace Buffer documentation. The signal direction is defined as viewed by the MTB_RAM controller.

Table 35-1. Private execution trace port from the core to MTB_RAM

Signal	Direction	Description
LOCKUP	Input	Indicates the processor is in the Lockup state. This signal is driven LOW for cycles when the processor is executing normally and driven HIGH for every cycle the processor is waiting in the Lockup state. This signal is valid on every cycle.
IAESEQ	Input	Indicates the next instruction address in execute, IAEX, is sequential, that is non-branching.
IAEXEN	Input	IAEX register enable.
IAEX[30:0]	Input	Registered address of the instruction in the execution stage, shifted right by one bit, that is, PC >> 1.
ATOMIC	Input	Indicates the processor is performing non-instruction related activities.
EDBGRQ	Output	Request for the processor to enter the Debug state, if enabled, and halt.

In addition, there are two signals formed by the MTB_DWT module and driven to the MTB_RAM controller: TSTART (trace start) and TSTOP (trace stop). These signals can be configured using the trace watchpoints to define programmable addresses and data values to affect the program trace recording state.

35.3 Memory map and register definition

The MTB_RAM and MTB_DWT modules each support a sparsely-populated 4 KB address space for their programming models. For each address space, there are a variety of control and configurable registers near the base address, followed by a large unused address space and finally a set of CoreSight registers to support dynamic determination of the debug configuration for the device.

Accesses to the programming model follow standard ARM conventions. Taken from the ARM CoreSight Micro Trace Buffer documentation, these are:

- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in UNPREDICTABLE behavior.
- The behavior of the MTB is UNPREDICTABLE if the registers with UNKNOWN reset values are not programmed prior to enabling trace.
- Unless otherwise stated in the accompanying text:
  - Do not modify reserved register bits
  - Ignore reserved register bits on reads
  - All register bits are reset to a logic 0 by a system or power-on reset
  - Use only word size, 32-bit, transactions to access all registers

35.3.1 MTB_RAM Memory Map

MTB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_0000	MTB Position Register (MTB0_POSITION)	32	R/W	Undefined	35.3.1.1/ 886
F000_0004	MTB Master Register (MTB0_MASTER)	32	R/W	See section	35.3.1.2/ 887
F000_0008	MTB Flow Register (MTB0_FLOW)	32	R/W	Undefined	35.3.1.3/ 889
F000_000C	MTB Base Register (MTB0_BASE)	32	R	Undefined	35.3.1.4/ 891
F000_0F00	Integration Mode Control Register (MTB0_MODECTRL)	32	R	0000_0000h	35.3.1.5/ 891

Table continues on the next page...

MTB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_0FA0	Claim TAG Set Register (MTB0_TAGSET)	32	R	0000_0000h	35.3.1.6/ 892
F000_0FA4	Claim TAG Clear Register (MTB0_TAGCLEAR)	32	R	0000_0000h	35.3.1.7/ 892
F000_0FB0	Lock Access Register (MTB0_LOCKACCESS)	32	R	0000_0000h	35.3.1.8/ 893
F000_0FB4	Lock Status Register (MTB0_LOCKSTAT)	32	R	0000_0000h	35.3.1.9/ 893
F000_0FB8	Authentication Status Register (MTB0_AUTHSTAT)	32	R	0000_0000h	35.3.1.10/ 893
F000_0FBC	Device Architecture Register (MTB0_DEVICEARCH)	32	R	4770_0A31h	35.3.1.11/ 894
F000_0FC8	Device Configuration Register (MTB0_DEVICECFG)	32	R	0000_0000h	35.3.1.12/ 895
F000_0FCC	Device Type Identifier Register (MTB0_DEVICETYPID)	32	R	0000_0031h	35.3.1.13/ 895
F000_0FD0	Peripheral ID Register (MTB0_PERIPHID4)	32	R	See section	35.3.1.14/ 896
F000_0FD4	Peripheral ID Register (MTB0_PERIPHID5)	32	R	See section	35.3.1.14/ 896
F000_0FD8	Peripheral ID Register (MTB0_PERIPHID6)	32	R	See section	35.3.1.14/ 896
F000_0FDC	Peripheral ID Register (MTB0_PERIPHID7)	32	R	See section	35.3.1.14/ 896
F000_0FE0	Peripheral ID Register (MTB0_PERIPHID0)	32	R	See section	35.3.1.14/ 896
F000_0FE4	Peripheral ID Register (MTB0_PERIPHID1)	32	R	See section	35.3.1.14/ 896
F000_0FE8	Peripheral ID Register (MTB0_PERIPHID2)	32	R	See section	35.3.1.14/ 896
F000_0FEC	Peripheral ID Register (MTB0_PERIPHID3)	32	R	See section	35.3.1.14/ 896
F000_0FF0	Component ID Register (MTB0_COMPID0)	32	R	See section	35.3.1.15/ 896
F000_0FF4	Component ID Register (MTB0_COMPID1)	32	R	See section	35.3.1.15/ 896
F000_0FF8	Component ID Register (MTB0_COMPID2)	32	R	See section	35.3.1.15/ 896
F000_0FFC	Component ID Register (MTB0_COMPID3)	32	R	See section	35.3.1.15/ 896

35.3.1.1 MTB Position Register (MTB*x*_POSITION)

The MTB_POSITION register contains the Trace Write Address Pointer and Wrap fields. This register can be modified by the explicit programming model writes. It is also automatically updated by the MTB hardware when trace packets are being recorded.

The base address of the system RAM in the memory map dictates special consideration for the placement of the MTB. Consider the following guidelines:

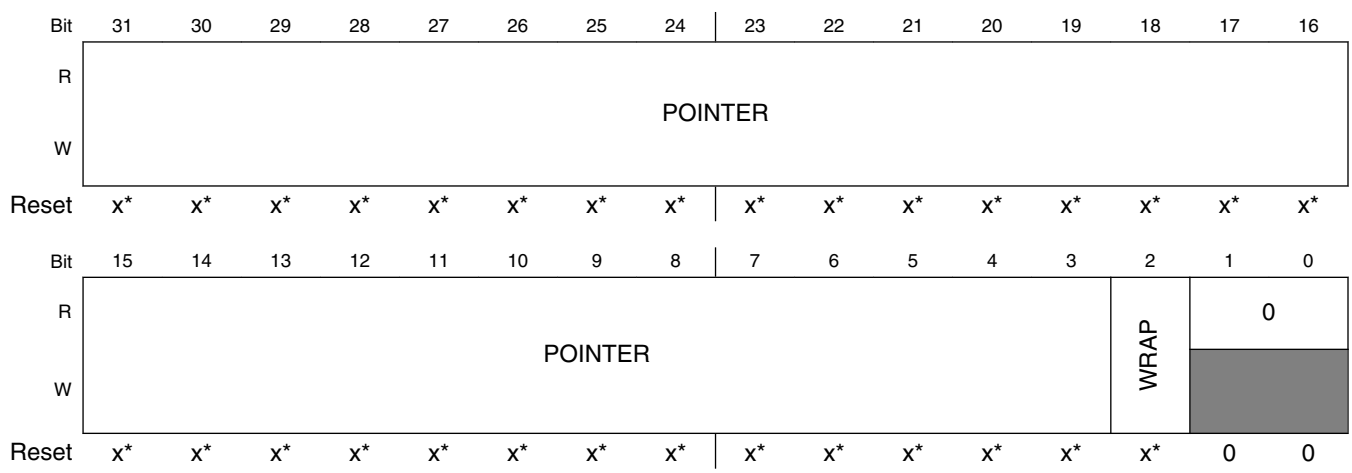
For the standard configuration where the size of the MTB is ≤ 25% of the total RAM capacity, it is recommended the MTB be based at the address defined by the MTB_BASE register. The read-only MTB_BASE register is defined by the expression (0x2000_0000 - (RAM_Size/4)). For this configuration, the MTB0_POSITION register is initialized to MTB0_BASE & 0x0000_1FF8.

If the size of the MTB is more than 25% but less than or equal to 50% of the total RAM capacity, then it is recommended the MTB be based at address 0x2000_0000. In this configuration, the MTB_POSITION register is initialized to (0x2000_0000 & 0x0000_7FF8) = 0x0000_00000.

Following these two suggested placements provides a full-featured circular memory buffer containing program trace packets.

In the unlikely event an even larger trace buffer is required, a write-once capacity of 75% of the total RAM capacity can be based at address 0x2000_0000. The MTB_POSITION register is initialized to (0x2000_0000 & 0x0000_7FF8) = 0x0000_0000. However, this configuration cannot support operation as a circular queue and instead requires the use of the MTB_FLOW[WATERMARK] capability to automatically disable tracing or halting the processor as the number of packet writes approach the buffer capacity. See the MTB_FLOW register description for more details.

Address: F000_0000h base + 0h offset = F000_0000h



* Notes:
- x = Undefined at reset.

MTBx_POSITION field descriptions

Field	Description

31–3
POINTER
	Trace Packet Address Pointer[28:0]
	
	Because a packet consists of 2 words, the POINTER field is the address of the first word of a packet. This field contains bits[31:3] of the RAM address where the next trace packet is written. Therefore, it points to an unused location and is automatically incremented.
	
	A debug agent can calculate the system memory map address for the current location in the MTB, using the following "generic" equation:
	
	Given mtb_size = 1 << (MTB_MASTER[MASK] + 4),
	
	systemAddress = MTB_BASE + (((MTB_POSITION & 0xFFFF_FFF8) + (mtb_size - (MTB_BASE & (mtb_size-1)))) & 0x0000_7FF8);
	
	For this device, a simpler expression (in pseudo-code) also applies:
	
	if ((MTB_POSITION >> 15) == 0x3) systemAddress = (0x1FFF << 16) + (MTB_POSITION & 0xFFF8);
	else systemAddress = (0x2000 << 16) + (MTB_POSITION & 0x1FFF8);
	
	NOTE: The size of the RAM is parameterized and the most significant bits of the POINTER field are RAZ/WI.
	
	For these devices, POSITION[31:17] == POSITION[POINTER[28:14]] are RAZ/WI. Therefore, the active bits in this field are POSITION[16:3] == POSITION[POINTER[13:0]].

2 WRAP	WRAP This field is set to 1 automatically when the POINTER value wraps (as determined by the MTB_MASTER[MASK] field in the MASTER Trace Control Register). A debug agent might use the WRAP field to determine whether the trace information above and below the pointer address is valid.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

35.3.1.2 MTB Master Register (MTBx_MASTER)

The MTB_MASTER register contains the main program trace enable plus other trace controls. This register can be modified by the explicit programming model writes. MTB_MASTER[EN] and MTB_MASTER[HALTREQ] fields are also automatically updated by the MTB hardware.

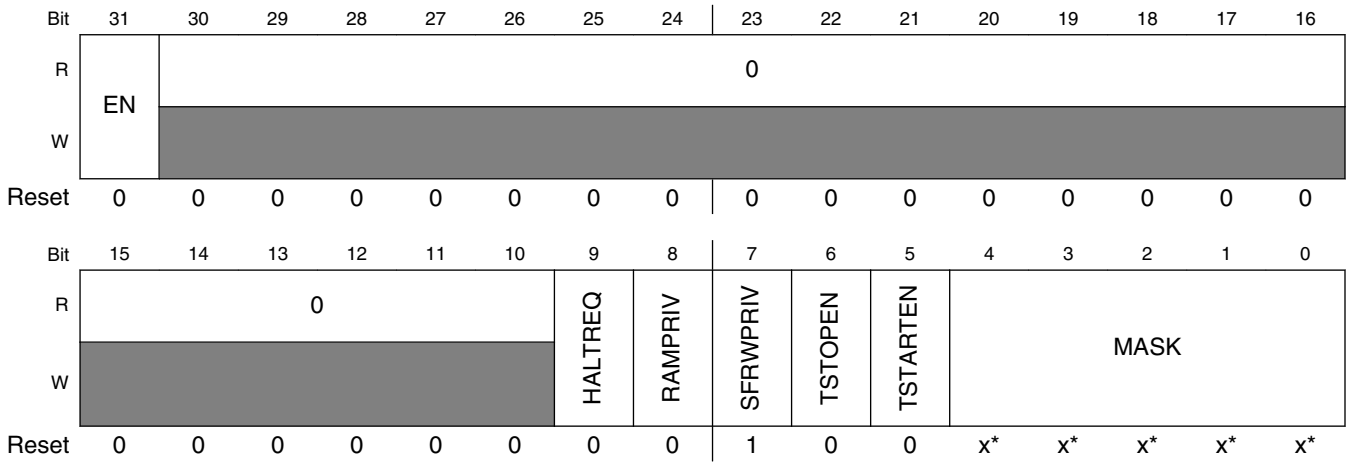
Before MTB_MASTER[EN] or MTB_MASTER[TSTARTEN] are set to 1, the software must initialize the MTB_POSITION and MTB_FLOW registers.

If MTB_FLOW[WATERMARK] is used to stop tracing or to halt the processor, MTB_MASTER[MASK] must still be set to a value that prevents MTB_POSITION[POINTER] from wrapping before it reaches the MTB_FLOW[WATERMARK] value.

NOTE

The format of this mask field is different than MTBDWT_MASKn[MASK].

Address: F000_0000h base + 4h offset = F000_0004h



* Notes:
- x = Undefined at reset.

MTBx_MASTER field descriptions

Field	Description
31 EN	Main Trace Enable When this field is 1, trace data is written into the RAM memory location addressed by MTB_POSITION[POINTER]. The MTB_POSITION[POINTER] value auto increments after the trace data packet is written. EN can be automatically set to 0 using the MTB_FLOW[WATERMARK] field and the MTB_FLOW[AUTOSTOP] bit. EN is automatically set to 1 if TSTARTEN is 1 and the TSTART signal is HIGH. EN is automatically set to 0 if TSTOPEN is 1 and the TSTOP signal is HIGH. **NOTE:** If EN is set to 0 because MTB_FLOW[WATERMARK] is set, then it is not automatically set to 1 if TSTARTEN is 1 and the TSTART input is HIGH. In this case, tracing can only be restarted if MTB_FLOW[WATERMARK] or MTB_POSITION[POINTER] value is changed by software.
30–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 HALTREQ	Halt Request This field is connected to the halt request signal of the trace logic, EDBGRQ. When HALTREQ is set to 1, the EDBFGRQ is asserted if DBGEN (invasive debug enable, one of the debug authentication interface signals) is also HIGH. HALTREQ can be automatically set to 1 using MTB_FLOW[WATERMARK].
8 RAMPRIV	RAM Privilege If this field is 0, then user or privileged AHB read and write accesses to the RAM are permitted. If this field is 1, then only privileged AHB read and write accesses to the RAM are permitted and user accesses are RAZ/WI. The HPROT[1] signal determines if an access is a user or privileged mode reference.

Table continues on the next page...

MTBx_MASTER field descriptions (continued)

Field	Description
7 SFRWPRIV	Special Function Register Write Privilege If this field is 0, then user or privileged AHB read and write accesses to the MTB_RAM Special Function Registers (programming model) are permitted. If this field is 1, then only privileged write accesses are permitted; user write accesses are ignored. The HPROT[1] signal determines if an access is user or privileged. Note MTB_RAM SFR read access are not controlled by this bit and are always permitted.
6 TSTOPEN	Trace Stop Input Enable If this field is 1 and the TSTOP signal is HIGH, then EN is set to 0. If a trace packet is being written to memory, the write is completed before tracing is stopped.
5 TSTARTEN	Trace Start Input Enable If this field is 1 and the TSTART signal is HIGH, then EN is set to 1. Tracing continues until a stop condition occurs.
MASK	Mask This value determines the maximum size of the trace buffer in RAM. It specifies the most-significant bit of the MTB_POSITION[POINTER] field that can be updated by automatic increment. If the trace tries to advance past this power of 2, the MTB_POSITION[WRAP] bit is set to 1, the MTB_POSITION[MASK+3:3] == MTB_POSITION[POINTER[MASK:0]] bits are set to 0, and the MTB_POSITION[14:MASK+3] == MTB_POSITION[POINTER[11:MASK+1]] bits remain unchanged. This field causes the trace packet information to be stored in a circular buffer of size 2^[MASK+4] bytes, that can be positioned in memory at multiples of this size. As detailed in the MTB_POSITION description, typical "upper limits" for the MTB size are RAM_Size/4 or RAM_Size/2. Values greater than the maximum have the same effect as the maximum.

35.3.1.3 MTB Flow Register (MTBx_FLOW)

The MTB_FLOW register contains the watermark address and the autostop/autohalt control bits.

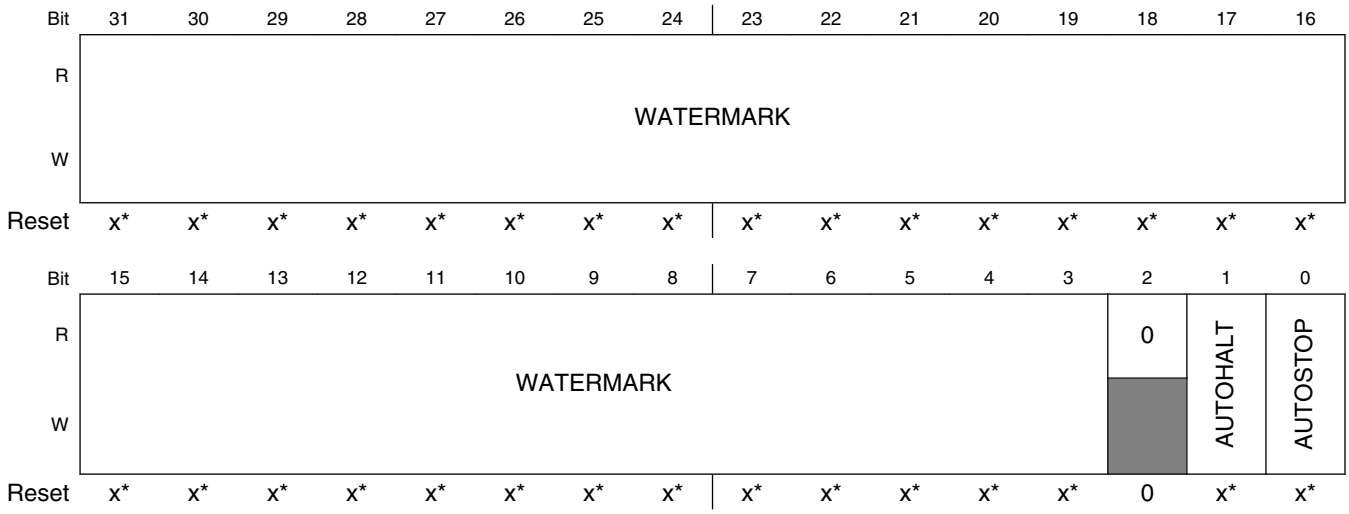
If tracing is stopped using the watermark autostop feature, it cannot be restarted until software clears the watermark autostop. This can be achieved in one of the following ways:

- Changing the MTB_POSITION[POINTER] field value to point to the beginning of the trace buffer, or
- Setting MTB_FLOW[AUTOSTOP] = 0.

A debug agent can use MTB_FLOW[AUTOSTOP] to fill the trace buffer once only without halting the processor.

A debug agent can use MTB_FLOW[AUTOHALT] to fill the trace buffer once before causing the Cortex-M0+ processor to enter the Debug state. To enter Debug state, the Cortex-M0+ processor might have to perform additional branch type operations. Therefore, the MTB_FLOW[WATERMARK] field must be set below the final entry in the trace buffer region.

Address: F000_0000h base + 8h offset = F000_0008h



* Notes:
- x = Undefined at reset.

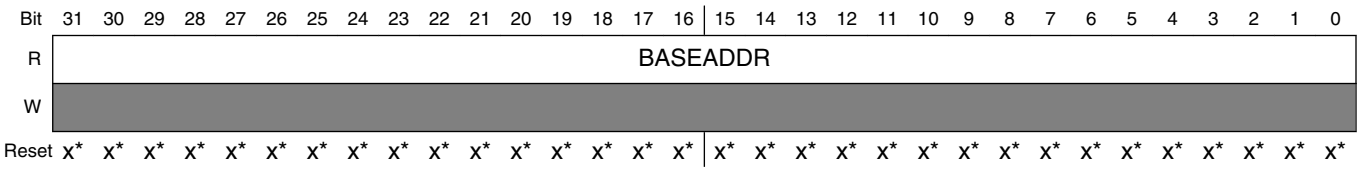
MTBx_FLOW field descriptions

Field	Description
31–3 WATERMARK	WATERMARK[28:0] This field contains an address in the same format as the MTB_POSITION[POINTER] field. When MTB_POSITION[POINTER] matches the WATERMARK field value, actions defined by the AUTOHALT and AUTOSTOP bits are performed.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 AUTOHALT	AUTOHALT If this field is 1 and WATERMARK is equal to MTB_POSITION[POINTER], then MTB_MASTER[HALTREQ] is automatically set to 1. If the DBGEN signal is HIGH, the MTB asserts this halt request to the Cortex-M0+ processor by asserting the EDBGRQ signal.
0 AUTOSTOP	AUTOSTOP If this field is 1 and WATERMARK is equal to MTB_POSITION[POINTER], then MTB_MASTER[EN] is automatically set to 0. This stops tracing.

35.3.1.4 MTB Base Register (MTBx_BASE)

The read-only MTB_BASE Register indicates where the RAM is located in the system memory map. This register is provided to enable auto discovery of the MTB RAM location by a debug agent, and is defined by a hardware design parameter.

Address: F000_0000h base + Ch offset = F000_000Ch



* Notes:
- x = Undefined at reset.

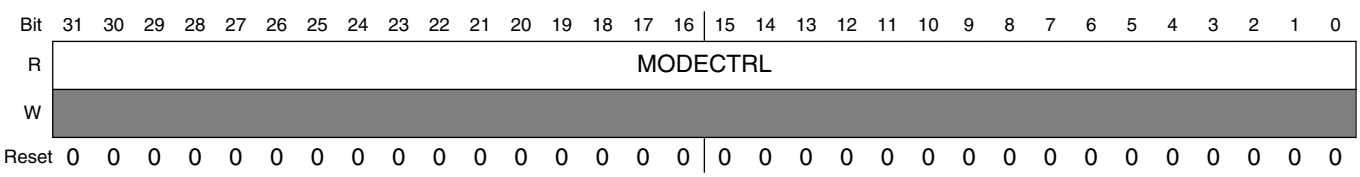
MTBx_BASE field descriptions

Field	Description
BASEADDR	BASEADDR This value is defined (hardwired): 0x1FFF_8000 (Core 0)

35.3.1.5 Integration Mode Control Register (MTBx_MODECTRL)

This register enables the device to switch from a functional mode, or default behavior, into integration mode. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + F00h offset = F000_0F00h



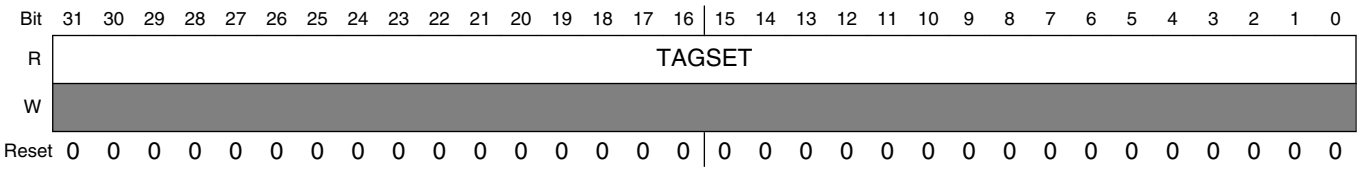
MTBx_MODECTRL field descriptions

Field	Description
MODECTRL	MODECTRL Hardwired to 0x0000_0000

35.3.1.6 Claim TAG Set Register (MTBx_TAGSET)

The Claim Tag Set Register returns the number of bits that can be set on a read, and enables individual bits to be set on a write. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FA0h offset = F000_0FA0h



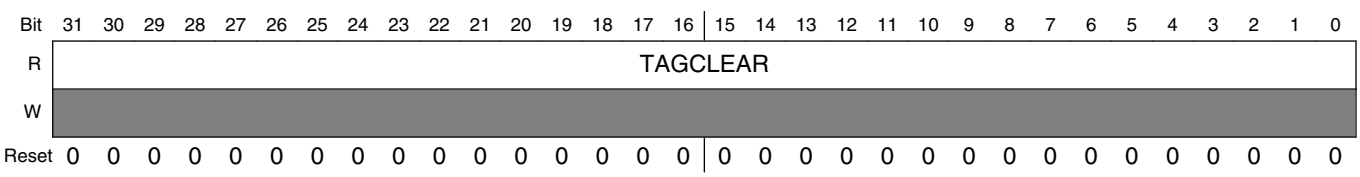
MTBx_TAGSET field descriptions

Field	Description
TAGSET	TAGSET Hardwired to 0x0000_0000

35.3.1.7 Claim TAG Clear Register (MTBx_TAGCLEAR)

The read/write Claim Tag Clear Register is used to read the claim status on debug resources. A read indicates the claim tag status. Writing 1 to a specific bit clears the corresponding claim tag to 0. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FA4h offset = F000_0FA4h



MTBx_TAGCLEAR field descriptions

Field	Description
TAGCLEAR	TAGCLEAR Hardwired to 0x0000_0000

35.3.1.8 Lock Access Register (MTBx_LOCKACCESS)

The Lock Access Register enables a write access to component registers. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FB0h offset = F000_0FB0h

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

MTBx_LOCKACCESS field descriptions

Field	Description
LOCKACCESS	Hardwired to 0x0000_0000

35.3.1.9 Lock Status Register (MTBx_LOCKSTAT)

The Lock Status Register indicates the status of the lock control mechanism. This register is used in conjunction with the Lock Access Register. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FB4h offset = F000_0FB4h

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

MTBx_LOCKSTAT field descriptions

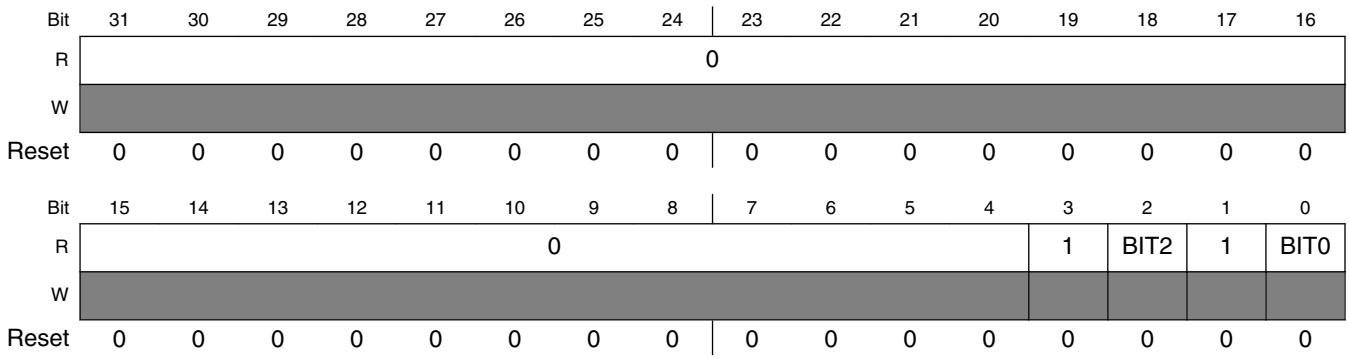
Field	Description
LOCKSTAT	LOCKSTAT Hardwired to 0x0000_0000

35.3.1.10 Authentication Status Register (MTBx_AUTHSTAT)

The Authentication Status Register reports the required security level and current status of the security enable bit pairs. Where functionality changes on a given security level, this change must be reported in this register. It is connected to specific signals used during the auto-discovery process by an external debug agent.

MTB_AUTHSTAT[3:2] indicates if nonsecure, noninvasive debug is enabled or disabled, while MTB_AUTHSTAT[1:0] indicates the enabled/disabled state of nonsecure, invasive debug. For both 2-bit fields, 0b10 indicates the functionality is disabled and 0b11 indicates it is enabled.

Address: F000_0000h base + FB8h offset = F000_0FB8h



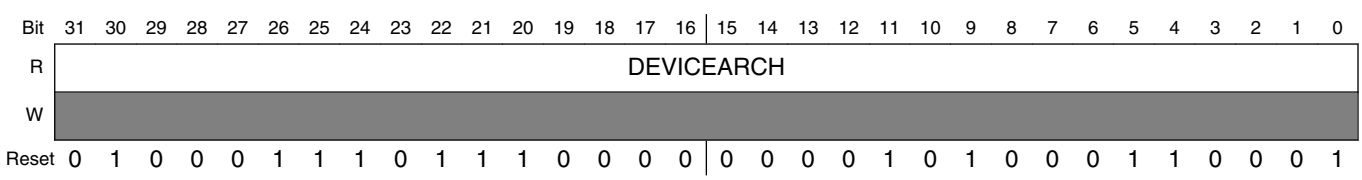
MTBx_AUTHSTAT field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	BIT3 This read-only field is reserved and always has the value 1.
2 BIT2	BIT2 Connected to NIDEN or DBGEN signal.
1 Reserved	BIT1 This read-only field is reserved and always has the value 1.
0 BIT0	Connected to DBGEN.

35.3.1.11 Device Architecture Register (MTBx_DEVICEARCH)

This register indicates the device architecture. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FBCh offset = F000_0FBCh



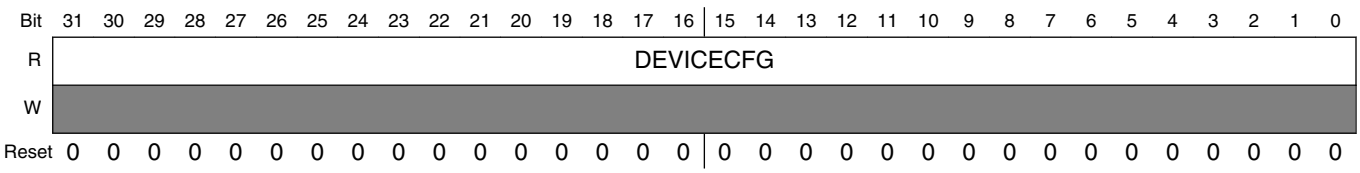
MTBx_DEVICEARCH field descriptions

Field	Description
DEVICEARCH	DEVICEARCH Hardwired to 0x4770_0A31.

35.3.1.12 Device Configuration Register (MTBx_DEVICECFG)

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FC8h offset = F000_0FC8h



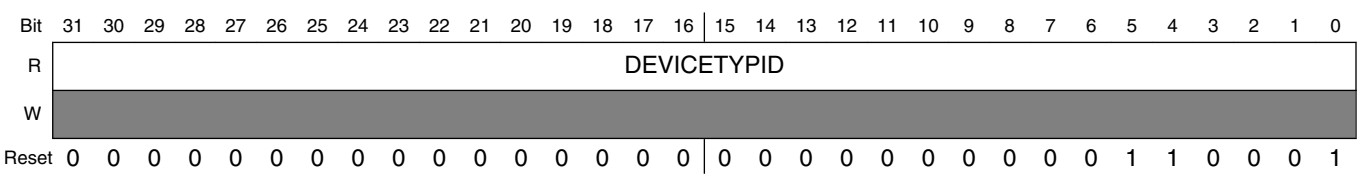
MTBx_DEVICECFG field descriptions

Field	Description
DEVICECFG	DEVICECFG Hardwired to 0x0000_0000.

35.3.1.13 Device Type Identifier Register (MTBx_DEVICETYPID)

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FCCh offset = F000_0FCCh



MTBx_DEVICETYPID field descriptions

Field	Description
DEVICETYPID	DEVICETYPID Hardwired to 0x0000_0031.

35.3.1.14 Peripheral ID Register (MTB*x*_PERIPHID*n*)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FD0h offset + (4d × i), where i=0d to 7d

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

MTB*x*_PERIPHID*n* field descriptions

Field	Description
PERIPHID	PERIPHID Peripheral ID4 is hardwired to 0x0000_0004; ID0 to 0x0000_0032; ID1 to 0x0000_00B9; ID2 to 0x0000_001B; and all the others to 0x0000_0000.

35.3.1.15 Component ID Register (MTB*x*_COMPID*n*)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_0000h base + FF0h offset + (4d × i), where i=0d to 3d

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

MTB*x*_COMPID*n* field descriptions

Field	Description
COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

35.3.2 MTB_DWT Memory Map

The MTB_DWT programming model supports a very simplified subset of the v7M debug architecture and follows the standard ARM DWT definition.

MTB_DWT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_1000	MTB DWT Control Register (MTB0_DWT_CTRL)	32	R	2F00_0000h	35.3.2.1/ 898
F000_1020	MTB_DWT Comparator Register (MTB0_DWT_COMP0)	32	R/W	0000_0000h	35.3.2.2/ 899
F000_1024	MTB_DWT Comparator Mask Register (MTB0_DWT_MASK0)	32	R/W	0000_0000h	35.3.2.3/ 899
F000_1028	MTB_DWT Comparator Function Register 0 (MTB0_DWT_FCT0)	32	R/W	0000_0000h	35.3.2.4/ 900
F000_1030	MTB_DWT Comparator Register (MTB0_DWT_COMP1)	32	R/W	0000_0000h	35.3.2.2/ 899
F000_1034	MTB_DWT Comparator Mask Register (MTB0_DWT_MASK1)	32	R/W	0000_0000h	35.3.2.3/ 899
F000_1038	MTB_DWT Comparator Function Register 1 (MTB0_DWT_FCT1)	32	R/W	0000_0000h	35.3.2.5/ 902
F000_1200	MTB_DWT Trace Buffer Control Register (MTB0_DWT_TBCTRL)	32	R/W	2000_0000h	35.3.2.6/ 903
F000_1FC8	Device Configuration Register (MTB0_DWT_DEVICECFG)	32	R	0000_0000h	35.3.2.7/ 905
F000_1FCC	Device Type Identifier Register (MTB0_DWT_DEVICETYPID)	32	R	0000_0004h	35.3.2.8/ 905
F000_1FD0	Peripheral ID Register (MTB0_DWT_PERIPHID4)	32	R	See section	35.3.2.9/ 906
F000_1FD4	Peripheral ID Register (MTB0_DWT_PERIPHID5)	32	R	See section	35.3.2.9/ 906
F000_1FD8	Peripheral ID Register (MTB0_DWT_PERIPHID6)	32	R	See section	35.3.2.9/ 906
F000_1FDC	Peripheral ID Register (MTB0_DWT_PERIPHID7)	32	R	See section	35.3.2.9/ 906
F000_1FE0	Peripheral ID Register (MTB0_DWT_PERIPHID0)	32	R	See section	35.3.2.9/ 906
F000_1FE4	Peripheral ID Register (MTB0_DWT_PERIPHID1)	32	R	See section	35.3.2.9/ 906
F000_1FE8	Peripheral ID Register (MTB0_DWT_PERIPHID2)	32	R	See section	35.3.2.9/ 906
F000_1FEC	Peripheral ID Register (MTB0_DWT_PERIPHID3)	32	R	See section	35.3.2.9/ 906
F000_1FF0	Component ID Register (MTB0_DWT_COMPID0)	32	R	See section	35.3.2.10/ 906
F000_1FF4	Component ID Register (MTB0_DWT_COMPID1)	32	R	See section	35.3.2.10/ 906
F000_1FF8	Component ID Register (MTB0_DWT_COMPID2)	32	R	See section	35.3.2.10/ 906
F000_1FFC	Component ID Register (MTB0_DWT_COMPID3)	32	R	See section	35.3.2.10/ 906

35.3.2.1 MTB DWT Control Register (MTBx0_DWT_CTRL)

The MTBDWT_CTRL register provides read-only information on the watchpoint configuration for the MTB_DWT.

Address: F000_1000h base + 0h offset = F000_1000h

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

MTBx0_DWT_CTRL field descriptions

Field	Description
31–28 NUMCMP	Number of comparators The MTB_DWT implements two comparators.
DWTCFGCTRL	DWT configuration controls This field is hardwired to 0xF00_0000, disabling all the remaining DWT functionality. The specific fields and their state are: MTBDWT_CTRL[27] = NOTRCPKT = 1, trace sample and exception trace is not supported MTBDWT_CTRL[26] = NOEXTTRIG = 1, external match signals are not supported MTBDWT_CTRL[25] = NOCYCCNT = 1, cycle counter is not supported MTBDWT_CTRL[24] = NOPRFCNT = 1, profiling counters are not supported MTBDWT_CTRL[22] = CYCEBTENA = 0, no POSTCNT underflow packets generated MTBDWT_CTRL[21] = FOLDEVTENA = 0, no folded instruction counter overflow events MTBDWT_CTRL[20] = LSUEVTENA = 0, no LSU counter overflow events MTBDWT_CTRL[19] = SLEEPEVTENA = 0, no sleep counter overflow events MTBDWT_CTRL[18] = EXCEVTENA = 0, no exception overhead counter events MTBDWT_CTRL[17] = CPIEVTENA = 0, no CPI counter overflow events MTBDWT_CTRL[16] = EXCTRCENA = 0, generation of exception trace disabled MTBDWT_CTRL[12] = PCSAMPLENA = 0, no periodic PC sample packets generated MTBDWT_CTRL[11:10] = SYNCTAP = 0, no synchronization packets MTBDWT_CTRL[9] = CYCTAP = 0, cycle counter is not supported MTBDWT_CTRL[8:5] = POSTINIT = 0, cycle counter is not supported MTBDWT_CTRL[4:1] = POSTPRESET = 0, cycle counter is not supported MTBDWT_CTRL[0] = CYCCNTENA = 0, cycle counter is not supported

35.3.2.2 MTB_DWT Comparator Register (MTBx0_DWT_COMPn)

The MTBDWT_COMPn registers provide the reference value for comparator n.

Address: F000_1000h base + 20h offset + (16d × i), where i=0d to 1d

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

MTBx0_DWT_COMPn field descriptions

Field	Description
COMP	Reference value for comparison If MTBDWT_COMP0 is used for a data value comparator and the access size is byte or halfword, the data value must be replicated across all appropriate byte lanes of this register. For example, if the data is a byte-sized "x" value, then COMP[31:24] = COMP[23:16] = COMP[15:8] = COMP[7:0] = "x". Likewise, if the data is a halfword-size "y" value, then COMP[31:16] = COMP[15:0] = "y".

35.3.2.3 MTB_DWT Comparator Mask Register (MTBx0_DWT_MASKn)

The MTBDWT_MASKn registers define the size of the ignore mask applied to the reference address for address range matching by comparator n. Note the format of this mask field is different than the MTB_MASTER[MASK].

Address: F000_1000h base + 24h offset + (16d × i), where i=0d to 1d

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

MTBx0_DWT_MASKn field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MASK	MASK The value of the ignore mask, 0-31 bits, is applied to address range matching. MASK = 0 is used to include all bits of the address in the comparison, except if MASK = 0 and the comparator is configured to watch instruction fetch addresses, address bit [0] is ignored by the hardware since all fetches must be at least halfword aligned. For MASK != 0 and regardless of watch type, address bits [x-1:0] are ignored in the address comparison.

Table continues on the next page...

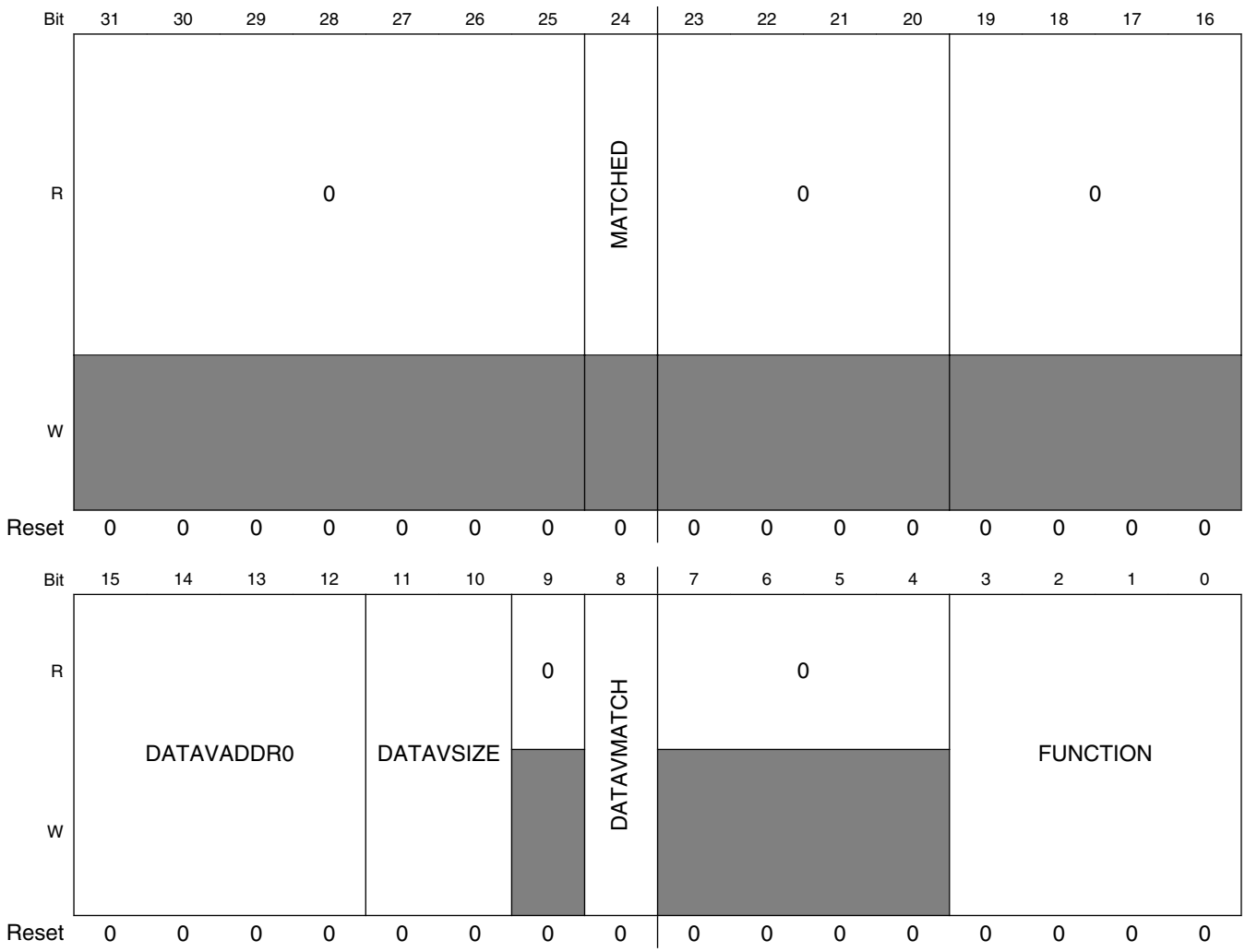
MTB*x*0_DWT_MASK*n* field descriptions (continued)

Field	Description
	Using a mask means the comparator matches on a range of addresses, defined by the unmasked most significant bits of the address, bits [31:x]. The maximum MASK value is 24, producing a 16 Mbyte mask. An attempted write of a MASK value > 24 is limited by the MTBDWT hardware to 24. If MTBDWT_COMP0 is used as a data value comparator, then MTBDWT_MASK0 should be programmed to zero.

35.3.2.4 MTB_DWT Comparator Function Register 0 (MTB*x*0_DWT_FCT0)

The MTBDWT_FCTn registers control the operation of comparator n.

Address: F000_1000h base + 28h offset = F000_1028h



MTBx0_DWT_FCT0 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 MATCHED	Comparator match If this read-only flag is asserted, it indicates the operation defined by the FUNCTION field occurred since the last read of the register. Reading the register clears this bit. 0 No match. 1 Match occurred.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 DATAVADDR0	Data Value Address 0 Since the MTB_DWT implements two comparators, the DATAVADDR0 field is restricted to values {0,1}. When the DATAVMATCH bit is asserted, this field defines the comparator number to use for linked address comparison. If MTBDWT_COMP0 is used as a data watchpoint and MTBDWT_COMP1 as an address watchpoint, DATAVADDR0 must be set.
11–10 DATAVSIZE	Data Value Size For data value matching, this field defines the size of the required data comparison. 00 Byte. 01 Halfword. 10 Word. 11 Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 DATAVMATCH	Data Value Match When this field is 1, it enables data value comparison. For this implementation, MTBDWT_COMP0 supports address or data value comparisons; MTBDWT_COMP1 only supports address comparisons. 0 Perform address comparison. 1 Perform data value comparison.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FUNCTION	Function Selects the action taken on a comparator match. If MTBDWT_COMP0 is used for a data value and MTBDWT_COMP1 for an address value, then MTBDWT_FCT1[FUNCTION] must be set to zero. For this configuration, MTBDWT_MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses. 0000 Disabled. 0100 Instruction fetch. 0101 Data operand read. 0110 Data operand write. 0111 Data operand (read + write). others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

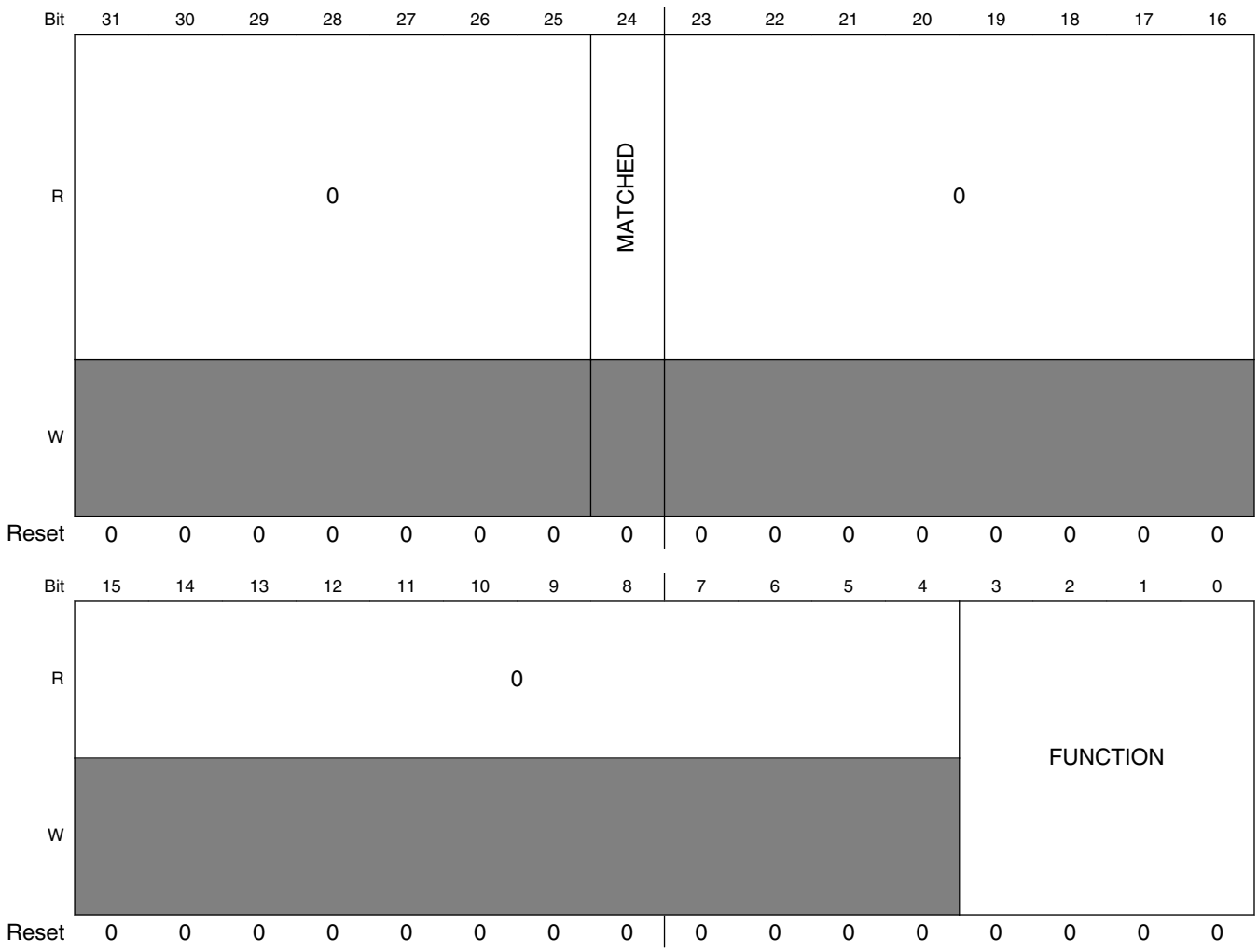
MTBx0_DWT_FCT0 field descriptions (continued)

Field	Description

35.3.2.5 MTB_DWT Comparator Function Register 1 (MTBx0_DWT_FCT1)

The MTBDWT_FCTn registers control the operation of comparator n. Since the MTB_DWT only supports data value comparisons on comparator 0, there are several fields in the MTBDWT_FCT1 register that are RAZ/WI (bits 12, 11:10, 8).

Address: F000_1000h base + 38h offset = F000_1038h



MTBx0_DWT_FCT1 field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

MTBx0_DWT_FCT1 field descriptions (continued)

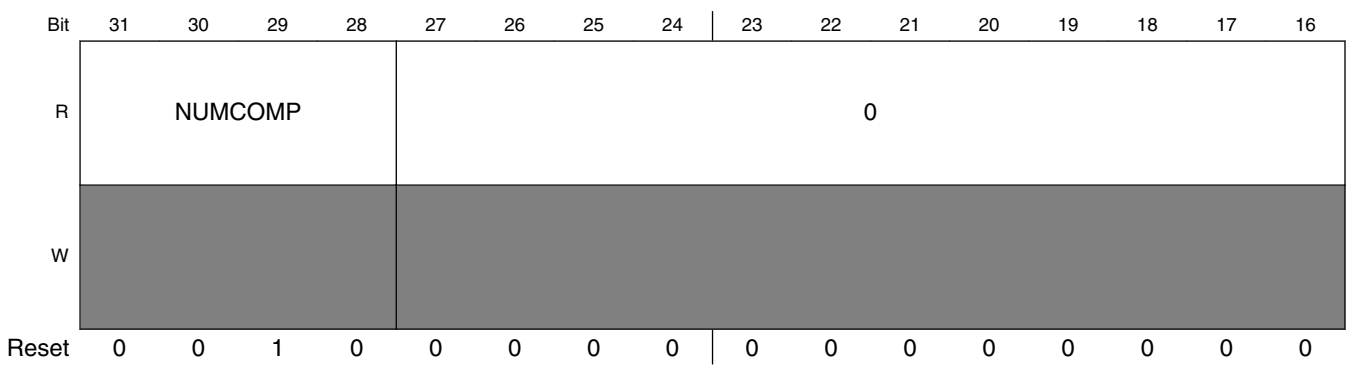
Field	Description
24 MATCHED	Comparator match If this read-only flag is asserted, it indicates the operation defined by the FUNCTION field occurred since the last read of the register. Reading the register clears this bit. 0 No match. 1 Match occurred.
23–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FUNCTION	Function Selects the action taken on a comparator match. If MTBDWT_COMP0 is used for a data value and MTBDWT_COMP1 for an address value, then MTBDWT_FCT1[FUNCTION] must be set to zero. For this configuration, MTBDWT_MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses. 0000 Disabled. 0100 Instruction fetch. 0101 Data operand read. 0110 Data operand write. 0111 Data operand (read + write). others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

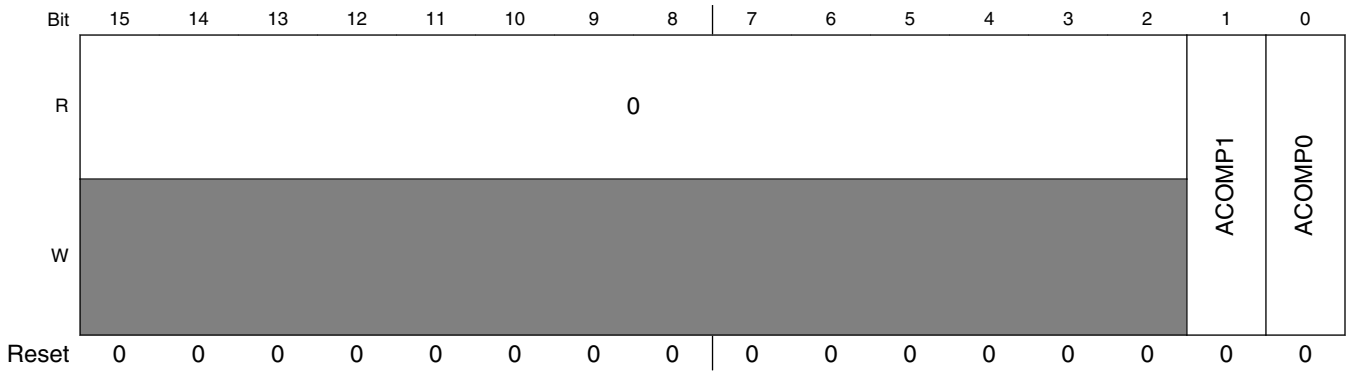
35.3.2.6 MTB_DWT Trace Buffer Control Register (MTBx0_DWT_TBCTRL)

The MTBDWT_TBCTRL register defines how the watchpoint comparisons control the actual trace buffer operation.

Recall the MTB supports starting and stopping the program trace based on the watchpoint comparisons signaled via TSTART and TSTOP. The watchpoint comparison signals are enabled in the MTB's control logic by setting the appropriate enable bits, MTB_MASTER[TSTARTEN, TSTOPEN]. In the event of simultaneous assertion of both TSTART and TSTOP, TSTART takes priority.

Address: F000_1000h base + 200h offset = F000_1200h





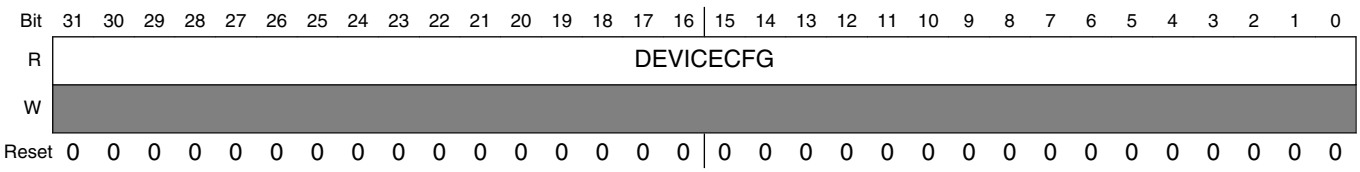
MTBx0_DWT_TBCTRL field descriptions

Field	Description
31–28 NUMCOMP	Number of Comparators This read-only field specifies the number of comparators in the MTB_DWT. This implementation includes two registers.
27–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 ACOMP1	Action based on Comparator 1 match When the MTBDWT_FCT1[MATCHED] is set, it indicates MTBDWT_COMP1 address compare has triggered and the trace buffer's recording state is changed. 0 Trigger TSTOP based on the assertion of MTBDWT_FCT1[MATCHED]. 1 Trigger TSTART based on the assertion of MTBDWT_FCT1[MATCHED].
0 ACOMP0	Action based on Comparator 0 match When the MTBDWT_FCT0[MATCHED] is set, it indicates MTBDWT_COMP0 address compare has triggered and the trace buffer's recording state is changed. The assertion of MTBDWT_FCT0[MATCHED] is caused by the following conditions: • Address match in MTBDWT_COMP0 when MTBDWT_FCT0[DATAVMATCH] = 0 • Data match in MTBDWT_COMP0 when MTBDWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,0} • Data match in MTBDWT_COMP0 and address match in MTBDWT_COMP1 when MTBDWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,1} 0 Trigger TSTOP based on the assertion of MTBDWT_FCT0[MATCHED]. 1 Trigger TSTART based on the assertion of MTBDWT_FCT0[MATCHED].

35.3.2.7 Device Configuration Register (MTBx0_DWT_DEVICECFG)

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FC8h offset = F000_1FC8h



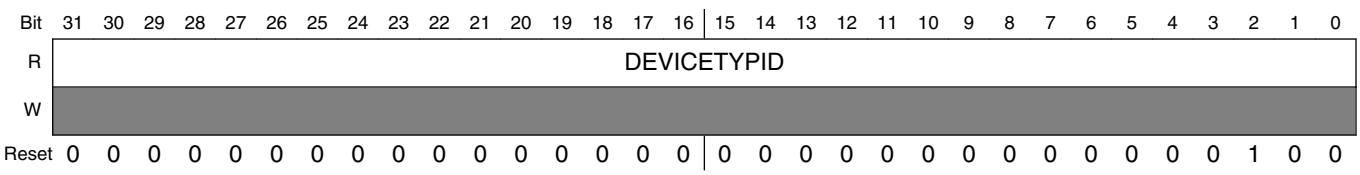
MTBx0_DWT_DEVICECFG field descriptions

Field	Description
DEVICECFG	DEVICECFG Hardwired to 0x0000_0000.

35.3.2.8 Device Type Identifier Register (MTBx0_DWT_DEVICETYPID)

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FCCh offset = F000_1FCCh



MTBx0_DWT_DEVICETYPID field descriptions

Field	Description
DEVICETYPID	DEVICETYPID Hardwired to 0x0000_0004.

35.3.2.9 Peripheral ID Register (MTBx0_DWT_PERIPHIDn)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FD0h offset + (4d × i), where i=0d to 7d

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

MTBx0_DWT_PERIPHIDn field descriptions

Field	Description
PERIPHID	PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000.

35.3.2.10 Component ID Register (MTBx0_DWT_COMPIDn)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_1000h base + FF0h offset + (4d × i), where i=0d to 3d

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

MTBx0_DWT_COMPIDn field descriptions

Field	Description
COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

35.3.3 System ROM Memory Map

The System ROM Table registers are also mapped into a sparsely-populated 4 KB address space.

For core configurations like that supported by Cortex-M0+, ARM recommends that a debugger identifies and connects to the debug components using the CoreSight debug infrastructure.

ARM recommends that a debugger follows the flow as shown in the following figure to discover the components in the CoreSight debug infrastructure. In this case, a debugger reads the peripheral and component ID registers for each CoreSight component in the CoreSight system.

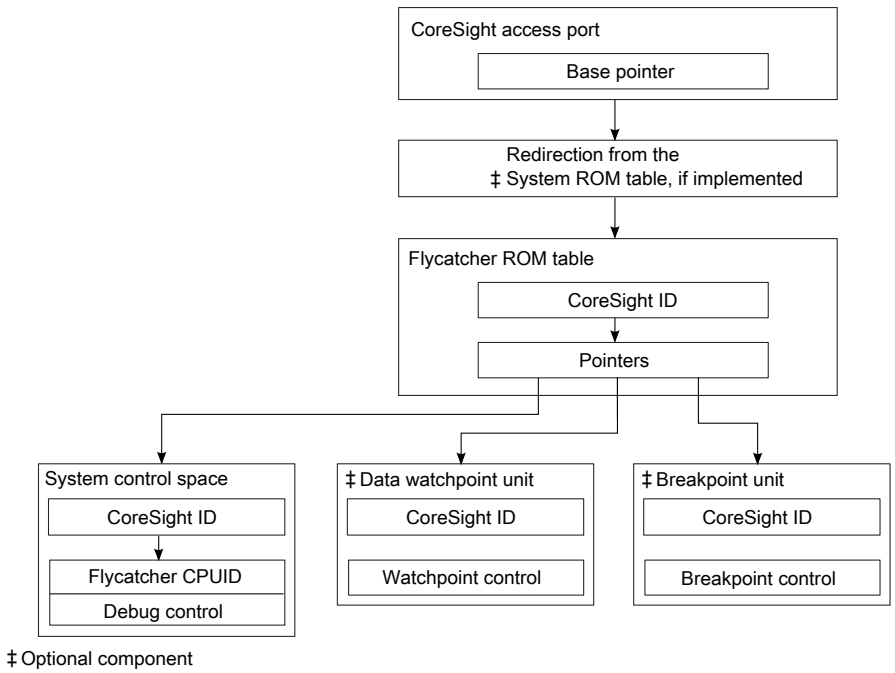


Figure 35-3. CoreSight discovery process

MTB_ROM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_2000	Entry (MTB0_ROM_ENTRY0)	32	R	See section	35.3.3.1/ 908
F000_2004	Entry (MTB0_ROM_ENTRY1)	32	R	See section	35.3.3.1/ 908
F000_2008	Entry (MTB0_ROM_ENTRY2)	32	R	See section	35.3.3.1/ 908
F000_200C	Entry (MTB0_ROM_ENTRY3)	32	R	See section	35.3.3.1/ 908
F000_2010	End of Table Marker Register (MTB0_ROM_TABLEMARK)	32	R	0000_0000h	35.3.3.2/ 909
F000_2FCC	System Access Register (MTB0_ROM_SYSACCESS)	32	R	0000_0001h	35.3.3.3/ 909

Table continues on the next page...

MTB_ROM memory map (continued)

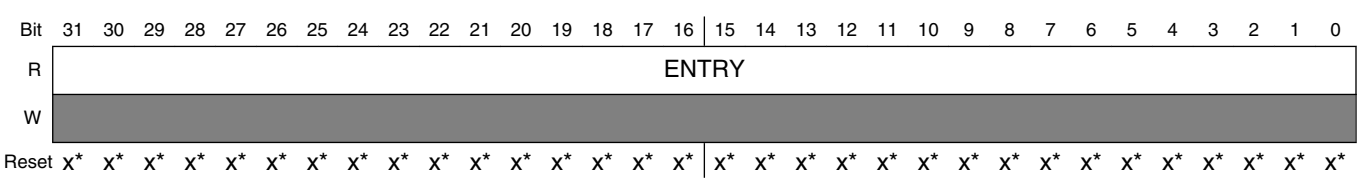
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F000_2FD0	Peripheral ID Register (MTB0_ROM_PERIPHID4)	32	R	See section	35.3.3.4/ 910
F000_2FD4	Peripheral ID Register (MTB0_ROM_PERIPHID5)	32	R	See section	35.3.3.4/ 910
F000_2FD8	Peripheral ID Register (MTB0_ROM_PERIPHID6)	32	R	See section	35.3.3.4/ 910
F000_2FDC	Peripheral ID Register (MTB0_ROM_PERIPHID7)	32	R	See section	35.3.3.4/ 910
F000_2FE0	Peripheral ID Register (MTB0_ROM_PERIPHID0)	32	R	See section	35.3.3.4/ 910
F000_2FE4	Peripheral ID Register (MTB0_ROM_PERIPHID1)	32	R	See section	35.3.3.4/ 910
F000_2FE8	Peripheral ID Register (MTB0_ROM_PERIPHID2)	32	R	See section	35.3.3.4/ 910
F000_2FEC	Peripheral ID Register (MTB0_ROM_PERIPHID3)	32	R	See section	35.3.3.4/ 910
F000_2FF0	Component ID Register (MTB0_ROM_COMPID0)	32	R	See section	35.3.3.5/ 910
F000_2FF4	Component ID Register (MTB0_ROM_COMPID1)	32	R	See section	35.3.3.5/ 910
F000_2FF8	Component ID Register (MTB0_ROM_COMPID2)	32	R	See section	35.3.3.5/ 910
F000_2FFC	Component ID Register (MTB0_ROM_COMPID3)	32	R	See section	35.3.3.5/ 910

35.3.3.1 Entry (MTBx0_ROM_ENTRY*n*)

The System ROM Table begins with "n" relative 32-bit addresses, one for each debug component present in the device and terminating with an all-zero value signaling the end of the table at the "n+1"-th value.

It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + 0h offset + (4d × i), where i=0d to 3d



MTBx0_ROM_ENTRY*n* field descriptions

Field	Description
ENTRY	ENTRY Entry*n* values are hardwired: Entry 0 (MTB) is hardwired to 0xFFFF_E003 Entry 1 (MTBDWT) to 0xFFFF_F003 Entry 2 (CM0PCTI) to 0x0000_4003 Entry 3 (CM0+ ROM Table) to 0xF00F_D003

35.3.3.2 End of Table Marker Register (MTBx0_ROM_TABLEMARK)

This register indicates end of table marker. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + 10h offset = F000_2010h

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

MTBx0_ROM_TABLEMARK field descriptions

Field	Description
MARK	MARK Hardwired to 0x0000_0000

35.3.3.3 System Access Register (MTBx0_ROM_SYSACCESS)

This register indicates system access. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + FCCh offset = F000_2FCCh

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

MTBx0_ROM_SYSACCESS field descriptions

Field	Description
SYSACCESS	SYSACCESS

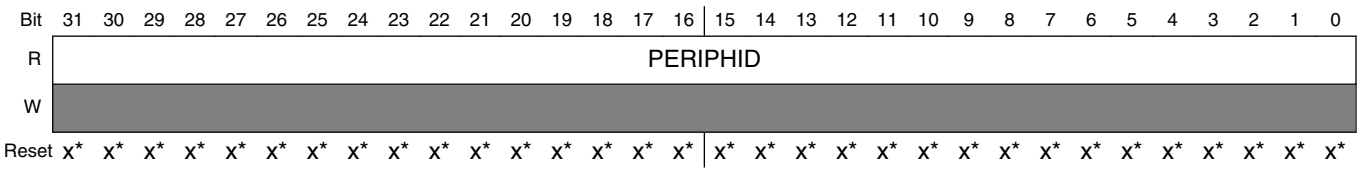
MTB*x*0_ROM_SYSACCESS field descriptions (continued)

Field	Description
	Hardwired to 0x0000_0001

35.3.3.4 Peripheral ID Register (MTB*x*0_ROM_PERIPHID*n*)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + FD0h offset + (4d × i), where i=0d to 7d



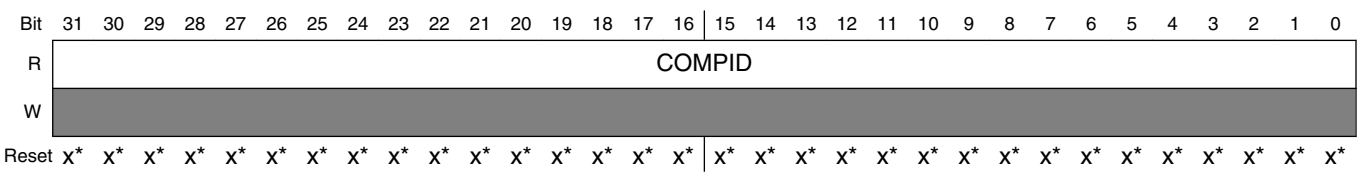
MTB*x*0_ROM_PERIPHID*n* field descriptions

Field	Description
PERIPHID	PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000.

35.3.3.5 Component ID Register (MTB*x*0_ROM_COMPID*n*)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000_2000h base + FF0h offset + (4d × i), where i=0d to 3d



MTB*x*0_ROM_COMPID*n* field descriptions

Field	Description
COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0010; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

Chapter 36
Peripheral Clock Control (PCC)

36.1 Introduction

The Peripheral Clock Control module (PCC) provides peripheral clock control and configuration registers.

In addition to the peripheral clock gates, clock multiplexors, and clock dividers, the PCC contains an interface between the peripherals and the system to control and acknowledge the Stop, Doze, and Debug signals.

36.1.1 Features

The PCC module enables software to configure the following clocking options for each peripheral:

- Clock gating
- Clock source selection
- Clock divide values

The following figure is a block diagram of the PCC clock source selection and clock gating. Some peripherals also have a clock divider available. See the peripheral's PCC register for more information.

PCC - Peripheral Clock Muxing and Gating

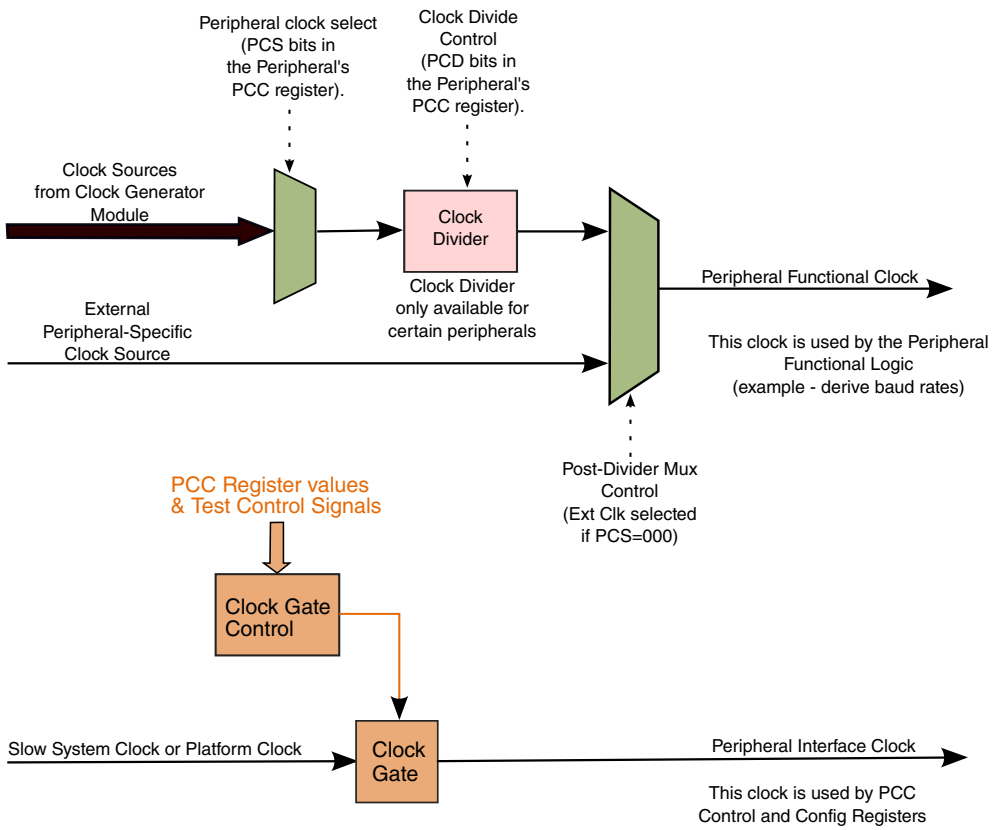


Figure 36-1. PCC Clock Source Selection and Gating

36.2 Memory map and register definition

Each peripheral has its own dedicated PCC Register, which controls the clock gating, clock source and dividers for that specific peripheral.

PCC registers can only be written in supervisor mode.

NOTE

Write accesses to unused PCC registers are blocked and result in a bus error.

36.2.1 PCC Register Descriptions

These register may not be applicable to all instances of PCC. For more details on the registers supported on each module instance, please refer to "The PCC as implemented on the chip."

Table 36-1. PCC Memory Map

Offset	Register	Width (In bits)	Access	Reset value
4007A020h	PCC DMA0 (PCC_DMA0)	32	RW	80000000h
4007A080h	PCC FLASH (PCC_FLASH)	32	RW	C0000000h
4007A084h	PCC DMAMUX0 (PCC_DMAMUX0)	32	RW	80000000h
4007A090h	PCC INTMUX0 (PCC_INTMUX0)	32	RW	80000000h
4007A0B8h	PCC TPM2 (PCC_TPM2)	32	RW	80000000h
4007A0C0h	PCC LPIT0 (PCC_LPIT0)	32	RW	80000000h
4007A0D0h	PCC LPTMR0 (PCC_LPTMR0)	32	RW	80000000h
4007A0E0h	PCC RTC (PCC_RTC)	32	RW	80000000h
4007A0F8h	PCC LPSPI2 (PCC_LPSPI2)	32	RW	80000000h
4007A108h	PCC LPI2C2 (PCC_LPI2C2)	32	RW	80000000h
4007A118h	PCC LPUART2 (PCC_LPUART2)	32	RW	80000000h
4007A130h	PCC SAI0 (PCC_SAI0)	32	RW	80000000h
4007A138h	PCC EMVSIM0 (PCC_EMVSIM0)	32	RW	80000000h
4007A154h	PCC USB0FS (PCC_USB0FS)	32	RW	80000000h
4007A168h	PCC PORTA (PCC_PORTA)	32	RW	80000000h
4007A16Ch	PCC PORTB (PCC_PORTB)	32	RW	80000000h
4007A170h	PCC PORTC (PCC_PORTC)	32	RW	80000000h
4007A174h	PCC PORTD (PCC_PORTD)	32	RW	80000000h
4007A178h	PCC PORTE (PCC_PORTE)	32	RW	80000000h
4007A188h	PCC TSI0 (PCC_TSI0)	32	RW	80000000h
4007A198h	PCC ADC0 (PCC_ADC0)	32	RW	80000000h
4007A1A8h	PCC DAC0 (PCC_DAC0)	32	RW	80000000h
4007A1B8h	PCC CMP0 (PCC_CMP0)	32	RW	80000000h
4007A1C8h	PCC VREF (PCC_VREF)	32	RW	80000000h
4007A1E0h	PCC CRC (PCC_CRC)	32	RW	80000000h

36.2.1.1 PCC Register Descriptions

36.2.1.2 PCC DMA0 (PCC_DMA0)

36.2.1.2.1 Address

Register	Offset
PCC_DMA0	4007A020h

36.2.1.2.2 Function

PCC Register

36.2.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.2.4 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.3 PCC FLASH (PCC_FLASH)

36.2.1.3.1 Address

Register	Offset
PCC_FLASH	4007A080h

PCC Register

36.2.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.3.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled

Table continues on the next page...

Field	Function
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.4 PCC DMAMUX0 (PCC_DMAMUX0)

36.2.1.4.1 Address

Register	Offset
PCC_DMAMUX0	4007A084h

PCC Register

36.2.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.4.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.5 PCC INTMUX0 (PCC_INTMUX0)

36.2.1.5.1 Address

Register	Offset
PCC_INTMUX0	4007A090h

PCC Register

36.2.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.5.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.6 PCC TPM2 (PCC_TPM2)

36.2.1.6.1 Address

Register	Offset
PCC_TPM2	4007A0B8h

PCC Register

36.2.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.6.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock.

Table continues on the next page...

Field	Function
	011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.7 PCC LPIT0 (PCC_LPIT0)

36.2.1.7.1 Address

Register	Offset
PCC_LPIT0	4007A0C0h

PCC Register

36.2.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.7.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.8 PCC LPTMR0 (PCC_LPTMR0)

36.2.1.8.1 Address

Register	Offset
PCC_LPTMR0	4007A0D0h

PCC Register

36.2.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.8.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.9 PCC RTC (PCC_RTC)

36.2.1.9.1 Address

Register	Offset
PCC_RTC	4007A0E0h

PCC Register

36.2.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.9.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.10 PCC LPSPI2 (PCC_LPSPI2)

36.2.1.10.1 Address

Register	Offset
PCC_LPSPI2	4007A0F8h

PCC Register

36.2.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.10.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled

Table continues on the next page...

Field	Function
	1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.11 PCC LPI2C2 (PCC_LPI2C2)

36.2.1.11.1 Address

Register	Offset
PCC_LPI2C2	4007A108h

PCC Register

36.2.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.11.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.12 PCC LPUART2 (PCC_LPUART2)

36.2.1.12.1 Address

Register	Offset
PCC_LPUART2	4007A118h

PCC Register

36.2.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.12.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29	This read-only bit indicates the peripheral is being used.

Table continues on the next page...

Field	Function
INUSE	0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.13 PCC SAI0 (PCC_SAI0)

36.2.1.13.1 Address

Register	Offset
PCC_SAI0	4007A130h

PCC Register

36.2.1.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.13.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled) An external clock can be enabled for this peripheral. 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.14 PCC EMVSIM0 (PCC_EMVSIM0)

36.2.1.14.1 Address

Register	Offset
PCC_EMVSIM0	4007A138h

PCC Register

36.2.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.14.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29	This read-only bit indicates the peripheral is being used.

Table continues on the next page...

Field	Function
INUSE	0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.15 PCC USB0FS (PCC_USB0FS)

36.2.1.15.1 Address

Register	Offset
PCC_USB0FS	4007A154h

PCC Register

36.2.1.15.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												FRAC	PCD		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.15.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled) An external clock can be enabled for this peripheral. 001 - OSCCLK - System Oscillator Platform Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 FRAC	This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(DIV+1)]. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 0 - Fractional value is 0. 1 - Fractional value is 1.
2-0 PCD	This read/write bit field is used for peripherals that require a clock divider. At SOC integration, each peripheral is assigned either a divider or not. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Divide by 1 (pass-through, no clock divide). 001 - Divide by 2. 010 - Divide by 3. 011 - Divide by 4. 100 - Divide by 5. 101 - Divide by 6. 110 - Divide by 7. 111 - Divide by 8.

36.2.1.16 PCC PORTA (PCC_PORTA)

36.2.1.16.1 Address

Register	Offset
PCC_PORTA	4007A168h

PCC Register

36.2.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.16.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.17 PCC PORTB (PCC_PORTB)

36.2.1.17.1 Address

Register	Offset
PCC_PORTB	4007A16Ch

PCC Register

36.2.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.17.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.18 PCC PORTC (PCC_PORTC)

36.2.1.18.1 Address

Register	Offset
PCC_PORTC	4007A170h

PCC Register

36.2.1.18.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.18.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used.

Table continues on the next page...

Field	Function
	1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

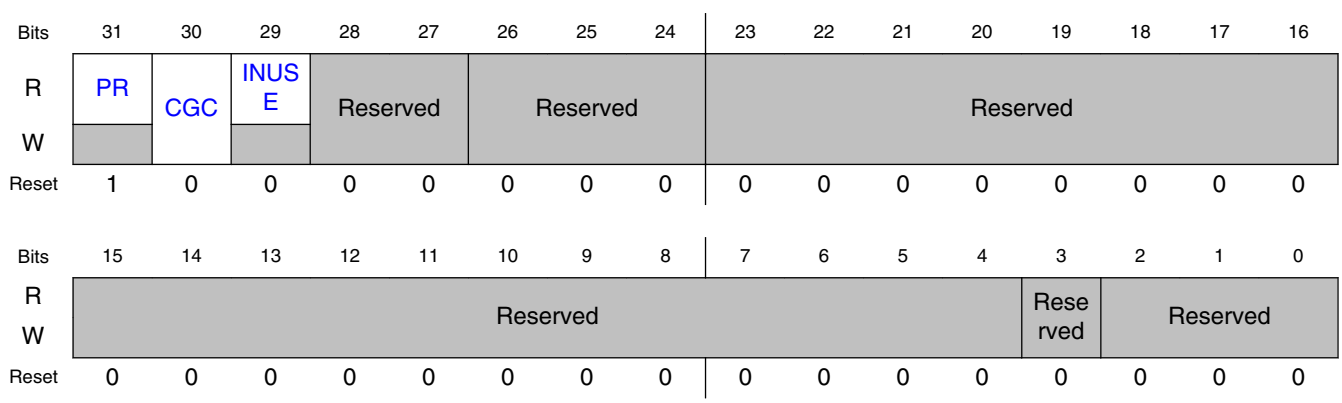
36.2.1.19 PCC PORTD (PCC_PORTD)

36.2.1.19.1 Address

Register	Offset
PCC_PORTD	4007A174h

PCC Register

36.2.1.19.2 Diagram



36.2.1.19.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.20 PCC PORTE (PCC_PORTE)

36.2.1.20.1 Address

Register	Offset
PCC_PORTE	4007A178h

PCC Register

36.2.1.20.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.20.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.21 PCC TSI0 (PCC_TSI0)

36.2.1.21.1 Address

Register	Offset
PCC_TSI0	4007A188h

PCC Register

36.2.1.21.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.21.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.22 PCC ADC0 (PCC_ADC0)

36.2.1.22.1 Address

Register	Offset
PCC_ADC0	4007A198h

PCC Register

36.2.1.22.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.22.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used.

Table continues on the next page...

Field	Function
	1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.23 PCC DAC0 (PCC_DAC0)

36.2.1.23.1 Address

Register	Offset
PCC_DAC0	4007A1A8h

PCC Register

36.2.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.23.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.24 PCC CMP0 (PCC_CMP0)

36.2.1.24.1 Address

Register	Offset
PCC_CMP0	4007A1B8h

PCC Register

36.2.1.24.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.24.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.25 PCC VREF (PCC_VREF)

36.2.1.25.1 Address

Register	Offset
PCC_VREF	4007A1C8h

PCC Register

36.2.1.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.25.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used.

Table continues on the next page...

Field	Function
	1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.1.26 PCC CRC (PCC_CRC)

36.2.1.26.1 Address

Register	Offset
PCC_CRC	4007A1E0h

PCC Register

36.2.1.26.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.1.26.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - Peripheral is not present. 1 - Peripheral is present.
30 CGC	This read/write bit enables the clock for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2 PCC Register Descriptions

These register may not be applicable to all instances of PCC. For more details on the registers supported on each module instance, please refer to "The PCC as implemented on the chip."

Table 36-2. PCC Memory Map

Offset	Register	Width (In bits)	Access	Reset value
400FA094h	PCC TRNG (PCC_TRNG)	32	RW	80000000h
400FA0B0h	PCC TPM0 (PCC_TPM0)	32	RW	80000000h
400FA0B4h	PCC TPM1 (PCC_TPM1)	32	RW	80000000h
400FA0D4h	PCC LPTMR1 (PCC_LPTMR1)	32	RW	80000000h
400FA0F0h	PCC LPSPI0 (PCC_LPSPI0)	32	RW	80000000h
400FA0F4h	PCC LPSPI1 (PCC_LPSPI1)	32	RW	80000000h
400FA100h	PCC LPI2C0 (PCC_LPI2C0)	32	RW	80000000h

Table continues on the next page...

Table 36-2. PCC Memory Map (continued)

Offset	Register	Width (In bits)	Access	Reset value
400FA104h	PCC LPI2C1 (PCC_LPI2C1)	32	RW	80000000h
400FA110h	PCC LPUART0 (PCC_LPUART0)	32	RW	80000000h
400FA114h	PCC LPUART1 (PCC_LPUART1)	32	RW	80000000h
400FA128h	PCC FLEXIO0 (PCC_FLEXIO0)	32	RW	80000000h
400FA1BCh	PCC CMP1 (PCC_CMP1)	32	RW	80000000h

36.2.2.1 PCC Register Descriptions

36.2.2.2 PCC TRNG (PCC_TRNG)

36.2.2.2.1 Address

Register	Offset
PCC_TRNG	400FA094h

36.2.2.2.2 Function

PCC Register

36.2.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.2.4 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.3 PCC TPM0 (PCC_TPM0)

36.2.2.3.1 Address

Register	Offset
PCC_TPM0	400FA0B0h

PCC Register

36.2.2.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.3.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.4 PCC TPM1 (PCC_TPM1)

36.2.2.4.1 Address

Register	Offset
PCC_TPM1	400FA0B4h

PCC Register

36.2.2.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.4.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled

Table continues on the next page...

Field	Function
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.5 PCC LPTMR1 (PCC_LPTMR1)

36.2.2.5.1 Address

Register	Offset
PCC_LPTMR1	400FA0D4h

PCC Register

36.2.2.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.5.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

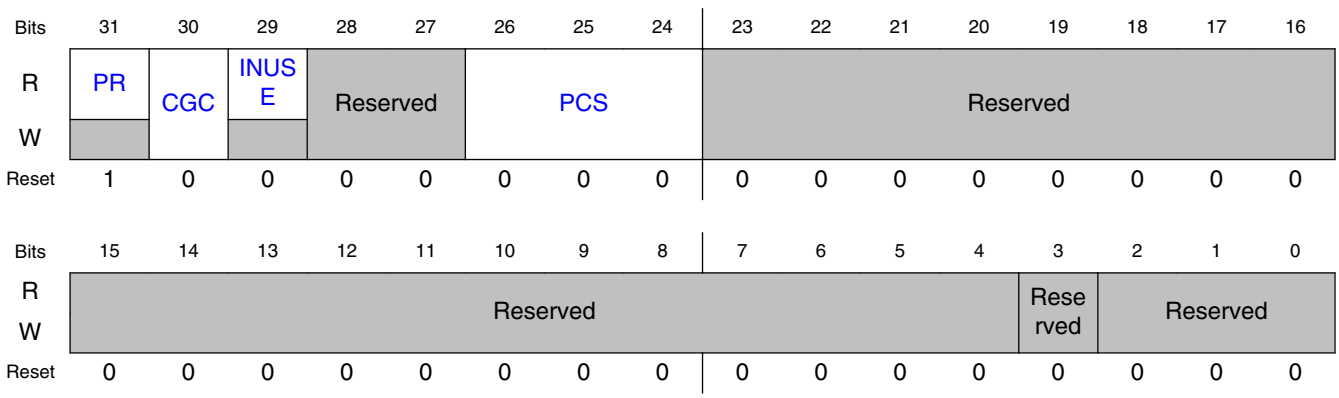
36.2.2.6 PCC LPSPI0 (PCC_LPSPI0)

36.2.2.6.1 Address

Register	Offset
PCC_LPSPI0	400FA0F0h

PCC Register

36.2.2.6.2 Diagram



36.2.2.6.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock.

Table continues on the next page...

Field	Function
	010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.7 PCC LPSPI1 (PCC_LPSPI1)

36.2.2.7.1 Address

Register	Offset
PCC_LPSPI1	400FA0F4h

PCC Register

36.2.2.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.7.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.8 PCC LPI2C0 (PCC_LPI2C0)

36.2.2.8.1 Address

Register	Offset
PCC_LPI2C0	400FA100h

PCC Register

36.2.2.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.8.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock .

Table continues on the next page...

Field	Function
	111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.9 PCC LPI2C1 (PCC_LPI2C1)

36.2.2.9.1 Address

Register	Offset
PCC_LPI2C1	400FA104h

PCC Register

36.2.2.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.9.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled

Table continues on the next page...

Field	Function
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.10 PCC LPUART0 (PCC_LPUART0)

36.2.2.10.1 Address

Register	Offset
PCC_LPUART0	400FA110h

PCC Register

36.2.2.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.10.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.11 PCC LPUART1 (PCC_LPUART1)

36.2.2.11.1 Address

Register	Offset
PCC_LPUART1	400FA114h

PCC Register

36.2.2.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.11.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled

Table continues on the next page...

Field	Function
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.12 PCC FLEXIO0 (PCC_FLEXIO0)

36.2.2.12.1 Address

Register	Offset
PCC_FLEXIO0	400FA128h

PCC Register

36.2.2.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		PCS			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.12.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	This read/write bit field is used for peripherals that support various clock selections. At SOC integration, each peripheral is assigned either the 'slow' clock source choices or the 'fast' clock source choices. This field can only be written when the CGC bit is 0 (clock disabled). Likewise, if the INUSE flag is set, this field is locked. 000 - Clock is off (or test clock is enabled). 001 - OSCCLK - System Oscillator Bus Clock. 010 - SCGIRCLK - Slow IRC Clock. 011 - SCGFIRCLK - Fast IRC Clock. 100 - Reserved. 101 - Reserved. 110 - SCGPCLK System PLL clock . 111 - Reserved.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.2.2.13 PCC CMP1 (PCC_CMP1)

36.2.2.13.1 Address

Register	Offset
PCC_CMP1	400FA1BCh

PCC Register

36.2.2.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	Reserved		Reserved			Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

36.2.2.13.3 Fields

Field	Function
31 PR	This bit shows whether the peripheral is present on this device. 0 - disabled 1 - enabled
30 CGC	This read/write bit enables the clock gate for the peripheral. 0 - Clock disabled 1 - Clock enabled

Table continues on the next page...

Field	Function
29 INUSE	This read-only bit indicates the peripheral is being used. 0 - Peripheral is not being used. 1 - Peripheral is being used.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

36.3 Functional description

The Peripheral Clock Control (PCC) module provides clock gating and clock source selection to each peripheral.

Each peripheral has it's own unique PCCn register that provide clock gating for the peripheral's interface clock and where applicable it's functional clock.

Optional peripheral functional clock sources can come from the SCG (*xxx*DIV*y*_CLK) or other modules eg LPO. Not all peripherals will make use of all available peripheral functional clocks.

The peripheral interface clock can be either DIVSLOW_CLK or DIVCORE_CLK.

See each peripherals PCCn register for details.

Chapter 37
Power Management Controller (PMC)

37.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The power management controller (PMC) contains the internal voltage regulator, power on reset (POR), and low voltage detect system (LVD) and high voltage detect system (HVD).

See AN4503: Power Management for Kinetis MCUs for further details on using the PMC.

37.2 Features

A list of included PMC features can be found here.

- Internal voltage regulator
- Active POR providing brown-out detect
- Low-voltage detect supporting two low-voltage trip points with four warning levels per trip point
- High-voltage detect supporting two high-voltage trip points

37.3 Low-voltage detect (LVD) system

This device includes a system to guard against low-voltage conditions. This protects memory contents and controls MCU system states during supply voltage variations.

The system is comprised of a power-on reset (POR) circuit and a LVD circuit with a user-selectable trip voltage: high (V_{LVDH}) or low (V_{LVDL}). The trip voltage is selected by LVDSC1[LVDV]. The LVD is disabled upon entering VLPx, LLS, and VLLSx modes.

Two flags are available to indicate the status of the low-voltage detect system:

- The Low Voltage Detect Flag in the Low Voltage Status and Control 1 Register (LVDSC1[LVDF]) operates in a level sensitive manner. LVDSC1[LVDF] is set when the supply voltage falls below the selected trip point (VLVD). LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC1[LVDF] remains set.
- The Low Voltage Warning Flag (LVWF) in the Low Voltage Status and Control 2 Register (LVDSC2[LVWF]) operates in a level sensitive manner. LVDSC2[LVWF] is set when the supply voltage falls below the selected monitor trip point (VLVW). LVDSC2[LVWF] is cleared by writing one to LVDSC2[LVWACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC2[LVWF] remains set.

37.3.1 LVD reset operation

By setting LVDSC1[LVDRE], the LVD generates a reset upon detection of a low-voltage condition. The low-voltage detection threshold is determined by LVDSC1[LVDV]. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above this threshold. The LVD field in the SRS register of the RCM module (RCM_SRS[LVD]) is set following an LVD or power-on reset.

37.3.2 LVD interrupt operation

By configuring the LVD circuit for interrupt operation (LVDSC1[LVDIE] set and LVDSC1[LVDRE] clear), LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK].

37.3.3 Low-voltage warning (LVW) interrupt operation

The LVD system contains a Low-Voltage Warning Flag (LVWF) in the Low Voltage Detect Status and Control 2 Register to indicate that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt, which is enabled by

setting LVDSC2[LVWIE]. If enabled, an LVW interrupt request occurs when LVDSC2[LVWF] is set. LVDSC2[LVWF] is cleared by writing 1 to LVDSC2[LVWACK].

LVDSC2[LVWV] selects one of the four trip voltages:

- Highest: V_{LVW4}
- Two mid-levels: V_{LVW3} and V_{LVW2}
- Lowest: V_{LVW1}

37.4 High-voltage detect (HVD) system

This device includes a system to guard against high-voltage conditions.

The system is comprised of a HVD circuit with a user-selectable trip voltage: high (V_{HVDH}) or low (V_{HVDL}). The trip voltage is selected by HVDSC1[HVDV]. The HVD is disabled upon entering VLPx, LLS, and VLLSx modes.

A flag is available to indicate the status of the high-voltage detect system:

- The High Voltage Detect Flag in the High Voltage Status and Control 1 Register (HVDSC1[HVDF]) operates in a level sensitive manner. HVDSC1[HVDF] is set when the supply voltage rises above the selected trip point (VHVD). HVDSC1[HVDF] is cleared by writing 1 to HVDSC1[HVDACK], but only if the internal supply has returned below the trip point; otherwise, HVDSC1[LVDF] remains set.

37.4.1 HVD reset operation

By setting HVDSC1[HVDRE], the HVD generates a reset upon detection of a high-voltage condition. The high-voltage detection threshold is determined by HVDSC1[HVDV]. After an HVD reset occurs, the HVD system holds the MCU in reset until the supply voltage falls below this threshold. The LVD field in the SRS register of the RCM module (RCM_SRS[LVD]) is set following an HVD reset.

37.4.2 HVD interrupt operation

By configuring the HVD circuit for interrupt operation (HVDSC1[HVDIE] set and HVDSC1[HVDRE] clear), HVDSC1[HVDF] is set and an HVD interrupt request occurs upon detection of a high voltage condition. HVDSC1[HVDF] is cleared by writing 1 to HVDSC1[HVDACK].

37.5 I/O retention

When in LLS mode, the I/O pins are held in their input or output state.

Upon wakeup, the PMC is re-enabled, goes through a power up sequence to full regulation, and releases the logic from state retention mode. The I/O are released immediately after a wake-up or reset event. In the case of LLS exit via a RESET pin, the I/O default to their reset state.

When in VLLS modes, the I/O states are held on a wake-up event (with the exception of wake-up by reset event) until the wake-up has been acknowledged via a write to REGSC[ACKISO]. In the case of VLLS exit via a RESET pin, the I/O are released and default to their reset state. In this case, no write to REGSC[ACKISO] is needed.

37.6 Memory map and register descriptions

Details about the PMC registers can be found here.

NOTE

Different portions of PMC registers are reset only by particular reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

The PMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

PMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_D000	Version ID register (PMC_VERID)	32	R	0400_0000h	37.6.1/971
4007_D004	Parameter register (PMC_PARAM)	32	R	See section	37.6.2/972
4007_D008	Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)	32	R/W	0000_0010h	37.6.3/972
4007_D00C	Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)	32	R/W	0000_0000h	37.6.4/974
4007_D010	Regulator Status And Control register (PMC_REGSC)	32	R/W	0000_0024h	37.6.5/976
4007_D034	High Voltage Detect Status And Control 1 register (PMC_HVDSC1)	32	R/W	0000_0001h	37.6.6/978

37.6.1 Version ID register (PMC_VERID)

Address: 4007_D000h base + 0h offset = 4007_D000h

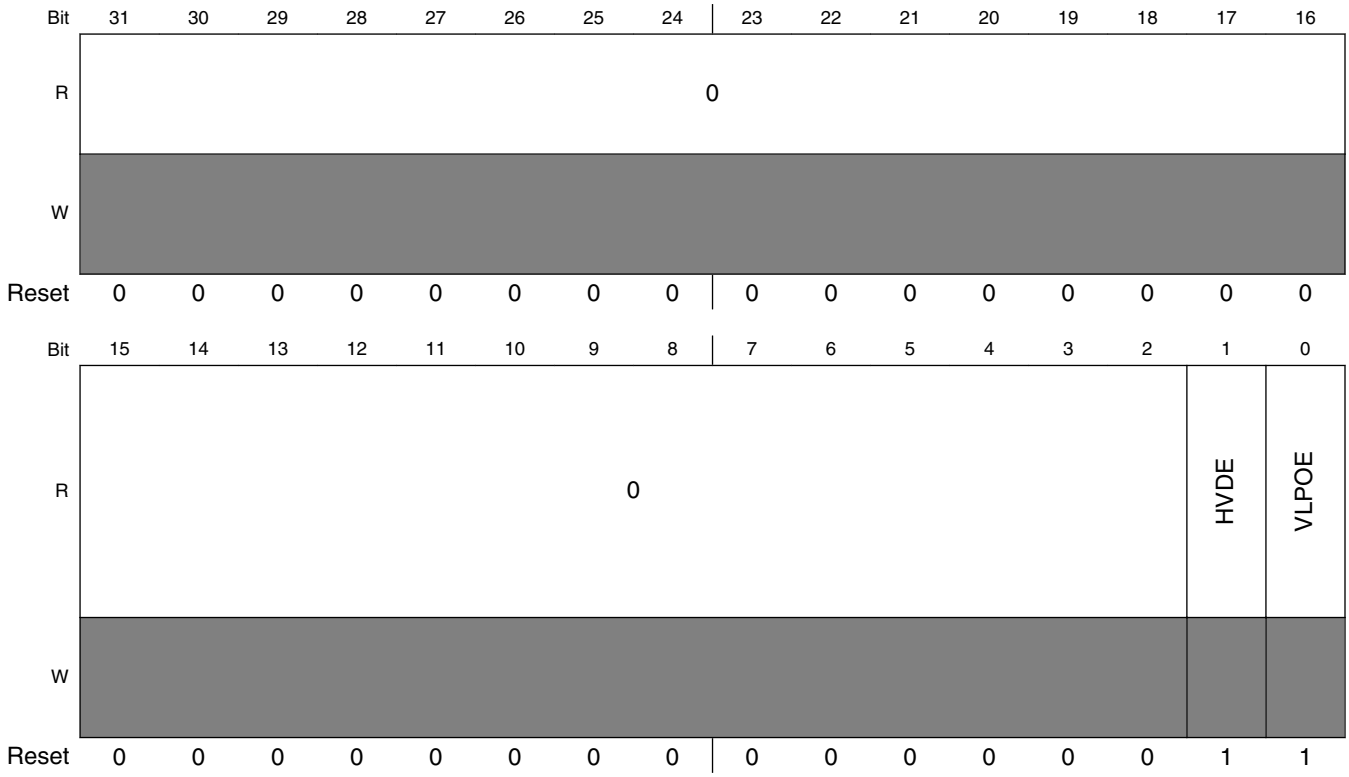
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PMC_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0000 Standard features implemented

37.6.2 Parameter register (PMC_PARAM)

Address: 4007_D000h base + 4h offset = 4007_D004h



PMC_PARAM field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 HVDE	HVD Enabled If set, indicates support for High Voltage Detect is enabled.
0 VLPOE	VLPO Enable If set, indicates support for VLPO bit is enabled in the PMC_REGSC register.

37.6.3 Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)

This register contains status and control bits to support the low voltage detect function. This register should be written during the reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

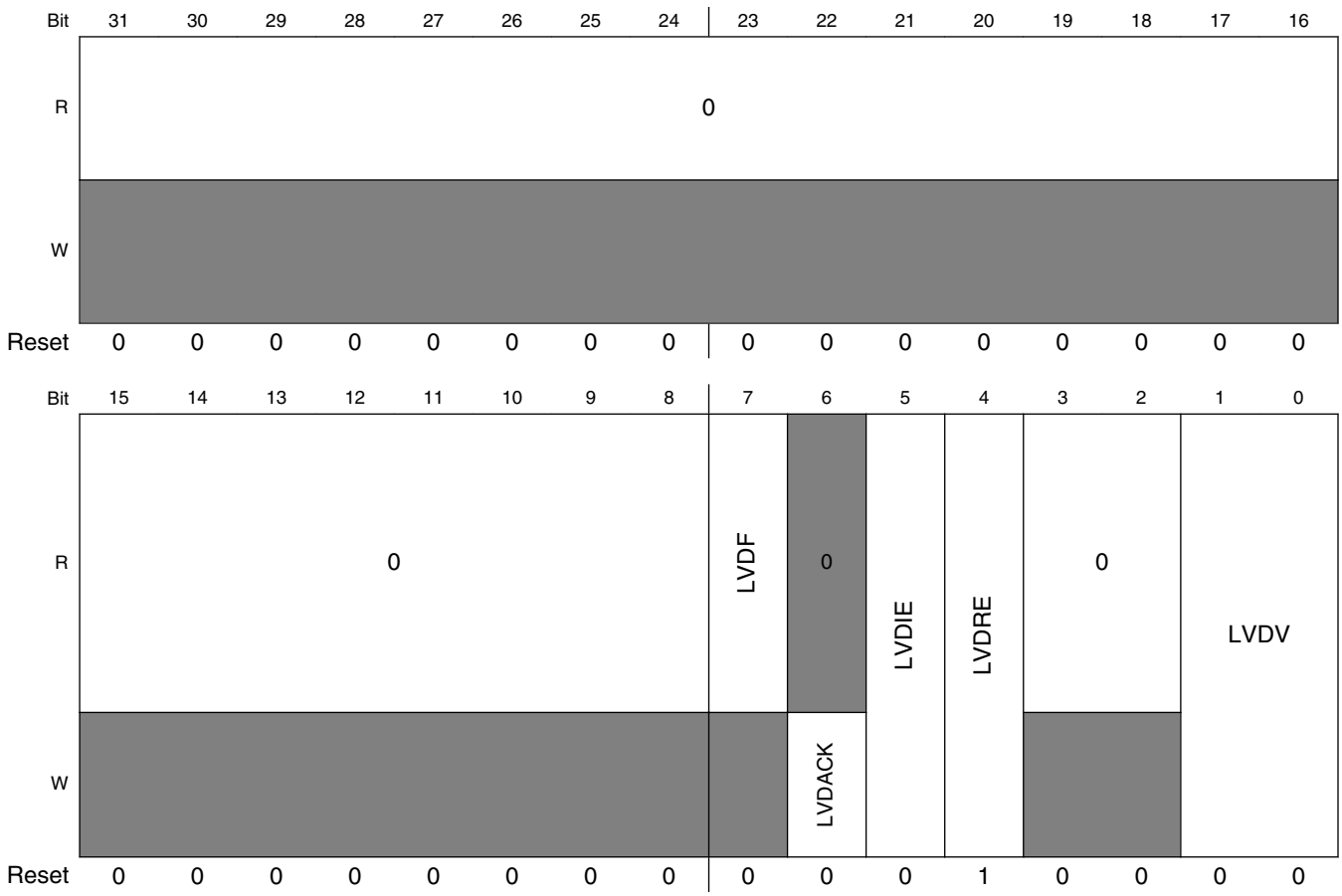
While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC1 settings. To protect systems that must have LVD always on, configure the Power Mode Protection (PMPROT) register of the SMC module (SMC_PMPROT) to disallow any very low power or low leakage modes from being enabled.

See the device's data sheet for the exact LVD trip voltages.

NOTE

The LVDV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + 8h offset = 4007_D008h



PMC_LVDSC1 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

PMC_LVDSC1 field descriptions (continued)

Field	Description
7 LVDF	Low-Voltage Detect Flag This read-only status field indicates a low-voltage detect event. 0 Low-voltage event not detected 1 Low-voltage event detected
6 LVDACK	Low-Voltage Detect Acknowledge This write-only field is used to acknowledge low voltage detection errors. Write 1 to clear LVDF. Reads always return 0.
5 LVDIE	Low-Voltage Detect Interrupt Enable Enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1
4 LVDRE	Low-Voltage Detect Reset Enable This write-once bit enables LVDF events to generate a hardware reset. Additional writes are ignored. 0 LVDF does not generate hardware resets 1 Force an MCU reset when LVDF = 1
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVDV	Low-Voltage Detect Voltage Select Selects the LVD trip point voltage (V_{LVD}). 00 Low trip point selected ($V_{LVD} = V_{LVDL}$) 01 High trip point selected ($V_{LVD} = V_{LVDH}$) 10 Reserved 11 Reserved

37.6.4 Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)

This register contains status and control bits to support the low voltage warning function.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC2 settings.

See the device's data sheet for the exact LVD trip voltages.

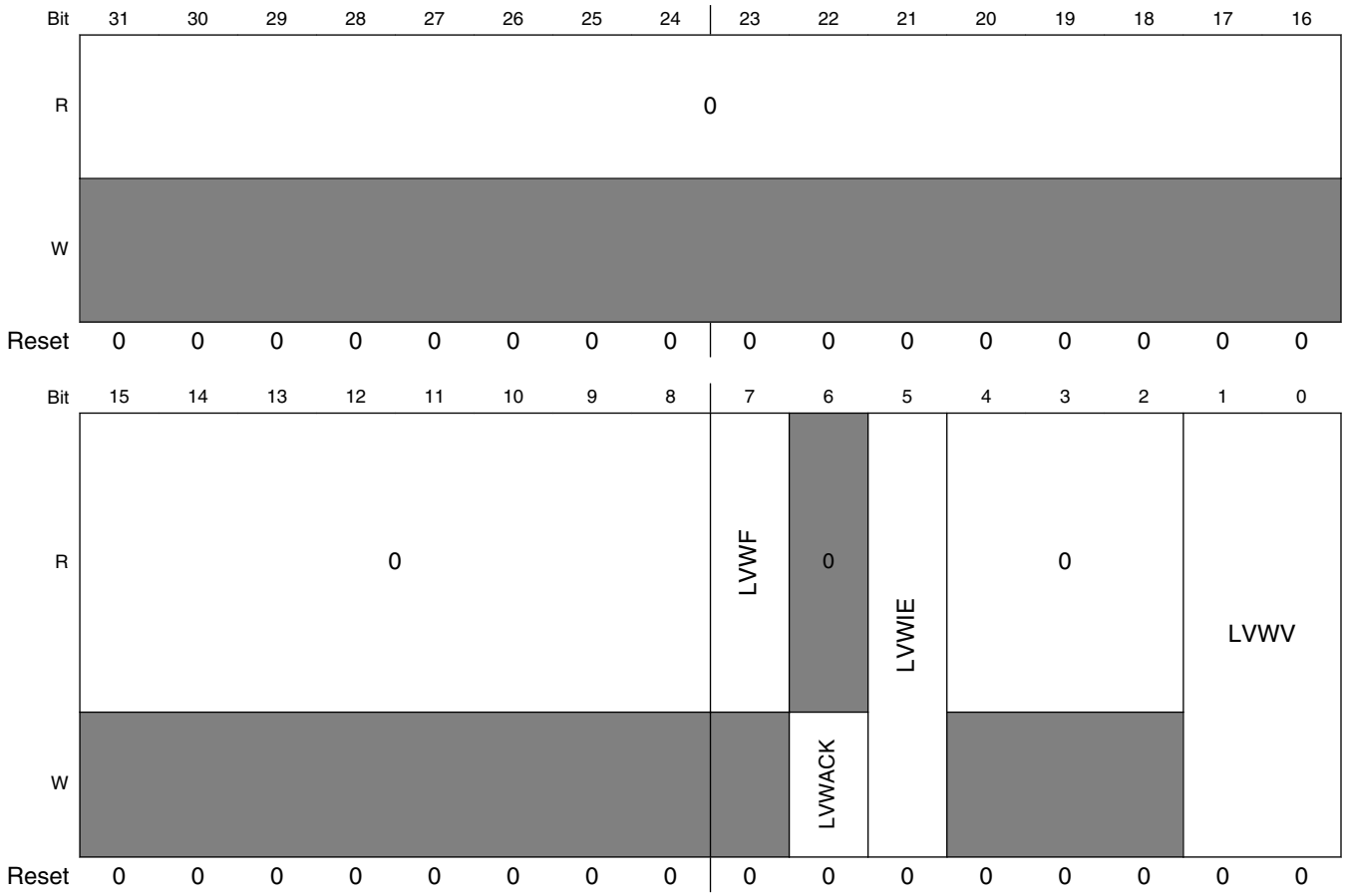
NOTE

The LVW trip voltages depend on LVWV and LVDV.

NOTE

LVWV is reset solely on a POR Only event. The other fields of the register are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + Ch offset = 4007_D00Ch



PMC_LVDSC2 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 LVWF	Low-Voltage Warning Flag This read-only status field indicates a low-voltage warning event. LVWF is set when V_{Supply} transitions below the trip point, or after reset and V_{Supply} is already below V_{LVW}. LVWF may be 1 after power-on reset, therefore, to use LVW interrupt function, before enabling LVWIE, LVWF must be cleared by writing LVWACK first. 0 Low-voltage warning event not detected 1 Low-voltage warning event detected
6 LVWACK	Low-Voltage Warning Acknowledge

Table continues on the next page...

PMC_LVDSC2 field descriptions (continued)

Field	Description
	This write-only field is used to acknowledge low voltage warning errors. Write 1 to clear LVWF. Reads always return 0.
5 LVWIE	Low-Voltage Warning Interrupt Enable Enables hardware interrupt requests for LVWF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVWF = 1
4–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVWV	Low-Voltage Warning Voltage Select Selects the LVW trip point voltage (V_{LVW}). The actual voltage for the warning depends on LVDSC1[LVDV]. 00 Low trip point selected ($V_{LVW} = V_{LVW1}$) 01 Mid 1 trip point selected ($V_{LVW} = V_{LVW2}$) 10 Mid 2 trip point selected ($V_{LVW} = V_{LVW3}$) 11 High trip point selected ($V_{LVW} = V_{LVW4}$)

37.6.5 Regulator Status And Control register (PMC_REGSC)

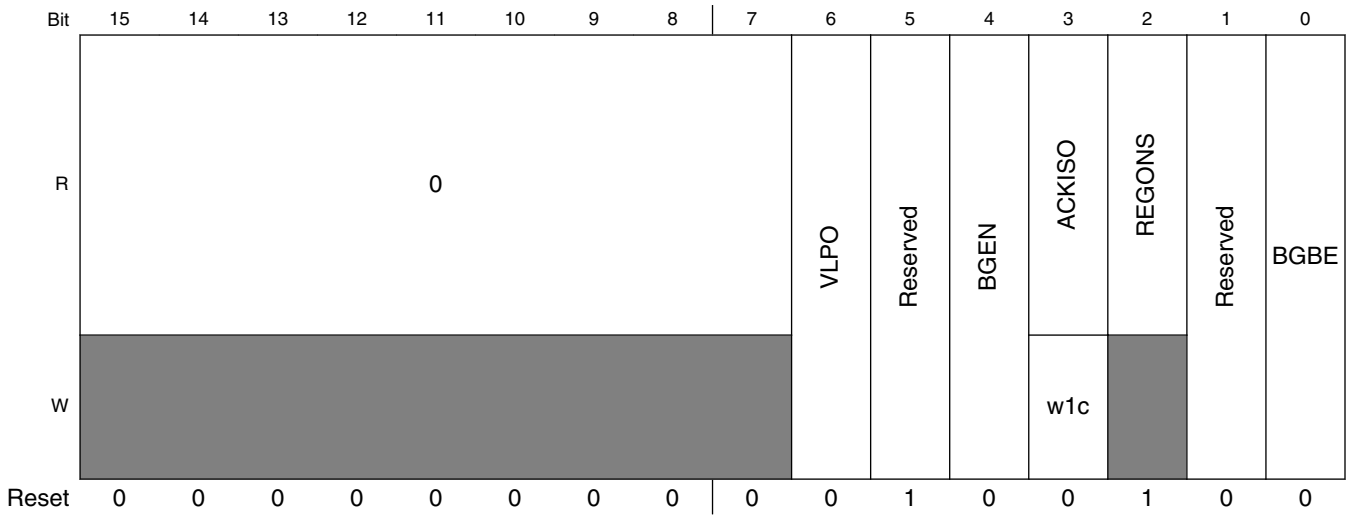
The PMC contains an internal voltage regulator. The voltage regulator design uses a bandgap reference that is also available through a buffer as input to certain internal peripherals, such as the CMP and ADC. The internal regulator provides a status bit (REGONS) indicating the regulator is in run regulation.

NOTE

This register is reset on Chip Reset Not VLLS and by reset types that trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007_D000h base + 10h offset = 4007_D010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



PMC_REGSC field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 VLPO	VLPx Option When used in conjunction with BGEN, this bit allows additional clock sources and higher frequency operation (at the cost of higher power) to be selected during VLPx modes. 0 Operating frequencies and SCG clocking modes are restricted during VLPx modes as listed in the Power Management chapter. 1 If BGEN is also set, operating frequencies and SCG clocking modes are unrestricted during VLPx modes. Note that flash access frequency is still restricted however.
5 Reserved	This field is reserved.
4 BGEN	Bandgap Enable In VLPx Operation BGEN controls whether the bandgap is enabled in lower power modes of operation (VLPx, LLS, and VLLSx). When on-chip peripherals require the bandgap voltage reference in low power modes of operation, set BGEN to continue to enable the bandgap operation. **NOTE:** When the bandgap voltage reference is not needed in low power modes, clear BGEN to avoid excess power consumption. 0 Bandgap voltage reference is disabled in VLPx , LLS , and VLLSx modes. 1 Bandgap voltage reference is enabled in VLPx , LLS , and VLLSx modes.
3 ACKISO	Acknowledge Isolation Reading this field indicates whether certain peripherals and the I/O pads are in a latched state as a result of having been in a VLLS mode. Writing 1 to this field when it is set releases the I/O pads and certain peripherals to their normal run mode state. **NOTE:** After recovering from a VLLS mode, user should restore chip configuration before clearing ACKISO. In particular, pin configuration for enabled LLWU wakeup pins should be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared.

Table continues on the next page...

PMC_REGSC field descriptions (continued)

Field	Description
	0 Peripherals and I/O pads are in normal run state. 1 Certain peripherals and I/O pads are in an isolated and latched state.
2 REGONS	Regulator In Run Regulation Status This read-only field provides the current status of the internal voltage regulator. 0 Regulator is in stop regulation or in transition to/from it 1 Regulator is in run regulation
1 Reserved	This field is reserved. **NOTE:** This reserved bit must remain cleared (set to 0).
0 BGBE	Bandgap Buffer Enable Enables the bandgap buffer. 0 Bandgap buffer not enabled 1 Bandgap buffer enabled

37.6.6 High Voltage Detect Status And Control 1 register (PMC_HVDSC1)

This register contains status and control bits to support the high voltage detect function. This register should be written during the reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

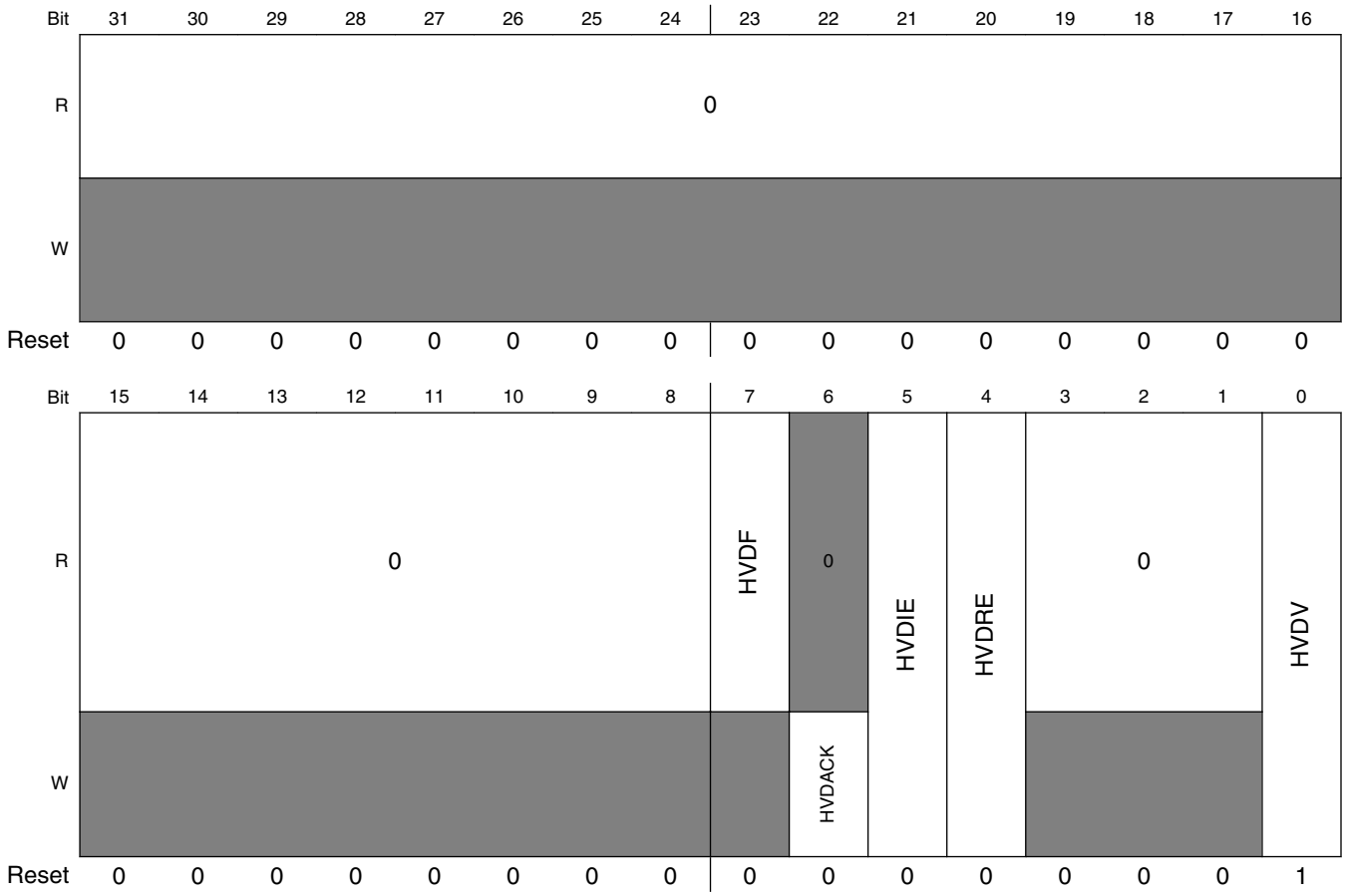
While the device is in the very low power or low leakage modes, the HVD system is disabled regardless of HVDSC1 settings. To protect systems that must have HVD always on, configure the Power Mode Protection (PMPROT) register of the SMC module (SMC_PMPROT) to disallow any very low power or low leakage modes from being enabled.

See the device's data sheet for the exact HVD trip voltages.

NOTE

This register is reset solely on a POR Only event.For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + 34h offset = 4007_D034h



PMC_HVDSC1 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 HVDF	High-Voltage Detect Flag This read-only status field indicates a high-voltage detect event. 0 High-voltage event not detected 1 High-voltage event detected
6 HVDACK	High-Voltage Detect Acknowledge This write-only field is used to acknowledge high voltage detection errors. Write 1 to clear HVDF. Reads always return 0.
5 HVDIE	High-Voltage Detect Interrupt Enable Enables hardware interrupt requests for HVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when HVDF = 1
4 HVDRE	High-Voltage Detect Reset Enable

Table continues on the next page...

PMC_HVDSC1 field descriptions (continued)

Field	Description
	This write-once bit enables HVDF events to generate a hardware reset. Additional writes are ignored until the next chip reset. 0 HVDF does not generate hardware resets 1 Force an MCU reset when HVDF = 1
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HVDV	High-Voltage Detect Voltage Select Selects the HVD trip point voltage (V_{HVD}). 0 Low trip point selected ($V_{HVD} = V_{HVDL}$) 1 High trip point selected ($V_{HVD} = V_{HVDH}$)

Chapter 38
Port Control and Interrupts (PORT)

38.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

38.2 Overview

The Port Control and Interrupt (PORT) module provides support for port control, digital filtering, and external interrupt functions.

Most functions can be configured independently for each pin in the 32-bit port and affect the pin regardless of its pin muxing state.

There is one instance of the PORT module for each port. Not all pins within each port are implemented on a specific device.

38.2.1 Features

The PORT module has the following features:

- Pin interrupt
  - Interrupt flag and enable registers for each pin
  - Support for edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wake-up in low-power modes
  - Pin interrupt is functional in all digital pin muxing modes
  - Peripheral trigger output (active high, low) configured per pin
- Digital input filter on selected pins

- Digital input filter for each pin, usable by any digital peripheral muxed onto the pin
- Individual enable or bypass control field per pin
- Selectable clock source for digital input filter with a five bit resolution on filter size
- Functional in all digital pin multiplexing modes

- Port control
  - Individual pull control fields with pullup, pulldown, and pull-disable support on selected pins
  - Individual drive strength field supporting high and low drive strength on selected pins
  - Individual slew rate field supporting fast and slow slew rates on selected pins
  - Individual input passive filter field supporting enable and disable of the individual input passive filter on selected pins
  - Individual open drain field supporting enable and disable of the individual open drain output on selected pins
  - Individual mux control field supporting analog or pin disabled, GPIO, and up to six chip-specific digital functions
  - Pad configuration fields are functional in all digital pin muxing modes.

38.2.2 Modes of operation

38.2.2.1 Run mode

In Run mode, the PORT operates normally.

38.2.2.2 Wait mode

In Wait mode, PORT continues to operate normally and may be configured to exit the Low-Power mode if an enabled interrupt is detected. DMA requests are still generated during the Wait mode, but do not cause an exit from the Low-Power mode.

38.2.2.3 Stop mode

In Stop mode, the PORT can be configured to exit the Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

In Stop mode, the digital input filters are bypassed unless they are configured to run from the LPO clock source.

38.2.2.4 Debug mode

In Debug mode, PORT operates normally.

38.3 External signal description

The table found here describes the PORT external signal.

Table 38-1. Signal properties

Name	Function	I/O	Reset	Pull
PORTx[31:0]	External interrupt	I/O	0	-

NOTE

Not all pins within each port are implemented on each device.

38.4 Detailed signal description

The table found here contains the detailed signal description for the PORT interface.

Table 38-2. PORT interface—detailed signal description

Signal	I/O	Description	
PORTx[31:0]	I/O	External interrupt.	
		State meaning	Asserted—pin is logic 1. Negated—pin is logic 0.
		Timing	Assertion—may occur at any time and can assert asynchronously to the system clock. Negation—may occur at any time and can assert asynchronously to the system clock.

38.5 Memory map and register definition

Any read or write access to the PORT memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states.

PORT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_A000	Pin Control Register n (PORTA_PCR0)	32	R/W	See section	38.5.1/990
4005_A004	Pin Control Register n (PORTA_PCR1)	32	R/W	See section	38.5.1/990
4005_A008	Pin Control Register n (PORTA_PCR2)	32	R/W	See section	38.5.1/990
4005_A00C	Pin Control Register n (PORTA_PCR3)	32	R/W	See section	38.5.1/990
4005_A010	Pin Control Register n (PORTA_PCR4)	32	R/W	See section	38.5.1/990
4005_A014	Pin Control Register n (PORTA_PCR5)	32	R/W	See section	38.5.1/990
4005_A018	Pin Control Register n (PORTA_PCR6)	32	R/W	See section	38.5.1/990
4005_A01C	Pin Control Register n (PORTA_PCR7)	32	R/W	See section	38.5.1/990
4005_A020	Pin Control Register n (PORTA_PCR8)	32	R/W	See section	38.5.1/990
4005_A024	Pin Control Register n (PORTA_PCR9)	32	R/W	See section	38.5.1/990
4005_A028	Pin Control Register n (PORTA_PCR10)	32	R/W	See section	38.5.1/990
4005_A02C	Pin Control Register n (PORTA_PCR11)	32	R/W	See section	38.5.1/990
4005_A030	Pin Control Register n (PORTA_PCR12)	32	R/W	See section	38.5.1/990
4005_A034	Pin Control Register n (PORTA_PCR13)	32	R/W	See section	38.5.1/990
4005_A038	Pin Control Register n (PORTA_PCR14)	32	R/W	See section	38.5.1/990
4005_A03C	Pin Control Register n (PORTA_PCR15)	32	R/W	See section	38.5.1/990
4005_A040	Pin Control Register n (PORTA_PCR16)	32	R/W	See section	38.5.1/990
4005_A044	Pin Control Register n (PORTA_PCR17)	32	R/W	See section	38.5.1/990
4005_A048	Pin Control Register n (PORTA_PCR18)	32	R/W	See section	38.5.1/990
4005_A04C	Pin Control Register n (PORTA_PCR19)	32	R/W	See section	38.5.1/990
4005_A050	Pin Control Register n (PORTA_PCR20)	32	R/W	See section	38.5.1/990
4005_A054	Pin Control Register n (PORTA_PCR21)	32	R/W	See section	38.5.1/990
4005_A058	Pin Control Register n (PORTA_PCR22)	32	R/W	See section	38.5.1/990
4005_A05C	Pin Control Register n (PORTA_PCR23)	32	R/W	See section	38.5.1/990
4005_A060	Pin Control Register n (PORTA_PCR24)	32	R/W	See section	38.5.1/990
4005_A064	Pin Control Register n (PORTA_PCR25)	32	R/W	See section	38.5.1/990
4005_A068	Pin Control Register n (PORTA_PCR26)	32	R/W	See section	38.5.1/990
4005_A06C	Pin Control Register n (PORTA_PCR27)	32	R/W	See section	38.5.1/990
4005_A070	Pin Control Register n (PORTA_PCR28)	32	R/W	See section	38.5.1/990
4005_A074	Pin Control Register n (PORTA_PCR29)	32	R/W	See section	38.5.1/990
4005_A078	Pin Control Register n (PORTA_PCR30)	32	R/W	See section	38.5.1/990
4005_A07C	Pin Control Register n (PORTA_PCR31)	32	R/W	See section	38.5.1/990
4005_A080	Global Pin Control Low Register (PORTA_GPCLR)	32	W (always reads 0)	0000_0000h	38.5.2/993
4005_A084	Global Pin Control High Register (PORTA_GPCHR)	32	W (always reads 0)	0000_0000h	38.5.3/993

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_A088	Global Interrupt Control Low Register (PORTA_GICLR)	32	W (always reads 0)	0000_0000h	38.5.4/994
4005_A08C	Global Interrupt Control High Register (PORTA_GICHR)	32	W (always reads 0)	0000_0000h	38.5.5/994
4005_A0A0	Interrupt Status Flag Register (PORTA_ISFR)	32	w1c	0000_0000h	38.5.6/995
4005_A0C0	Digital Filter Enable Register (PORTA_DFER)	32	R/W	0000_0000h	38.5.7/995
4005_A0C4	Digital Filter Clock Register (PORTA_DFCR)	32	R/W	0000_0000h	38.5.8/996
4005_A0C8	Digital Filter Width Register (PORTA_DFWR)	32	R/W	0000_0000h	38.5.9/997
4005_B000	Pin Control Register n (PORTB_PCR0)	32	R/W	See section	38.5.1/990
4005_B004	Pin Control Register n (PORTB_PCR1)	32	R/W	See section	38.5.1/990
4005_B008	Pin Control Register n (PORTB_PCR2)	32	R/W	See section	38.5.1/990
4005_B00C	Pin Control Register n (PORTB_PCR3)	32	R/W	See section	38.5.1/990
4005_B010	Pin Control Register n (PORTB_PCR4)	32	R/W	See section	38.5.1/990
4005_B014	Pin Control Register n (PORTB_PCR5)	32	R/W	See section	38.5.1/990
4005_B018	Pin Control Register n (PORTB_PCR6)	32	R/W	See section	38.5.1/990
4005_B01C	Pin Control Register n (PORTB_PCR7)	32	R/W	See section	38.5.1/990
4005_B020	Pin Control Register n (PORTB_PCR8)	32	R/W	See section	38.5.1/990
4005_B024	Pin Control Register n (PORTB_PCR9)	32	R/W	See section	38.5.1/990
4005_B028	Pin Control Register n (PORTB_PCR10)	32	R/W	See section	38.5.1/990
4005_B02C	Pin Control Register n (PORTB_PCR11)	32	R/W	See section	38.5.1/990
4005_B030	Pin Control Register n (PORTB_PCR12)	32	R/W	See section	38.5.1/990
4005_B034	Pin Control Register n (PORTB_PCR13)	32	R/W	See section	38.5.1/990
4005_B038	Pin Control Register n (PORTB_PCR14)	32	R/W	See section	38.5.1/990
4005_B03C	Pin Control Register n (PORTB_PCR15)	32	R/W	See section	38.5.1/990
4005_B040	Pin Control Register n (PORTB_PCR16)	32	R/W	See section	38.5.1/990
4005_B044	Pin Control Register n (PORTB_PCR17)	32	R/W	See section	38.5.1/990
4005_B048	Pin Control Register n (PORTB_PCR18)	32	R/W	See section	38.5.1/990
4005_B04C	Pin Control Register n (PORTB_PCR19)	32	R/W	See section	38.5.1/990
4005_B050	Pin Control Register n (PORTB_PCR20)	32	R/W	See section	38.5.1/990
4005_B054	Pin Control Register n (PORTB_PCR21)	32	R/W	See section	38.5.1/990
4005_B058	Pin Control Register n (PORTB_PCR22)	32	R/W	See section	38.5.1/990
4005_B05C	Pin Control Register n (PORTB_PCR23)	32	R/W	See section	38.5.1/990
4005_B060	Pin Control Register n (PORTB_PCR24)	32	R/W	See section	38.5.1/990
4005_B064	Pin Control Register n (PORTB_PCR25)	32	R/W	See section	38.5.1/990
4005_B068	Pin Control Register n (PORTB_PCR26)	32	R/W	See section	38.5.1/990
4005_B06C	Pin Control Register n (PORTB_PCR27)	32	R/W	See section	38.5.1/990
4005_B070	Pin Control Register n (PORTB_PCR28)	32	R/W	See section	38.5.1/990

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_B074	Pin Control Register n (PORTB_PCR29)	32	R/W	See section	38.5.1/990
4005_B078	Pin Control Register n (PORTB_PCR30)	32	R/W	See section	38.5.1/990
4005_B07C	Pin Control Register n (PORTB_PCR31)	32	R/W	See section	38.5.1/990
4005_B080	Global Pin Control Low Register (PORTB_GPCLR)	32	W (always reads 0)	0000_0000h	38.5.2/993
4005_B084	Global Pin Control High Register (PORTB_GPCHR)	32	W (always reads 0)	0000_0000h	38.5.3/993
4005_B088	Global Interrupt Control Low Register (PORTB_GICLR)	32	W (always reads 0)	0000_0000h	38.5.4/994
4005_B08C	Global Interrupt Control High Register (PORTB_GICHR)	32	W (always reads 0)	0000_0000h	38.5.5/994
4005_B0A0	Interrupt Status Flag Register (PORTB_ISFR)	32	w1c	0000_0000h	38.5.6/995
4005_B0C0	Digital Filter Enable Register (PORTB_DFER)	32	R/W	0000_0000h	38.5.7/995
4005_B0C4	Digital Filter Clock Register (PORTB_DFCR)	32	R/W	0000_0000h	38.5.8/996
4005_B0C8	Digital Filter Width Register (PORTB_DFWR)	32	R/W	0000_0000h	38.5.9/997
4005_C000	Pin Control Register n (PORTC_PCR0)	32	R/W	See section	38.5.1/990
4005_C004	Pin Control Register n (PORTC_PCR1)	32	R/W	See section	38.5.1/990
4005_C008	Pin Control Register n (PORTC_PCR2)	32	R/W	See section	38.5.1/990
4005_C00C	Pin Control Register n (PORTC_PCR3)	32	R/W	See section	38.5.1/990
4005_C010	Pin Control Register n (PORTC_PCR4)	32	R/W	See section	38.5.1/990
4005_C014	Pin Control Register n (PORTC_PCR5)	32	R/W	See section	38.5.1/990
4005_C018	Pin Control Register n (PORTC_PCR6)	32	R/W	See section	38.5.1/990
4005_C01C	Pin Control Register n (PORTC_PCR7)	32	R/W	See section	38.5.1/990
4005_C020	Pin Control Register n (PORTC_PCR8)	32	R/W	See section	38.5.1/990
4005_C024	Pin Control Register n (PORTC_PCR9)	32	R/W	See section	38.5.1/990
4005_C028	Pin Control Register n (PORTC_PCR10)	32	R/W	See section	38.5.1/990
4005_C02C	Pin Control Register n (PORTC_PCR11)	32	R/W	See section	38.5.1/990
4005_C030	Pin Control Register n (PORTC_PCR12)	32	R/W	See section	38.5.1/990
4005_C034	Pin Control Register n (PORTC_PCR13)	32	R/W	See section	38.5.1/990
4005_C038	Pin Control Register n (PORTC_PCR14)	32	R/W	See section	38.5.1/990
4005_C03C	Pin Control Register n (PORTC_PCR15)	32	R/W	See section	38.5.1/990
4005_C040	Pin Control Register n (PORTC_PCR16)	32	R/W	See section	38.5.1/990
4005_C044	Pin Control Register n (PORTC_PCR17)	32	R/W	See section	38.5.1/990
4005_C048	Pin Control Register n (PORTC_PCR18)	32	R/W	See section	38.5.1/990
4005_C04C	Pin Control Register n (PORTC_PCR19)	32	R/W	See section	38.5.1/990
4005_C050	Pin Control Register n (PORTC_PCR20)	32	R/W	See section	38.5.1/990

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_C054	Pin Control Register n (PORTC_PCR21)	32	R/W	See section	38.5.1/990
4005_C058	Pin Control Register n (PORTC_PCR22)	32	R/W	See section	38.5.1/990
4005_C05C	Pin Control Register n (PORTC_PCR23)	32	R/W	See section	38.5.1/990
4005_C060	Pin Control Register n (PORTC_PCR24)	32	R/W	See section	38.5.1/990
4005_C064	Pin Control Register n (PORTC_PCR25)	32	R/W	See section	38.5.1/990
4005_C068	Pin Control Register n (PORTC_PCR26)	32	R/W	See section	38.5.1/990
4005_C06C	Pin Control Register n (PORTC_PCR27)	32	R/W	See section	38.5.1/990
4005_C070	Pin Control Register n (PORTC_PCR28)	32	R/W	See section	38.5.1/990
4005_C074	Pin Control Register n (PORTC_PCR29)	32	R/W	See section	38.5.1/990
4005_C078	Pin Control Register n (PORTC_PCR30)	32	R/W	See section	38.5.1/990
4005_C07C	Pin Control Register n (PORTC_PCR31)	32	R/W	See section	38.5.1/990
4005_C080	Global Pin Control Low Register (PORTC_GPCLR)	32	W (always reads 0)	0000_0000h	38.5.2/993
4005_C084	Global Pin Control High Register (PORTC_GPCHR)	32	W (always reads 0)	0000_0000h	38.5.3/993
4005_C088	Global Interrupt Control Low Register (PORTC_GICLR)	32	W (always reads 0)	0000_0000h	38.5.4/994
4005_C08C	Global Interrupt Control High Register (PORTC_GICHR)	32	W (always reads 0)	0000_0000h	38.5.5/994
4005_C0A0	Interrupt Status Flag Register (PORTC_ISFR)	32	w1c	0000_0000h	38.5.6/995
4005_C0C0	Digital Filter Enable Register (PORTC_DFER)	32	R/W	0000_0000h	38.5.7/995
4005_C0C4	Digital Filter Clock Register (PORTC_DFCR)	32	R/W	0000_0000h	38.5.8/996
4005_C0C8	Digital Filter Width Register (PORTC_DFWR)	32	R/W	0000_0000h	38.5.9/997
4005_D000	Pin Control Register n (PORTD_PCR0)	32	R/W	See section	38.5.1/990
4005_D004	Pin Control Register n (PORTD_PCR1)	32	R/W	See section	38.5.1/990
4005_D008	Pin Control Register n (PORTD_PCR2)	32	R/W	See section	38.5.1/990
4005_D00C	Pin Control Register n (PORTD_PCR3)	32	R/W	See section	38.5.1/990
4005_D010	Pin Control Register n (PORTD_PCR4)	32	R/W	See section	38.5.1/990
4005_D014	Pin Control Register n (PORTD_PCR5)	32	R/W	See section	38.5.1/990
4005_D018	Pin Control Register n (PORTD_PCR6)	32	R/W	See section	38.5.1/990
4005_D01C	Pin Control Register n (PORTD_PCR7)	32	R/W	See section	38.5.1/990
4005_D020	Pin Control Register n (PORTD_PCR8)	32	R/W	See section	38.5.1/990
4005_D024	Pin Control Register n (PORTD_PCR9)	32	R/W	See section	38.5.1/990
4005_D028	Pin Control Register n (PORTD_PCR10)	32	R/W	See section	38.5.1/990
4005_D02C	Pin Control Register n (PORTD_PCR11)	32	R/W	See section	38.5.1/990
4005_D030	Pin Control Register n (PORTD_PCR12)	32	R/W	See section	38.5.1/990

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_D034	Pin Control Register n (PORTD_PCR13)	32	R/W	See section	38.5.1/990
4005_D038	Pin Control Register n (PORTD_PCR14)	32	R/W	See section	38.5.1/990
4005_D03C	Pin Control Register n (PORTD_PCR15)	32	R/W	See section	38.5.1/990
4005_D040	Pin Control Register n (PORTD_PCR16)	32	R/W	See section	38.5.1/990
4005_D044	Pin Control Register n (PORTD_PCR17)	32	R/W	See section	38.5.1/990
4005_D048	Pin Control Register n (PORTD_PCR18)	32	R/W	See section	38.5.1/990
4005_D04C	Pin Control Register n (PORTD_PCR19)	32	R/W	See section	38.5.1/990
4005_D050	Pin Control Register n (PORTD_PCR20)	32	R/W	See section	38.5.1/990
4005_D054	Pin Control Register n (PORTD_PCR21)	32	R/W	See section	38.5.1/990
4005_D058	Pin Control Register n (PORTD_PCR22)	32	R/W	See section	38.5.1/990
4005_D05C	Pin Control Register n (PORTD_PCR23)	32	R/W	See section	38.5.1/990
4005_D060	Pin Control Register n (PORTD_PCR24)	32	R/W	See section	38.5.1/990
4005_D064	Pin Control Register n (PORTD_PCR25)	32	R/W	See section	38.5.1/990
4005_D068	Pin Control Register n (PORTD_PCR26)	32	R/W	See section	38.5.1/990
4005_D06C	Pin Control Register n (PORTD_PCR27)	32	R/W	See section	38.5.1/990
4005_D070	Pin Control Register n (PORTD_PCR28)	32	R/W	See section	38.5.1/990
4005_D074	Pin Control Register n (PORTD_PCR29)	32	R/W	See section	38.5.1/990
4005_D078	Pin Control Register n (PORTD_PCR30)	32	R/W	See section	38.5.1/990
4005_D07C	Pin Control Register n (PORTD_PCR31)	32	R/W	See section	38.5.1/990
4005_D080	Global Pin Control Low Register (PORTD_GPCLR)	32	W (always reads 0)	0000_0000h	38.5.2/993
4005_D084	Global Pin Control High Register (PORTD_GPCHR)	32	W (always reads 0)	0000_0000h	38.5.3/993
4005_D088	Global Interrupt Control Low Register (PORTD_GICLR)	32	W (always reads 0)	0000_0000h	38.5.4/994
4005_D08C	Global Interrupt Control High Register (PORTD_GICHR)	32	W (always reads 0)	0000_0000h	38.5.5/994
4005_D0A0	Interrupt Status Flag Register (PORTD_ISFR)	32	w1c	0000_0000h	38.5.6/995
4005_D0C0	Digital Filter Enable Register (PORTD_DFER)	32	R/W	0000_0000h	38.5.7/995
4005_D0C4	Digital Filter Clock Register (PORTD_DFCR)	32	R/W	0000_0000h	38.5.8/996
4005_D0C8	Digital Filter Width Register (PORTD_DFWR)	32	R/W	0000_0000h	38.5.9/997
4005_E000	Pin Control Register n (PORTE_PCR0)	32	R/W	See section	38.5.1/990
4005_E004	Pin Control Register n (PORTE_PCR1)	32	R/W	See section	38.5.1/990
4005_E008	Pin Control Register n (PORTE_PCR2)	32	R/W	See section	38.5.1/990
4005_E00C	Pin Control Register n (PORTE_PCR3)	32	R/W	See section	38.5.1/990
4005_E010	Pin Control Register n (PORTE_PCR4)	32	R/W	See section	38.5.1/990

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_E014	Pin Control Register n (PORTE_PCR5)	32	R/W	See section	38.5.1/990
4005_E018	Pin Control Register n (PORTE_PCR6)	32	R/W	See section	38.5.1/990
4005_E01C	Pin Control Register n (PORTE_PCR7)	32	R/W	See section	38.5.1/990
4005_E020	Pin Control Register n (PORTE_PCR8)	32	R/W	See section	38.5.1/990
4005_E024	Pin Control Register n (PORTE_PCR9)	32	R/W	See section	38.5.1/990
4005_E028	Pin Control Register n (PORTE_PCR10)	32	R/W	See section	38.5.1/990
4005_E02C	Pin Control Register n (PORTE_PCR11)	32	R/W	See section	38.5.1/990
4005_E030	Pin Control Register n (PORTE_PCR12)	32	R/W	See section	38.5.1/990
4005_E034	Pin Control Register n (PORTE_PCR13)	32	R/W	See section	38.5.1/990
4005_E038	Pin Control Register n (PORTE_PCR14)	32	R/W	See section	38.5.1/990
4005_E03C	Pin Control Register n (PORTE_PCR15)	32	R/W	See section	38.5.1/990
4005_E040	Pin Control Register n (PORTE_PCR16)	32	R/W	See section	38.5.1/990
4005_E044	Pin Control Register n (PORTE_PCR17)	32	R/W	See section	38.5.1/990
4005_E048	Pin Control Register n (PORTE_PCR18)	32	R/W	See section	38.5.1/990
4005_E04C	Pin Control Register n (PORTE_PCR19)	32	R/W	See section	38.5.1/990
4005_E050	Pin Control Register n (PORTE_PCR20)	32	R/W	See section	38.5.1/990
4005_E054	Pin Control Register n (PORTE_PCR21)	32	R/W	See section	38.5.1/990
4005_E058	Pin Control Register n (PORTE_PCR22)	32	R/W	See section	38.5.1/990
4005_E05C	Pin Control Register n (PORTE_PCR23)	32	R/W	See section	38.5.1/990
4005_E060	Pin Control Register n (PORTE_PCR24)	32	R/W	See section	38.5.1/990
4005_E064	Pin Control Register n (PORTE_PCR25)	32	R/W	See section	38.5.1/990
4005_E068	Pin Control Register n (PORTE_PCR26)	32	R/W	See section	38.5.1/990
4005_E06C	Pin Control Register n (PORTE_PCR27)	32	R/W	See section	38.5.1/990
4005_E070	Pin Control Register n (PORTE_PCR28)	32	R/W	See section	38.5.1/990
4005_E074	Pin Control Register n (PORTE_PCR29)	32	R/W	See section	38.5.1/990
4005_E078	Pin Control Register n (PORTE_PCR30)	32	R/W	See section	38.5.1/990
4005_E07C	Pin Control Register n (PORTE_PCR31)	32	R/W	See section	38.5.1/990
4005_E080	Global Pin Control Low Register (PORTE_GPCLR)	32	W (always reads 0)	0000_0000h	38.5.2/993
4005_E084	Global Pin Control High Register (PORTE_GPCHR)	32	W (always reads 0)	0000_0000h	38.5.3/993
4005_E088	Global Interrupt Control Low Register (PORTE_GICLR)	32	W (always reads 0)	0000_0000h	38.5.4/994
4005_E08C	Global Interrupt Control High Register (PORTE_GICHR)	32	W (always reads 0)	0000_0000h	38.5.5/994
4005_E0A0	Interrupt Status Flag Register (PORTE_ISFR)	32	w1c	0000_0000h	38.5.6/995

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_E0C0	Digital Filter Enable Register (PORTE_DFER)	32	R/W	0000_0000h	38.5.7/995
4005_E0C4	Digital Filter Clock Register (PORTE_DFCR)	32	R/W	0000_0000h	38.5.8/996
4005_E0C8	Digital Filter Width Register (PORTE_DFWR)	32	R/W	0000_0000h	38.5.9/997

38.5.1 Pin Control Register n (PORTx_PCR*n*)

NOTE

See the Signal Multiplexing and Pin Assignment chapter for the reset value of this device.

See the GPIO Configuration section for details on the available functions for each pin.

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 0h offset + (4d × i), where i=0d to 31d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							ISF	0				IRQC			
W								w1c								
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
R	LK	0				MUX			0	DSE	ODE	PFE	0	SRE	PE	PS
W																
Reset	0	0	0	0	0	*	*	*	0	*	0	*	0	*	*	*

* Notes:
- MUX field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- DSE field: Varies by port. See the Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PFE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- SRE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PS field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.

PORTx_PCR*n* field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag The pin interrupt configuration is valid in all digital pin muxing modes. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 IRQC	Interrupt Configuration The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows: 0000 Interrupt Status Flag (ISF) is disabled. 0001 ISF flag and DMA request on rising edge. 0010 ISF flag and DMA request on falling edge. 0011 ISF flag and DMA request on either edge. 0100 Reserved. 0101 Flag sets on rising edge. 0110 Flag sets on falling edge. 0111 Flag sets on either edge. 1000 ISF flag and Interrupt when logic 0. 1001 ISF flag and Interrupt on rising-edge. 1010 ISF flag and Interrupt on falling-edge. 1011 ISF flag and Interrupt on either edge. 1100 ISF flag and Interrupt when logic 1. 1101 Enable active high trigger output, flag is disabled. [The trigger output goes to the trigger mux, which allows pins to trigger other peripherals (configurable polarity; 1 pin per port; if multiple pins are configured, then they are ORed together to create the trigger)] 1110 Enable active low trigger output, flag is disabled. 1111 Reserved.
15 LK	Lock Register 0 Pin Control Register fields [15:0] are not locked. 1 Pin Control Register fields [15:0] are locked and cannot be updated until the next system reset.
14–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 MUX	Pin Mux Control Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot. The corresponding pin is configured in the following pin muxing slot as follows: 000 Pin disabled (Alternative 0) (analog).

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	001 Alternative 1 (GPIO). 010 Alternative 2 (chip-specific). 011 Alternative 3 (chip-specific). 100 Alternative 4 (chip-specific). 101 Alternative 5 (chip-specific). 110 Alternative 6 (chip-specific). 111 Alternative 7 (chip-specific).
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DSE	Drive Strength Enable This field is read-only for pins that do not support a configurable drive strength. Drive strength configuration is valid in all digital pin muxing modes. 0 Low drive strength is configured on the corresponding pin, if pin is configured as a digital output. 1 High drive strength is configured on the corresponding pin, if pin is configured as a digital output.
5 ODE	Open Drain Enable This field is read-only for pins that do not support a configurable open drain output. Open drain configuration is valid in all digital pin muxing modes. 0 Open drain output is disabled on the corresponding pin. 1 Open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
4 PFE	Passive Filter Enable This field is read-only for pins that do not support a configurable passive input filter. Passive filter configuration is valid in all digital pin muxing modes. 0 Passive input filter is disabled on the corresponding pin. 1 Passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input. Refer to the device data sheet for filter characteristics.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SRE	Slew Rate Enable This field is read-only for pins that do not support a configurable slew rate. Slew rate configuration is valid in all digital pin muxing modes. 0 Fast slew rate is configured on the corresponding pin, if the pin is configured as a digital output. 1 Slow slew rate is configured on the corresponding pin, if the pin is configured as a digital output.
1 PE	Pull Enable This field is read-only for pins that do not support a configurable pull resistor. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support a configurable pull resistor. Pull configuration is valid in all digital pin muxing modes. 0 Internal pullup or pulldown resistor is not enabled on the corresponding pin. 1 Internal pullup or pulldown resistor is enabled on the corresponding pin, if the pin is configured as a digital input.

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
0 PS	Pull Select This bit is read only for pins that do not support a configurable pull resistor direction. Pull configuration is valid in all digital pin muxing modes. 0 Internal pulldown resistor is enabled on the corresponding pin, if the corresponding PE field is set. 1 Internal pullup resistor is enabled on the corresponding pin, if the corresponding PE field is set.

38.5.2 Global Pin Control Low Register (PORTx_GPCLR)

Only 32-bit writes are supported to this register.

Address: Base address + 80h offset

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

PORTx_GPCLR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (15 through 0) bits [15:0] update with the value in GPWD. If a selected Pin Control Register is locked then the write to that register is ignored. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.
GPWD	Global Pin Write Data Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.

38.5.3 Global Pin Control High Register (PORTx_GPCHR)

Only 32-bit writes are supported to this register.

Address: Base address + 84h offset

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

PORTx_GPCHR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (31 through 16) bits [15:0] update with the value in GPWD. If a selected Pin Control Register is locked then the write to that register is ignored. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.
GPWD	Global Pin Write Data Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.

38.5.4 Global Interrupt Control Low Register (PORTx_GICLR)

Only 32-bit writes are supported to this register.

Address: Base address + 88h offset

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

PORTx_GICLR field descriptions

Field	Description
31–16 GIWD	Global Interrupt Write Data Write value that is written to all Pin Control Registers bits [31:16] that are selected by GIWE.
GIWE	Global Interrupt Write Enable Selects which Pin Control Registers (15 through 0) bits [31:16] update with the value in GIWD. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.

38.5.5 Global Interrupt Control High Register (PORTx_GICHR)

Only 32-bit writes are supported to this register.

Address: Base address + 8Ch offset

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

PORTx_GICHR field descriptions

Field	Description
31–16 GIWD	Global Interrupt Write Data Write value that is written to all Pin Control Registers bits [31:16] that are selected by GIWE.
GIWE	Global Interrupt Write Enable Selects which Pin Control Registers (31 through 16) bits [31:16] update with the value in GIWD. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.

38.5.6 Interrupt Status Flag Register (PORTx_ISFR)

The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Pin Control Register, and each flag can be cleared in either location.

Address: Base address + A0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISF																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_ISFR field descriptions

Field	Description
ISF	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt of the same number as the field. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

38.5.7 Digital Filter Enable Register (PORTx_DFER)

The corresponding bit is read only for pins that do not support a digital filter. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support digital filter.

The digital filter configuration is valid in all digital pin muxing modes.

Address: Base address + C0h offset

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

PORTx_DFER field descriptions

Field	Description
DFE	Digital Filter Enable The digital filter configuration is valid in all digital pin muxing modes. The output of each digital filter is reset to zero at system reset and whenever the digital filter is disabled. Each bit in the field enables the digital filter of the same number as the field. 0 Digital filter is disabled on the corresponding pin and output of the digital filter is reset to zero. 1 Digital filter is enabled on the corresponding pin, if the pin is configured as a digital input.

38.5.8 Digital Filter Clock Register (PORTx_DFCR)

This register is read only for ports that do not support a digital filter.

The digital filter configuration is valid in all digital pin muxing modes.

Address: Base address + C4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0															CS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_DFCR field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CS	Clock Source The digital filter configuration is valid in all digital pin muxing modes. Configures the clock source for the digital input filters. Changing the filter clock source must be done only when all digital filters are disabled. 0 Digital filters are clocked by the bus clock. 1 Digital filters are clocked by the LPO clock.

38.5.9 Digital Filter Width Register (PORTx_DFWR)

This register is read only for ports that do not support a digital filter.

The digital filter configuration is valid in all digital pin muxing modes.

Address: Base address + C8h offset

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

PORTx_DFWR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FILT	Filter Length The digital filter configuration is valid in all digital pin muxing modes. Configures the maximum size of the glitches, in clock cycles, that the digital filter absorbs for the enabled digital filters. Glitches that are longer than this register setting will pass through the digital filter, and glitches that are equal to or less than this register setting are filtered. Changing the filter length must be done only after all filters are disabled.

38.6 Functional description

38.6.1 Pin control

Each port pin has a corresponding Pin Control register, PORT_PCRn, associated with it.

The upper half of the Pin Control register configures the pin's capability to either interrupt the CPU or request a DMA transfer, on a rising/falling edge or both edges as well as a logic level occurring on the port pin. It also includes a flag to indicate that an interrupt has occurred.

The lower half of the Pin Control register configures the following functions for each pin within the 32-bit port.

- Pullup or pulldown enable on selected pins
- Drive strength and slew rate configuration on selected pins
- Open drain enable on selected pins
- Passive input filter enable on selected pins
- Pin Muxing mode

The functions apply across all digital pin muxing modes and individual peripherals do not override the configuration in the Pin Control register. For example, if an I^2C function is enabled on a pin, that does not override the pullup or open drain configuration for that pin.

When the Pin Muxing mode is configured for analog or is disabled, all the digital functions on that pin are disabled. This includes the pullup and pulldown enables, output buffer enable, input buffer enable, and passive filter enable.

The LK bit (bit 15 of Pin Control Register PCR*n*) allows the configuration for each pin to be locked until the next system reset. When locked, writes to the lower half of that pin control register are ignored, although a bus error is not generated on an attempted write to a locked register.

The configuration of each Pin Control register is retained when the PORT module is disabled.

Whenever a pin is configured in any digital pin muxing mode, the input buffer for that pin is enabled allowing the pin state to be read via the corresponding GPIO Port Data Input Register (GPIO_PDIR) or allowing a pin interrupt or DMA request to be generated. If a pin is ever floating when its input buffer is enabled, then this can cause an increase in power consumption and must be avoided. A pin can be floating due to an input pin that is not connected or an output pin that has tri-stated (output buffer is disabled).

Enabling the internal pull resistor (or implementing an external pull resistor) will ensure a pin does not float when its input buffer is enabled; note that the internal pull resistor is automatically disabled whenever the output buffer is enabled allowing the Pull Enable bit to remain set. Configuring the Pin Muxing mode to disabled or analog will disable the pin's input buffer and results in the lowest power consumption.

38.6.2 Global pin control

The two global pin control registers allow a single register write to update the lower half of the pin control register on up to 16 pins, all with the same value. Registers that are locked cannot be written using the global pin control registers.

The global pin control registers are designed to enable software to quickly configure multiple pins within the one port for the same peripheral function. However, the interrupt functions cannot be configured using the global pin control registers.

The global pin control registers are write-only registers, that always read as 0.

38.6.3 Global interrupt control

The two global interrupt control registers allow a single register write to update the upper half of the pin control register on up to 16 pins, all with the same value.

The global interrupt control registers are designed to enable software to quickly configure multiple pins within the one port for the same interrupt configuration. However, the pin control functions cannot be configured using the global interrupt control registers.

The global interrupt control registers are write-only registers and always read as 0.

38.6.4 External interrupts

The external interrupt capability of the PORT module is available in all digital pin muxing modes provided the PORT module is enabled.

Each pin can be individually configured for any of the following external interrupt modes:

- Interrupt disabled, default out of reset
- Rising edge flag (for software polling)
- Falling edge flag (for software polling)
- Rising and falling edge flag (for software polling)
- Active high level peripheral trigger (status flag disabled)
- Active low level peripheral trigger (status flag disabled)
- Active high level sensitive interrupt
- Active low level sensitive interrupt
- Rising edge sensitive interrupt
- Falling edge sensitive interrupt
- Rising and falling edge sensitive interrupt
- Rising edge sensitive DMA request
- Falling edge sensitive DMA request
- Rising and falling edge sensitive DMA request

The interrupt status flag is set when the configured edge or level is detected on the pin or at the output of the digital input filter, if the digital input digital filter is enabled. When not in Stop mode, the input is first synchronized to the bus clock to detect the configured level or edge transition.

The PORT module generates a single interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that port. The interrupt negates after the interrupt status flags for all enabled interrupts have been cleared by writing a logic 1 to the ISF flag in either the PORT_ISFR or PORT_PCRn registers.

The PORT module generates a single DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that port. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

During Stop mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wake-up signal to exit the Low-Power mode.

The PORT module generates a single peripheral trigger output that asserts if any pin configured for active high trigger is logic one, or any pin triggered for active low trigger is logic zero. The peripheral trigger output asynchronously updates from the value on the configured pins or at the output of the digital input filter, if the digital input digital filter is enabled.

38.6.5 Digital filter

The digital filter capabilities of the PORT module are available in all digital Pin Muxing modes if the PORT module is enabled.

The clock used for all digital filters within one port can be configured between the bus clock or the LPO clock. This selection must be changed only when all digital filters for that port are disabled. If the digital filters for a port are configured to use the bus clock, then the digital filters are bypassed for the duration of Stop mode. While the digital filters are bypassed, the output of each digital filter always equals the input pin, but the internal state of the digital filters remains static and does not update due to any change on the input pin.

The filter width in clock size is the same for all enabled digital filters within one port and must be changed only when all digital filters for that port are disabled.

The output of each digital filter is logic zero after system reset and whenever a digital filter is disabled. After a digital filter is enabled, the input is synchronized to the filter clock, either the bus clock or the LPO clock. If the synchronized input and the output of the digital filter remain different for a number of filter clock cycles equal to the filter width register configuration, then the output of the digital filter updates to equal the synchronized filter input.

The maximum latency through a digital filter equals three filter clock cycles plus the filter width configuration register.

Chapter 39
Reset Control Module (RCM)

39.1 Introduction

Information found here describes the registers of the Reset Control Module (RCM). The RCM implements many of the reset functions for the chip. See the chip's reset chapter for more information.

See AN4503: Power Management for Kinetis and ColdFire+ MCUs for further details on using the RCM.

39.2 Reset memory map and register descriptions

The RCM Memory Map/Register Definition can be found here.

The Reset Control Module (RCM) registers provide reset status information and reset filter control.

NOTE

The RCM registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

RCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_F000	Version ID Register (RCM_VERID)	32	R	0300_0003h	39.2.1/1002
4007_F004	Parameter Register (RCM_PARAM)	32	R	See section	39.2.2/1003
4007_F008	System Reset Status Register (RCM_SRS)	32	R	0000_0082h	39.2.3/1005
4007_F00C	Reset Pin Control register (RCM_RPC)	32	R/W	0000_0000h	39.2.4/1008

Table continues on the next page...

RCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_F010	Mode Register (RCM_MR)	32	R/W	See section	39.2.5/1010
4007_F014	Force Mode Register (RCM_FM)	32	R/W	0000_0000h	39.2.6/1011
4007_F018	Sticky System Reset Status Register (RCM_SSRS)	32	R/W	0000_0082h	39.2.7/1012
4007_F01C	System Reset Interrupt Enable Register (RCM_SRIE)	32	R/W	0000_0000h	39.2.8/1015

39.2.1 Version ID Register (RCM_VERID)

Address: 4007_F000h base + 0h offset = 4007_F000h

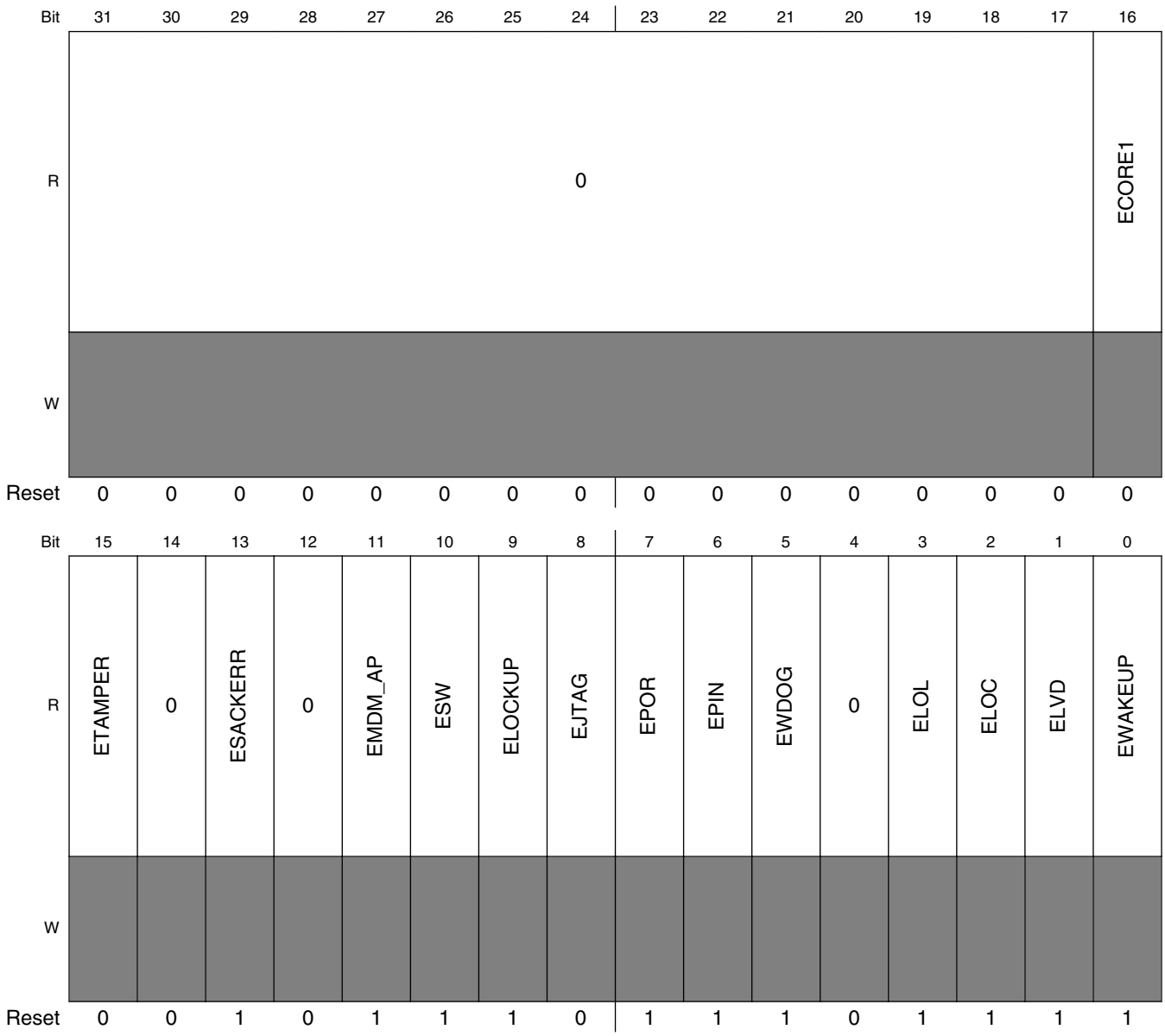
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

RCM_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0003 Standard feature set.

39.2.2 Parameter Register (RCM_PARAM)

Address: 4007_F000h base + 4h offset = 4007_F004h



RCM_PARAM field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ECORE1	Existence of SRS[CORE1] status indication feature

Table continues on the next page...

RCM_PARAM field descriptions (continued)

Field	Description
	This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
15 ETAMPER	Existence of SRS[TAMPER] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 ESACKERR	Existence of SRS[SACKERR] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 EMDM_AP	Existence of SRS[MDM_AP] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
10 ESW	Existence of SRS[SW] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
9 ELOCKUP	Existence of SRS[LOCKUP] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
8 EJTAG	Existence of SRS[JTAG] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
7 EPOR	Existence of SRS[POR] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

Table continues on the next page...

RCM_PARAM field descriptions (continued)

Field	Description
6 EPIN	Existence of SRS[PIN] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
5 EWDOG	Existence of SRS[WDOG] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ELOL	Existence of SRS[LOL] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
2 ELOC	Existence of SRS[LOC] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
1 ELVD	Existence of SRS[LVD] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
0 EWAKEUP	Existence of SRS[WAKEUP] status indication feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

39.2.3 System Reset Status Register (RCM_SRS)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x82
- LVD (without POR) — 0x02

- VLLS mode wakeup due to $\overline{\text{RESET}}$ pin assertion — 0x41
- VLLS mode wakeup due to other wakeup sources — 0x01
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007_F000h base + 8h offset = 4007_F008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															0
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
R	0	0	SACKERR	0	MDM_AP	SW	LOCKUP	0	POR	PIN	WDOG	0	LOL	LOC	LVD	WAKEUP
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0

RCM_SRS field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SACKERR	Stop Acknowledge Error Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode.

Table continues on the next page...

RCM_SRS field descriptions (continued)

Field	Description
	0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 MDM_AP	MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset was not caused by host debugger system setting of the System Reset Request bit 1 Reset was caused by host debugger system setting of the System Reset Request bit
10 SW	Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
9 LOCKUP	Core Lockup Indicates a reset has been caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 POR	Power-On Reset Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 PIN	External Reset Pin Indicates a reset has been caused by an active-low level on the external $\overline{\text{RESET}}$ (RESET_b) pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog Indicates a reset has been caused by the watchdog timer timing out. This reset source can be blocked by disabling the watchdog. 0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LOL	Loss-of-Lock Reset

Table continues on the next page...

RCM_SRS field descriptions (continued)

Field	Description
	Indicates a reset has been caused by a loss of lock in the SCG PLL/FLL. 0 Reset not caused by a loss of lock in the PLL/FLL 1 Reset caused by a loss of lock in the PLL/FLL
2 LOC	Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The SCG SOSC clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed SCG description for information on enabling the clock monitor. 0 Reset not caused by a loss of external clock. 1 Reset caused by a loss of external clock.
1 LVD	Low-Voltage Detect Reset or High-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. If PMC_HVDSC1[HVDRE] is set and the supply rises above the HVD trip voltage, an HVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip, HVD trip or POR 1 Reset caused by LVD trip, HVD trip or POR
0 WAKEUP	VLLS Wakeup Reset Indicates a reset has been caused by a wakeup from VLLS mode. 0 Reset not caused by wakeup from VLLS mode. 1 Reset caused by wakeup from VLLS mode.

39.2.4 Reset Pin Control register (RCM_RPC)

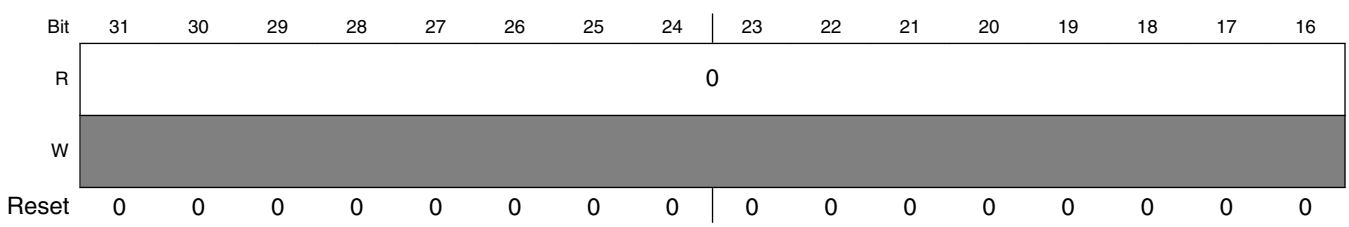
NOTE

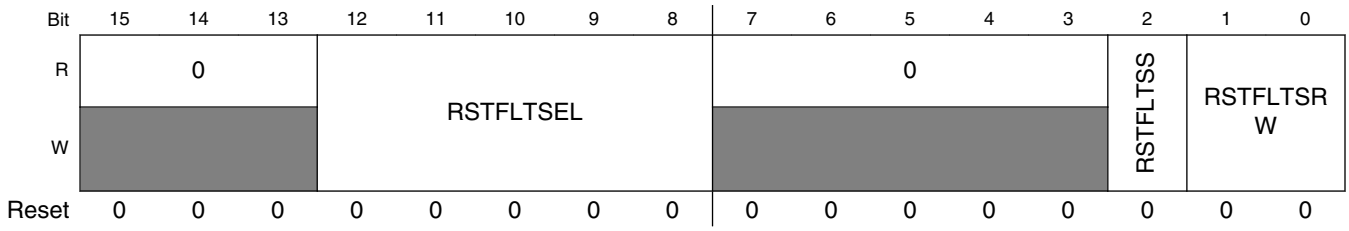
This register is reset on Chip POR only, it is unaffected by other reset types.

NOTE

The bus clock filter is reset when disabled or when entering stop mode. The LPO filter is reset when disabled.

Address: 4007_F000h base + Ch offset = 4007_F00Ch





RCM_RPC field descriptions

Field	Description
31–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 RSTFLTSEL	Reset Pin Filter Bus Clock Select Selects the reset pin bus clock filter width: • Transition lengths less than RSTFLTSEL cycles are filtered. • Transition lengths between RSTFLTSEL and (RSTFLTSEL+1) cycles (inclusive) may be filtered. • Transition lengths greater than (RSTFLTSEL+1) cycles are not filtered.
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RSTFLTSS	Reset Pin Filter Select in Stop Mode Selects how the reset pin filter is enabled in any stop mode. On exit from VLLS mode, this bit should be reconfigured before clearing PMC_REGSC[ACKISO]. 0 All filtering disabled 1 LPO clock filter enabled
RSTFLTSRW	Reset Pin Filter Select in Run and Wait Modes Selects how the reset pin filter is enabled in run and wait modes. 00 All filtering disabled 01 Bus clock filter enabled for normal operation 10 LPO clock filter enabled for normal operation 11 Reserved

39.2.5 Mode Register (RCM_MR)

This register includes status flags to indicate the state of the mode pins during the last Chip Reset.

Address: 4007_F000h base + 10h offset = 4007_F010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0													BOOTROM		0
W														w1c		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	*	*	0

* Notes:
- BOOTROM field: The reset state of this register depends on the boot mode.

RCM_MR field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2–1 BOOTROM	Boot ROM Configuration Indicates the boot source, the boot source remains set until the next System Reset or software can write logic one to clear the corresponding mode bit. While either bit is set, the NMI input is disabled and the vector table is relocated to the ROM base address at $1C00_0000. These bits should be cleared by writing logic one before executing any code from either Flash or SRAM. 00 Boot from Flash 01 Boot from ROM due to BOOTCFG0 pin assertion / Reserved if no Boot pin 10 Boot form ROM due to FOPT[7] configuration 11 Boot from ROM due to both BOOTCFG0 pin assertion and FOPT[7] configuration
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

39.2.6 Force Mode Register (RCM_FM)

NOTE

The reset values of the bits in the FORCEROM field are for Chip POR only. They are unaffected by other reset types.

Address: 4007_F000h base + 14h offset = 4007_F014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0													FORCEROM		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

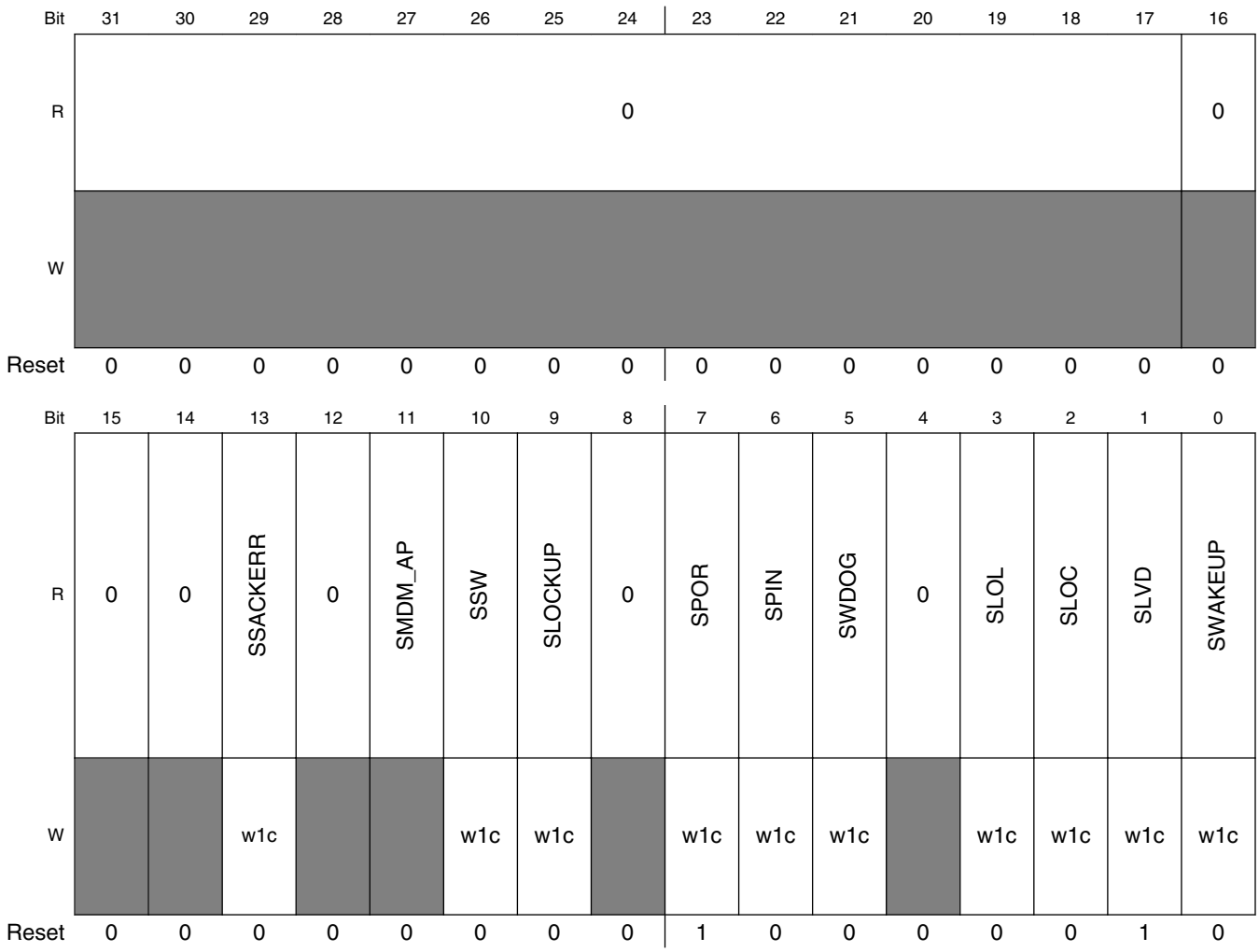
RCM_FM field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2–1 FORCEROM	Force ROM Boot When either bit is set, will force boot from ROM during all subsequent system resets. 00 No effect 01 Force boot from ROM with RCM_MR[1] set. 10 Force boot from ROM with RCM_MR[2] set. 11 Force boot from ROM with RCM_MR[2:1] set.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

39.2.7 Sticky System Reset Status Register (RCM_SSRS)

This register includes status flags to indicate all reset sources since the last POR or LVD, or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_F000h base + 18h offset = 4007_F018h



RCM_SSRS field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

RCM_SSRS field descriptions (continued)

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SSACKERR	Sticky Stop Acknowledge Error Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 SMDM_AP	Sticky MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset was not caused by host debugger system setting of the System Reset Request bit 1 Reset was caused by host debugger system setting of the System Reset Request bit
10 SSW	Sticky Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
9 SLOCKUP	Sticky Core Lockup Indicates a reset has been caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 SPOR	Sticky Power-On Reset Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 SPIN	Sticky External Reset Pin Indicates a reset has been caused by an active-low level on the external $\overline{\text{RESET}}$ (RESET_b) pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 SWDOG	Sticky Watchdog

Table continues on the next page...

RCM_SSRS field descriptions (continued)

Field	Description
	Indicates a reset has been caused by the watchdog timer timing out. This reset source can be blocked by disabling the watchdog. 0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SLOL	Sticky Loss-of-Lock Reset Indicates a reset has been caused by a loss of lock in the SCG PLL/FLL. See the SCG description for information on the loss-of-lock event. 0 Reset not caused by a loss of lock in the PLL/FLL 1 Reset caused by a loss of lock in the PLL/FLL
2 SLOC	Sticky Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The SCG SOSC clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed SCG description for information on enabling the clock monitor. 0 Reset not caused by a loss of external clock. 1 Reset caused by a loss of external clock.
1 SLVD	Sticky Low-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR
0 SWAKEUP	Sticky VLLS Wakeup Reset Indicates a reset has been caused by a wakeup from VLLS mode. 0 Reset not caused by wakeup from VLLS mode. 1 Reset caused by wakeup from VLLS mode.

39.2.8 System Reset Interrupt Enable Register (RCM_SRIE)

This register provides the option to delay the assertion of a system reset for a period of time (DELAY field) while an interrupt is generated. When an interrupt for a reset source is enabled, software has time to perform a graceful shutdown. A Chip POR source cannot be delayed by this feature, and entering LLS/VLLS mode terminates the delay. The SRS updates only after the system reset occurs.

Address: 4007_F000h base + 1Ch offset = 4007_F01Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															0
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
R	0	0	SACKERR	0	MDM_AP	SW	LOCKUP	0	GIE	PIN	WDOG	0	LOL	LOC	DELAY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RCM_SRIE field descriptions

Field	Description
31–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SACKERR	Stop Acknowledge Error Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 MDM_AP	MDM-AP System Reset Request 0 Interrupt disabled. 1 Interrupt enabled.

Table continues on the next page...

RCM_SRIE field descriptions (continued)

Field	Description
10 SW	Software Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
9 LOCKUP	Core Lockup Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 GIE	Global Interrupt Enable 0 All interrupt sources disabled. 1 All interrupt sources enabled.
6 PIN	External Reset Pin Interrupt 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LOL	Loss-of-Lock Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
2 LOC	Loss-of-Clock Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
DELAY	Reset Delay Time Configures the maximum reset delay time from when the interrupt is asserted and the system reset occurs. 00 10 LPO cycles 01 34 LPO cycles 10 130 LPO cycles 11 514 LPO cycles

Chapter 40
General-Purpose Input/Output (GPIO)

40.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The general-purpose input and output (GPIO) module is accessible via the peripheral bus and also communicates to the processor core via a zero wait state interface (IOPORT) for maximum pin performance. The GPIO registers support 8-bit, 16-bit or 32-bit accesses.

The GPIO data direction and output data registers control the direction and output data of each pin when the pin is configured for the GPIO function. The GPIO input data register displays the logic value on each pin when the pin is configured for any digital function, provided the corresponding Port Control and Interrupt module for that pin is enabled.

Efficient bit manipulation of the general-purpose outputs is supported through the addition of set, clear, and toggle write-only registers for each port output data register.

40.1.1 Features

Features of the GPIO module include:

- Port Data Input register visible in all digital pin-multiplexing modes
- Port Data Output register with corresponding set/clear/toggle registers
- Port Data Direction register
- Zero wait state access to GPIO registers through IOPORT

NOTE

The GPIO module is clocked by system clock.

40.1.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 40-1. Modes of operation

Modes of operation	Description
Run	The GPIO module operates normally.
Wait	The GPIO module operates normally.
Stop	The GPIO module is disabled.
Debug	The GPIO module operates normally.

40.1.3 GPIO signal descriptions

Table 40-2. GPIO signal descriptions

GPIO signal descriptions	Description	I/O
PORTA31–PORTA0	General-purpose input/output	I/O
PORTB31–PORTB0	General-purpose input/output	I/O
PORTC31–PORTC0	General-purpose input/output	I/O
PORTD31–PORTD0	General-purpose input/output	I/O
PORTE31–PORTE0	General-purpose input/output	I/O

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

40.1.3.1 Detailed signal description

Table 40-3. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
PORTA31–PORTA0	I/O	General-purpose input/output	
PORTB31–PORTB0 PORTC31–PORTC0		State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.
PORTD31–PORTD0 PORTE31–PORTE0		Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time

Table 40-3. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
			and input may be asserted asynchronously to the system clock. Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

40.2 Memory map and register definition

Any read or write access to the GPIO memory space that is outside the valid memory map results in a bus error.

GPIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_F000	Port Data Output Register (GPIOA_PDOR)	32	R/W	0000_0000h	40.2.1/1021
4000_F004	Port Set Output Register (GPIOA_PSOR)	32	W (always reads 0)	0000_0000h	40.2.2/1021
4000_F008	Port Clear Output Register (GPIOA_PCOR)	32	W (always reads 0)	0000_0000h	40.2.3/1022
4000_F00C	Port Toggle Output Register (GPIOA_PTOR)	32	W (always reads 0)	0000_0000h	40.2.4/1022
4000_F010	Port Data Input Register (GPIOA_PDIR)	32	R	0000_0000h	40.2.5/1023
4000_F014	Port Data Direction Register (GPIOA_PDDR)	32	R/W	0000_0000h	40.2.6/1023
4000_F040	Port Data Output Register (GPIOB_PDOR)	32	R/W	0000_0000h	40.2.1/1021
4000_F044	Port Set Output Register (GPIOB_PSOR)	32	W (always reads 0)	0000_0000h	40.2.2/1021

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_F048	Port Clear Output Register (GPIOB_PCOR)	32	W (always reads 0)	0000_0000h	40.2.3/1022
4000_F04C	Port Toggle Output Register (GPIOB_PTOR)	32	W (always reads 0)	0000_0000h	40.2.4/1022
4000_F050	Port Data Input Register (GPIOB_PDIR)	32	R	0000_0000h	40.2.5/1023
4000_F054	Port Data Direction Register (GPIOB_PDDR)	32	R/W	0000_0000h	40.2.6/1023
4000_F080	Port Data Output Register (GPIOC_PDOR)	32	R/W	0000_0000h	40.2.1/1021
4000_F084	Port Set Output Register (GPIOC_PSOR)	32	W (always reads 0)	0000_0000h	40.2.2/1021
4000_F088	Port Clear Output Register (GPIOC_PCOR)	32	W (always reads 0)	0000_0000h	40.2.3/1022
4000_F08C	Port Toggle Output Register (GPIOC_PTOR)	32	W (always reads 0)	0000_0000h	40.2.4/1022
4000_F090	Port Data Input Register (GPIOC_PDIR)	32	R	0000_0000h	40.2.5/1023
4000_F094	Port Data Direction Register (GPIOC_PDDR)	32	R/W	0000_0000h	40.2.6/1023
4000_F0C0	Port Data Output Register (GPIOD_PDOR)	32	R/W	0000_0000h	40.2.1/1021
4000_F0C4	Port Set Output Register (GPIOD_PSOR)	32	W (always reads 0)	0000_0000h	40.2.2/1021
4000_F0C8	Port Clear Output Register (GPIOD_PCOR)	32	W (always reads 0)	0000_0000h	40.2.3/1022
4000_F0CC	Port Toggle Output Register (GPIOD_PTOR)	32	W (always reads 0)	0000_0000h	40.2.4/1022
4000_F0D0	Port Data Input Register (GPIOD_PDIR)	32	R	0000_0000h	40.2.5/1023
4000_F0D4	Port Data Direction Register (GPIOD_PDDR)	32	R/W	0000_0000h	40.2.6/1023
4000_F100	Port Data Output Register (GPIOE_PDOR)	32	R/W	0000_0000h	40.2.1/1021
4000_F104	Port Set Output Register (GPIOE_PSOR)	32	W (always reads 0)	0000_0000h	40.2.2/1021
4000_F108	Port Clear Output Register (GPIOE_PCOR)	32	W (always reads 0)	0000_0000h	40.2.3/1022
4000_F10C	Port Toggle Output Register (GPIOE_PTOR)	32	W (always reads 0)	0000_0000h	40.2.4/1022
4000_F110	Port Data Input Register (GPIOE_PDIR)	32	R	0000_0000h	40.2.5/1023
4000_F114	Port Data Direction Register (GPIOE_PDDR)	32	R/W	0000_0000h	40.2.6/1023

40.2.1 Port Data Output Register (GPIOx_PDOR)

This register configures the logic levels that are driven on each general-purpose output pins.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 0h offset

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

GPIOx_PDOR field descriptions

Field	Description
PDO	Port Data Output Register bits for unbonded pins return a undefined value when read. 0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

40.2.2 Port Set Output Register (GPIOx_PSOR)

This register configures whether to set the fields of the PDOR.

Address: Base address + 4h offset

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

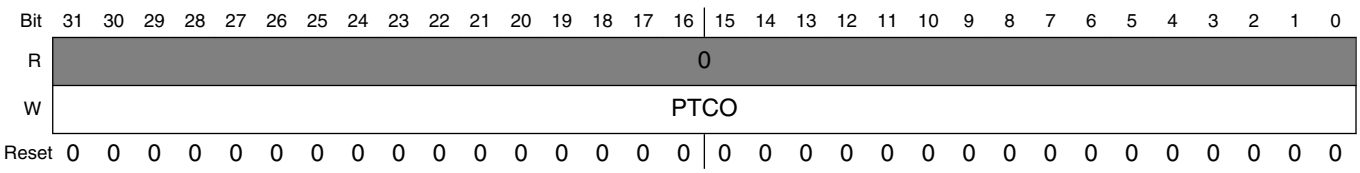
GPIOx_PSOR field descriptions

Field	Description
PTSO	Port Set Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic 1.

40.2.3 Port Clear Output Register (GPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

Address: Base address + 8h offset

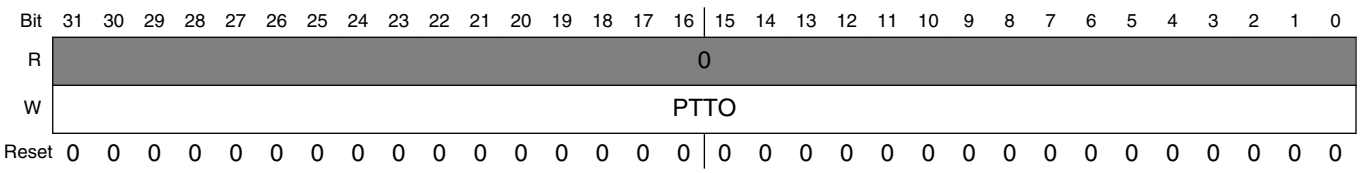


GPIOx_PCOR field descriptions

Field	Description
PTCO	Port Clear Output Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0.

40.2.4 Port Toggle Output Register (GPIOx_PTOR)

Address: Base address + Ch offset



GPIOx_PTOR field descriptions

Field	Description
PTTO	Port Toggle Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

40.2.5 Port Data Input Register (GPIOx_PDIR)

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 10h offset

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

GPIOx_PDIR field descriptions

Field	Description
PDI	Port Data Input Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update. 0 Pin logic level is logic 0, or is not configured for use by digital function. 1 Pin logic level is logic 1.

40.2.6 Port Data Direction Register (GPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Address: Base address + 14h offset

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

GPIOx_PDDR field descriptions

Field	Description
PDD	Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

GPIOx_PDDR field descriptions (continued)

Field	Description

40.3 FGPIO memory map and register definition

The GPIO registers are also aliased to the IOPORT interface on the Cortex-M0+ from address 0xF800_0000.

Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle. This aliased Fast GPIO memory map is called FGPIO.

Any read or write access to the FGPIO memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states, except error accesses which complete with one wait state.

FGPIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F800_0000	Port Data Output Register (FGPIOA_PDOR)	32	R/W	0000_0000h	40.3.1/1024
F800_0004	Port Set Output Register (FGPIOA_PSOR)	32	W (always reads 0)	0000_0000h	40.3.2/1025
F800_0008	Port Clear Output Register (FGPIOA_PCOR)	32	W (always reads 0)	0000_0000h	40.3.3/1025
F800_000C	Port Toggle Output Register (FGPIOA_PTOR)	32	W (always reads 0)	0000_0000h	40.3.4/1026
F800_0010	Port Data Input Register (FGPIOA_PDIR)	32	R	0000_0000h	40.3.5/1026
F800_0014	Port Data Direction Register (FGPIOA_PDDR)	32	R/W	0000_0000h	40.3.6/1027

40.3.1 Port Data Output Register (FGPIOx_PDOR)

This register configures the logic levels that are driven on each general-purpose output pins.

Address: F800_0000h base + 0h offset = F800_0000h

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

FGPIOx_PDOR field descriptions

Field	Description
PDO	Port Data Output Unimplemented pins for a particular device read as zero. 0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

40.3.2 Port Set Output Register (FGPIOx_PSOR)

This register configures whether to set the fields of the PDOR.

Address: F800_0000h base + 4h offset = F800_0004h

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

FGPIOx_PSOR field descriptions

Field	Description
PTSO	Port Set Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic 1.

40.3.3 Port Clear Output Register (FGPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

Address: F800_0000h base + 8h offset = F800_0008h

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

FGPIOx_PCOR field descriptions

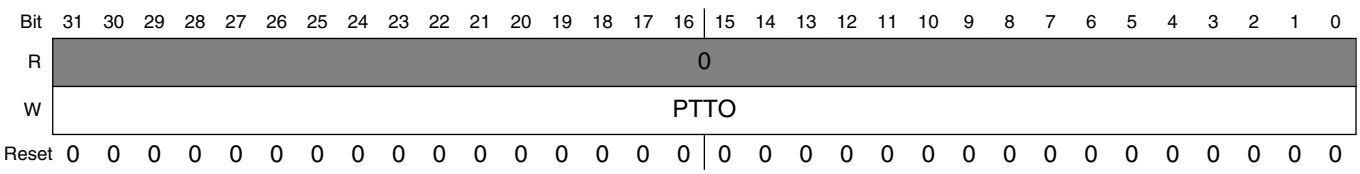
Field	Description
PTCO	Port Clear Output

FGPIOx_PCOR field descriptions (continued)

Field	Description
	Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0.

40.3.4 Port Toggle Output Register (FGPIOx_PTOR)

Address: F800_0000h base + Ch offset = F800_000Ch

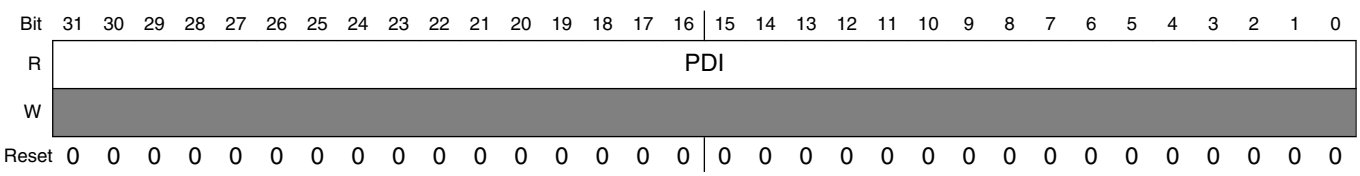


FGPIOx_PTOR field descriptions

Field	Description
PTTO	Port Toggle Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

40.3.5 Port Data Input Register (FGPIOx_PDIR)

Address: F800_0000h base + 10h offset = F800_0010h



FGPIOx_PDIR field descriptions

Field	Description
PDI	Port Data Input Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update. 0 Pin logic level is logic 0, or is not configured for use by digital function. 1 Pin logic level is logic 1.

40.3.6 Port Data Direction Register (FGPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Address: F800_0000h base + 14h offset = F800_0014h

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

FGPIOx_PDDR field descriptions

Field	Description
PDD	Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

40.4 Functional description

40.4.1 General-purpose input

The logic state of each pin is available via the Port Data Input registers, provided the pin is configured for a digital function and the corresponding Port Control and Interrupt module is enabled.

40.4.2 General-purpose output

The logic state of each pin can be controlled via the port data output registers and port data direction registers, provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

If	Then
A pin is configured for the GPIO function and the corresponding port data direction register bit is clear.	The pin is configured as an input.
A pin is configured for the GPIO function and the corresponding port data direction register bit is set.	The pin is configured as an output and and the logic state of the pin is equal to the corresponding port data output register.

To facilitate efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers exist to allow one or more outputs within one port to be set, cleared, or toggled from a single register write.

The corresponding Port Control and Interrupt module does not need to be enabled to update the state of the port data direction registers and port data output registers including the set/clear/toggle registers.

40.4.3 IOPORT

The GPIO registers are also aliased to the IOPORT interface on the Cortex-M0+ from address 0xF800_0000. Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle. If the DMA attempts to access the GPIO registers on the same cycle as an IOPORT access, then the DMA access will stall until any IOPORT accesses have completed.

During Compute Operation, the GPIO registers remain accessible via the IOPORT interface only. Since the clocks to the Port Control and Interrupt modules are disabled during Compute Operation, the Pin Data Input Registers do not update with the current state of the pins.

Chapter 41
Real Time Clock (RTC)

41.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

41.1.1 Features

The RTC module features include:

- 32-bit seconds counter with roll-over protection and 32-bit alarm
- 16-bit prescaler with compensation that can correct errors between 0.12 ppm and 3906 ppm
- Option to increment prescaler using the LPO (prescaler increments by 32 every clock edge)
- Register write protection
  - Lock register requires POR or software reset to enable write access
- Configurable 1, 2, 4, 8, 16, 32, 64 or 128 Hz square wave output with optional interrupt

41.1.2 Modes of operation

The RTC remains functional in all low power modes and can generate an interrupt to exit any low power mode.

41.1.3 RTC signal descriptions

Table 41-1. RTC signal descriptions

Signal	Description	I/O
RTC_CLKOUT	Prescaler square-wave output or 32kHz crystal clock	O

41.1.3.1 RTC clock output

The RTC_CLKOUT signal can output either a square wave prescaler output (configurable to 1, 2, 4, 8, 16, 32, 64 or 128 Hz) or the 32 kHz crystal clock.

41.2 Register definition

All registers must be accessed using 32-bit writes and all register accesses incur three wait states.

Write accesses to any register by non-supervisor mode software, when the supervisor access bit in the control register is clear, will terminate with a bus error.

Read accesses by non-supervisor mode software complete as normal.

Writing to a register protected by the lock register does not generate a bus error, but the write will not complete.

RTC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8000	RTC Time Seconds Register (RTC_TSR)	32	R/W	0000_0000h	41.2.1/1031
4003_8004	RTC Time Prescaler Register (RTC_TPR)	32	R/W	0000_0000h	41.2.2/1031
4003_8008	RTC Time Alarm Register (RTC_TAR)	32	R/W	0000_0000h	41.2.3/1032
4003_800C	RTC Time Compensation Register (RTC_TCR)	32	R/W	0000_0000h	41.2.4/1032
4003_8010	RTC Control Register (RTC_CR)	32	R/W	0000_0000h	41.2.5/1033
4003_8014	RTC Status Register (RTC_SR)	32	R/W	0000_0001h	41.2.6/1035
4003_8018	RTC Lock Register (RTC_LR)	32	R/W	0000_00FFh	41.2.7/1036
4003_801C	RTC Interrupt Enable Register (RTC_IER)	32	R/W	0000_0007h	41.2.8/1037

41.2.1 RTC Time Seconds Register (RTC_TSR)

Address: 4003_8000h base + 0h offset = 4003_8000h

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

RTC_TSR field descriptions

Field	Description
TSR	Time Seconds Register When the time counter is enabled, the TSR is read only and increments once a second provided SR[TOF] or SR[TIF] are not set. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TSR can be read or written. Writing to the TSR when the time counter is disabled will clear the SR[TOF] and/or the SR[TIF]. Writing to TSR with zero is supported, but not recommended because TSR will read as zero when SR[TIF] or SR[TOF] are set (indicating the time is invalid).

41.2.2 RTC Time Prescaler Register (RTC_TPR)

Address: 4003_8000h base + 4h offset = 4003_8004h

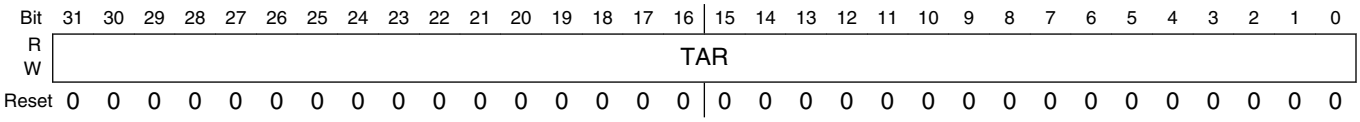
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TPR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RTC_TPR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TPR	Time Prescaler Register When the time counter is enabled, the TPR is read only and increments every 32.768 kHz clock cycle. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TPR can be read or written. The TSR[TSR] increments when bit 14 of the TPR transitions from a logic one to a logic zero.

41.2.3 RTC Time Alarm Register (RTC_TAR)

Address: 4003_8000h base + 8h offset = 4003_8008h

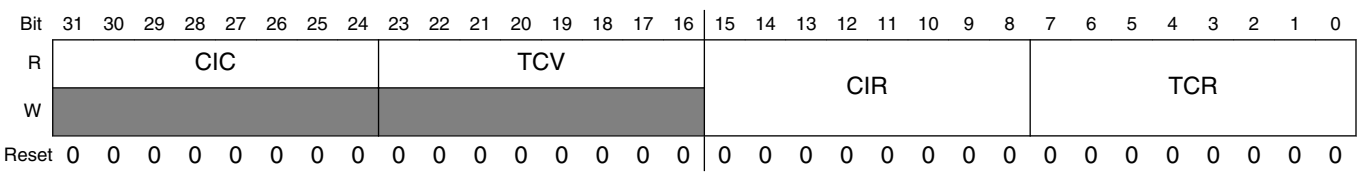


RTC_TAR field descriptions

Field	Description
TAR	Time Alarm Register When the time counter is enabled, the SR[TAF] is set whenever the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. Writing to the TAR clears the SR[TAF].

41.2.4 RTC Time Compensation Register (RTC_TCR)

Address: 4003_8000h base + Ch offset = 4003_800Ch



RTC_TCR field descriptions

Field	Description
31–24 CIC	Compensation Interval Counter Current value of the compensation interval counter. If the compensation interval counter equals zero then it is loaded with the contents of the CIR. If the CIC does not equal zero then it is decremented once a second.
23–16 TCV	Time Compensation Value Current value used by the compensation logic for the present second interval. Updated once a second if the CIC equals 0 with the contents of the TCR field. If the CIC does not equal zero then it is loaded with zero (compensation is not enabled for that second increment).
15–8 CIR	Compensation Interval Register Configures the compensation interval in seconds from 1 to 256 to control how frequently the TCR should adjust the number of 32.768 kHz cycles in each second. The value written should be one less than the number of seconds. For example, write zero to configure for a compensation interval of one second. This register is double buffered and writes do not take affect until the end of the current compensation interval.
TCR	Time Compensation Register Configures the number of 32.768 kHz clock cycles in each second. This register is double buffered and writes do not take affect until the end of the current compensation interval.

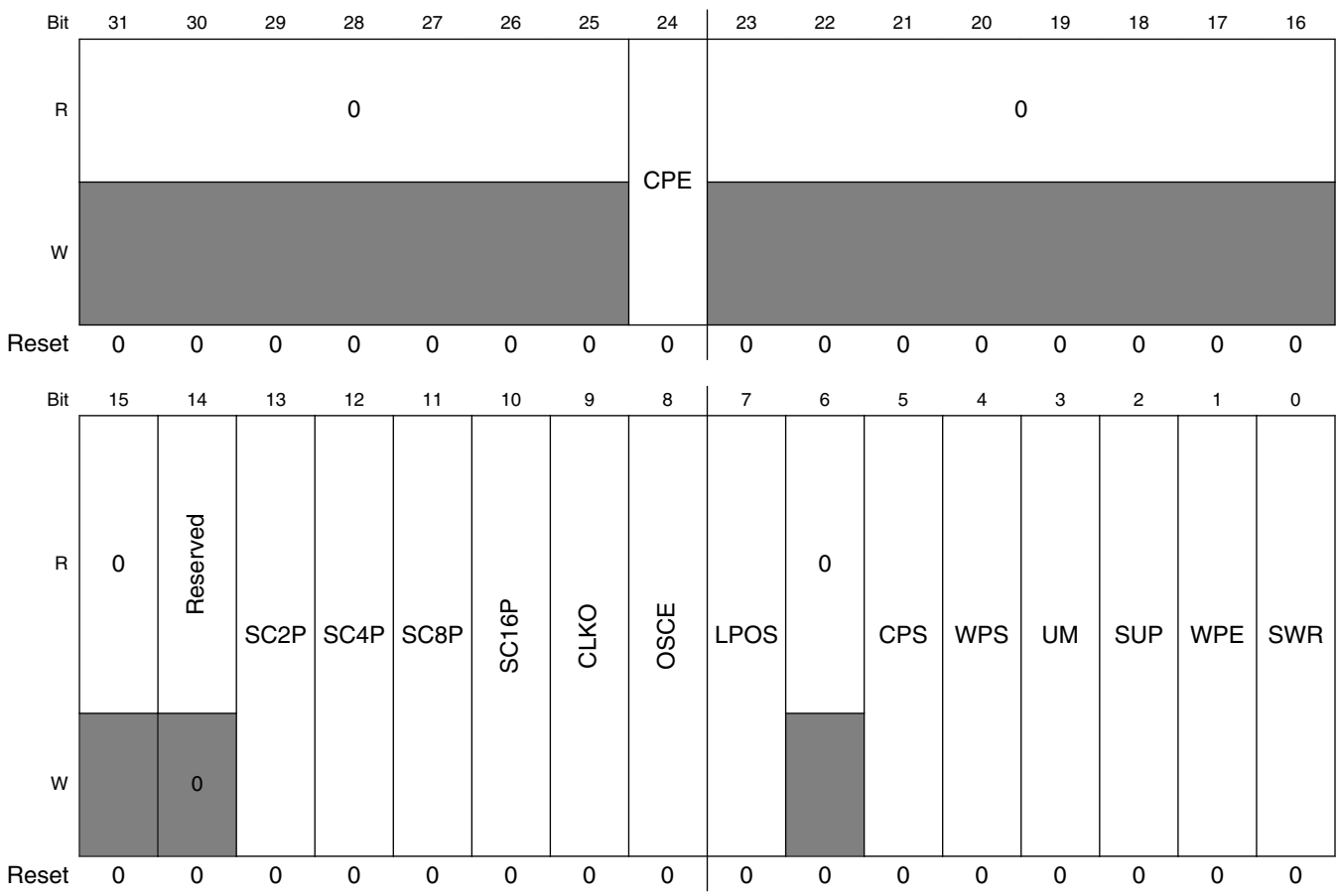
Table continues on the next page...

RTC_TCR field descriptions (continued)

Field	Description
	80h Time Prescaler Register overflows every 32896 clock cycles. ... ... FFh Time Prescaler Register overflows every 32769 clock cycles. 00h Time Prescaler Register overflows every 32768 clock cycles. 01h Time Prescaler Register overflows every 32767 clock cycles. .... 7Fh Time Prescaler Register overflows every 32641 clock cycles.

41.2.5 RTC Control Register (RTC_CR)

Address: 4003_8000h base + 10h offset = 4003_8010h



RTC_CR field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

RTC_CR field descriptions (continued)

Field	Description
24 CPE	Clock Pin Enable 0 RTC_CLKOUT is disabled. 01 RTC_CLKOUT is enabled.
23–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 Reserved	This field is reserved. It must always be written to 0.
13 SC2P	Oscillator 2pF Load Configure 0 Disable the load. 1 Enable the additional load.
12 SC4P	Oscillator 4pF Load Configure 0 Disable the load. 1 Enable the additional load.
11 SC8P	Oscillator 8pF Load Configure 0 Disable the load. 1 Enable the additional load.
10 SC16P	Oscillator 16pF Load Configure 0 Disable the load. 1 Enable the additional load.
9 CLKO	Clock Output 0 The 32 kHz clock is output to other peripherals. 1 The 32 kHz clock is not output to other peripherals.
8 OSCE	Oscillator Enable 0 32.768 kHz oscillator is disabled. 1 32.768 kHz oscillator is enabled. After setting this bit, wait the oscillator startup time before enabling the time counter to allow the 32.768 kHz clock time to stabilize.
7 LPOS	LPO Select When set, the RTC prescaler increments using the LPO 1kHz clock and not the RTC 32kHz crystal clock. The LPO increments the prescaler from bit TPR[5] (TPR[4:0] are ignored), supporting close to 1 second increment of the seconds register. Although compensation is supported when clocked from the LPO, TCR[4:0] of the compensation register are also ignored and only TCR[7:5] set the compensation value (can overflow after 1020 to 1027 cycles). 0 RTC prescaler increments using 32kHz crystal. 1 RTC prescaler increments using 1kHz LPO, bits [4:0] of the prescaler are bypassed.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 CPS	Clock Pin Select 0 The prescaler output clock (as configured by TSIC) is output on RTC_CLKOUT. 1 The RTC 32kHz crystal clock is output on RTC_CLKOUT.

Table continues on the next page...

RTC_CR field descriptions (continued)

Field	Description
4 WPS	Wakeup Pin Select The wakeup pin is optional and not available on all devices. 0 Wakeup pin asserts (active low, open drain) if the RTC interrupt asserts or the wakeup pin is turned on. 1 Wakeup pin instead outputs the RTC 32kHz clock, provided the wakeup pin is turned on and the 32kHz clock is output to other peripherals.
3 UM	Update Mode Allows SR[TCE] to be written even when the Status Register is locked. When set, the SR[TCE] can always be written if the SR[TIF] or SR[TOF] are set or if the SR[TCE] is clear. 0 Registers cannot be written when locked. 1 Registers can be written when locked under limited conditions.
2 SUP	Supervisor Access 0 Non-supervisor mode write accesses are not supported and generate a bus error. 1 Non-supervisor mode write accesses are supported.
1 WPE	Wakeup Pin Enable The wakeup pin is optional and not available on all devices. 0 Wakeup pin is disabled. 1 Wakeup pin is enabled and wakeup pin asserts if the RTC interrupt asserts or the wakeup pin is turned on.
0 SWR	Software Reset 0 No effect. 1 Resets all RTC registers except for the SWR bit . The SWR bit is cleared by POR and by software explicitly clearing it.

41.2.6 RTC Status Register (RTC_SR)

Address: 4003_8000h base + 14h offset = 4003_8014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0											TCE	0	TAF	TOF	TIF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

RTC_SR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 TCE	Time Counter Enable When time counter is disabled the TSR register and TPR register are writeable, but do not increment. When time counter is enabled the TSR register and TPR register are not writeable, but increment. 0 Time counter is disabled. 1 Time counter is enabled.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 TAF	Time Alarm Flag Time alarm flag is set when the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. This bit is cleared by writing the TAR register. 0 Time alarm has not occurred. 1 Time alarm has occurred.
1 TOF	Time Overflow Flag Time overflow flag is set when the time counter is enabled and overflows. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. 0 Time overflow has not occurred. 1 Time overflow has occurred and time counter is read as zero.
0 TIF	Time Invalid Flag The time invalid flag is set on POR or software reset. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. 0 Time is valid. 1 Time is invalid and time counter is read as zero.

41.2.7 RTC Lock Register (RTC_LR)

Address: 4003_8000h base + 18h offset = 4003_8018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								1	LRL	SRL	CRL	TCL	1		
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

RTC_LR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
6 LRL	Lock Register Lock After being cleared, this bit can be set only by POR or software reset. 0 Lock Register is locked and writes are ignored. 1 Lock Register is not locked and writes complete as normal.
5 SRL	Status Register Lock After being cleared, this bit can be set only by POR or software reset. 0 Status Register is locked and writes are ignored. 1 Status Register is not locked and writes complete as normal.
4 CRL	Control Register Lock After being cleared, this bit can only be set by POR. 0 Control Register is locked and writes are ignored. 1 Control Register is not locked and writes complete as normal.
3 TCL	Time Compensation Lock After being cleared, this bit can be set only by POR or software reset. 0 Time Compensation Register is locked and writes are ignored. 1 Time Compensation Register is not locked and writes complete as normal.
Reserved	This field is reserved. This read-only field is reserved and always has the value 1.

41.2.8 RTC Interrupt Enable Register (RTC_IER)

Address: 4003_8000h base + 1Ch offset = 4003_801Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													TSIC		
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
R	0								WPON	Reserved		TSIE	Reserved	TAIE	TOIE	TIIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

RTC_IER field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 TSIC	Timer Seconds Interrupt Configuration Configures the frequency of the RTC Seconds interrupt and the RTC_CLKOUT prescaler output. This field should only be altered when TSIE is clear. 000 1 Hz. 001 2 Hz. 010 4 Hz. 011 8 Hz. 100 16 Hz. 101 32 Hz. 110 64 Hz. 111 128 Hz.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 WPON	Wakeup Pin On The wakeup pin is optional and not available on all devices. Whenever the wakeup pin is enabled and this bit is set, the wakeup pin will assert. 0 No effect. 1 If the wakeup pin is enabled, then the wakeup pin will assert.
6–5 Reserved	This field is reserved.
4 TSIE	Time Seconds Interrupt Enable The seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector. It is generated at least once a second and requires no software overhead (there is no corresponding status flag to clear). The frequency of the seconds interrupt is configured by TSIC. 0 Seconds interrupt is disabled. 1 Seconds interrupt is enabled.
3 Reserved	This field is reserved.
2 TAIE	Time Alarm Interrupt Enable 0 Time alarm flag does not generate an interrupt. 1 Time alarm flag does generate an interrupt.
1 TOIE	Time Overflow Interrupt Enable 0 Time overflow flag does not generate an interrupt. 1 Time overflow flag does generate an interrupt.
0 TIIE	Time Invalid Interrupt Enable 0 Time invalid flag does not generate an interrupt. 1 Time invalid flag does generate an interrupt.

41.3 Functional description

41.3.1 Power, clocking, and reset

The RTC is an always powered block that remains active in all low power modes.

The time counter within the RTC is clocked by a 32.768 kHz clock sourced from an external crystal using the oscillator. Alternatively, the time counter can be clocked by the LPO and the prescaler will increment by 32 for each LPO clock.

The power-on-reset signal initializes all RTC registers to their default state. A software reset bit can also initialize all RTC registers.

41.3.1.1 Oscillator control

The 32.768 kHz crystal oscillator is disabled at POR and must be enabled by software. After enabling the cystal oscillator, wait the oscillator startup time before setting SR[TCE] or using the oscillator clock external to the RTC.

The crystal oscillator includes tunable capacitors that can be configured by software. Do not change the capacitance unless the oscillator is disabled.

41.3.1.2 Software reset

Writing 1 to CR[SWR] forces the equivalent of a POR to the rest of the RTC module. CR[SWR] is not affected by the software reset and must be cleared by software.

41.3.1.3 Supervisor access

When the supervisor access control bit is clear, only supervisor mode software can write to the RTC registers, non-supervisor mode software will generate a bus error. Both supervisor and non-supervisor mode software can always read the RTC registers.

41.3.2 Time counter

The time counter consists of a 32-bit seconds counter that increments once every second and a 16-bit prescaler register that increments once every 32.768 kHz clock cycle. There is also the option to clock the prescaler using a 1 kHz LPO that increments the prescaler by 32 on every clock cycle.

Reading the time counter (either seconds or prescaler) while it is incrementing may return invalid data due to synchronization of the read data bus. If it is necessary for software to read the prescaler or seconds counter when they could be incrementing, it is recommended that two read accesses are performed and that software verifies that the same data was returned for both reads.

The time seconds register and time prescaler register can be written only when SR[TCE] is clear. Always write to the prescaler register before writing to the seconds register, because the seconds register increments on the falling edge of bit 14 of the prescaler register.

The time prescaler register increments provided SR[TCE] is set, SR[TIF] is clear, SR[TOF] is clear, and the 32.768 kHz (or 1 kHz) clock source is present. After enabling the oscillator, wait the oscillator startup time before setting SR[TCE] to allow time for the oscillator clock output to stabilize.

If the time seconds register overflows then the SR[TOF] will set and the time prescaler register will stop incrementing. Clear SR[TOF] by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TOF] is set.

SR[TIF] is set on POR and software reset and is cleared by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TIF] is set.

41.3.3 Compensation

The compensation logic provides an accurate and wide compensation range and can correct errors as high as 3906 ppm and as low as 0.12 ppm. The compensation factor must be calculated externally to the RTC and supplied by software to the compensation register. The RTC itself does not calculate the amount of compensation that is required, although the 1 Hz clock is output to an external pin in support of external calibration logic.

Crystal compensation can be supported by using firmware and crystal characteristics to determine the compensation amount. Temperature compensation can be supported by firmware that periodically measures the external temperature via ADC and updates the compensation register based on a look-up table that specifies the change in crystal frequency over temperature.

The compensation logic alters the number of 32.768 kHz clock cycles it takes for the prescaler register to overflow and increment the time seconds counter. The time compensation value is used to adjust the number of clock cycles between -127 and +128. Cycles are added or subtracted from the prescaler register when the prescaler register equals 0x3FFF and then increments. The compensation interval is used to adjust the frequency at which the time compensation value is used, that is, from once a second to once every 256 seconds.

Updates to the time compensation register will not take effect until the next time the time seconds register increments and provided the previous compensation interval has expired. When the compensation interval is set to other than once a second then the compensation is applied in the first second interval and the remaining second intervals receive no compensation.

Compensation is disabled by configuring the time compensation register to zero.

When the prescaler is configured to increment using the 1 kHz LPO, the effective compensation value is divided by 32 and can only adjust the number of clock cycles between -4 and +3.

41.3.4 Time alarm

The Time Alarm register (TAR), SR[TAF], and IER[TAIE] allow the RTC to generate an interrupt at a predefined time. The 32-bit TAR is compared with the 32-bit Time Seconds register (TSR) each time it increments. SR[TAF] will set when TAR equals TSR and TSR increments.

SR[TAF] is cleared by writing TAR. This will usually be the next alarm value, although writing a value that is less than TSR, such as 0, will prevent SR[TAF] from setting again. SR[TAF] cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

41.3.5 Update mode

The Update Mode field in the Control register (CR[UM]) configures software write access to the Time Counter Enable (SR[TCE]) field. When CR[UM] is clear, SR[TCE] can be written only when LR[SRL] is set. When CR[UM] is set, SR[TCE] can also be written when SR[TCE] is clear or when SR[TIF] or SR[TOF] are set. This allows the time seconds and prescaler registers to be initialized whenever time is invalidated, while preventing the time seconds and prescaler registers from being changed on the fly. When LR[SRL] is set, CR[UM] has no effect on SR[TCE].

41.3.6 Register lock

The Lock register (LR) can be used to block write accesses to certain registers until the next POR or software reset. Locking the Control register (CR) will disable the software reset. Locking LR will block future updates to LR.

Write accesses to a locked register are ignored and do not generate a bus error.

41.3.7 Interrupt

The RTC interrupt is asserted whenever a status flag and the corresponding interrupt enable bit are both set. It is always asserted on POR, and software reset. The RTC interrupt is enabled at the chip level by enabling the chip-specific RTC clock gate control bit. The RTC interrupt can be used to wakeup the chip from any low-power mode.

The optional RTC seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector that is generated once a second and requires no software overhead (there is no corresponding status flag to clear). It is enabled in the RTC by the time seconds interrupt enable bit and enabled at the chip level by setting the chip-specific RTC clock gate control bit. The frequency of the seconds interrupt defaults to 1 Hz, but can instead be configured to trigger every 2, 4, 8, 16, 32, 64 or 128 Hz. This interrupt is optional and may not be implemented on all devices.

Chapter 42
Integrated Interchip Sound (I2S) / Synchronous Audio Interface (SAI)

42.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The I^2S (or I2S) module provides a synchronous audio interface (SAI) that supports full-duplex serial interfaces with frame synchronization such as I^2S, AC97, TDM, and codec/DSP interfaces.

42.1.1 Features

Note that some of the features are not supported across all SAI instances; see the chip-specific information in the first section of this chapter.

- Transmitter with independent bit clock and frame sync supporting 1 data line
- Receiver with independent bit clock and frame sync supporting 1 data line
- Maximum Frame Size of 16 words
- Word size of between 8-bits and 32-bits
- Word size configured separately for first word and remaining words in frame
- Supports graceful restart after FIFO error
- Supports automatic restart after FIFO error without software intervention
- Supports packing of 8-bit and 16-bit data into each 32-bit FIFO word

42.1.2 Block diagram

The following block diagram also shows the module clocks.

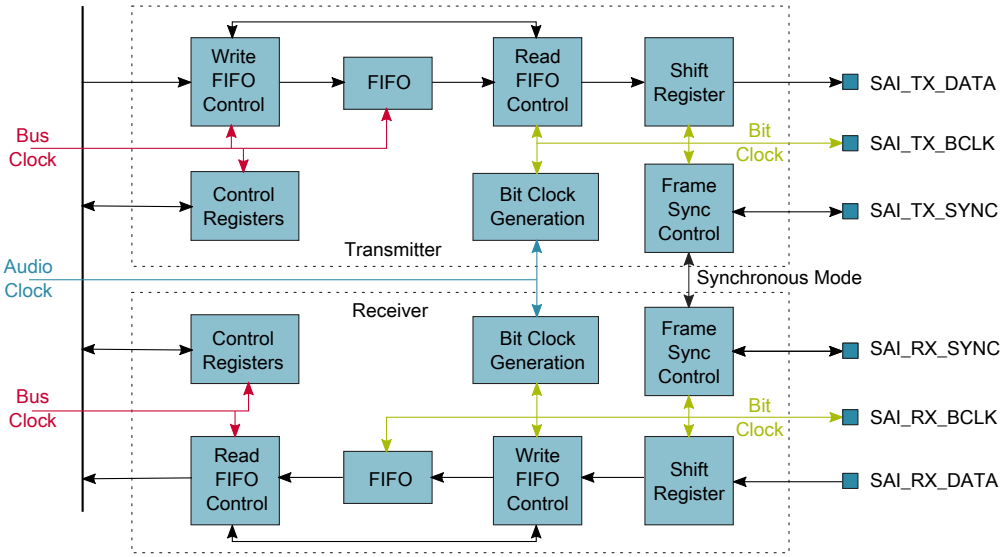


Figure 42-1. I²S/SAI block diagram

42.1.3 Modes of operation

The module operates in these power modes: Run mode, stop modes, low-leakage modes, and Debug mode.

42.1.3.1 Run mode

In Run mode, the SAI transmitter and receiver operate normally.

42.1.3.2 Stop modes

In Stop mode, the SAI transmitter and/or receiver can continue operating provided the appropriate Stop Enable bit is set (TCSR[STOPE] and/or RCSR[STOPE], respectively), and provided the transmitter and/or receiver is/are using an externally generated bit clock or an Audio Master Clock that remains operating in Stop mode. The SAI transmitter and/or receiver can generate an asynchronous interrupt to wake the CPU from Stop mode.

In Stop mode, if the Transmitter Stop Enable (TCSR[STOPE]) bit is clear, the transmitter is disabled after completing the current transmit frame, and, if the Receiver Stop Enable (RCSR[STOPE]) bit is clear, the receiver is disabled after completing the current receive frame. Entry into Stop mode is prevented–not acknowledged–while waiting for the transmitter and receiver to be disabled at the end of the current frame.

42.1.3.3 Low-leakage modes

When entering low-leakage modes, the Stop Enable (TCSR[STOPE] and RCSR[STOPE]) bits are ignored and the SAI is disabled after completing the current transmit and receive Frames. Entry into stop mode is prevented (not acknowledged) while waiting for the transmitter and receiver to be disabled at the end of the current frame.

42.1.3.4 Debug mode

In Debug mode, the SAI transmitter and/or receiver can continue operating provided the Debug Enable bit is set. When TCSR[DBGE] or RCSR[DBGE] bit is clear and Debug mode is entered, the SAI is disabled after completing the current transmit or receive frame. The transmitter and receiver bit clocks are not affected by Debug mode.

42.2 External signals

Name	Function	I/O
SAI_TX_BCLK	Transmit Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
SAI_TX_SYNC	Transmit Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
SAI_TX_DATA	Transmit Data. The transmit data is generated synchronously by the bit clock and is tristated whenever not transmitting a word.	O
SAI_RX_BCLK	Receive Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
SAI_RX_SYNC	Receive Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
SAI_RX_DATA	Receive Data. The receive data is sampled synchronously by the bit clock.	I
SAI_MCLK	Audio Master Clock.	I

42.3 Memory map and register definition

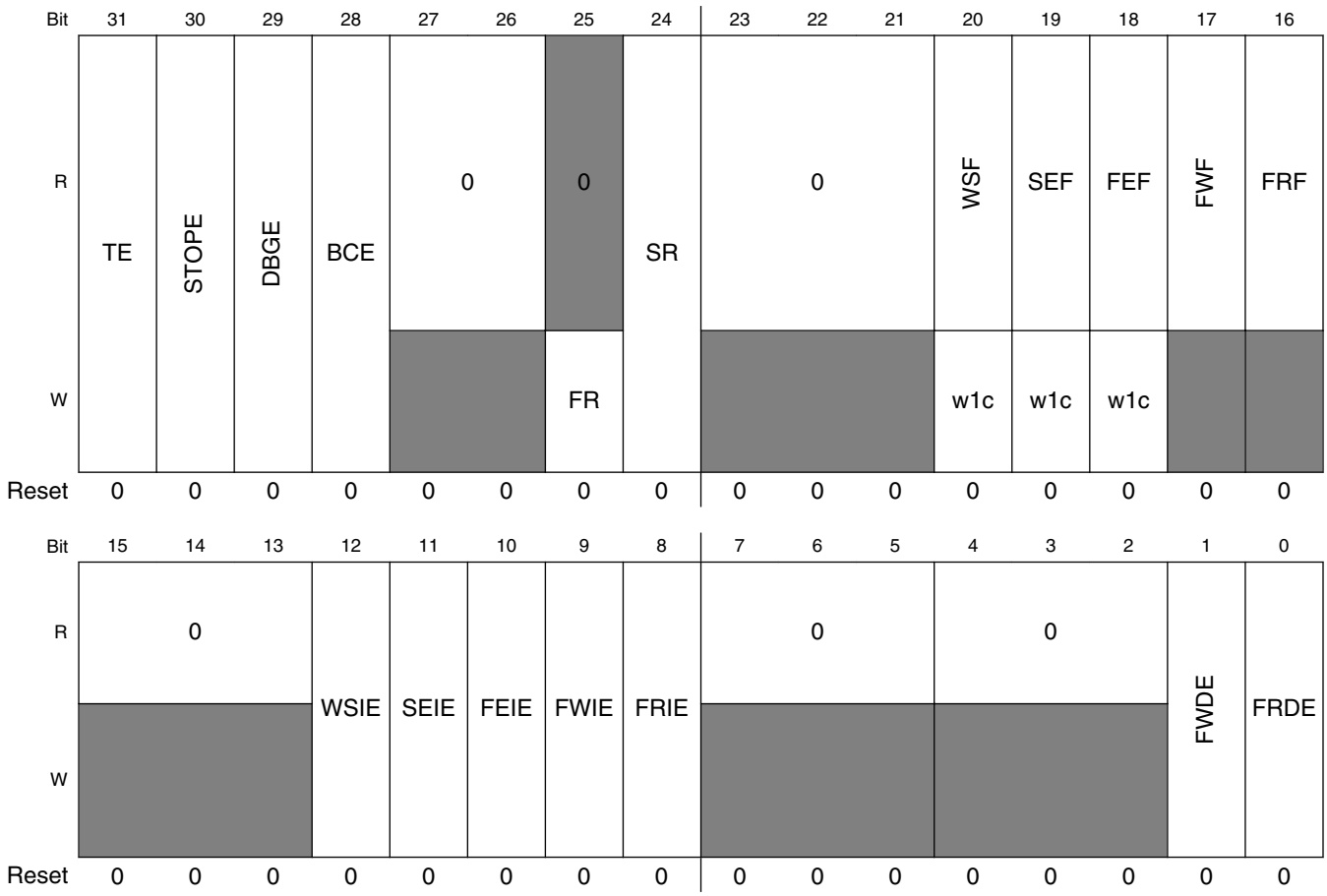
A read or write access to an address from offset 0x100 and above will result in a bus error.

I2S memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_C000	SAI Transmit Control Register (I2S0_TCSR)	32	R/W	0000_0000h	42.3.1/1047
4004_C004	SAI Transmit Configuration 1 Register (I2S0_TCR1)	32	R/W	0000_0000h	42.3.2/1050
4004_C008	SAI Transmit Configuration 2 Register (I2S0_TCR2)	32	R/W	0000_0000h	42.3.3/1050
4004_C00C	SAI Transmit Configuration 3 Register (I2S0_TCR3)	32	R/W	0000_0000h	42.3.4/1052
4004_C010	SAI Transmit Configuration 4 Register (I2S0_TCR4)	32	R/W	0000_0000h	42.3.5/1053
4004_C014	SAI Transmit Configuration 5 Register (I2S0_TCR5)	32	R/W	0000_0000h	42.3.6/1055
4004_C020	SAI Transmit Data Register (I2S0_TDR0)	32	W (always reads 0)	0000_0000h	42.3.7/1055
4004_C040	SAI Transmit FIFO Register (I2S0_TFR0)	32	R	0000_0000h	42.3.8/1056
4004_C060	SAI Transmit Mask Register (I2S0_TMR)	32	R/W	0000_0000h	42.3.9/1056
4004_C080	SAI Receive Control Register (I2S0_RCSR)	32	R/W	0000_0000h	42.3.10/ 1058
4004_C084	SAI Receive Configuration 1 Register (I2S0_RCR1)	32	R/W	0000_0000h	42.3.11/ 1061
4004_C088	SAI Receive Configuration 2 Register (I2S0_RCR2)	32	R/W	0000_0000h	42.3.12/ 1061
4004_C08C	SAI Receive Configuration 3 Register (I2S0_RCR3)	32	R/W	0000_0000h	42.3.13/ 1063
4004_C090	SAI Receive Configuration 4 Register (I2S0_RCR4)	32	R/W	0000_0000h	42.3.14/ 1064
4004_C094	SAI Receive Configuration 5 Register (I2S0_RCR5)	32	R/W	0000_0000h	42.3.15/ 1066
4004_C0A0	SAI Receive Data Register (I2S0_RDR0)	32	R	0000_0000h	42.3.16/ 1066
4004_C0C0	SAI Receive FIFO Register (I2S0_RFR0)	32	R	0000_0000h	42.3.17/ 1067
4004_C0E0	SAI Receive Mask Register (I2S0_RMR)	32	R/W	0000_0000h	42.3.18/ 1067

42.3.1 SAI Transmit Control Register (I2Sx_TCSR)

Address: 4004_C000h base + 0h offset = 4004_C000h



I2Sx_TCSR field descriptions

Field	Description
31 TE	Transmitter Enable Enables/disables the transmitter. When software clears this field, the transmitter remains enabled, and this bit remains set, until the end of the current frame. 0 Transmitter is disabled. 1 Transmitter is enabled, or transmitter has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures transmitter operation in Stop mode. This field is ignored and the transmitter is disabled in all low-leakage stop modes. 0 Transmitter disabled in Stop mode. 1 Transmitter enabled in Stop mode.
29 DBGE	Debug Enable

Table continues on the next page...

I2Sx_TCSR field descriptions (continued)

Field	Description
	Enables/disables transmitter operation in Debug mode. The transmit bit clock is not affected by debug mode. 0 Transmitter is disabled in Debug mode, after completing the current frame. 1 Transmitter is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the transmit bit clock, separately from the TE. This field is automatically set whenever TE is set. When software clears this field, the transmit bit clock remains enabled, and this bit remains set, until the end of the current frame. 0 Transmit bit clock is disabled. 1 Transmit bit clock is enabled.
27–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 FR	FIFO Reset Resets the FIFO pointers. Reading this field will always return zero. FIFO pointers should only be reset when the transmitter is disabled or the FIFO error flag is set. 0 No effect. 1 FIFO reset.
24 SR	Software Reset When set, resets the internal transmitter logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. 0 No effect. 1 Software reset.
23–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0 Start of word not detected. 1 Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0 Sync error not detected. 1 Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled transmit FIFO has underrun. Write a logic 1 to this field to clear this flag. 0 Transmit underrun not detected. 1 Transmit underrun detected.

Table continues on the next page...

I2Sx_TCSR field descriptions (continued)

Field	Description
17 FWF	FIFO Warning Flag Indicates that an enabled transmit FIFO is empty. 0 No enabled transmit FIFO is empty. 1 Enabled transmit FIFO is empty.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0 Transmit FIFO watermark has not been reached. 1 Transmit FIFO watermark has been reached.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0 Disables interrupt. 1 Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0 Disables interrupt. 1 Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FWDE	FIFO Warning DMA Enable

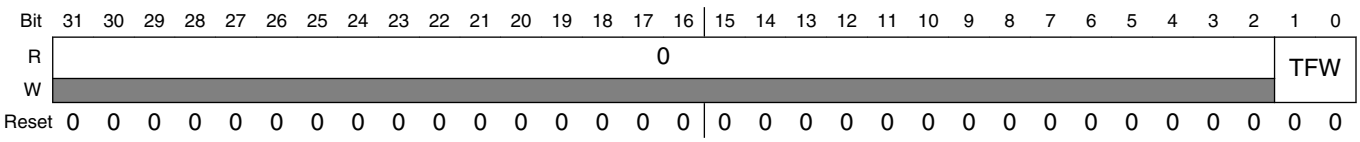
Table continues on the next page...

I2Sx_TCSR field descriptions (continued)

Field	Description
	Enables/disables DMA requests. 0 Disables the DMA request. 1 Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0 Disables the DMA request. 1 Enables the DMA request.

42.3.2 SAI Transmit Configuration 1 Register (I2Sx_TCR1)

Address: 4004_C000h base + 4h offset = 4004_C004h



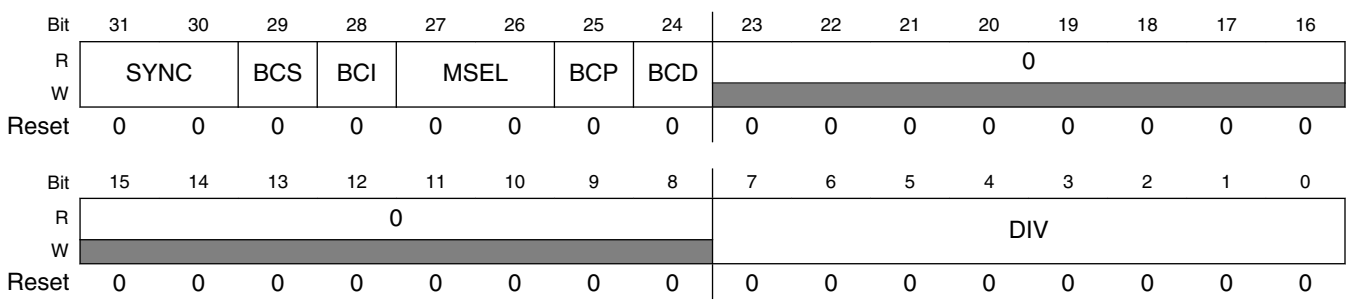
I2Sx_TCR1 field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TFW	Transmit FIFO Watermark Configures the watermark level for all enabled transmit channels.

42.3.3 SAI Transmit Configuration 2 Register (I2Sx_TCR2)

This register must not be altered when TCSR[TE] is set.

Address: 4004_C000h base + 8h offset = 4004_C008h



I2Sx_TCR2 field descriptions

Field	Description
31–30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the receiver must be configured for asynchronous operation. 00 Asynchronous mode. 01 Synchronous with receiver.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the transmitter. When the transmitter is configured in asynchronous mode and this bit is set, the transmitter is clocked by the receiver bit clock (SAI_RX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the transmitter continues to use the transmit frame sync (SAI_TX_SYNC). When the transmitter is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the transmitter bit clock (SAI_TX_BCLK) but use the receiver frame sync (SAI_RX_SYNC). 0 Use the normal bit clock source. 1 Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the transmitter is delayed by the pad output delay (the transmitter is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the transmitter when this bit is set. In synchronous mode, this bit allows the transmitter to use the slave mode timing from the datasheet, while the receiver uses the master mode timing. This field has no effect when configured for an externally generated bit clock . 0 No effect. 1 Internal logic is clocked as if bit clock was externally generated.
27–26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the meaning of each option. 01 Master Clock (MCLK) 1 option selected. 10 Master Clock (MCLK) 2 option selected. 11 Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0 Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1 Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock.

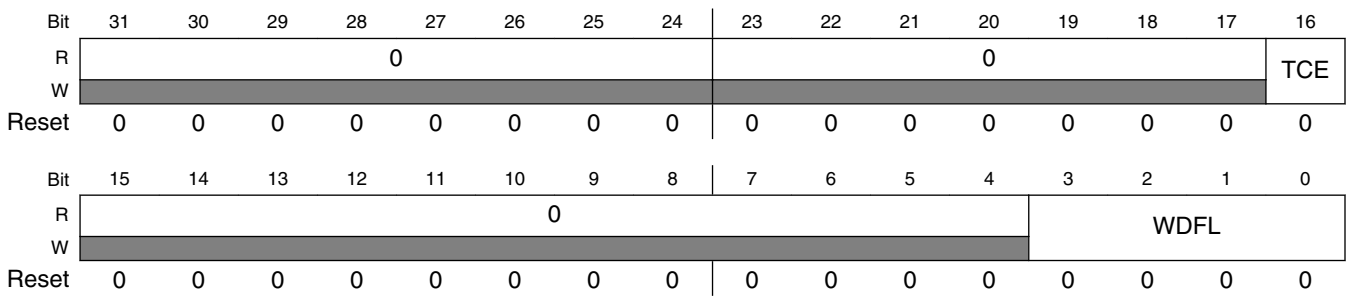
Table continues on the next page...

I2Sx_TCR2 field descriptions (continued)

Field	Description
	0 Bit clock is generated externally in Slave mode. 1 Bit clock is generated internally in Master mode.
23–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

42.3.4 SAI Transmit Configuration 3 Register (I2Sx_TCR3)

Address: 4004_C000h base + Ch offset = 4004_C00Ch



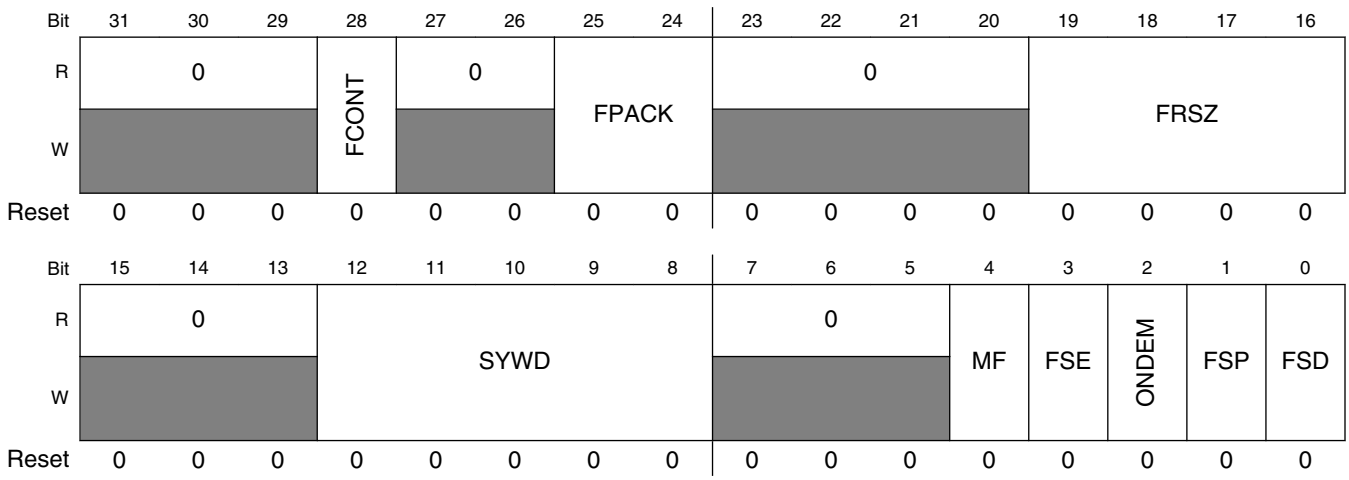
I2Sx_TCR3 field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 TCE	Transmit Channel Enable Enables the corresponding data channel for transmit operation. A channel must be enabled before its FIFO is accessed. Changing this field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for transmit operation. 0 Transmit data channel N is disabled. 1 Transmit data channel N is enabled.
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
WDFL	Word Flag Configuration Configures which word sets the start of word flag. The value written must be one less than the word number. For example, writing 0 configures the first word in the frame. When configured to a value greater than TCR4[FRSZ], then the start of word flag is never set.

42.3.5 SAI Transmit Configuration 4 Register (I2Sx_TCR4)

This register must not be altered when TCSR[TE] is set.

Address: 4004_C000h base + 10h offset = 4004_C010h



I2Sx_TCR4 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 FCONT	FIFO Continue on Error Configures when the SAI will continue transmitting after a FIFO error has been detected. 0 On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1 On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO write pointer will only increment when the full 32-bit FIFO word has been written by software. 00 FIFO packing is disabled 01 Reserved 10 8-bit FIFO packing is enabled 11 16-bit FIFO packing is enabled
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

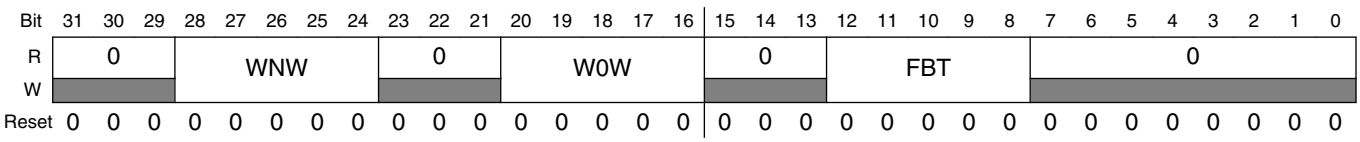
I2Sx_TCR4 field descriptions (continued)

Field	Description
19–16 FRSZ	Frame size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 16 words.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 MF	MSB First Configures whether the LSB or the MSB is transmitted first. 0 LSB is transmitted first. 1 MSB is transmitted first.
3 FSE	Frame Sync Early 0 Frame sync asserts with the first bit of the frame. 1 Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0 Internal frame sync is generated continuously. 1 Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0 Frame sync is active high. 1 Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0 Frame sync is generated externally in Slave mode. 1 Frame sync is generated internally in Master mode.

42.3.6 SAI Transmit Configuration 5 Register (I2Sx_TCR5)

This register must not be altered when TCSR[TE] is set.

Address: 4004_C000h base + 14h offset = 4004_C014h

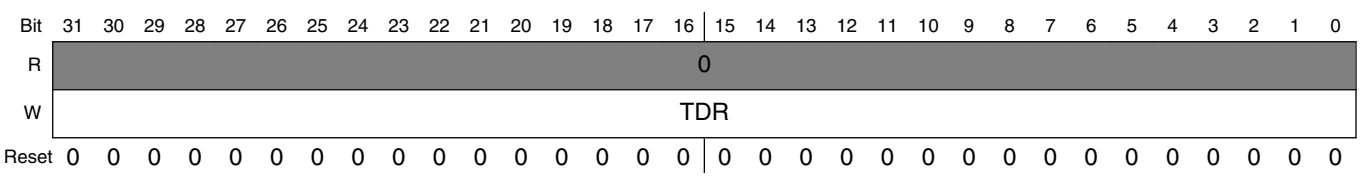


I2Sx_TCR5 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28–24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB First, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB First, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

42.3.7 SAI Transmit Data Register (I2Sx_TDR*n*)

Address: 4004_C000h base + 20h offset + (4d × i), where i=0d to 0d



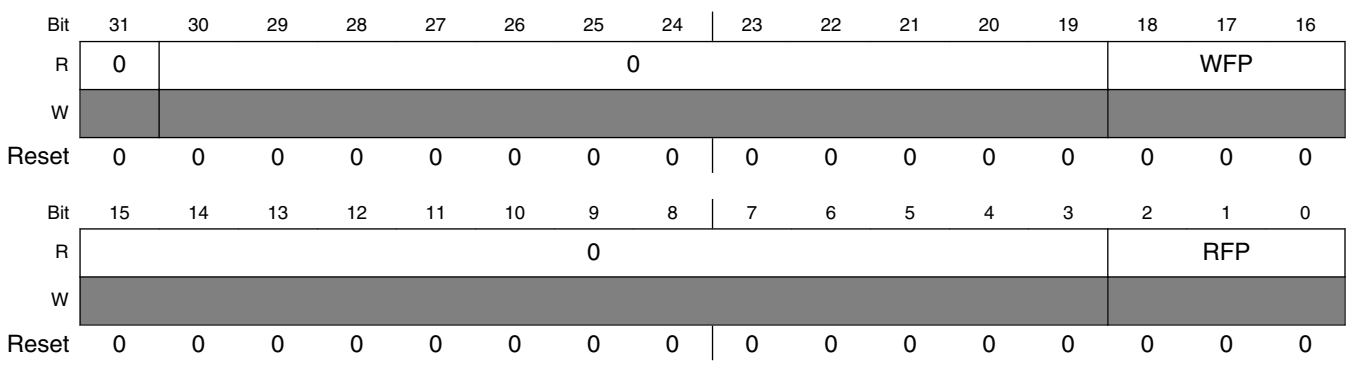
I2Sx_TDR*n* field descriptions

Field	Description
TDR	Transmit Data Register The corresponding TCR3[TCE] bit must be set before accessing the channel's transmit data register. Writes to this register when the transmit FIFO is not full will push the data written into the transmit data FIFO. Writes to this register when the transmit FIFO is full are ignored.

42.3.8 SAI Transmit FIFO Register (I2Sx_TFR*n*)

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

Address: 4004_C000h base + 40h offset + (4d × i), where i=0d to 0d



I2Sx_TFR*n* field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 WFP	Write FIFO Pointer FIFO write pointer for transmit data channel.
15–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RFP	Read FIFO Pointer FIFO read pointer for transmit data channel.

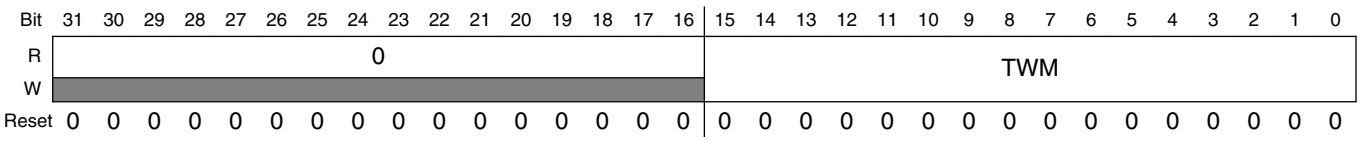
42.3.9 SAI Transmit Mask Register (I2Sx_TMR)

This register is double-buffered and updates:

1. When TCSR[TE] is first set
2. At the end of each frame.

This allows the masked words in each frame to change from frame to frame.

Address: 4004_C000h base + 60h offset = 4004_C060h

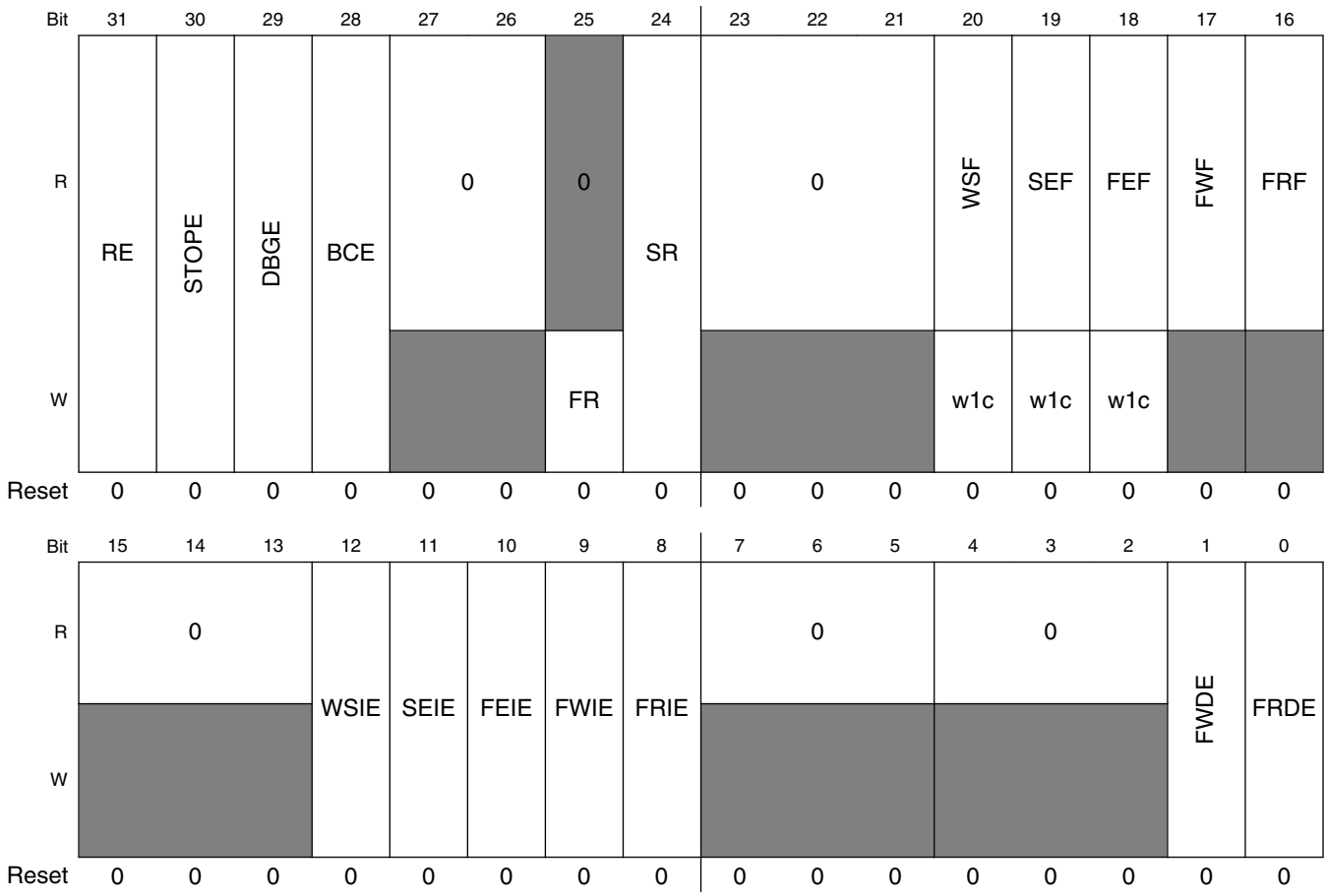


I2Sx_TMR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TWM	Transmit Word Mask Configures whether the transmit word is masked (transmit data pin tristated and transmit data not read from FIFO) for the corresponding word in the frame. 0 Word N is enabled. 1 Word N is masked. The transmit data pins are tri-stated when masked.

42.3.10 SAI Receive Control Register (I2Sx_RCSR)

Address: 4004_C000h base + 80h offset = 4004_C080h



I2Sx_RCSR field descriptions

Field	Description
31 RE	Receiver Enable Enables/disables the receiver. When software clears this field, the receiver remains enabled, and this bit remains set, until the end of the current frame. 0 Receiver is disabled. 1 Receiver is enabled, or receiver has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures receiver operation in Stop mode. This bit is ignored and the receiver is disabled in all low-leakage stop modes. 0 Receiver disabled in Stop mode. 1 Receiver enabled in Stop mode.
29 DBGE	Debug Enable

Table continues on the next page...

I2Sx_RCSR field descriptions (continued)

Field	Description
	Enables/disables receiver operation in Debug mode. The receive bit clock is not affected by Debug mode. 0 Receiver is disabled in Debug mode, after completing the current frame. 1 Receiver is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RE. This field is automatically set whenever RE is set. When software clears this field, the receive bit clock remains enabled, and this field remains set, until the end of the current frame. 0 Receive bit clock is disabled. 1 Receive bit clock is enabled.
27–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 FR	FIFO Reset Resets the FIFO pointers. Reading this field will always return zero. FIFO pointers should only be reset when the receiver is disabled or the FIFO error flag is set. 0 No effect. 1 FIFO reset.
24 SR	Software Reset Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. 0 No effect. 1 Software reset.
23–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0 Start of word not detected. 1 Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0 Sync error not detected. 1 Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled receive FIFO has overflowed. Write a logic 1 to this field to clear this flag. 0 Receive overflow not detected. 1 Receive overflow detected.
17 FWF	FIFO Warning Flag

Table continues on the next page...

I2Sx_RCSR field descriptions (continued)

Field	Description
	Indicates that an enabled receive FIFO is full. 0 No enabled receive FIFO is full. 1 Enabled receive FIFO is full.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0 Receive FIFO watermark not reached. 1 Receive FIFO watermark has been reached.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0 Disables interrupt. 1 Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0 Disables interrupt. 1 Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests.

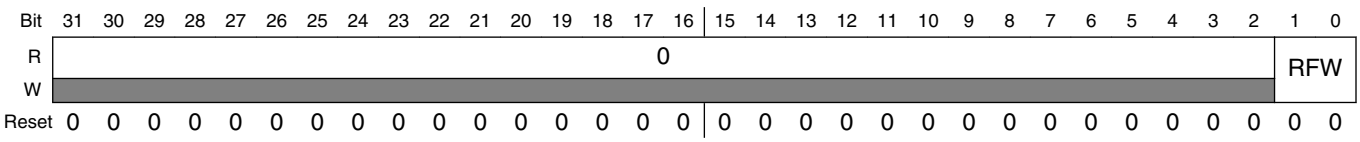
Table continues on the next page...

I2Sx_RCSR field descriptions (continued)

Field	Description
	0 Disables the DMA request. 1 Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0 Disables the DMA request. 1 Enables the DMA request.

42.3.11 SAI Receive Configuration 1 Register (I2Sx_RCR1)

Address: 4004_C000h base + 84h offset = 4004_C084h



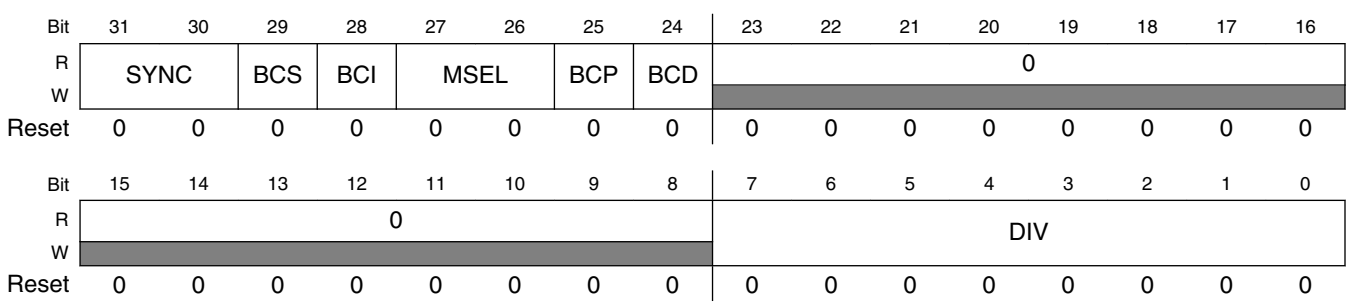
I2Sx_RCR1 field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RFW	Receive FIFO Watermark Configures the watermark level for all enabled receiver channels.

42.3.12 SAI Receive Configuration 2 Register (I2Sx_RCR2)

This register must not be altered when RCSR[RE] is set.

Address: 4004_C000h base + 88h offset = 4004_C088h



I2Sx_RCR2 field descriptions

Field	Description
31–30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the transmitter must be configured for asynchronous operation. 00 Asynchronous mode. 01 Synchronous with transmitter.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the receiver. When the receiver is configured in asynchronous mode and this bit is set, the receiver is clocked by the transmitter bit clock (SAI_TX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the receiver continues to use the receiver frame sync (SAI_RX_SYNC). When the receiver is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the receiver bit clock (SAI_RX_BCLK) but use the transmitter frame sync (SAI_TX_SYNC). 0 Use the normal bit clock source. 1 Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the receiver is delayed by the pad output delay (the receiver is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the receiver when this bit is set. In synchronous mode, this bit allows the receiver to use the slave mode timing from the datasheet, while the transmitter uses the master mode timing. This field has no effect when configured for an externally generated bit clock . 0 No effect. 1 Internal logic is clocked as if bit clock was externally generated.
27–26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the availability and chip-specific meaning of each option. 00 Bus Clock selected. 01 Master Clock (MCLK) 1 option selected. 10 Master Clock (MCLK) 2 option selected. 11 Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0 Bit Clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1 Bit Clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction

Table continues on the next page...

I2Sx_RCR2 field descriptions (continued)

Field	Description
	Configures the direction of the bit clock. 0 Bit clock is generated externally in Slave mode. 1 Bit clock is generated internally in Master mode.
23–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

42.3.13 SAI Receive Configuration 3 Register (I2Sx_RCR3)

Address: 4004_C000h base + 8Ch offset = 4004_C08Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							RCE
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
R	0												WDFL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

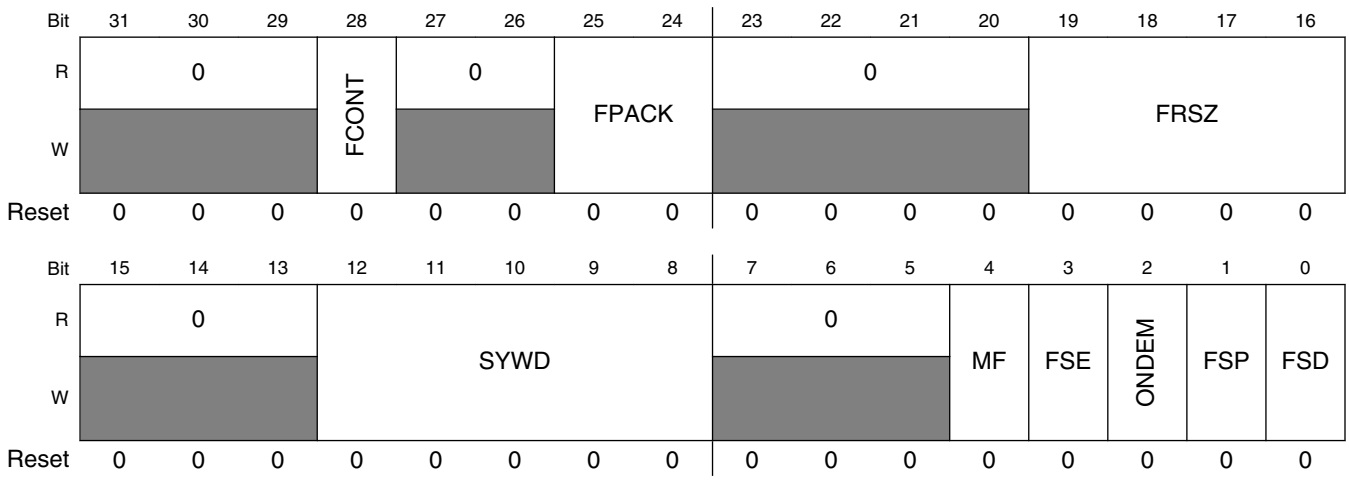
I2Sx_RCR3 field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 RCE	Receive Channel Enable Enables the corresponding data channel for receive operation. A channel must be enabled before its FIFO is accessed. Changing this field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for receive operation. 0 Receive data channel N is disabled. 1 Receive data channel N is enabled.
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
WDFL	Word Flag Configuration Configures which word the start of word flag is set. The value written should be one less than the word number (for example, write zero to configure for the first word in the frame). When configured to a value greater than the Frame Size field, then the start of word flag is never set.

42.3.14 SAI Receive Configuration 4 Register (I2Sx_RCR4)

This register must not be altered when RCSR[RE] is set.

Address: 4004_C000h base + 90h offset = 4004_C090h



I2Sx_RCR4 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 FCONT	FIFO Continue on Error Configures when the SAI will continue receiving after a FIFO error has been detected. 0 On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1 On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are stored to the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer will only increment when the full 32-bit FIFO word has been read by software. 00 FIFO packing is disabled 01 Reserved. 10 8-bit FIFO packing is enabled 11 16-bit FIFO packing is enabled
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

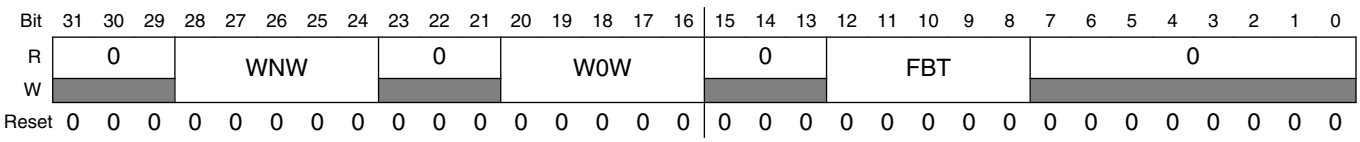
I2Sx_RCR4 field descriptions (continued)

Field	Description
19–16 FRSZ	Frame Size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 16 words.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 MF	MSB First Configures whether the LSB or the MSB is received first. 0 LSB is received first. 1 MSB is received first.
3 FSE	Frame Sync Early 0 Frame sync asserts with the first bit of the frame. 1 Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0 Internal frame sync is generated continuously. 1 Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0 Frame sync is active high. 1 Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0 Frame Sync is generated externally in Slave mode. 1 Frame Sync is generated internally in Master mode.

42.3.15 SAI Receive Configuration 5 Register (I2Sx_RCR5)

This register must not be altered when RCSR[RE] is set.

Address: 4004_C000h base + 94h offset = 4004_C094h



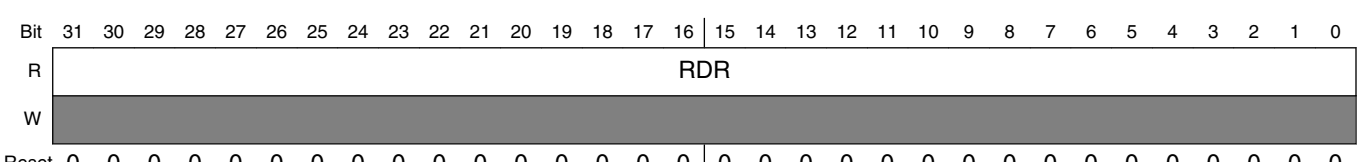
I2Sx_RCR5 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28–24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 FBT	First Bit Shifted Configures the bit index for the first bit received for each word in the frame. If configured for MSB First, the index of the next bit received is one less than the current bit received. If configured for LSB First, the index of the next bit received is one more than the current bit received. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

42.3.16 SAI Receive Data Register (I2Sx_RDR*n*)

Reading this register introduces one additional peripheral clock wait state on each read.

Address: 4004_C000h base + A0h offset + (4d × i), where i=0d to 0d



I2S*x*_RDR*n* field descriptions

Field	Description
RDR	Receive Data Register The corresponding RCR3[RCE] bit must be set before accessing the channel's receive data register. Reads from this register when the receive FIFO is not empty will return the data from the top of the receive FIFO. Reads from this register when the receive FIFO is empty are ignored.

42.3.17 SAI Receive FIFO Register (I2S*x*_RFR*n*)

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

Address: 4004_C000h base + C0h offset + (4d × i), where i=0d to 0d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													WFP		
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
R	0	0												RFP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2S*x*_RFR*n* field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 WFP	Write FIFO Pointer FIFO write pointer for receive data channel.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RFP	Read FIFO Pointer FIFO read pointer for receive data channel.

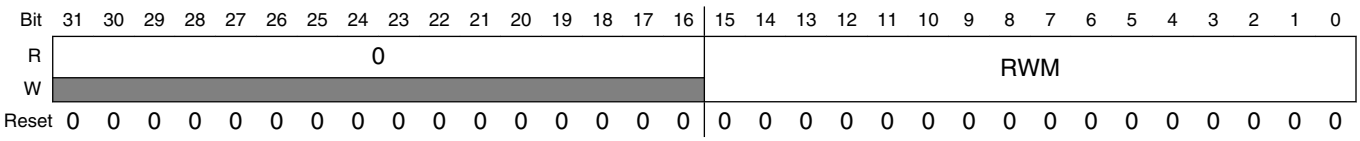
42.3.18 SAI Receive Mask Register (I2S*x*_RMR)

This register is double-buffered and updates:

1. When RCSR[RE] is first set
2. At the end of each frame

This allows the masked words in each frame to change from frame to frame.

Address: 4004_C000h base + E0h offset = 4004_C0E0h



I2Sx_RMR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 0 Word N is enabled. 1 Word N is masked.

42.4 Functional description

This section provides a complete functional description of the block.

42.4.1 SAI clocking

The SAI clocks include:

- The audio master clock
- The bit clock
- The bus clock

42.4.1.1 Audio master clock

The audio master clock is used to generate the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The transmitter and receiver can independently select between the bus clock and up to three audio master clocks to generate the bit clock.

The audio master clock generation and selection is chip-specific. Refer to chip-specific clocking information about how the audio master clocks are generated.

42.4.1.2 Bit clock

The SAI transmitter and receiver support asynchronous free-running bit clocks that can be generated internally from an audio master clock or supplied externally. There is also the option for synchronous bit clock and frame sync operation between the receiver and transmitter.

Externally generated bit clocks must be:

- Enabled before the SAI transmitter or receiver is enabled
- Disabled after the SAI transmitter or receiver is disabled and completes its current frames

If the SAI transmitter or receiver is using an externally generated bit clock in asynchronous mode and that bit clock is generated by an SAI that is disabled in stop mode, then the transmitter or receiver should be disabled by software before entering stop mode. This issue does not apply when the transmitter or receiver is in a synchronous mode because all synchronous SAIs are enabled and disabled simultaneously.

42.4.1.3 Bus clock

The bus clock is used by the control and configuration registers and to generate synchronous interrupts and DMA requests.

NOTE

Although there is no specific minimum bus clock frequency specified, the bus clock frequency must be fast enough (relative to the bit clock frequency) to ensure that the FIFOs can be serviced, without generating either a transmitter FIFO underrun or receiver FIFO overflow condition.

42.4.2 SAI resets

The SAI is asynchronously reset on system reset. The SAI has a software reset and a FIFO reset.

42.4.2.1 Software reset

The SAI transmitter includes a software reset that resets all transmitter internal logic, including the bit clock generation, status flags, and FIFO pointers. It does not reset the configuration registers. The software reset remains asserted until cleared by software.

The SAI receiver includes a software reset that resets all receiver internal logic, including the bit clock generation, status flags and FIFO pointers. It does not reset the configuration registers. The software reset remains asserted until cleared by software.

42.4.2.2 FIFO reset

The SAI transmitter includes a FIFO reset that synchronizes the FIFO write pointer to the same value as the FIFO read pointer. This empties the FIFO contents and is to be used after TCSR[FEF] is set, and before the FIFO is re-initialized and TCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI receiver includes a FIFO reset that synchronizes the FIFO read pointer to the same value as the FIFO write pointer. This empties the FIFO contents and is to be used after the RCSR[FEF] is set and any remaining data has been read from the FIFO, and before the RCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

42.4.3 Synchronous modes

The SAI transmitter and receiver can operate synchronously to each other.

42.4.3.1 Synchronous mode

The SAI transmitter and receiver can be configured to operate with synchronous bit clock and frame sync.

If the transmitter bit clock and frame sync are to be used by both the transmitter and receiver:

- The transmitter must be configured for asynchronous operation and the receiver for synchronous operation.
- In synchronous mode, the receiver is enabled only when both the transmitter and receiver are enabled.
- It is recommended that the transmitter is the last enabled and the first disabled.

If the receiver bit clock and frame sync are to be used by both the transmitter and receiver:

- The receiver must be configured for asynchronous operation and the transmitter for synchronous operation.
- In synchronous mode, the transmitter is enabled only when both the receiver and transmitter are both enabled.
- It is recommended that the receiver is the last enabled and the first disabled.

When operating in synchronous mode, only the bit clock, frame sync, and transmitter/receiver enable are shared. The transmitter and receiver otherwise operate independently, although configuration registers must be configured consistently across both the transmitter and receiver.

42.4.4 Frame sync configuration

When enabled, the SAI continuously transmits and/or receives frames of data. Each frame consists of a fixed number of words and each word consists of a fixed number of bits. Within each frame, any given word can be masked causing the receiver to ignore that word and the transmitter to tri-state for the duration of that word.

The frame sync signal is used to indicate the start of each frame. A valid frame sync requires a rising edge (if active high) or falling edge (if active low) to be detected and the transmitter or receiver cannot be busy with a previous frame. A valid frame sync is also ignored (slave mode) or not generated (master mode) for the first four bit clock cycles after enabling the transmitter or receiver.

The transmitter and receiver frame sync can be configured independently with any of the following options:

- Externally generated or internally generated
- Active high or active low
- Assert with the first bit in frame or asserts one bit early
- Assert for a duration between 1 bit clock and the first word length
- Frame length from 1 to 16 words per frame
- Word length to support 8 to 32 bits per word

    - First word length and remaining word lengths can be configured separately
- Words can be configured to transmit/receive MSB first or LSB first

These configuration options cannot be changed after the SAI transmitter or receiver is enabled.

42.4.5 Data FIFO

Each transmit and receive channel includes a FIFO of size 4 × 32-bit. The FIFO data is accessed using the SAI Transmit/Receive Data Registers.

42.4.5.1 Data alignment

Data in the FIFO can be aligned anywhere within the 32-bit wide register through the use of the First Bit Shifted configuration field, which selects the bit index (between 31 and 0) of the first bit shifted.

Examples of supported data alignment and the required First Bit Shifted configuration are illustrated in Figure 42-2 for LSB First configurations and Figure 42-3 for MSB First configurations.

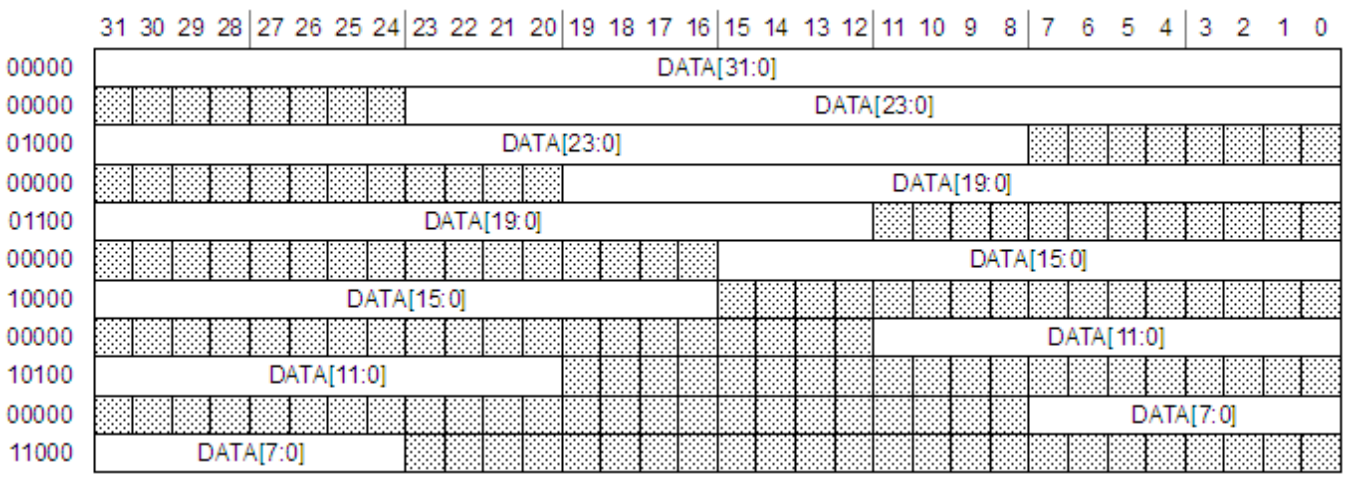


Figure 42-2. SAI first bit shifted, LSB first

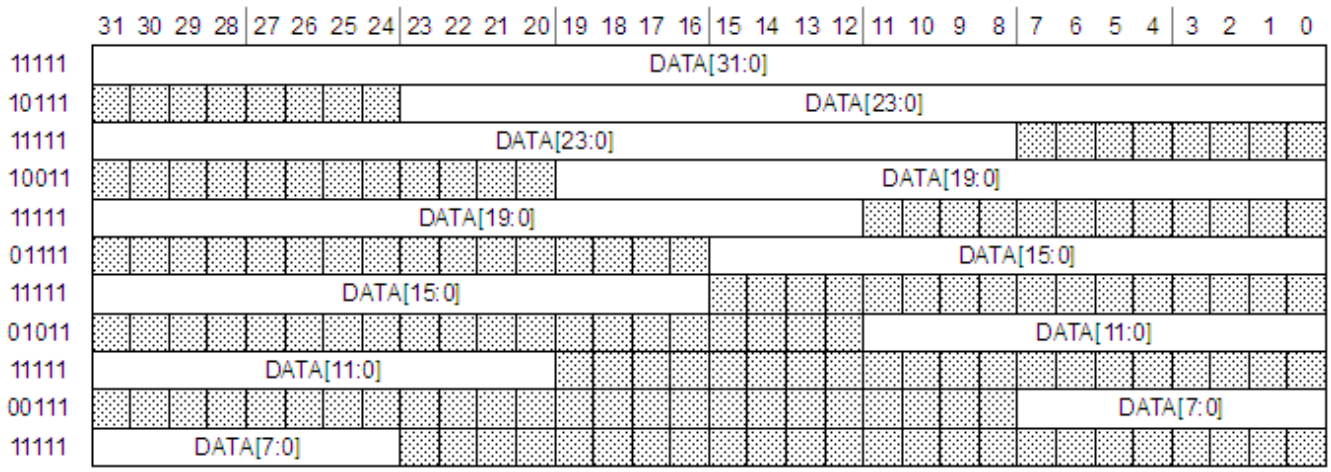


Figure 42-3. SAI first bit shifted, MSB first

42.4.5.2 FIFO pointers

When writing to a TDR, the WFP of the corresponding TFR increments after each valid write. The SAI supports 8-bit, 16-bit and 32-bit writes to the TDR and the FIFO pointer will increment after each individual write. Note that 8-bit writes should only be used when transmitting up to 8-bit data and 16-bit writes should only be used when transmitting up to 16-bit data.

Writes to a TDR are ignored if the corresponding bit of TCR3[TCE] is clear or if the FIFO is full. If the Transmit FIFO is empty, the TDR must be written at least three bit clocks before the start of the next unmasked word to avoid a FIFO underrun.

When reading an RDR, the RFP of the corresponding RFR increments after each valid read. The SAI supports 8-bit, 16-bit and 32-bit reads from the RDR and the FIFO pointer will increment after each individual read. Note that 8-bit reads should only be used when receiving up to 8-bit data and 16-bit reads should only be used when receiving up to 16-bit data.

Reads from an RDR are ignored if the corresponding bit of RCR3[RCE] is clear or if the FIFO is empty. If the Receive FIFO is full, the RDR must be read at least three bit clocks before the end of an unmasked word to avoid a FIFO overrun.

42.4.5.3 FIFO packing

FIFO packing supports storing multiple 8-bit or 16-bit data words in one 32-bit FIFO word for the transmitter and/or receiver. While this can be emulated by adjusting the number of bits per word and number of words per frame (for example, one 32-bit word

per frame versus two 16-bit words per frame), FIFO packing does not require even multiples of words per frame and fully supports word masking. When FIFO packing is enabled, the FIFO pointers only increment when the full 32-bit FIFO word has been written (transmit) or read (receive) by software, supporting scenarios where different words within each frame are loaded/stored in different areas of memory.

When 16-bit FIFO packing is enabled for transmit, the transmit shift register is loaded at the start of each frame and after every second unmasked transmit word. The first word transmitted is taken from 16-bit word at byte offset $0 (first bit is selected by TCFG5[FBT] must be configured within this 16-bit word) and the second word transmitted is taken from the 16-bit word at byte offset $2 (first bit is selected by TCSR5[FBT][3:0]). The transmitter will transmit logic zero until the start of the next word once the 16-bit word has been transmitted.

When 16-bit FIFO packing is enabled for receive, the receive shift register is stored after every second unmasked received word, and at the end of each frame if there is an odd number of unmasked received words in each frame. The first word received is stored in the 16-bit word at byte offset $0 (first bit is selected by RCFG5[FBT] and must be configured within this 16-bit word) and the second word received is stored in the 16-bit word at byte offset $2 (first bit is selected by RCSR5[FBT][3:0]). The receiver will ignore received data until the start of the next word once the 16-bit word has been received.

The 8-bit FIFO packing is similar to 16-bit packing except four words are loaded or stored into each 32-bit FIFO word. The first word is loaded/stored in byte offset $0, second word in byte offset $1, third word in byte offset $2 and fourth word in byte offset $3. The TCFG5[FBT] and/or RCFG5[FBT] must be configured within byte offset $0.

42.4.6 Word mask register

The SAI transmitter and receiver each contain a word mask register, namely TMR and RMR, that can be used to mask any word in the frame. Because the word mask register is double buffered, software can update it before the end of each frame to mask a particular word in the next frame.

The TMR causes the Transmit Data pin to be tri-stated for the length of each selected word and the transmit FIFO is not read for masked words.

The RMR causes the received data for each selected word to be discarded and not written to the receive FIFO.

42.4.7 Interrupts and DMA requests

The SAI transmitter and receiver generate separate interrupts and separate DMA requests, but support the same status flags. Asynchronous versions of the transmitter and receiver interrupts are generated to wake up the CPU from stop mode.

42.4.7.1 FIFO request flag

The FIFO request flag is set based on the number of entries in the FIFO and the FIFO watermark configuration.

The transmit FIFO request flag is set when the number of entries in any of the enabled transmit FIFOs is less than or equal to the transmit FIFO watermark configuration and is cleared when the number of entries in each enabled transmit FIFO is greater than the transmit FIFO watermark configuration.

The receive FIFO request flag is set when the number of entries in any of the enabled receive FIFOs is greater than the receive FIFO watermark configuration and is cleared when the number of entries in each enabled receive FIFO is less than or equal to the receive FIFO watermark configuration.

The FIFO request flag can generate an interrupt or a DMA request.

42.4.7.2 FIFO warning flag

The FIFO warning flag is set based on the number of entries in the FIFO.

The transmit warning flag is set when the number of entries in any of the enabled transmit FIFOs is empty and is cleared when the number of entries in each enabled transmit FIFO is not empty.

The receive warning flag is set when the number of entries in any of the enabled receive FIFOs is full and is cleared when the number of entries in each enabled receive FIFO is not full.

The FIFO warning flag can generate an Interrupt or a DMA request.

42.4.7.3 FIFO error flag

The transmit FIFO error flag is set when the any of the enabled transmit FIFOs underflow. After it is set, all enabled transmit channels will transmit zero until TCSR[FEF] is cleared and the next transmit frame starts. All enabled transmit FIFOs must be reset and initialized with new data before TCSR[FEF] is cleared.

When TCR4[FCONT] is set, the FIFO will continue transmitting data following an underflow without software intervention. To ensure that data is transmitted in the correct order, the transmitter will continue from the same word number in the frame that caused the FIFO to underflow, but only after new data has been written to the transmit FIFO. Software should still clear the TCSR[FEF] flag, but without reinitializing the transmit FIFOs.

RCSR[FEF] is set when the any of the enabled receive FIFOs overflow. After it is set, all enabled receive channels discard received data until RCSR[FEF] is cleared and the next next receive frame starts. All enabled receive FIFOs should be emptied before RCSR[FEF] is cleared.

When RCR4[FCONT] is set, the FIFO will continue receiving data following an overflow without software intervention. To ensure that data is received in the correct order, the receiver will continue from the same word number in the frame that caused the FIFO to overflow, but only after data has been read from the receive FIFO. Software should still clear the RCSR[FEF] flag, but without emptying the receive FIFOs.

The FIFO error flag can generate only an interrupt.

42.4.7.4 Sync error flag

The sync error flag, TCSR[SEF] or RCSR[SEF], is set when configured for an externally generated frame sync and the external frame sync asserts when the transmitter or receiver is busy with the previous frame. The external frame sync assertion is ignored and the sync error flag is set. When the sync error flag is set, the transmitter or receiver continues checking for frame sync assertion when idle or at the end of each frame.

The sync error flag can generate an interrupt only.

42.4.7.5 Word start flag

The word start flag is set at the start of the second bit clock for the selected word, as configured by the Word Flag register field.

The word start flag can generate an interrupt only.

Chapter 43
System Clock Generator (SCG)

43.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The system clock generator (SCG) module provides the system clocks of the MCU. The SCG contains a system phase-locked loop (SPLL) , a slow internal reference clock (SIRC), a fast internal reference clock (FIRC), and the system oscillator clock (SOSC). The PLLs are sourced by either the SOSC reference clock or the FIRC reference clock. The SCG can select either the output clock of the SPLL or a SCG reference clock (SIRC, FIRC, and SOSC) as the source for the MCU system clocks. The SCG also supports operation with crystal oscillators, which allows an external crystal, ceramic resonator, or another external clock source to produce the external reference clock (which are also available as clock sources for the MCU systems clocks).

43.1.1 Features

Key features of the SCG module are:

- System Phase-locked loop (SPLL):
  - Voltage-controlled oscillator (VCO)
  - Selectable Internal or External reference clock is used as the PLL source
  - Modulo VCO frequency divider
  - Phase/Frequency detector
  - Integrated loop filter

    - Can be selected as the clock source for the MCU system clocks
    - 3 programmable post-dividers clock outputs, which can be used as clock sources for other on-chip peripherals
- 2 Internal reference clock (IRC) generators:
    - Slow IRC clock with programmable High and Low frequency range, with each range having a set of 8 trim bits for accuracy
    - Fast IRC clock with programmable High and Low frequency range, with 3 sets of trim bits for accuracy
    - Fast clock can be used as a source for System PLL
    - Either the slow or the fast clock can be selected as the clock source for the MCU system clocks
    - 3 programmable post-divider clock outputs for each IRC, which can be used as clock sources for other on-chip peripherals
- System Crystal Oscillator:
    - Can be used as a source for the System PLL
    - Can be selected as the clock source for the MCU system clocks
- Clock monitor with reset and interrupt request capability for SPLL, SOSC, clocks
- Each of the clock sources have reference dividers for clocking on-chip modules and peripherals, namely:
    - SPLLDIV1_CLK / SPLLDIV2_CLK / SPLLDIV3_CLK
    - FIRCDIV1_CLK / SCG_FIRCDIV2_CLK / SCG_FIRCDIV3_CLK
    - SIRCDIV1_CLK / SIRCDIV2_CLK / SIRCDIV3_CLK
    - SOSCDIV1_CLK / SOSCDIV2_CLK / SOSCDIV3_CLK

The next figure shows the SCG module block diagram.

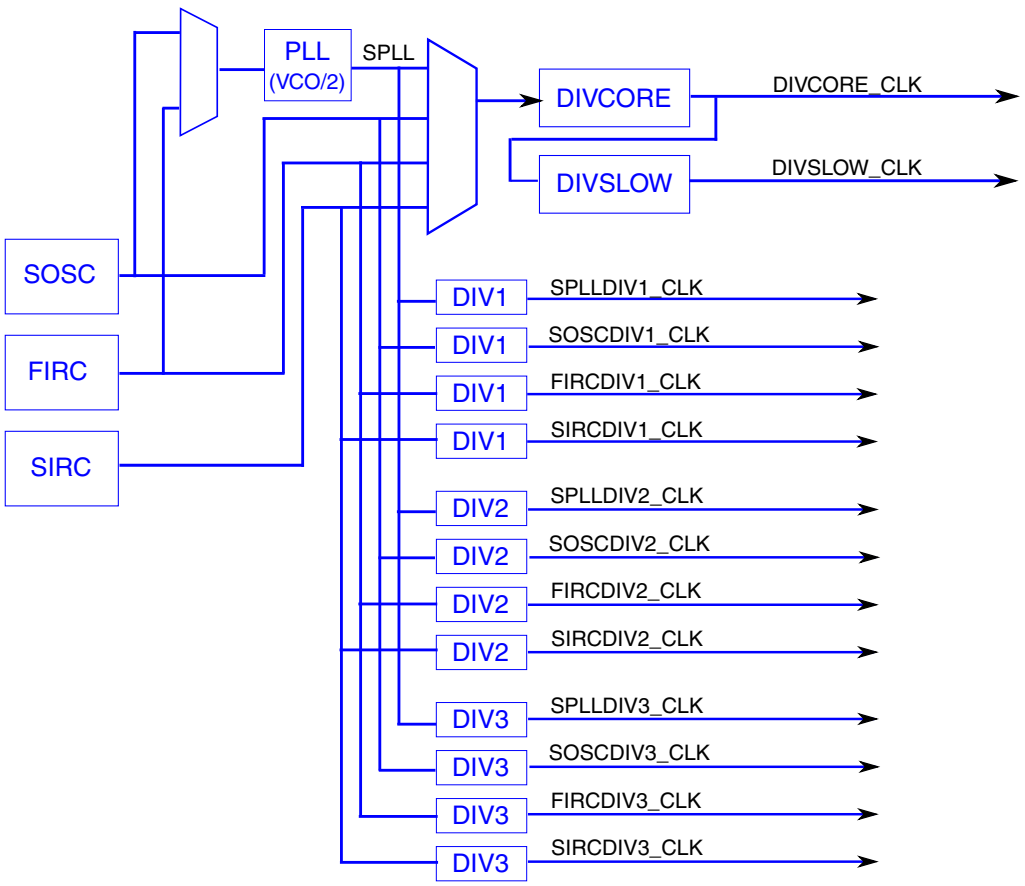


Figure 43-1. System Clock Generator (SCG) block diagram

NOTE

To identify the oscillator used in your specific MCU device, see the chip configuration chapter.

43.2 Memory Map/Register Definition

This section includes the memory map and register definition.

The SCG registers can only be written when in supervisor mode. Write accesses when in user mode will result in a bus transfer error. Read accesses may be performed in both supervisor and user mode.

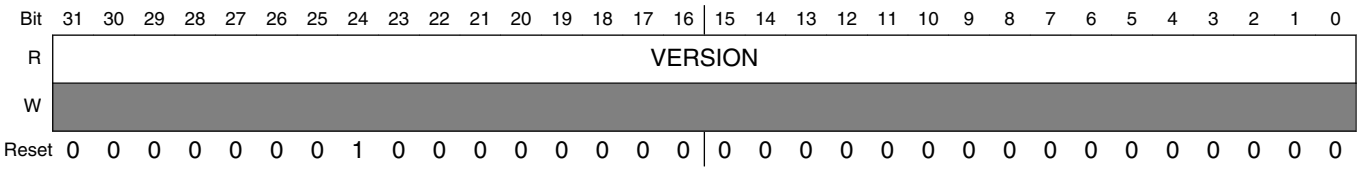
SCG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_B000	Version ID Register (SCG_VERID)	32	R	0100_0000h	43.2.1/1081
4007_B004	Parameter Register (SCG_PARAM)	32	R	See section	43.2.2/1081
4007_B010	Clock Status Register (SCG_CSR)	32	R	See section	43.2.3/1082
4007_B014	Run Clock Control Register (SCG_RCCR)	32	R/W	See section	43.2.4/1084
4007_B018	VLPR Clock Control Register (SCG_VCCR)	32	R/W	See section	43.2.5/1086
4007_B01C	HSRUN Clock Control Register (SCG_HCCR)	32	R/W	See section	43.2.6/1088
4007_B020	SCG CLKOUT Configuration Register (SCG_CLKOUTCNFG)	32	R/W	0200_0000h	43.2.7/1089
4007_B100	System OSC Control Status Register (SCG_SOSCCSR)	32	R/W	See section	43.2.8/1091
4007_B104	System OSC Divide Register (SCG_SOSCDIV)	32	R/W	0000_0000h	43.2.9/1093
4007_B108	System Oscillator Configuration Register (SCG_SOSCCFG)	32	R/W	0000_0010h	43.2.10/ 1094
4007_B200	Slow IRC Control Status Register (SCG_SIRCCSR)	32	R/W	0300_0005h	43.2.11/ 1096
4007_B204	Slow IRC Divide Register (SCG_SIRCDIV)	32	R/W	0000_0000h	43.2.12/ 1097
4007_B208	Slow IRC Configuration Register (SCG_SIRCCFG)	32	R/W	0000_0001h	43.2.13/ 1099
4007_B300	Fast IRC Control Status Register (SCG_FIRCCSR)	32	R/W	See section	43.2.14/ 1100
4007_B304	Fast IRC Divide Register (SCG_FIRCDIV)	32	R/W	0000_0000h	43.2.15/ 1102
4007_B308	Fast IRC Configuration Register (SCG_FIRCCFG)	32	R/W	0000_0000h	43.2.16/ 1103
4007_B30C	Fast IRC Trim Configuration Register (SCG_FIRCTCFG)	32	R/W	0000_0000h	43.2.17/ 1104
4007_B318	Fast IRC Status Register (SCG_FIRCSTAT)	32	R	See section	43.2.18/ 1105
4007_B600	System PLL Control Status Register (SCG_SPLLCSR)	32	R/W	0000_0000h	43.2.19/ 1106
4007_B604	System PLL Divide Register (SCG_SPLLDIV)	32	R/W	0000_0000h	43.2.20/ 1108
4007_B608	System PLL Configuration Register (SCG_SPLLCFG)	32	R/W	0000_0000h	43.2.21/ 1109

43.2.1 Version ID Register (SCG_VERID)

Note: Writing to this register will result in a transfer error.

Address: 4007_B000h base + 0h offset = 4007_B000h



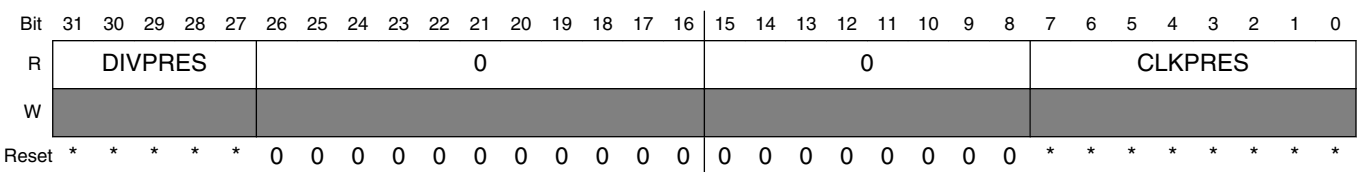
SCG_VERID field descriptions

Field	Description
VERSION	SCG Version Number

43.2.2 Parameter Register (SCG_PARAM)

Note: Writing to this register will result in a transfer error.

Address: 4007_B000h base + 4h offset = 4007_B004h



* Notes:
- DIVPRES field: The reset value is controlled by which SCG System Dividers are used by Soc.
- CLKPRES field: The reset value is controlled by which SCG Clock Sources are used by Soc. Please reference the Reference manual clocking chapter.

SCG_PARAM field descriptions

Field	Description
31–27 DIVPRES	Divider Present Indicates which system clock dividers are present in this instance of SCG. DIVPRES[27]=1 System DIVSLOW is present. DIVPRES[31]=1 System DIVCORE is present
26–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CLKPRES	Clock Present

Table continues on the next page...

SCG_PARAM field descriptions (continued)

Field	Description
	Indicates which clock sources are present in this instance of SCG. Any bits not defined in this bit field are Reserved and always has the value 0 when read. CLKPRES[0] Reserved CLKPRES[1]=1System OSC (SOSC) is present CLKPRES[2]=1Slow IRC (SIRC) is present CLKPRES[3]=1Fast IRC (FIRC) is present CLKPRES[6]=1System PLL (SPLL) is present

43.2.3 Clock Status Register (SCG_CSR)

This register returns the currently configured system clock source and the system clock dividers for the core (DIVCORE) and peripheral interface clock (DIVSLOW). The SCG_CSR reflects the configuration set by one of three clock control registers SCG_RCCR, SCG_VCCR, SCG_HCCR.

Note: Writing to this register will result in a transfer error.

Address: 4007_B000h base + 10h offset = 4007_B010h

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

* Notes:

- DIVCORE field: The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-1 or div-by-2 when resetting into RUN mode or div-by-4 or div-by-8 when resetting into VLPR mode

SCG_CSR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 SCS	System Clock Source Returns the currently configured clock source generating the system clock. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved

Table continues on the next page...

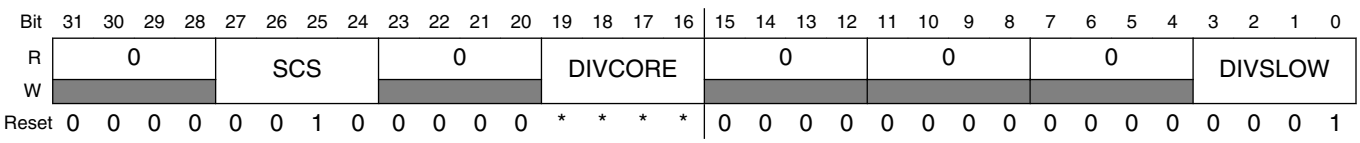
SCG_CSR field descriptions (continued)

Field	Description
	0110 System PLL (SPLL_CLK) 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIVSLOW	Slow Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16

43.2.4 Run Clock Control Register (SCG_RCCR)

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in Run mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in RUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in RUN, new system clock divide ratios will not take affect until new clock source is valid.

Address: 4007_B000h base + 14h offset = 4007_B014h



* Notes:
- DIVCORE field: The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-1 and div-by-2

SCG_RCCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 SCS	System Clock Source Selects the clock source generating the system clock in Run mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in Run mode requires that clock source to be enabled first and be valid before system clocks are allowed to switch to that clock source. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4

Table continues on the next page...

SCG_RCCR field descriptions (continued)

Field	Description
	0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIVSLOW	Slow Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16

43.2.5 VLPR Clock Control Register (SCG_VCCR)

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in VLPR mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in VLPR requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in VLPR, new system clock divide ratios will not take affect until new clock source is valid.

Address: 4007_B000h base + 18h offset = 4007_B018h

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

* Notes:

- DIVCORE field: The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-4 and div-by-8.

SCG_VCCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 SCS	System Clock Source Selects the clock source generating the system clock in VLPR mode. Attempting to select a clock that is not valid will be ignored. Selects the clock source generating the system clock. Selecting a different clock source when in VLPR mode requires that clock source to be enabled first and be valid before system clocks switch to that clock source. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3

Table continues on the next page...

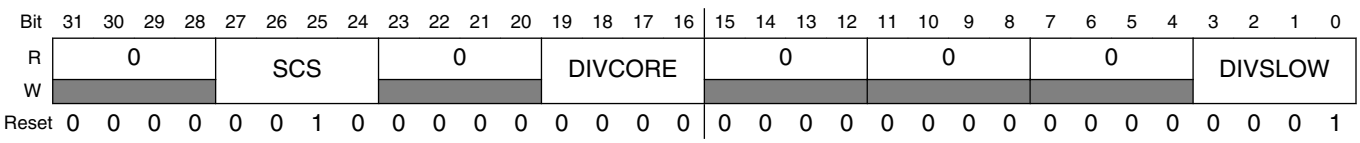
SCG_VCCR field descriptions (continued)

Field	Description
	0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIVSLOW	Slow Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16

43.2.6 HSRUN Clock Control Register (SCG_HCCR)

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in HSRUN mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in HSRUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in HSRUN, new system clock divide ratios will not take affect until new clock source is valid.

Address: 4007_B000h base + 1Ch offset = 4007_B01Ch



SCG_HCCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 SCS	System Clock Source Selects the clock source generating the system clock in HSRUN mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in HSRUN mode will enable that clock source and switch to that clock mode when it is valid. 0000 Reserved 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 DIVCORE	Core Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8

Table continues on the next page...

SCG_HCCR field descriptions (continued)

Field	Description
	1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DIVSLOW	Slow Clock Divide Ratio 0000 Divide-by-1 0001 Divide-by-2 0010 Divide-by-3 0011 Divide-by-4 0100 Divide-by-5 0101 Divide-by-6 0110 Divide-by-7 0111 Divide-by-8 1000 Divide-by-9 1001 Divide-by-10 1010 Divide-by-11 1011 Divide-by-12 1100 Divide-by-13 1101 Divide-by-14 1110 Divide-by-15 1111 Divide-by-16

43.2.7 SCG CLKOUT Configuration Register (SCG_CLKOUTCNFG)

This register controls which SCG clock source is selected to be ported out to the CLKOUT pin.

Address: 4007_B000h base + 20h offset = 4007_B020h

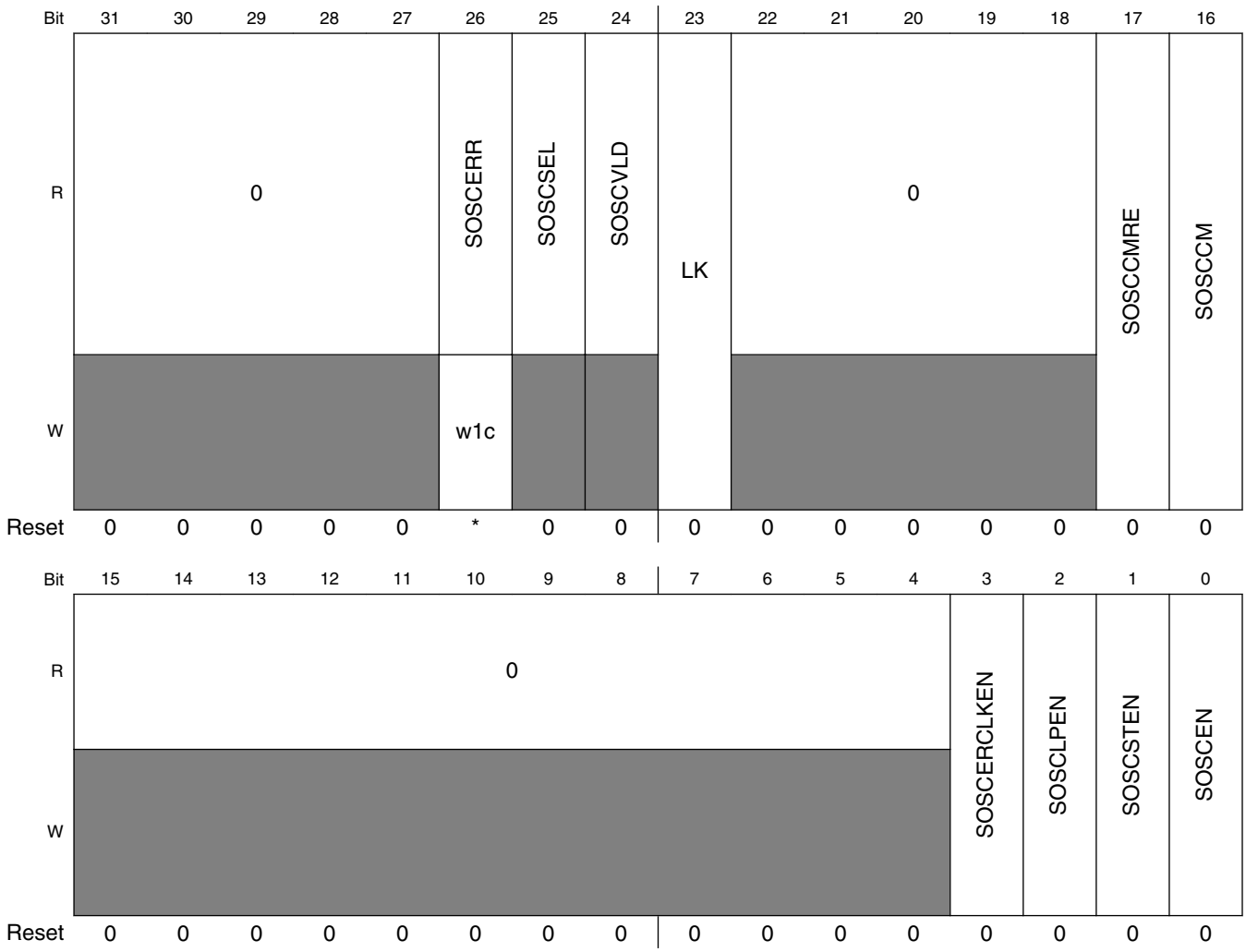
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				CLKOUTSEL				0																							
W																																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SCG_CLKOUTCNFG field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 CLKOUTSEL	SCG Clkout Select Selects the SCG system clock. 0000 SCG SLOW Clock 0001 System OSC (SOSC_CLK) 0010 Slow IRC (SIRC_CLK) 0011 Fast IRC (FIRC_CLK) 0100 Reserved 0101 Reserved 0110 System PLL (SPLL_CLK) 0111 Reserved 1111 Reserved
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.2.8 System OSC Control Status Register (SCG_SOSCCSR)

Address: 4007_B000h base + 100h offset = 4007_B100h



* Notes:
- SOSCERR field: This flag is reset on Chip POR only

SCG_SOSCCSR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SOSCERR	System OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one. 0 System OSC Clock Monitor is disabled or has not detected an error 1 System OSC Clock Monitor is enabled and detected an error

Table continues on the next page...

SCG_SOSCCSR field descriptions (continued)

Field	Description
25 SOSCSEL	System OSC Selected 0 System OSC is not the system clock source 1 System OSC is the system clock source
24 SOSCVLD	System OSC Valid The SOSC is considered valid after 4096 xtal counts. 0 System OSC is not enabled or clock is not valid 1 System OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0 This Control Status Register can be written. 1 This Control Status Register cannot be written.
22–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 SOSCCMRE	System OSC Clock Monitor Reset Enable 0 Clock Monitor generates interrupt when error detected 1 Clock Monitor generates reset when error detected
16 SOSCCM	System OSC Clock Monitor Enables the clock monitor when SOSCVLD is set. If the clock source is disabled in a low power mode then the clock monitor is also disabled in the low power mode. The clock monitor is always disabled in LLS/VLLS modes. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. 0 System OSC Clock Monitor is disabled 1 System OSC Clock Monitor is enabled
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SOSCERCLKEN	System OSC 3V ERCLK Enable SOSCERCLKEN is required for stop modes. 0 System OSC 3V ERCLK output clock is disabled. 1 System OSC 3V ERCLK output clock is enabled when SYSOSC is enabled.
2 SOSCLPEN	System OSC Low Power Enable SOSCLPEN is required for low power modes. In VLPS mode (low power stop mode), if you want the clock to remain ON, then both SOSCLPEN and SOSCSTEN bits must be enabled. 0 System OSC is disabled in VLP modes 1 System OSC is enabled in VLP modes
1 SOSCSTEN	System OSC Stop Enable 0 System OSC is disabled in Stop modes 1 System OSC is enabled in Stop modes if SOSCEN=1. In VLLS0, system oscillator is disabled even if SOSCSTEN=1 and SOSCEN=1.

Table continues on the next page...

SCG_SOSCCSR field descriptions (continued)

Field	Description
0 SOSCEN	System OSC Enable 0 System OSC is disabled 1 System OSC is enabled

43.2.9 System OSC Divide Register (SCG_SOSCDIV)

The SCG_SOSCDIV register provides the control of 3 clock trees which can be used to provide optional peripheral functional clocks, or alternative module clocks. Each clock tree has optional dividers of the input SOSC clock. Changes to SOSCDIV should be done when System OSC is disabled to prevent glitches to output divided clock.

Address: 4007_B000h base + 104h offset = 4007_B104h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													SOSCDIV3		
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
R	0					SOSCDIV2			0					SOSCDIV1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SCG_SOSCDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 SOSCDIV3	System OSC Clock Divide 3 Clock divider 3 for System OSC. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 SOSCDIV2	System OSC Clock Divide 2 Clock divider 2 for System OSC.

Table continues on the next page...

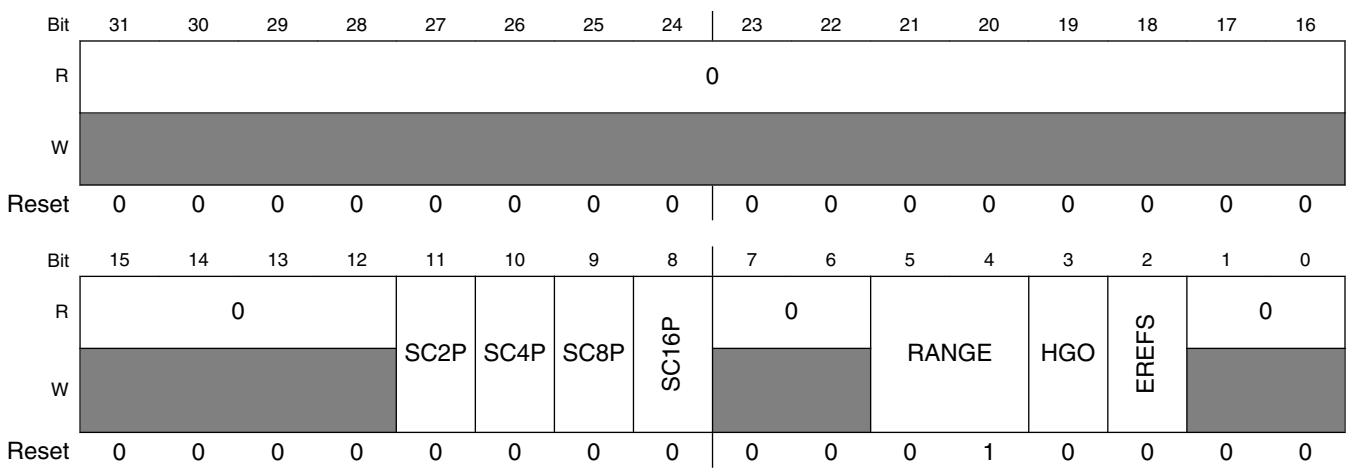
SCG_SOSCDIV field descriptions (continued)

Field	Description
	000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SOSCDIV1	System OSC Clock Divide 1 Clock divider 1 for System OSC. Used to generate the system clock source and by platform clock modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

43.2.10 System Oscillator Configuration Register (SCG_SOSCCFG)

The SOSCCFG register cannot be changed when the System OSC is enabled. When the System OSC is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4007_B000h base + 108h offset = 4007_B108h

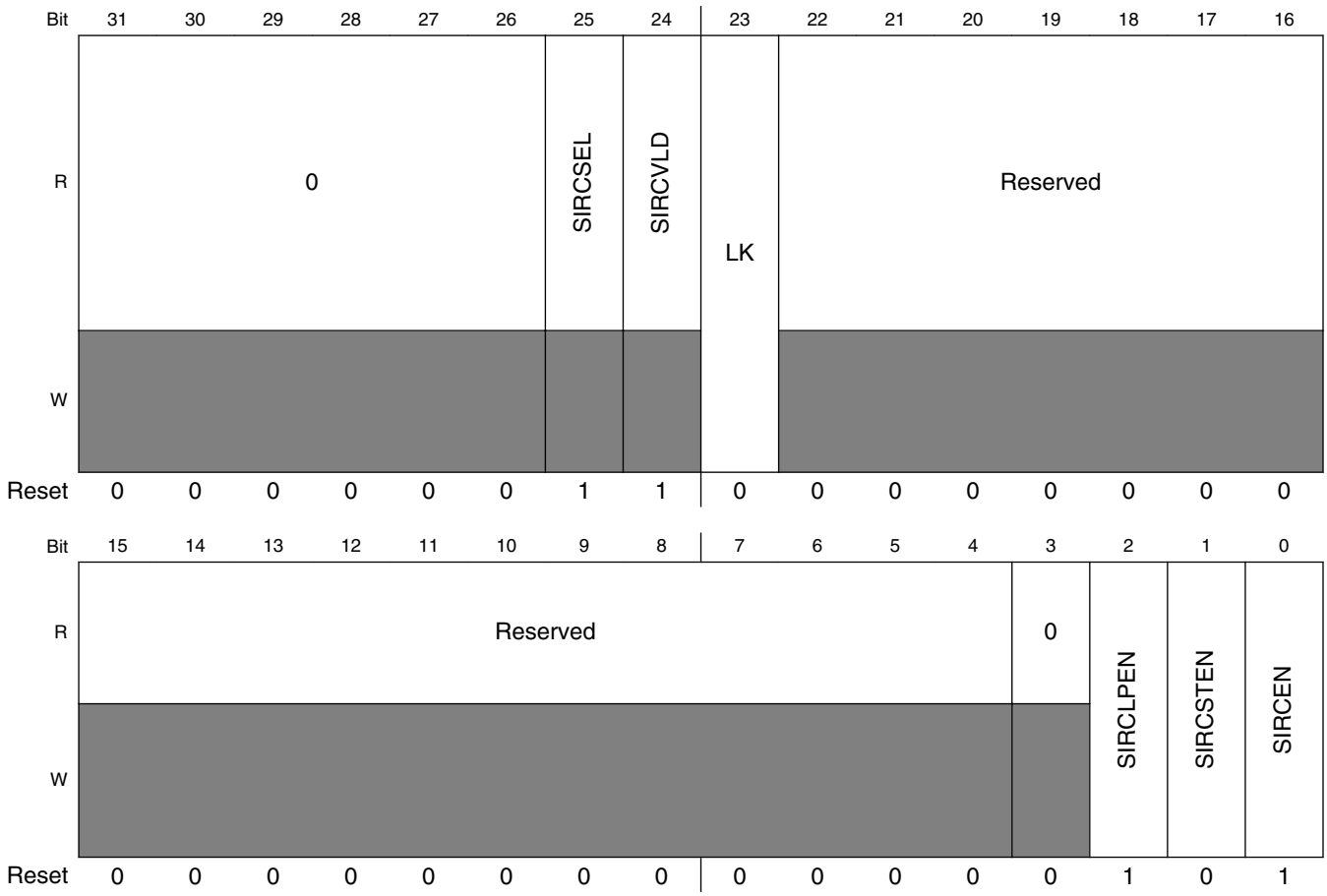


SCG_SOSCCFG field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 SC2P	Oscillator 2 pF Capacitor Load
10 SC4P	Oscillator 4 pF Capacitor Load
9 SC8P	Oscillator 8 pF Capacitor Load Configure
8 SC16P	Oscillator 16 pF Capacitor Load
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 RANGE	System OSC Range Select Selects the frequency range for the system crystal oscillator (OSC) 00 Reserved 01 Low frequency range selected for the crystal oscillator of 32 kHz to 40 kHz. 10 Medium frequency range selected for the crytstal oscillator of 1 Mhz to 8 Mhz. 11 High frequency range selected for the crystal oscillator of 8 Mhz to 32 Mhz.
3 HGO	High Gain Oscillator Select Controls the crystal oscillator power mode of operations. 0 Configure crystal oscillator for low-power operation 1 Configure crystal oscillator for high-gain operation
2 EREFS	External Reference Select Selects the source for the external reference clock. This bit selects which clock is output from the System OSC (SOSC) into the SCG, thus either the crystal oscillator or from an external clock input 0 External reference clock selected 1 Internal crystal oscillator of OSC requested. In VLLS0, the internal oscillator of OSC is disabled even if SOSCEN=1 and SOSCSTEN=1.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.2.11 Slow IRC Control Status Register (SCG_SIRCCSR)

Address: 4007_B000h base + 200h offset = 4007_B200h



SCG_SIRCCSR field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 SIRCSEL	Slow IRC Selected 0 Slow IRC is not the system clock source 1 Slow IRC is the system clock source
24 SIRCVLD	Slow IRC Valid 0 Slow IRC is not enabled or clock is not valid 1 Slow IRC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time.

Table continues on the next page...

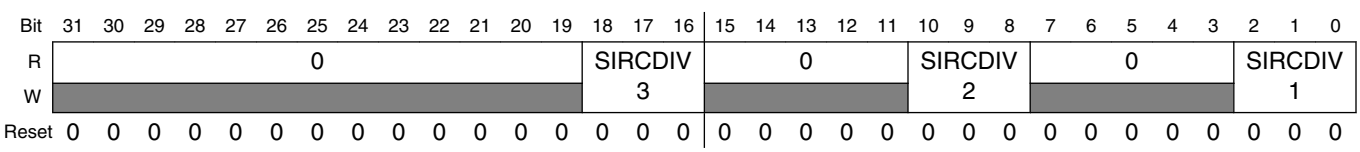
SCG_SIRCCSR field descriptions (continued)

Field	Description
	0 Control Status Register can be written. 1 Control Status Register cannot be written.
22–4 Reserved	This field is reserved and is always has the value 0 This field is reserved.
3 Reserved	This field is reserved and is always has the value 0 This field is reserved. This read-only field is reserved and always has the value 0.
2 SIRCLPEN	Slow IRC Low Power Enable 0 Slow IRC is disabled in VLP modes 1 Slow IRC is enabled in VLP modes
1 SIRCSTEN	Slow IRC Stop Enable 0 Slow IRC is disabled in Stop modes 1 Slow IRC is enabled in Stop modes
0 SIRCEN	Slow IRC Enable 0 Slow IRC is disabled 1 Slow IRC is enabled

43.2.12 Slow IRC Divide Register (SCG_SIRCDIV)

To prevent glitches to the output divided clock, change SIRDIV when the Slow IRC is disabled.

Address: 4007_B000h base + 204h offset = 4007_B204h



SCG_SIRCDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 SIRCDIV3	Slow IRC Clock Divider 3 Clock divider 3 for Slow IRC. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4

Table continues on the next page...

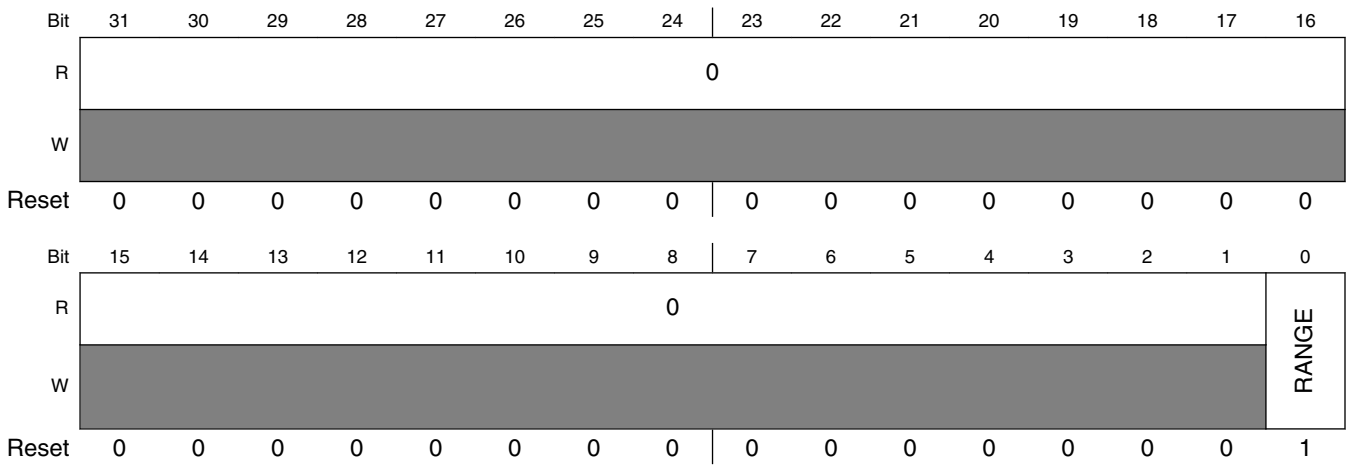
SCG_SIRCDIV field descriptions (continued)

Field	Description
	100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 SIRCDIV2	Slow IRC Clock Divide 2 Clock divider 2 for Slow IRC. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SIRCDIV1	Slow IRC Clock Divide 1 Clock divider 1 for Slow IRC. Used to generate the system clock source and by platform clock modules that need an asynchronous clock source. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

43.2.13 Slow IRC Configuration Register (SCG_SIRCCFG)

The SIRCCFG register cannot be changed when the slow IRC clock is enabled. When the slow IRC clock is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4007_B000h base + 208h offset = 4007_B208h

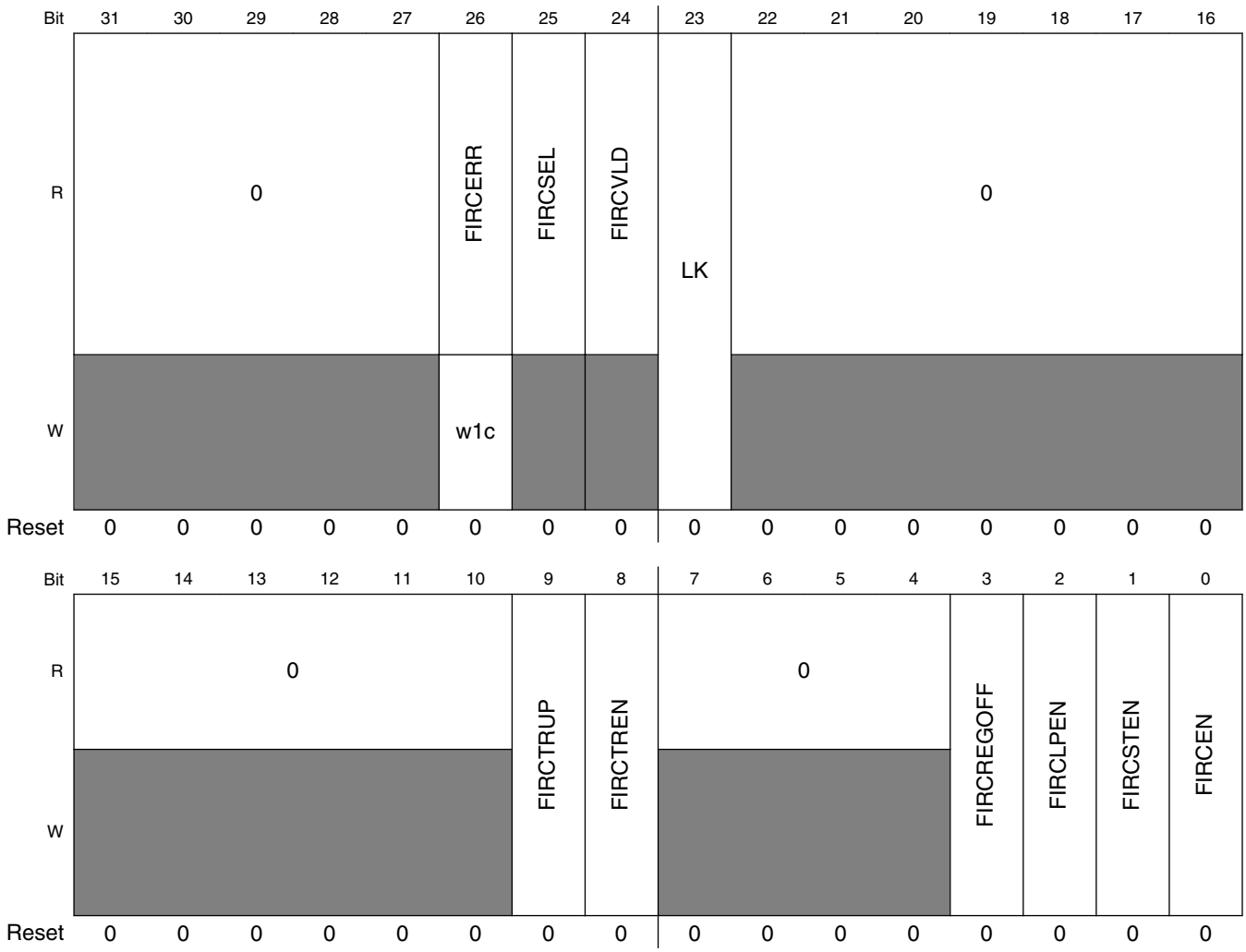


SCG_SIRCCFG field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 RANGE	Frequency Range 0 Slow IRC low range clock (2 MHz) 1 Slow IRC high range clock (8 MHz)

43.2.14 Fast IRC Control Status Register (SCG_FIRCCSR)

Address: 4007_B000h base + 300h offset = 4007_B300h



SCG_FIRCCSR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 FIRCERR	Fast IRC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0 Error not detected with the Fast IRC trimming. 1 Error detected with the Fast IRC trimming.
25 FIRCSEL	Fast IRC Selected status 0 Fast IRC is not the system clock source 1 Fast IRC is the system clock source

Table continues on the next page...

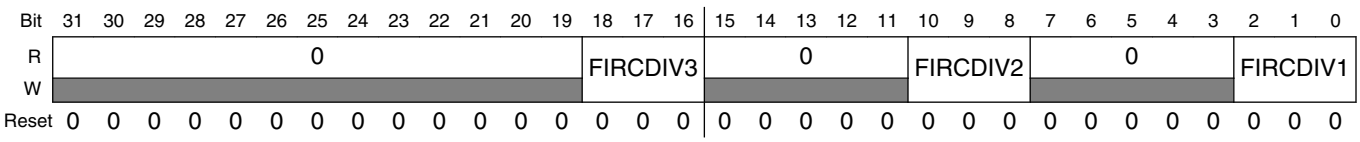
SCG_FIRCCSR field descriptions (continued)

Field	Description
24 FIRCVLD	Fast IRC Valid status 0 Fast IRC is not enabled or clock is not valid. 1 Fast IRC is enabled and output clock is valid. The clock is valid once there is an output clock from the FIRC analog.
23 LK	Lock Register This bit field can be cleared/set at any time. 0 Control Status Register can be written. 1 Control Status Register cannot be written.
22–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 FIRCTRUP	Fast IRC Trim Update 0 Disable Fast IRC trimming updates 1 Enable Fast IRC trimming updates
8 FIRCTREN	Fast IRC Trim Enable 0 Disable trimming Fast IRC to an external clock source 1 Enable trimming Fast IRC to an external clock source
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FIRCREGOFF	Fast IRC Regulator Enable 0 Fast IRC Regulator is enabled. 1 Fast IRC Regulator is disabled.
2 FIRCLPEN	Fast IRC Low Power Enable 0 Fast IRC is disabled in VLP modes 1 Fast IRC is enabled in VLP modes
1 FIRCSTEN	Fast IRC Stop Enable 0 Fast IRC is disabled in Stop modes. When selected as the reference clock to the System PLL and if the System PLL is enabled in STOP mode, the Fast IRC will stay enabled even if FIRCSTEN=0. 1 Fast IRC is enabled in Stop modes
0 FIRCEN	Fast IRC Enable 0 Fast IRC is disabled 1 Fast IRC is enabled

43.2.15 Fast IRC Divide Register (SCG_FIRCDIV)

Changes to FIRCDIV should be done when FAST IRC is disabled to prevent glitches to output divided clock.

Address: 4007_B000h base + 304h offset = 4007_B304h



SCG_FIRCDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 FIRCDIV3	Fast IRC Clock Divider 3 Clock divider 3 for the Fast IRC. 000 Clock disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 FIRCDIV2	Fast IRC Clock Divide 2 Clock divider 2 for the Fast IRC. 000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FIRCDIV1	Fast IRC Clock Divide 1 Clock divider 1 for Fast IRC. Used to generate the system clock source and by platform clock modules that need an asynchronous clock source.

Table continues on the next page...

SCG_FIRCDIV field descriptions (continued)

Field	Description
	000 Output disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

43.2.16 Fast IRC Configuration Register (SCG_FIRCCFG)

The FIRCCFG register cannot be changed when the Fast IRC is enabled. When the Fast IRC is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4007_B000h base + 308h offset = 4007_B308h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0														RANGE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SCG_FIRCCFG field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RANGE	Frequency Range 00 Fast IRC is trimmed to 48 MHz 01 Fast IRC is trimmed to 52 MHz 10 Fast IRC is trimmed to 56 MHz 11 Fast IRC is trimmed to 60 MHz

43.2.17 Fast IRC Trim Configuration Register (SCG_FIRCTCFG)

The FIRCTCFG register cannot be changed when Fast IRC tuning is enabled. When the Fast IRC tuning is enabled, writes to this register are ignored, and there is no transfer error.

Address: 4007_B000h base + 30Ch offset = 4007_B30Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0					TRIMDIV			0						TRIMSRC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

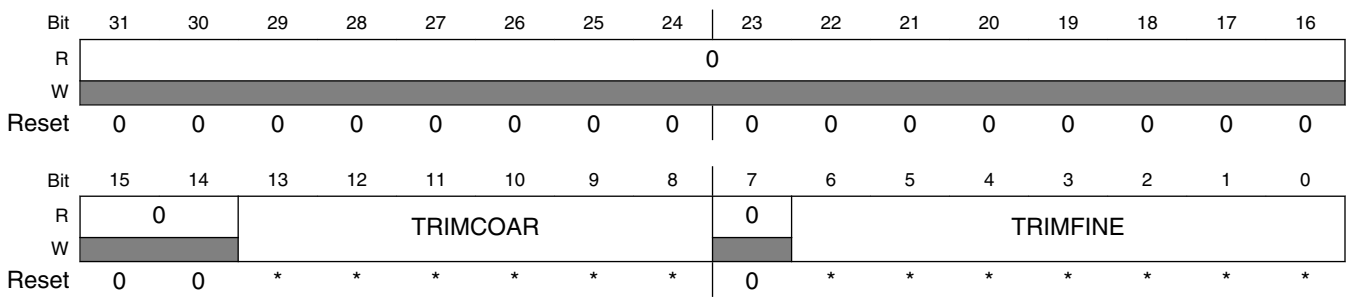
SCG_FIRCTCFG field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 TRIMDIV	Fast IRC Trim Predivide Divide the System OSC down for Fast IRC trimming. 000 Divide by 1 001 Divide by 128 010 Divide by 256 011 Divide by 512 100 Divide by 1024 101 Divide by 2048 110 Reserved. Writing this value will result in Divide by 1. 111 Reserved. Writing this value will result in a Divide by 1.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRIMSRC	Trim Source Configures the external clock source to tune the Fast IRC. TRMSRC must be configured before programming FIRCSTAT register for trim update 00 USB0 Start of Frame (1 kHz) 01 Reserved 10 System OSC 11 Reserved

43.2.18 Fast IRC Status Register (SCG_FIRCSTAT)

This register is loaded from IFR during reset. These register gets uploaded with the trim values generated by FIRC auto trimming which is enabled when FIRC is enabled and FIRCTREN=1 and FIRCTRUP=1. When FIRC auto trimming is enabled and FIRCTRUP is off (Note: TRIMSRC needs to be programmed to TRIMSRC=10 or TRIMSRC=11), writes to this register is allowed and values written to this register are used to trim FIRC clock.

Address: 4007_B000h base + 318h offset = 4007_B318h



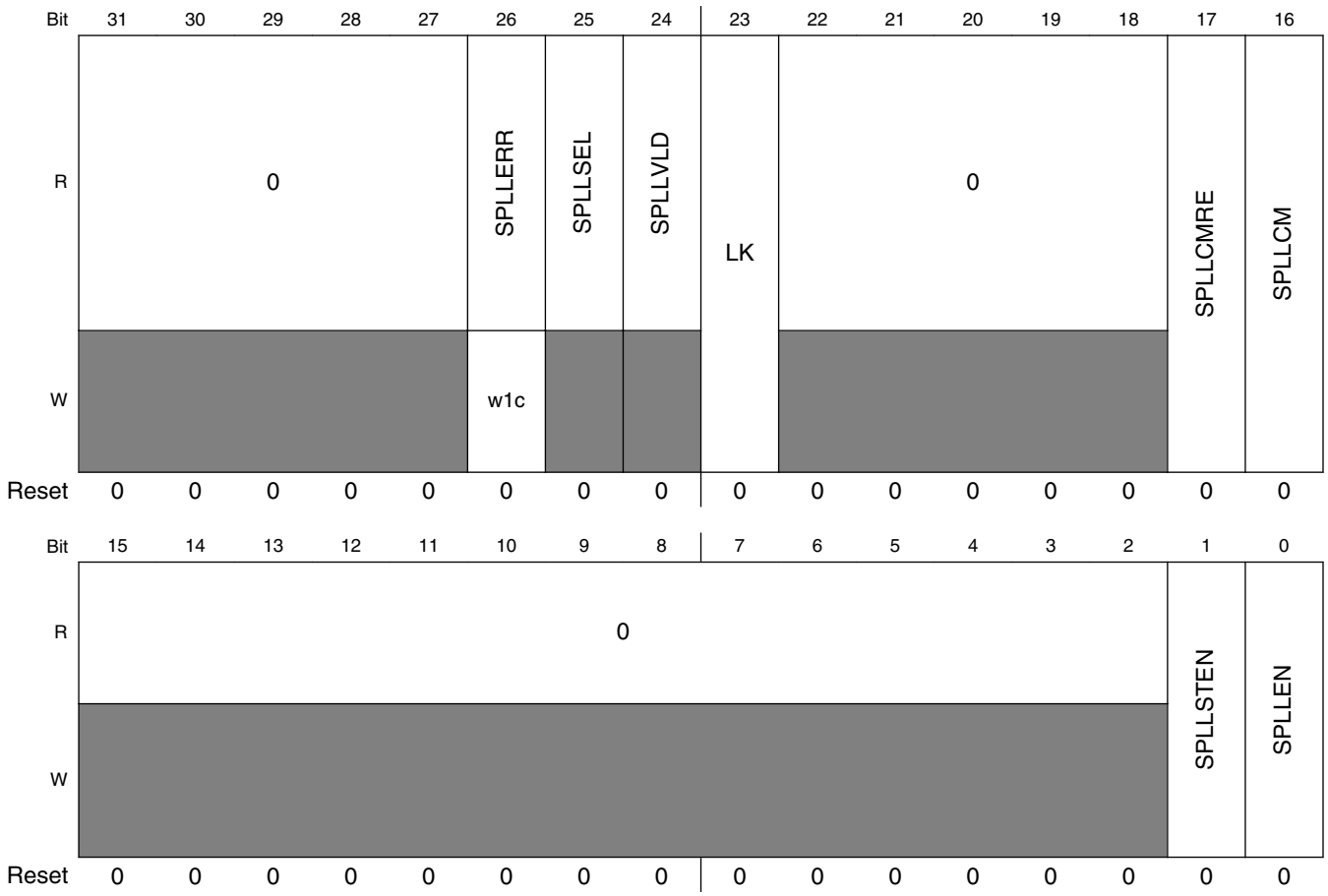
* Notes:
- TRIMCOAR field: Reset values are loaded out of IFR.
- TRIMFINE field: Reset values are loaded out of IFR.

SCG_FIRCSTAT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 TRIMCOAR	Trim Coarse TRIMCOAR bits are used to coursely trim the Fast IRC Clock to within approximately ±0.7% of the target frequency. When FIRC is enabled and auto trimming is enabled (FIRCTREN=1 and FIRCTRUP=1), then TRIMCOAR register gets uploaded with the trimmed coarse value. When FIRCTRUP=0, TRIMCOAR register is writable, to allow user programming of coarse trim values.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRIMFINE	Trim Fine Status Once the Fast IRC Clock is trimmed to ±0.7% of the target frequency using the TRIMCOAR bits, the TRIMFINE bits can be used to trim the Fast IRC Clock to within ±0.04% of the target frequency. When FIRC is enabled and auto trimming is enabled (FIRCTREN=1 and FIRCTRUP=1), TRIMFINE register gets uploaded with the trimmed fine value. When FIRCTRUP=0, TRIMFINE register is writeable, to allow user programming of fine trim values.

43.2.19 System PLL Control Status Register (SCG_SPLLCSR)

Address: 4007_B000h base + 600h offset = 4007_B600h



SCG_SPLLCSR field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SPLLERR	System PLL Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0 System PLL Clock Monitor is disabled or has not detected an error 1 System PLL Clock Monitor is enabled and detected an error. System PLL Clock Error flag will not set when System OSC is selected as its source and SOSCERR has set.
25 SPLLSEL	System PLL Selected 0 System PLL is not the system clock source 1 System PLL is the system clock source
24 SPLLVLD	System PLL Valid

Table continues on the next page...

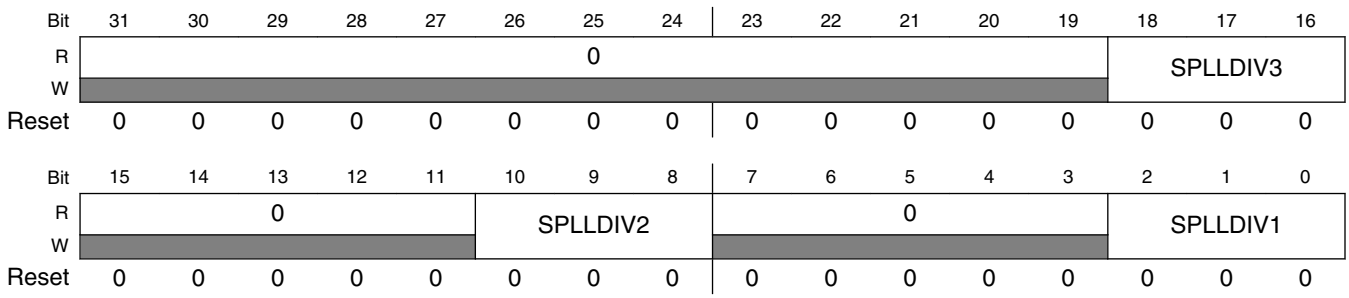
SCG_SPLLCSR field descriptions (continued)

Field	Description
	Indicates when the SPLL clock is valid. Disabling the SPLL or a SOSC error when selected as the reference clock to the SPLL will cause the SPLLVLD to clear without setting SPLLERROR. Lock detect is determined by a lock detect circuit. Three samples of lock detect determines whether or not the clock is valid. 0 System PLL is not enabled or clock is not valid 1 System PLL is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0 Control Status Register can be written. 1 Control Status Register cannot be written.
22–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 SPLLCMRE	System PLL Clock Monitor Reset Enable 0 Clock Monitor generates interrupt when error detected 1 Clock Monitor generates reset when error detected
16 SPLLCM	System PLL Clock Monitor Enables the clock monitor, if the clock source is disabled in a low power mode then the clock monitor is also disabled in the low power mode. The clock monitor is always disabled in LLS/VLLS modes. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. 0 System PLL Clock Monitor is disabled 1 System PLL Clock Monitor is enabled
15–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 SPLLSTEN	System PLL Stop Enable 0 System PLL is disabled in Stop modes 1 System PLL is enabled in Stop modes
0 SPLLEN	System PLL Enable 0 System PLL is disabled 1 System PLL is enabled

43.2.20 System PLL Divide Register (SCG_SPLLDIV)

Changes to SPLLDIV should be done when System PLL is disabled to prevent glitches to output divided clock.

Address: 4007_B000h base + 604h offset = 4007_B604h



SCG_SPLLDIV field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 SPLLDIV3	System PLL Clock Divide 3 Clock divider 3 for System PLL. 000 Clock disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 SPLLDIV2	System PLL Clock Divide 2 Clock divider 2 for System PLL. 000 Clock disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

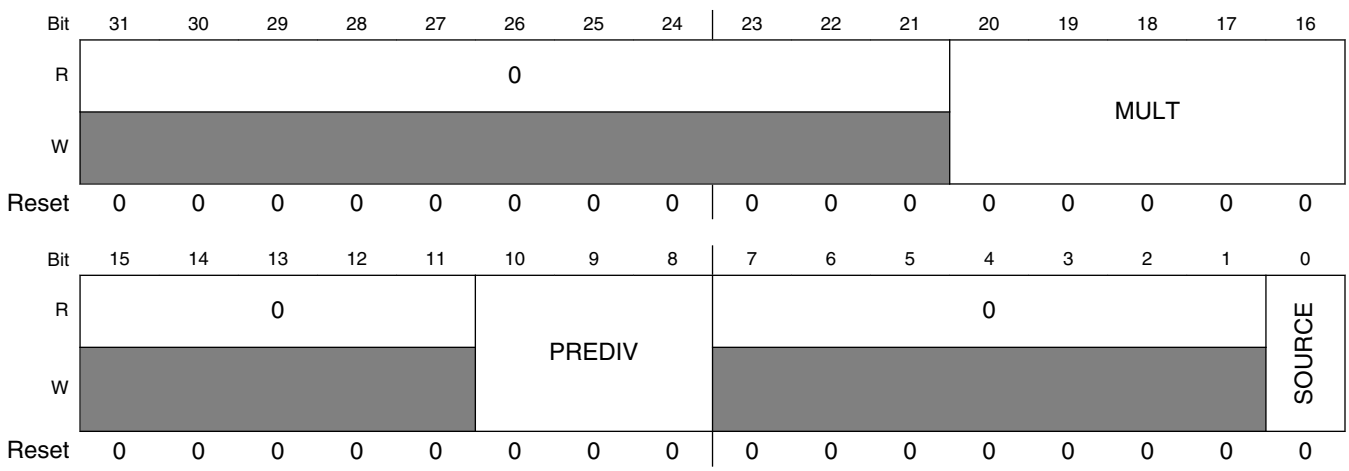
SCG_SPLLDIV field descriptions (continued)

Field	Description
SPLLDIV1	System PLL Clock Divide 1 Clock divider 1 for System PLL. Used to generate the system clock source used by platform clock modules that need an asynchronous clock source. 000 Clock disabled 001 Divide by 1 010 Divide by 2 011 Divide by 4 100 Divide by 8 101 Divide by 16 110 Divide by 32 111 Divide by 64

43.2.21 System PLL Configuration Register (SCG_SPLLCFG)

The SPLLCFG register cannot be changed when the System PLL is enabled. When the System PLL is enabled, writes to this register are ignored, and there is no transfer error. The below information applies to VCO_CLK_2x. The SPLL_CLK = (VCO_CLK_2x)/2. The VCO_CLK_2x = SOSC_CLK/(PREDIV + 1) X (MULT + 16)

Address: 4007_B000h base + 608h offset = 4007_B608h



SCG_SPLLCFG field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–16 MULT	System PLL Multiplier

Table continues on the next page...

SCG_SPLLCFG field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>Multiplier for the System PLL. The MULT bits establish the multiplication factor applied to the PLL reference clock frequency.

Table 43-1. PLL VCO Multiply Factor
<table><tr><td>MULT</td><td>Multiply Factor</td><td></td><td>MULT</td><td>Multiply Factor</td><td></td><td>MULT</td><td>Multiply Factor</td><td></td><td>MULT</td><td>Multiply Factor</td></tr><tr><td>00000</td><td>16</td><td></td><td>01000</td><td>24</td><td></td><td>10000</td><td>32</td><td></td><td>11000</td><td>40</td></tr><tr><td>00001</td><td>17</td><td></td><td>01001</td><td>25</td><td></td><td>10001</td><td>33</td><td></td><td>11001</td><td>41</td></tr><tr><td>00010</td><td>18</td><td></td><td>01010</td><td>26</td><td></td><td>10010</td><td>34</td><td></td><td>11010</td><td>42</td></tr><tr><td>00011</td><td>19</td><td></td><td>01011</td><td>27</td><td></td><td>10011</td><td>35</td><td></td><td>11011</td><td>43</td></tr><tr><td>00100</td><td>20</td><td></td><td>01100</td><td>28</td><td></td><td>10100</td><td>36</td><td></td><td>11100</td><td>44</td></tr><tr><td>00101</td><td>21</td><td></td><td>01101</td><td>29</td><td></td><td>10101</td><td>37</td><td></td><td>11101</td><td>45</td></tr><tr><td>00110</td><td>22</td><td></td><td>01110</td><td>30</td><td></td><td>10110</td><td>38</td><td></td><td>11110</td><td>46</td></tr><tr><td>00111</td><td>23</td><td></td><td>01111</td><td>31</td><td></td><td>10111</td><td>39</td><td></td><td>11111</td><td>47</td></tr></table></td></tr>
<tr><td>15–11
Reserved</td><td>This field is reserved.
This read-only field is reserved and always has the value 0.</td></tr>
<tr><td>10–8
PREDIV</td><td>PLL Reference Clock Divider

Selects the amount to divide down the reference clock for the System PLL. The resulting frequency must be in the range of 8 MHz to 32 MHz.

Table 43-2. System PLL Reference Divide Factor
<table><tr><td>PREDIV</td><td>Divide Factor</td></tr><tr><td>000</td><td>1</td></tr><tr><td>001</td><td>2</td></tr><tr><td>010</td><td>3</td></tr><tr><td>011</td><td>4</td></tr><tr><td>100</td><td>5</td></tr><tr><td>101</td><td>6</td></tr><tr><td>110</td><td>7</td></tr><tr><td>111</td><td>8</td></tr></table></td></tr>
<tr><td>7–1
Reserved</td><td>This field is reserved.
This read-only field is reserved and always has the value 0.</td></tr>
<tr><td>0
SOURCE</td><td>Clock Source

Configures the input clock source for the System PLL.

0 System OSC (SOSC)
1 Fast IRC (FIRC)</td></tr>
</table>

43.3 Functional description

43.3.1 SCG Clock Mode Transitions

The following figure shows the valid clock mode transitions supported by SCG.

Fast IRC (FIRC) boot mode is not supported on this device.

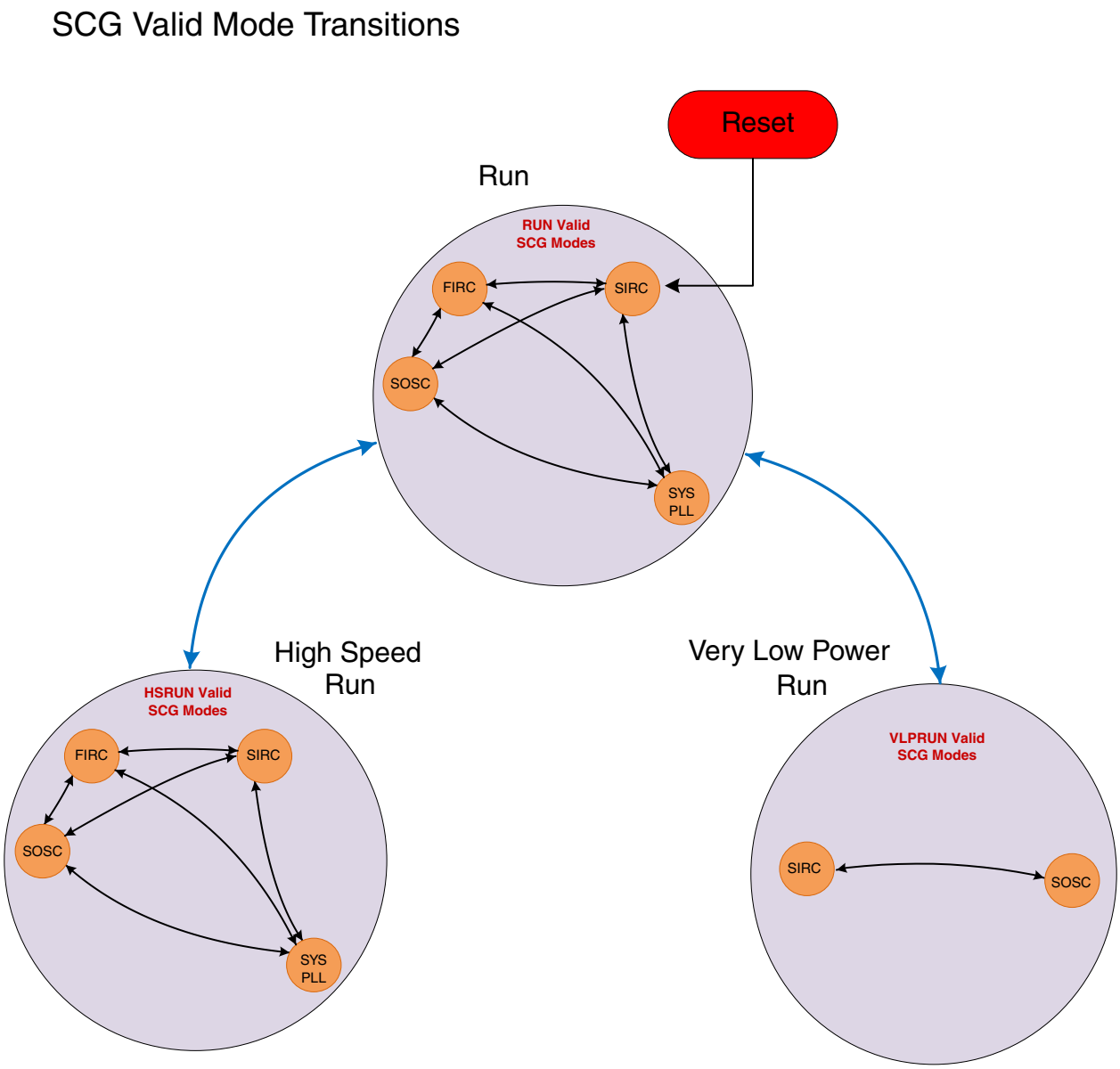


Figure 43-2. SCG Valid Mode Transition Diagram

The SCG will restrict programming into invalid clock modes and writes to SCS bits will be ignored. When a transition between run modes or a transition into wait mode occurs, the SCG completes the switch to the clock mode as defined in the SCG clock control register. When completed, the system switches to the selected run/wait mode.

The modes of operation listed in the following table are the valid modes for this implementation of the SCG.

Table 43-3. SCG modes of operation

Mode	Description
System Oscillator Clock (SOSC)	System Oscillator Clock (SOSC) mode is entered when all the following conditions occur: • RUN MODE: 0001 is written to RCCR[SCS]. VLRUN MODE: 0001 is written to VCCR[SCS]. HSRUN MODE: 0001 is written to HCCR[SCS]. • SOSCEN = 1 • SOSCVLD = 1 In SOSC mode, SCSCLKOUT and system clocks are derived from the external System Oscillator Clock (SOSC).
Slow Internal Reference Clock (SIRC)	Slow Internal Reference Clock (SIRC) mode is entered when all the following conditions occur: • RUN MODE: 0010 is written to RCCR[SCS]. VLRUN MODE: 0010 is written to VCCR[SCS] and 1 is written to SIRCCSR[SIRCLPEN]. HSRUN MODE: 0010 is written to HCCR[SCS]. • SIRCEN = 1 • SIRCVLD = 1 In SIRC mode, SCSCLKOUT and system clocks are derived from the slow internal reference clock. Two frequency ranges are available for SIRC clock as described in the SIRCCFG[RANGE] register definition. Changes to SIRC range settings will be ignored when SIRC clock is enabled.
Fast Internal Reference Clock (FIRC)	Fast Internal Reference Clock (FIRC) mode is entered when all the following conditions occur: • RUN MODE: 0011 is written to RCCR[SCS]. VLRUN MODE: Invalid mode. Programming SCG into FIRC mode will be ingored. HSRUN MODE: 0011 is written to HCCR[SCS]. • FIRCEN = 1 • FIRCVLD = 1 In FIRC mode, SCSCLKOUT and system clocks are derived from the fast internal reference clock. Two frequency range settings are available for FIRC clock as described in the FIRC[RANGE] register definition. Changes to FIRC range settings will be ignored when FIRC clock is enabled.
Sys PLL (SPLL)	Sys PLL (SPLL) mode is entered when all the following conditions occur: • RUN MODE: 0110 is written to RCCR[SCS]. VLRUN MODE: Invalid mode. Programming SCG into SPLL mode will be ingored. HSRUN MODE: 0110 is written to HCCR[SCS]. • SPLLEN = 1 • SPLLVLD = 1 In SPLL mode, the SCSCLKOUT and system clocks are derived from the output of PLL which is controlled by either the System Oscillator (SOSC) clock or the Fast internal reference clock (FIRC). The selected PLL clock frequency locks to a multiplication factor, as specified by its corresponding SCG_SPLLCFG[MULT], times the selected PLL reference frequency. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. This divide value is defined by the SCG_SPLLCFG[PREDIV] bits.

Table continues on the next page...

Table 43-3. SCG modes of operation (continued)

Mode	Description
Stop	Entered whenever the MCU enters a Stop state. The power modes are chip specific. For power mode assignments, see the chapter that describes how modules are configured and SCG behaviour during Stop recovery. Entering Stop mode, all SCG clock signals are static except the following clocks which can continue to run and stayed enabled in the following cases: SIRCCLK is available in Normal Stop and VLPS mode when all the following conditions become true: • SIRCCSR[SIRCEN] = 1 • SIRCCSR[SIRCSTEN] = 1 • SIRCCSR[SIRCLPEN] = 1 in VLPS FIRCCLK is available only in Normal Stop mode when all the following conditions become true: • FIRCCSR[FIRCEN] = 1 • FIRCCSR[FIRCSTEN] = 1 SOSCLK is available in following low power stop modes (Normal Stop, VLPS, LLS) when all the below conditions are true. In VLLS stop mode, SOSCLK is disabled. • SOSCCSR[SOSCEN] = 1 • SOSCCSR[SOSCSTEN] = 1 • SOSCCSR[SOSCLPEN] = 1 (required only for Low Power Stop modes (VLPS and LLS) SPLLCLK is available in Normal Stop mode when all the following conditions are true: • SPLLCSR[SPLLEN] = 1 • SPLLCSR[SPLLSTEN] = 1 • SPLLSTEN control bit has no affect in LLS or VLPS Power mode.

Chapter 44
System Integration Module (SIM)

44.1 Introduction

The system integration module (SIM) provides system control and chip configuration registers.

44.1.1 Features

- Flash and System RAM size configuration
- USB regulator configuration

44.2 Memory map and register definition

The SIM module contains many bit fields for miscellaneous configuration of the device.

NOTE

The SIM registers can be written only in supervisor mode. In user mode, write accesses are blocked and will result in a bus error.

NOTE

The SIM_SOPT1 and SIM_SOPT1CFG registers are located at a different base address than the other SIM registers.

SIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_4000	System Options Register 1 (SIM_SOPT1)	32	R/W	See section	44.2.1/1116
4007_4004	SOPT1 Configuration Register (SIM_SOPT1CFG)	32	R/W	0000_0000h	44.2.2/1117
4007_5024	System Device Identification Register (SIM_SDID)	32	R	See section	44.2.3/1118
4007_504C	Flash Configuration Register 1 (SIM_FCFG1)	32	R/W	See section	44.2.4/1121
4007_5050	Flash Configuration Register 2 (SIM_FCFG2)	32	R	See section	44.2.5/1123
4007_5058	Unique Identification Register Mid-High (SIM_UIDMH)	32	R	See section	44.2.6/1124
4007_505C	Unique Identification Register Mid Low (SIM_UIDML)	32	R	See section	44.2.7/1125
4007_5060	Unique Identification Register Low (SIM_UIDL)	32	R	See section	44.2.8/1125
4007_50EC	Peripheral Clock Status Register (SIM_PCSR)	32	R	0000_0000h	44.2.9/1126

44.2.1 System Options Register 1 (SIM_SOPT1)

NOTE

The SOPT1 register is only reset on POR or LVD.

Address: 4007_4000h base + 0h offset = 4007_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	USBREGEN	USBSSTBY	USBVSTBY	0											0	
W																
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
R	0										Reserved					
W																
Reset	0	0	0	0	0	0	0	0	0	0	*	*	*	*	*	*

* Notes:
- Reserved field: Device specific value.

SIM_SOPT1 field descriptions

Field	Description
31 USBREGEN	USB voltage regulator enable Controls whether the USB voltage regulator is enabled.

Table continues on the next page...

SIM_SOPT1 field descriptions (continued)

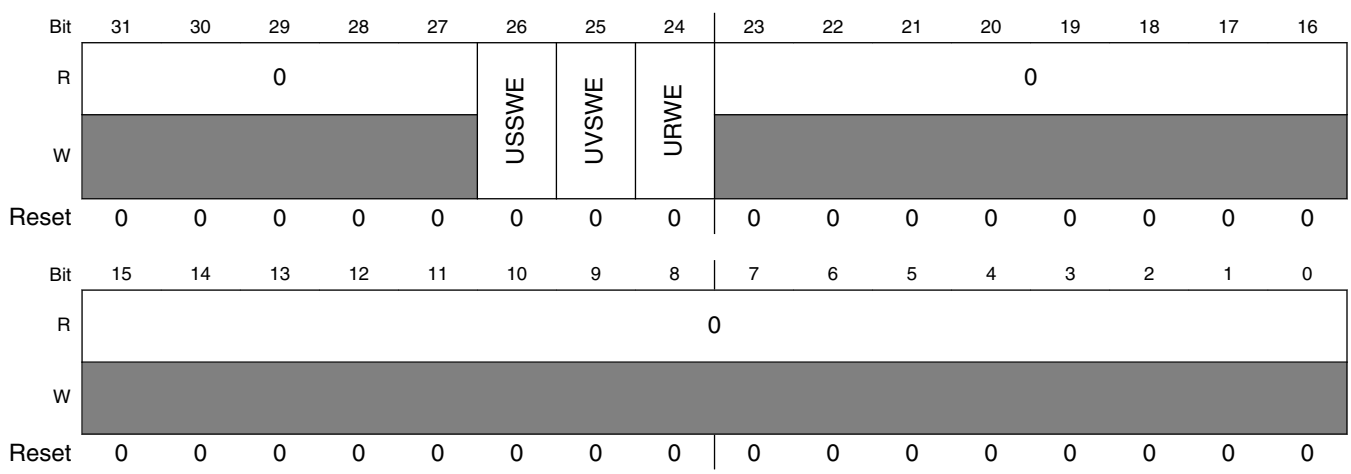
Field	Description
	0 USB voltage regulator is disabled. 1 USB voltage regulator is enabled.
30 USBSSTBY	USB voltage regulator in standby mode during Stop, VLPS, LLS and VLLS modes. Controls whether the USB voltage regulator is placed in standby mode during Stop, VLPS, LLS and VLLS modes. 0 USB voltage regulator not in standby during Stop, VLPS, LLS and VLLS modes. 1 USB voltage regulator in standby during Stop, VLPS, LLS and VLLS modes.
29 USBVSTBY	USB voltage regulator in standby mode during VLPR and VLPW modes Controls whether the USB voltage regulator is placed in standby mode during VLPR and VLPW modes. 0 USB voltage regulator not in standby during VLPR and VLPW modes. 1 USB voltage regulator in standby during VLPR and VLPW modes.
28–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved.

44.2.2 SOPT1 Configuration Register (SIM_SOPT1CFG)

NOTE

The SOPT1CFG register is reset on System Reset not VLLS.

Address: 4007_4000h base + 4h offset = 4007_4004h



SIM_SOPT1CFG field descriptions

Field	Description
31–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 USSWE	USB voltage regulator stop standby write enable Writing one to the USSWE bit allows the SOPT1 USBSSTBY bit to be written. This register bit clears after a write to USBSSTBY. 0 SOPT1 USBSSTB cannot be written. 1 SOPT1 USBSSTB can be written.
25 UVSWE	USB voltage regulator VLP standby write enable Writing one to the UVSWE bit allows the SOPT1 USBVSTBY bit to be written. This register bit clears after a write to USBVSTBY. 0 SOPT1 USBVSTB cannot be written. 1 SOPT1 USBVSTB can be written.
24 URWE	USB voltage regulator enable write enable Writing one to the URWE bit allows the SOPT1 USBREGEN bit to be written. This register bit clears after a write to USBREGEN. 0 SOPT1 USBREGEN cannot be written. 1 SOPT1 USBREGEN can be written.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

44.2.3 System Device Identification Register (SIM_SDID)

Address: 4007_4000h base + 1024h offset = 4007_5024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FAMID				SUBFAMID				SERIESID				SRAMSIZE				REVID				DIEID							KEYATT			PINID			
W																																
Reset	*	*	*	*	*	*	*	*	0	0	0	1	*	*	*	*	*	*	*	*	1	1	1	0	0	*	*	*	*	*	*	*

* Notes:
- FAMID field: Device specific value.
- SUBFAMID field: Device specific value.
- SRAMSIZE field: Device specific value.
- REVID field: Device specific value.
- KEYATT field: Device specific value.
- PINID field: Device specific value.

SIM_SDID field descriptions

Field	Description
31–28 FAMID	Kinetis family ID

Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
	Specifies the Kinetis family of the device. 0010 KL2x Family (USB)
27–24 SUBFAMID	Kinetis Sub-Family ID Specifies the Kinetis sub-family of the device. 0010 KLx2 Subfamily 0011 KLx3 Subfamily 0100 KLx4 Subfamily 0101 KLx5 Subfamily 0110 KLx6 Subfamily 0111 KLx7 Subfamily 1000 KLx8 Subfamily 1001 KLx9 Subfamily
23–20 SERIESID	Kinetis Series ID Specifies the Kinetis family of the device. 0001 KL family
19–16 SRAMSIZE	System SRAM Size Specifies the size of the System SRAM 1000 96 KB 1001 128 KB
15–12 REVID	Device Revision Number Specifies the silicon implementation number for the device. 0001 Revision 1.1
11–7 DIEID	Device Die Number Specifies the silicon implementation number for the device. This field always reads as 11100.
6–4 KEYATT	Core configuration of the device. Specifies the core configuration of the device. 000 Cortex CM0+ Core 001 Reserved 010 Reserved 011 Reserved 100 Reserved 101 Reserved 110 Reserved 111 Reserved
PINID	Pin count identification Specifies the pin count of the device.

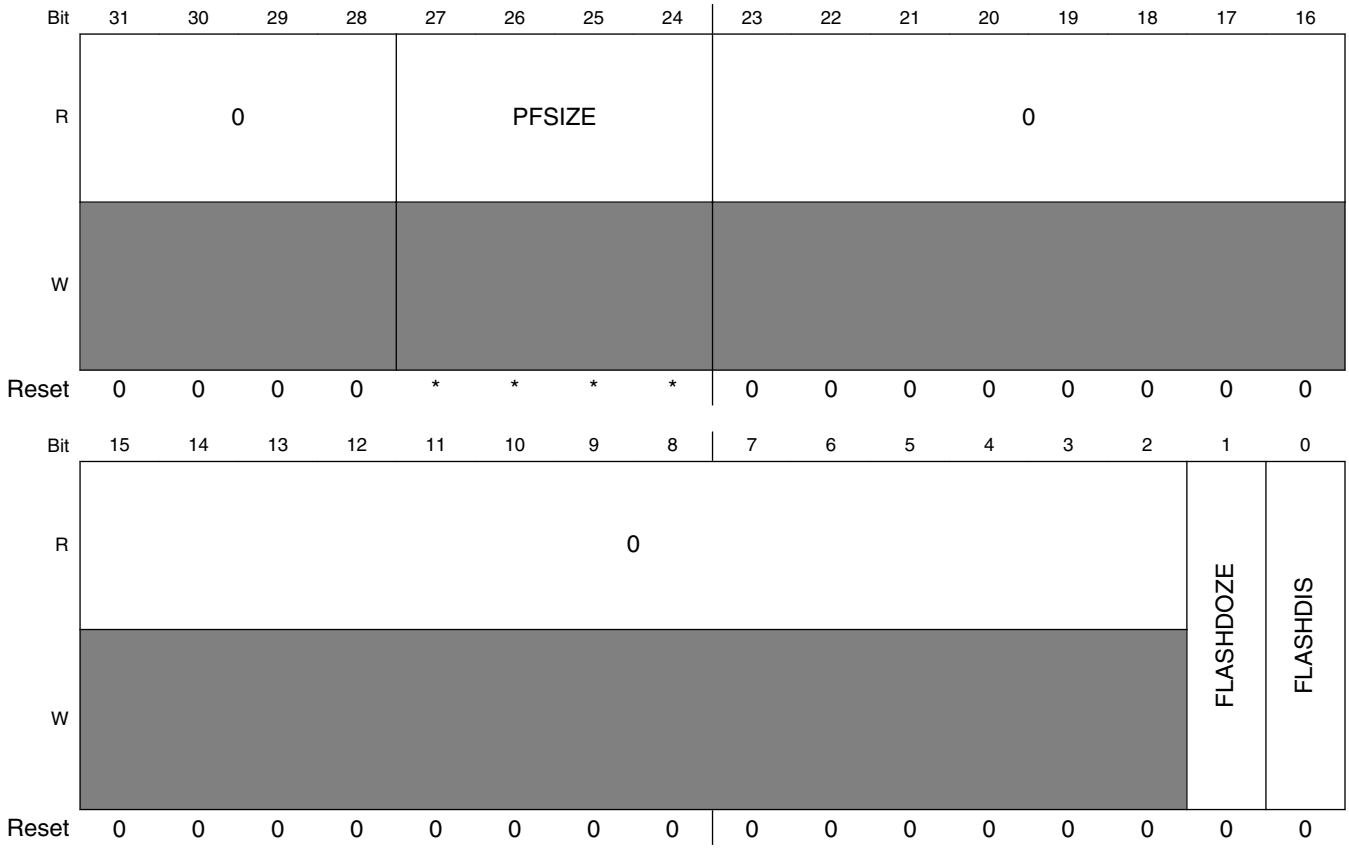
Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
	0000 Reserved 0001 Reserved 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Reserved 1000 100-pin 1001 121-pin 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

44.2.4 Flash Configuration Register 1 (SIM_FCFG1)

Address: 4007_4000h base + 104Ch offset = 4007_504Ch



* Notes:
- PFSIZE field: Device specific value.

SIM_FCFG1 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 PFSIZE	Program Flash Size Specifies the amount of program flash memory available on the device . Undefined values are reserved. **NOTE:** Your device may not support all of these possible values for Program Flash Size. To determine which program flash sizes are supported by your device, check the data sheet. 0101 64 KB of program flash memory, 2 KB protection region 0111 128 KB of program flash memory, 4 KB protection region 1001 256 KB of program flash memory, 8 KB protection region 1011 512 KB of program flash memory, 16 KB protection region

Table continues on the next page...

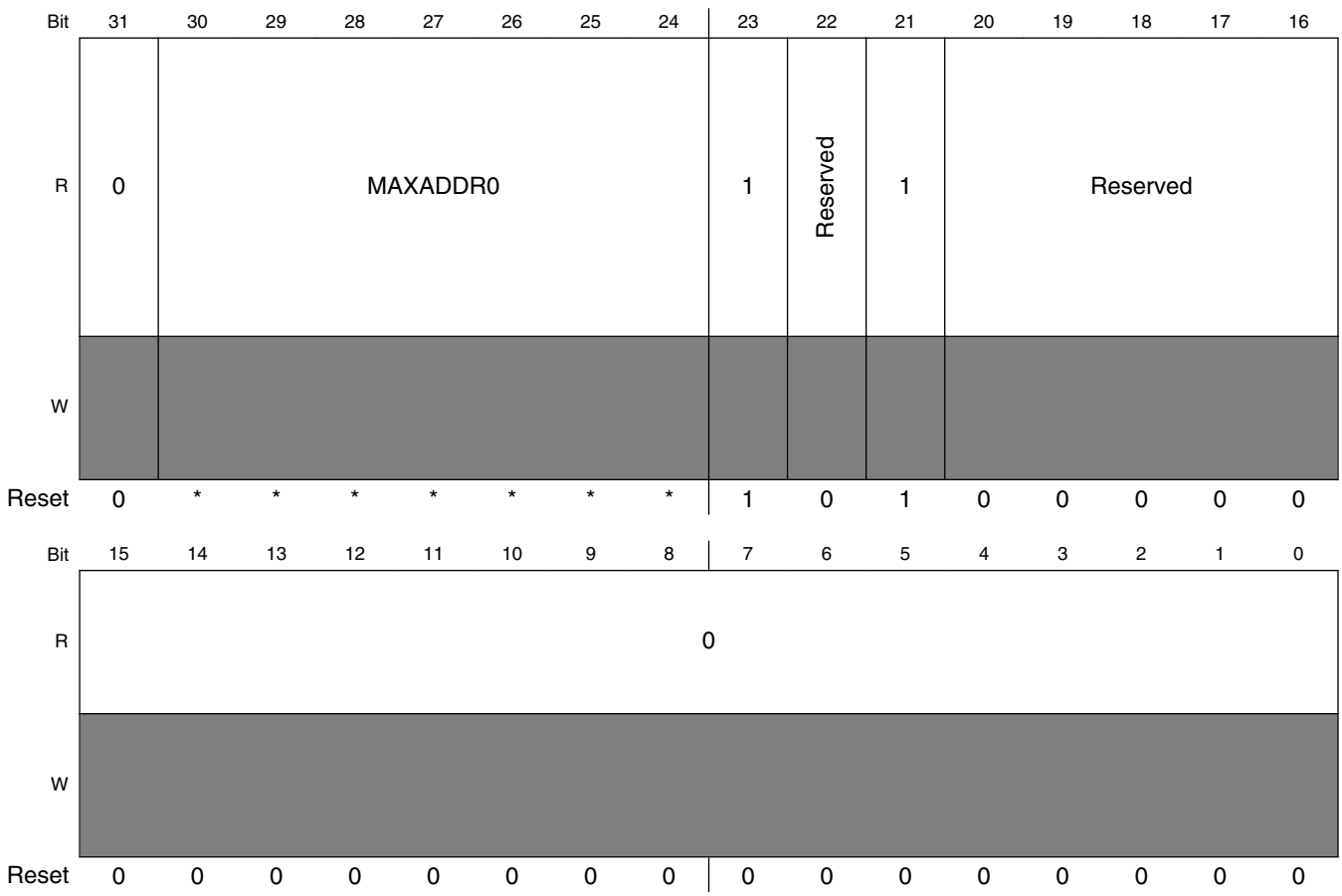
SIM_FCFG1 field descriptions (continued)

Field	Description
23–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FLASHDOZE	Flash Doze When set, flash memory is disabled for the duration of Doze mode. This field must be clear during VLP modes. The flash will be automatically enabled again at the end of Doze mode so interrupt vectors do not need to be relocated out of flash memory. The wake-up time from Doze mode is extended when this field is set. An attempt by the DMA or other bus master to access the flash memory when the flash is disabled will result in a bus error. 0 Flash remains enabled during Doze mode. 1 Flash is disabled for the duration of Doze mode.
0 FLASHDIS	Flash Disable Flash accesses are disabled (and generate a bus error) and the flash memory is placed in a low-power state. This field should not be changed during VLP modes. Relocate the interrupt vectors out of Flash memory before disabling the Flash. 0 Flash is enabled. 1 Flash is disabled.

44.2.5 Flash Configuration Register 2 (SIM_FCFG2)

This is read only register, any write to this register will cause transfer error.

Address: 4007_4000h base + 1050h offset = 4007_5050h



* Notes:
- MAXADDR0 field: Device specific value indicating amount of implemented flash.

SIM_FCFG2 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30–24 MAXADDR0	Max Address lock • MAXADDR0 field *concatenated with 13 trailing zeroes* indicates the first invalid address of **program flash block 0**. • MAXADDR0 field *concatenated with 13 trailing zeroes*, **then doubled**, indicates the first invalid address of **program flash block 1**.

Table continues on the next page...

SIM_FCFG2 field descriptions (continued)

Field	Description
	For example, if MAXADDR0 = 0x10, then the first invalid address of program flash block 0 is 0x0002_0000, and the first invalid address of program flash block 1 is 0x0004_0000.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
22 Reserved	This field is reserved.
21 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
20–16 Reserved	This field is reserved.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

44.2.6 Unique Identification Register Mid-High (SIM_UIDMH)

Address: 4007_4000h base + 1058h offset = 4007_5058h

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

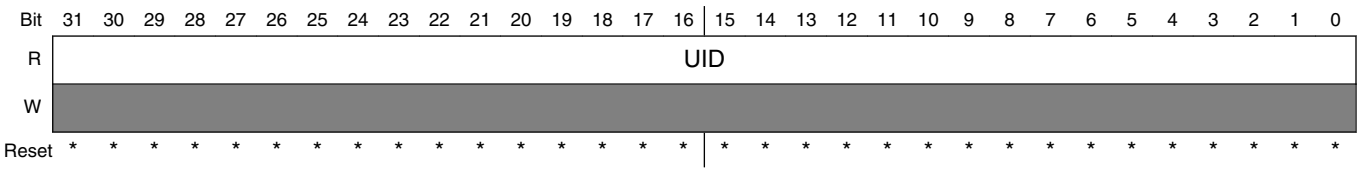
* Notes:
- UID field: Device specific value.

SIM_UIDMH field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
UID	Unique Identification Unique identification for the device.

44.2.7 Unique Identification Register Mid Low (SIM_UIDML)

Address: 4007_4000h base + 105Ch offset = 4007_505Ch



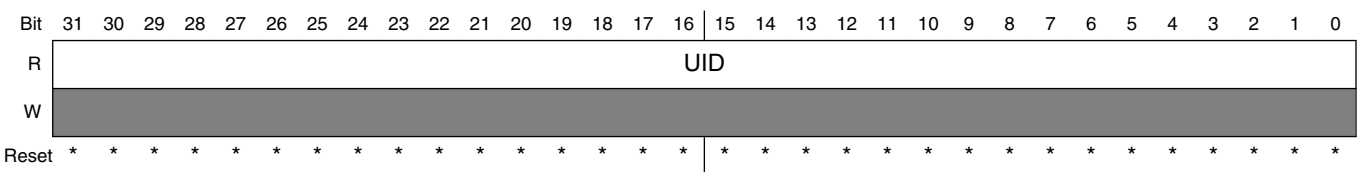
* Notes:
- UID field: Device specific value.

SIM_UIDML field descriptions

Field	Description
UID	Unique Identification Unique identification for the device.

44.2.8 Unique Identification Register Low (SIM_UIDL)

Address: 4007_4000h base + 1060h offset = 4007_5060h



* Notes:
- UID field: Device specific value.

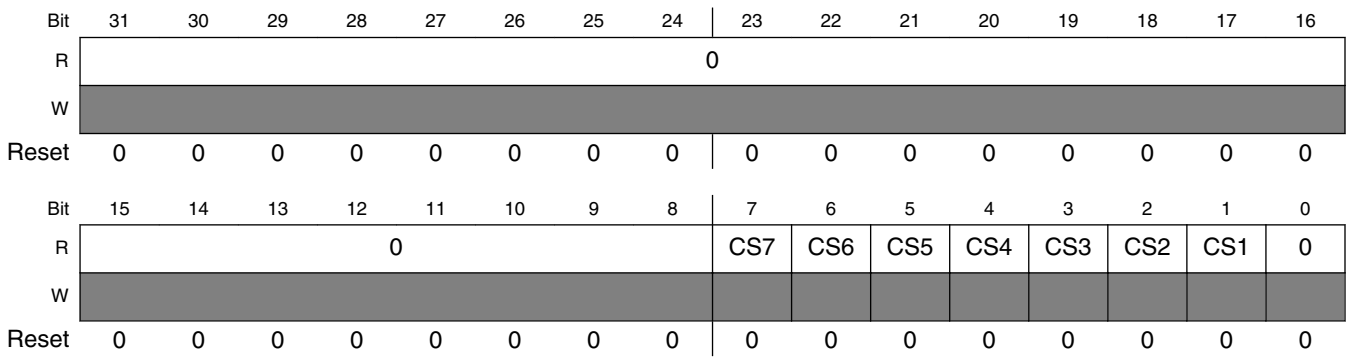
SIM_UIDL field descriptions

Field	Description
UID	Unique Identification Unique identification for the device.

44.2.9 Peripheral Clock Status Register (SIM_PCSR)

The System Clock Generator will provide a 'ready' status for all peripheral clock sources. The 'ready' status bits of the respective peripheral clock source can be polled by software prior to choosing a clock as the source clock for a peripheral.

Address: 4007_4000h base + 10ECh offset = 4007_50ECh



SIM_PCSR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CS7	Clock Source 7 Clock Source 7 (Peripheral PLL) Status 0 Clock not ready. 1 Clock ready.
6 CS6	Clock Source 6 Clock Source 6 (System PLL) Status 0 Clock not ready. 1 Clock ready.
5 CS5	Clock Source 5 Clock Source 5 (System FLL) Status 0 Clock not ready. 1 Clock ready.
4 CS4	Clock Source 4 Clock Source 4 (RTC Oscillator) Status 0 Clock not ready. 1 Clock ready.
3 CS3	Clock Source 3

Table continues on the next page...

SIM_PCSR field descriptions (continued)

Field	Description
	Clock Source 3 (Fast IRC Clock) Status 0 Clock not ready. 1 Clock ready.
2 CS2	Clock Source 2 Clock Source 2 (Slow IRC Clock) Status 0 Clock not ready. 1 Clock ready.
1 CS1	Clock Source 1 Clock Source 1 (System Oscillator) Status 0 Clock not ready. 1 Clock ready.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

44.3 Functional description

See Introduction section.

Chapter 45
System Mode Controller (SMC)

45.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The System Mode Controller (SMC) is responsible for sequencing the system into and out of all low-power Stop and Run modes.

Specifically, it monitors events to trigger transitions between power modes while controlling the power, clocks, and memories of the system to achieve the power consumption and functionality of that mode.

This chapter describes all the available low-power modes, the sequence followed to enter/exit each mode, and the functionality available while in each of the modes.

The SMC is able to function during even the deepest low power modes.

See AN4503: Power Management for Kinetis MCUs for further details on using the SMC.

45.2 Modes of operation

The ARM CPU has three primary modes of operation:

- Run
- Sleep
- Deep Sleep

The WFI or WFE instruction is used to invoke Sleep and Deep Sleep modes. Run, Wait, and Stop are the common terms used for the primary operating modes of Kinetis microcontrollers.

The following table shows the translation between the ARM CPU modes and the Kinetis MCU power modes.

ARM CPU mode	MCU mode
Sleep	Wait
Deep Sleep	Stop

Accordingly, the ARM CPU documentation refers to sleep and deep sleep, while the Kinetis MCU documentation normally uses wait and stop.

In addition, Kinetis MCUs also augment Stop, Wait, and Run modes in a number of ways. The power management controller (PMC) contains a run and a stop mode regulator. Run regulation is used in normal run, wait and stop modes. Stop mode regulation is used during all very low power and low leakage modes. During stop mode regulation, the bus frequencies are limited in the very low power modes.

The SMC provides the user with multiple power options. The Very Low Power Run (VLPR) mode can drastically reduce run time power when maximum bus frequency is not required to handle the application needs. From Normal Run mode, the Run Mode (RUNM) field can be modified to change the MCU into VLPR mode when limited frequency is sufficient for the application. From VLPR mode, a corresponding wait (VLPW) and stop (VLPS) mode can be entered.

Depending on the needs of the user application, a variety of stop modes are available that allow the state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. Several registers are used to configure the various modes of operation for the device.

The following table describes the power modes available for the device.

Table 45-1. Power modes

Mode	Description
RUN	The MCU can be run at full speed and the internal supply is fully regulated, that is, in run regulation. This mode is also referred to as Normal Run mode.
HSRUN	The MCU can be run at a faster frequency compared with RUN mode and the internal supply is fully regulated. See the Power Management chapter for details about the maximum allowable frequencies.
WAIT	The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate. Run regulation is maintained.
STOP	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
VLPR	The core, system, bus, and flash clock maximum frequencies are restricted in this mode. See the Power Management chapter for details about the maximum allowable frequencies.

Table continues on the next page...

Table 45-1. Power modes (continued)

Mode	Description
VLPW	The core clock is gated off. The system, bus, and flash clocks continue to operate, although their maximum frequency is restricted. See the Power Management chapter for details on the maximum allowable frequencies.
VLPS	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
LLS3	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by reducing the voltage to internal logic. All system RAM contents, internal logic and I/O states are retained.
LLS2	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by reducing voltage to internal logicand powering down the system RAM2 partition. The system RAM1 partition, internal logic and I/O states are retained.[1]
VLLS3	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic. All system RAM contents are retained and I/O states are held. Internal logic states are not retained.
VLLS2	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and the system RAM2 partition. The system RAM1 partition contents are retained in this mode. Internal logic states are not retained. [1]
VLLS1	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained.
VLLS0	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained. The 1kHz LPO clock is disabled and the power on reset (POR) circuit can be optionally enabled using STOPCTRL[PORPO].

1. See the devices' chip configuration details for the size and location of the system RAM partitions.

45.3 Memory map and register descriptions

Information about the registers related to the system mode controller can be found here.

Different SMC registers reset on different reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

NOTE

The SMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

NOTE

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

SMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_E000	SMC Version ID Register (SMC_VERID)	32	R	0100_0000h	45.3.1/1132
4007_E004	SMC Parameter Register (SMC_PARAM)	32	R	See section	45.3.2/1133
4007_E008	Power Mode Protection register (SMC_PMPROT)	32	R/W		45.3.3/1134
4007_E00C	Power Mode Control register (SMC_PMCTRL)	32	R/W		45.3.4/1136
4007_E010	Stop Control Register (SMC_STOPCTRL)	32	R/W	0000_0003h	45.3.5/1138
4007_E014	Power Mode Status register (SMC_PMSTAT)	32	R		45.3.6/1139

45.3.1 SMC Version ID Register (SMC_VERID)

Address: 4007_E000h base + 0h offset = 4007_E000h

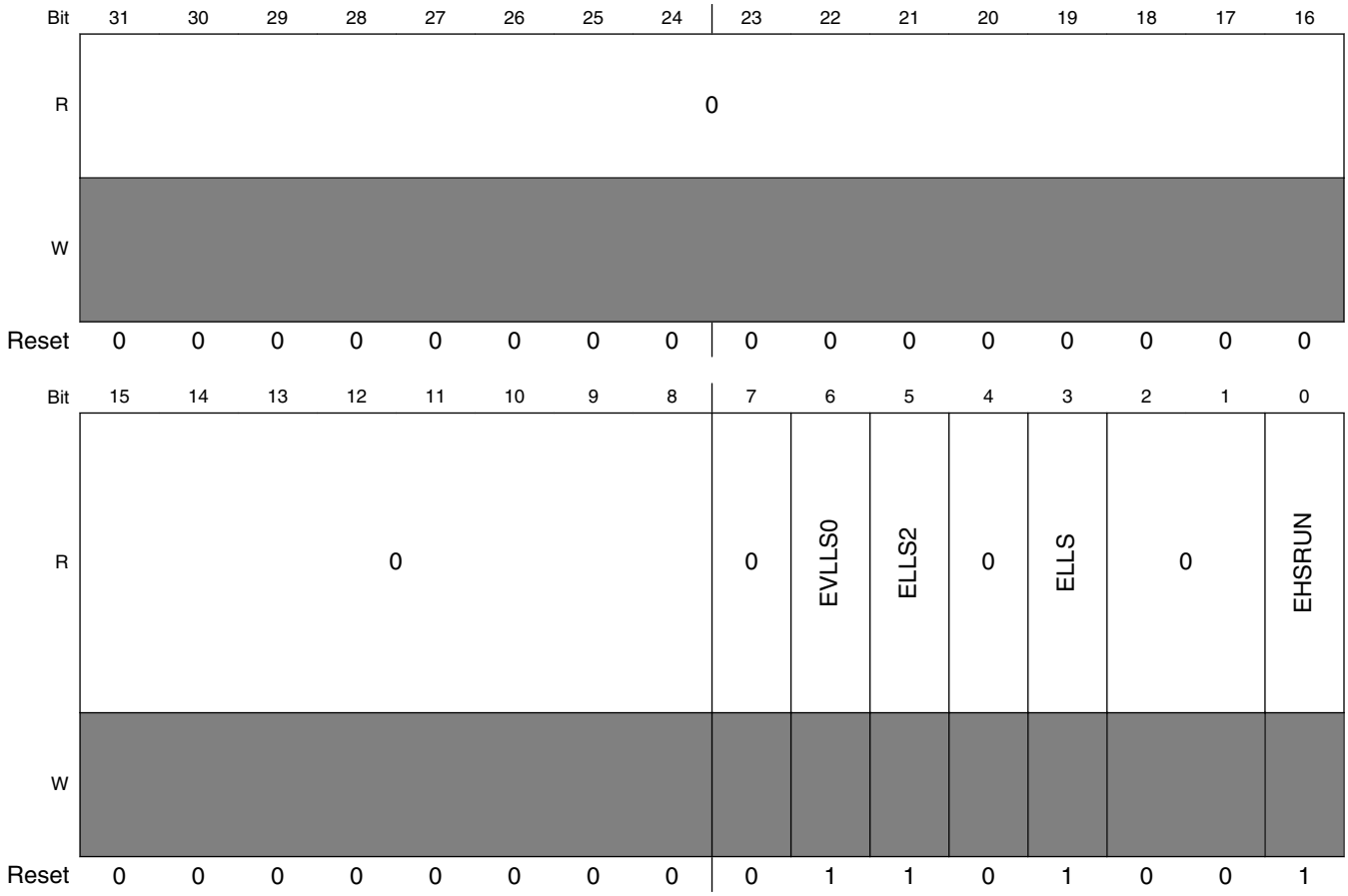
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								FEATURE															
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SMC_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Specification Number This read only field returns the feature set number. 0x0000 Standard features implemented

45.3.2 SMC Parameter Register (SMC_PARAM)

Address: 4007_E000h base + 4h offset = 4007_E004h



SMC_PARAM field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 EVLLS0	Existence of VLLS0 feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
5 ELLS2	Existence of LLS2 feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

Table continues on the next page...

SMC_PARAM field descriptions (continued)

Field	Description
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ELLS	Existence of LLS feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.
2–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 EHSRUN	Existence of HSRUN feature This static bit states whether or not the feature is available on the device. 0 The feature is not available. 1 The feature is available.

45.3.3 Power Mode Protection register (SMC_PMPROT)

This register provides protection for entry into any low-power run or stop mode. The enabling of the low-power run or stop mode occurs by configuring the Power Mode Control register (PMCTRL).

The PMPROT register can be written only once after any system reset.

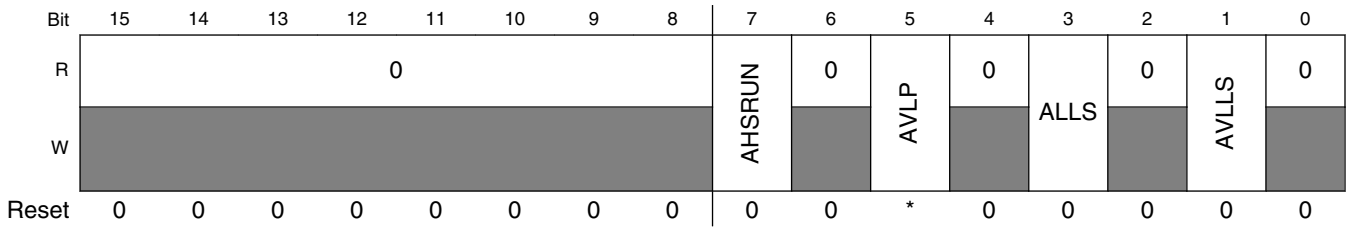
If the MCU is configured for a disallowed or reserved power mode, the MCU remains in its current power mode. For example, if the MCU is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and PMCTRL[RUNM] remains 00b, indicating the MCU is still in Normal Run mode.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 8h offset = 4007_E008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



* Notes:
- AVLP field: When booting in run mode, the reset value is 0. When booting in VLPR mode, the reset value is 1.

SMC_PMPROT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 AHSRUN	Allow High Speed Run mode Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter High Speed Run mode (HSRUN). 0 HSRUN is not allowed 1 HSRUN is allowed
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 AVLP	Allow Very-Low-Power Modes Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter any very-low-power mode (VLPR, VLPW, and VLPS). 0 VLPR, VLPW, and VLPS are not allowed. 1 VLPR, VLPW, and VLPS are allowed.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 ALLS	Allow Low-Leakage Stop Mode Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter any low-leakage stop mode (LLS). 0 Any LLSx mode is not allowed 1 Any LLSx mode is allowed
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 AVLLS	Allow Very-Low-Leakage Stop Mode Provided the appropriate control bits are set up in PMCTRL, this write once bit allows the MCU to enter any very-low-leakage stop mode (VLLSx). 0 Any VLLSx mode is not allowed 1 Any VLLSx mode is allowed
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

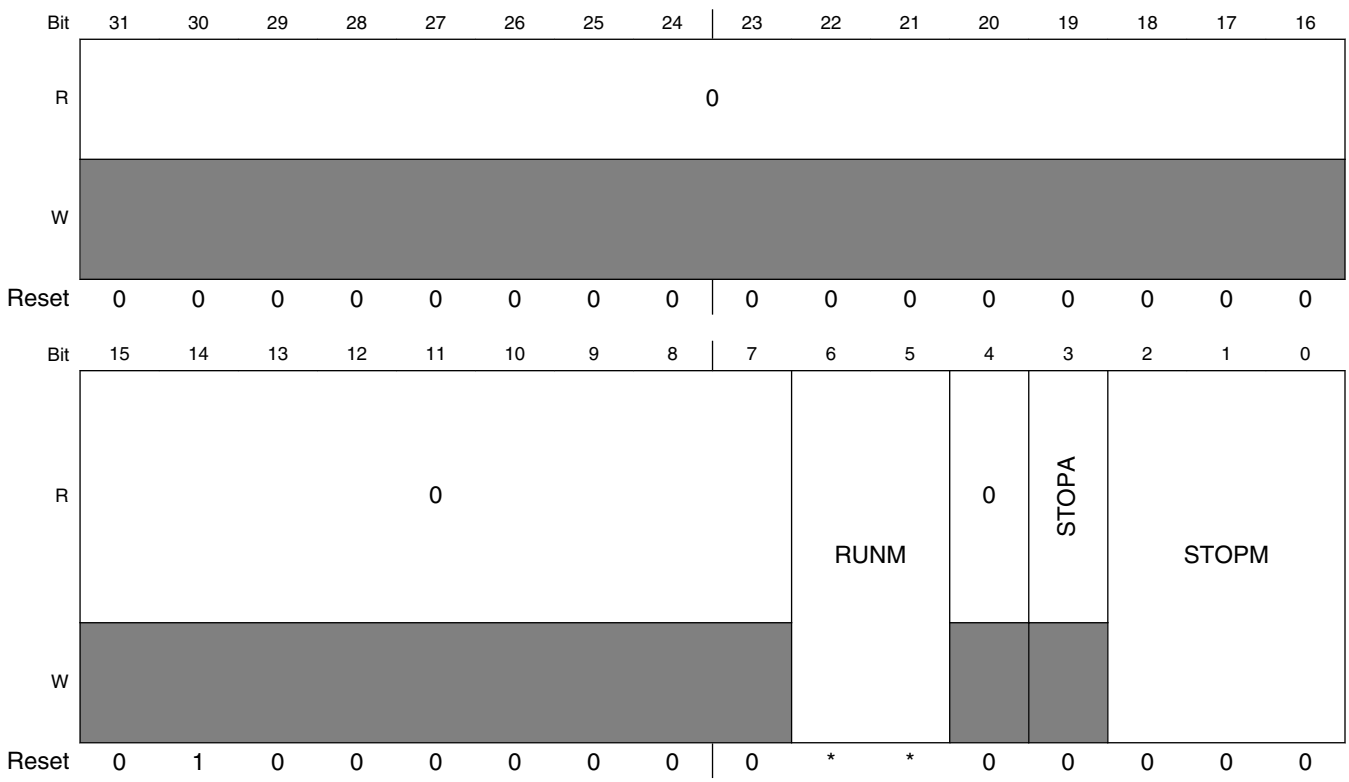
45.3.4 Power Mode Control register (SMC_PMCTRL)

The PMCTRL register controls entry into low-power Run and Stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + Ch offset = 4007_E00Ch



* Notes:
- RUNM field: When booting in run mode, the reset value is 00. When booting in VLPR mode, the reset value is 10.

SMC_PMCTRL field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–5 RUNM	Run Mode Control

Table continues on the next page...

SMC_PMCTRL field descriptions (continued)

Field	Description
	When written, causes entry into the selected run mode. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. **NOTE:** RUNM may be set to VLPR only when PMSTAT=RUN. After being written to VLPR, RUNM should not be written back to RUN until PMSTAT=VLPR. **NOTE:** RUNM may be set to HSRUN only when PMSTAT=RUN. After being programmed to HSRUN, RUNM should not be programmed back to RUN until PMSTAT=HSRUN. Also, stop mode entry should not be attempted while RUNM=HSRUN or PMSTAT=HSRUN. When in HSRUN mode, any reset clears RUNM and causes the system to exit to normal RUN mode after the MCU exits its reset flow. 00 Normal Run mode (RUN) 01 Reserved 10 Very-Low-Power Run mode (VLPR) 11 High Speed Run mode (HSRUN)
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 STOPA	Stop Aborted When set, this read-only status bit indicates an interrupt occured during the previous stop mode entry sequence, preventing the system from entering that mode. This field is cleared by reset or by hardware at the beginning of any stop mode entry sequence and is set if the sequence was aborted. 0 The previous stop mode entry was successful. 1 The previous stop mode entry was aborted.
STOPM	Stop Mode Control When written, controls entry into the selected stop mode when Sleep-Now or Sleep-On-Exit mode is entered with SLEEPDEEP=1 . Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register. **NOTE:** When set to VLLSxor LLSx, the LLSM in the STOPCTRL register is used to further select the particular VLLSor LLS submode which will be entered. **NOTE:** When set to STOP, the PSTOPO bits in the STOPCTRL register can be used to select a Partial Stop mode if desired. 000 Normal Stop (STOP) 001 Reserved 010 Very-Low-Power Stop (VLPS) 011 Low-Leakage Stop (LLSx) 100 Very-Low-Leakage Stop (VLLSx) 101 Reserved 110 Reseved 111 Reserved

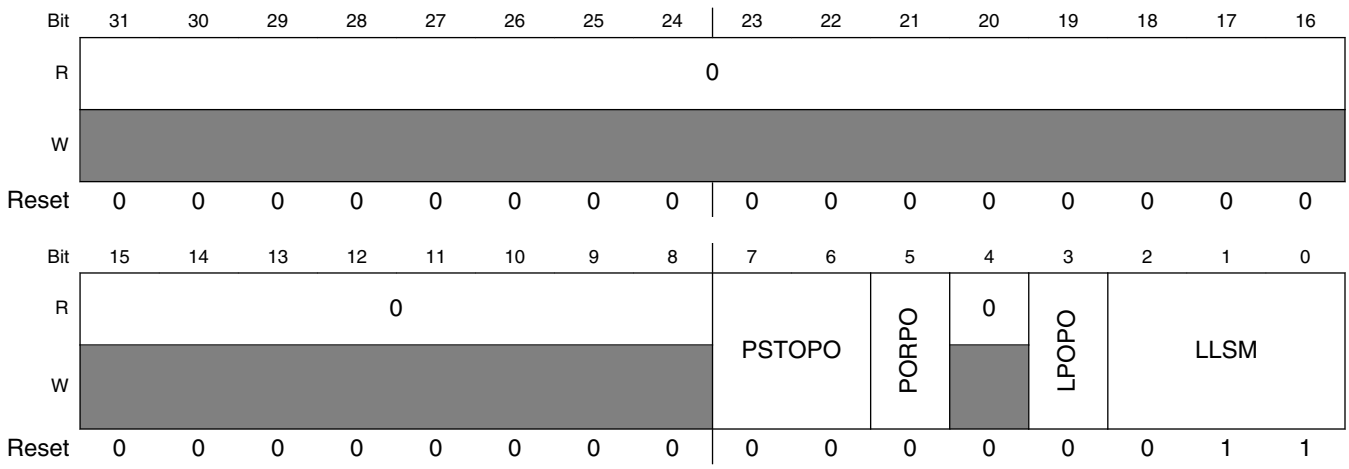
45.3.5 Stop Control Register (SMC_STOPCTRL)

The STOPCTRL register provides various control bits allowing the user to fine tune power consumption during the stop mode selected by the STOPM field.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 10h offset = 4007_E010h



SMC_STOPCTRL field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 PSTOPO	Partial Stop Option These bits control whether a Partial Stop mode is entered when STOPM=STOP. When entering a Partial Stop mode from RUN mode, the PMC, SCG and flash remain fully powered, allowing the device to wakeup almost instantaneously at the expense of higher power consumption. In PSTOP2, only system clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated. 00 STOP - Normal Stop mode 01 PSTOP1 - Partial Stop with both system and bus clocks disabled 10 PSTOP2 - Partial Stop with system clock disabled and bus clock enabled 11 Reserved
5 PORPO	POR Power Option This bit controls whether the POR detect circuit is enabled in VLLS0 mode.

Table continues on the next page...

SMC_STOPCTRL field descriptions (continued)

Field	Description
	0 POR detect circuit is enabled in VLLS0 1 POR detect circuit is disabled in VLLS0
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LPOPO	LPO Power Option Controls whether the 1 kHz LPO clock is enabled in LLS/VLLSx modes. **NOTE:** During VLLS0 mode, the LPO clock is disabled by hardware and this bit has no effect. 0 LPO clock is enabled in LLS/VLLSx 1 LPO clock is disabled in LLS/VLLSx
LLSM	LLS or VLLS Mode Control This field controls whichLLS or VLLS sub-mode to enter if STOPM=LLSx orVLLSx. 000 VLLS0 if PMCTRL[STOPM]=VLLSx, reserved if PMCTRL[STOPM]=LLSx 001 VLLS1 if PMCTRL[STOPM]=VLLSx, reserved if PMCTRL[STOPM]=LLSx 010 VLLS2 if PMCTRL[STOPM]=VLLSx, LLS2 if PMCTRL[STOPM]=LLSx 011 VLLS3 if PMCTRL[STOPM]=VLLSx, LLS3 if PMCTRL[STOPM]=LLSx 100 Reserved 101 Reserved 110 Reserved 111 Reserved

45.3.6 Power Mode Status register (SMC_PMSTAT)

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

When in HSRUN mode, any reset causes the system to exit to normal RUN mode after the MCU exits its reset flow. The PMSTAT field is then automatically updated to show RUN as the current power mode.

Address: 4007_E000h base + 14h offset = 4007_E014h

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

* Notes:
- PMSTAT field: When booting in run mode, the reset value is 0x01. When booting in VLPR mode, the reset value is 0x04.

SMC_PMSTAT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PMSTAT	Power Mode Status **NOTE:** When debug is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** When a PSTOP mode is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** Since the RUNM bits in the PMCTRL register are reset to VLPR on any Chip Reset not VLLS, the PMSTAT will update to VLPR shortly after the reset sequence is complete. 0000_0001 Current power mode is RUN. 0000_0010 Current power mode is STOP. 0000_0100 Current power mode is VLPR. 0000_1000 Current power mode is VLPW. 0001_0000 Current power mode is VLPS. 0010_0000 Current power mode is LLS. 0100_0000 Current power mode is VLLS. 1000_0000 Current power mode is HSRUN

45.4 Functional description

45.4.1 Power mode transitions

The following figure shows the power mode state transitions available on the chip. Any reset always brings the MCU back to the normal RUN state.

The following figure shows the power mode state transitions available on the chip. Any reset will initially bring the MCU back to the normal RUN state. However, in order to minimize peak power consumption, the RUNM bits in the PMCTRL register can be reset to VLPR via Flash IFR settings, causing the SMC to begin transitioning the MCU into VLPR mode during the reset recovery sequence.

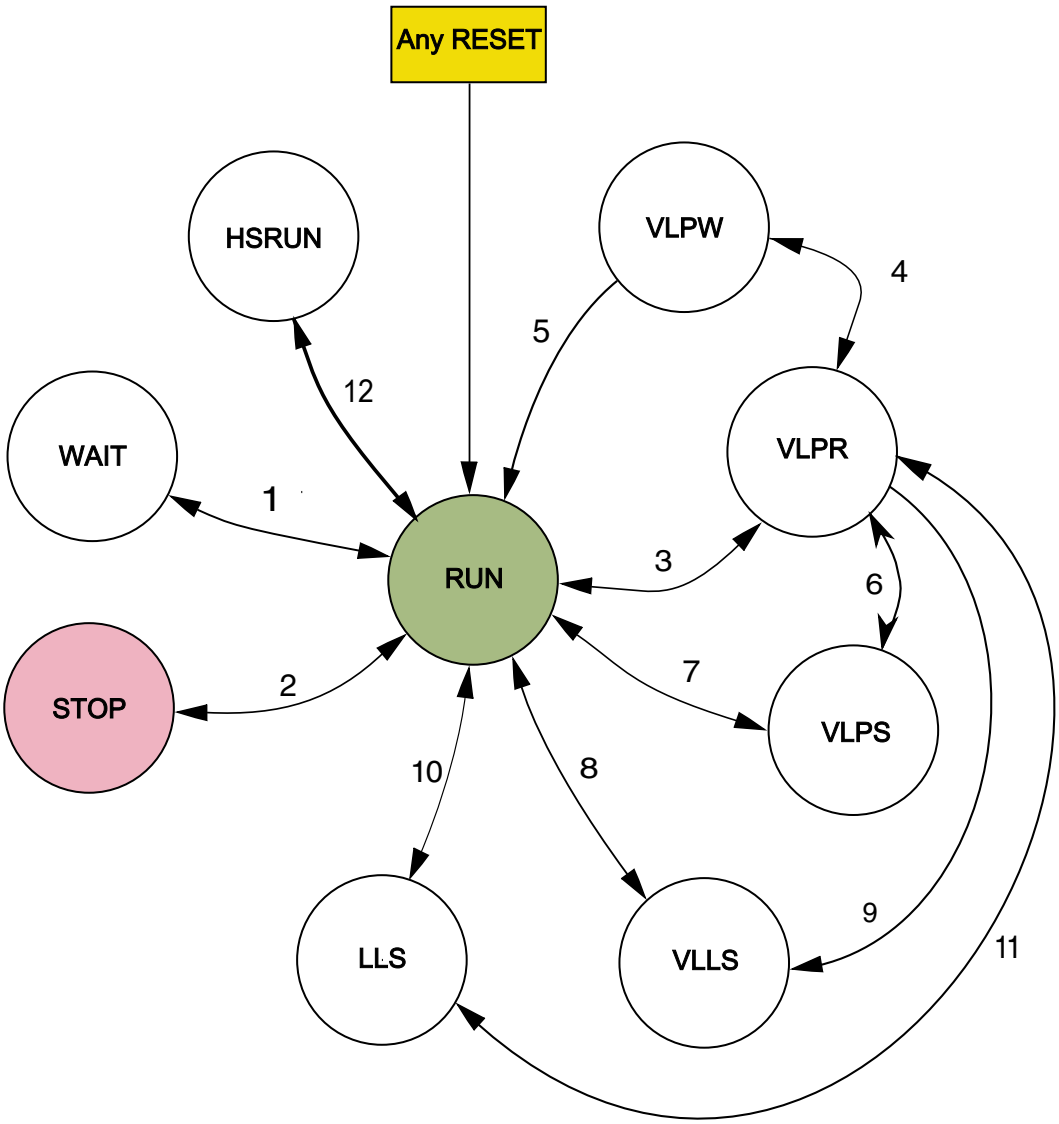


Figure 45-1. Power mode state diagram

The following table defines triggers for the various state transitions shown in the previous figure.

Table 45-2. Power mode transition triggers

Transition #	From	To	Trigger conditions
1	RUN	WAIT	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, controlled in System Control Register in ARM core. See note.[1]
	WAIT	RUN	Interrupt or Reset
2	RUN	STOP	PMCTRL[RUNM]=00, PMCTRL[STOPM]=000[2]

Table continues on the next page...

Table 45-2. Power mode transition triggers (continued)

Transition #	From	To	Trigger conditions
			Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note.[1]
	STOP	RUN	Interrupt or Reset
3	RUN	VLPR	The core, system, bus and flash clock frequencies and SCG clocking mode are restricted in this mode. See the Power Management chapter for the maximum allowable frequencies and SCG modes supported. Set PMPROT[AVLP]=1, PMCTRL[RUNM]=10. **NOTE:** In order to limit peak current, PMPROT[AVLP] and PMCTRL[RUNM] bits can be set on any Reset via Flash IFR settings, causing the SMC to transition the MCU from RUN->VLPR during the reset recovery sequence.
	VLPR	RUN	Set PMCTRL[RUNM]=00 or Reset.
4	VLPR	VLPW	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, which is controlled in System Control Register in ARM core. See note.[1]
	VLPW	VLPR	Interrupt
5	VLPW	RUN	Reset
6	VLPR	VLPS	PMCTRL[STOPM]=000[3] or 010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note.[1]
	VLPS	VLPR	Interrupt **NOTE:** If VLPS was entered directly from RUN (transition #7), hardware forces exit back to RUN and does not allow a transition to VLPR.
7	RUN	VLPS	PMPROT[AVLP]=1, PMCTRL[STOPM]=010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note.[1]
	VLPS	RUN	Interrupt and VLPS mode was entered directly from RUN or Reset
8	RUN	VLLSx	PMPROT[AVLLS]=1, PMCTRL[STOPM]=100, STOPCTRL[LLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.
	VLLSx	RUN	Wakeup from enabled LLWU input source or RESET pin

Table continues on the next page...

Table 45-2. Power mode transition triggers (continued)

Transition #	From	To	Trigger conditions
9	VLPR	VLLSx	PMPROT[AVLLS]=1, PMCTRL[STOPM]=100, STOPCTRL[LLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.
10	RUN	LLSx	PMPROT[ALLS]=1, PMCTRL[STOPM]=011, STOPCTRL[LLSM]=x (LLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.
	LLSx	RUN	Wakeup from enabled LLWU input source and LLSx mode was entered directly from RUN or RESET pin.
11	VLPR	LLSx	PMPROT[ALLS]=1, PMCTRL[STOPM]=011, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.
	LLSx	VLPR	Wakeup from enabled LLWU input source and LLSx mode was entered directly from VLPR **NOTE:** If LLSx was entered directly from RUN, hardware will not allow this transition and will force exit back to RUN
12	RUN	HSRUN	Set PMPROT[AHSRUN]=1, PMCTRL[RUNM]=11.
	HSRUN	RUN	Set PMCTRL[RUNM]=00 or Reset

1. If debug is enabled, the core clock remains to support debug.
2. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of STOP
3. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=00, then VLPS mode is entered instead of STOP. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of VLPS

45.4.2 Power mode entry/exit sequencing

When entering or exiting low-power modes, the system must conform to an orderly sequence to manage transitions safely.

The SMC manages the system's entry into and exit from all power modes. This diagram illustrates the connections of the SMC with other system components in the chip that are necessary to sequence the system through all power modes.

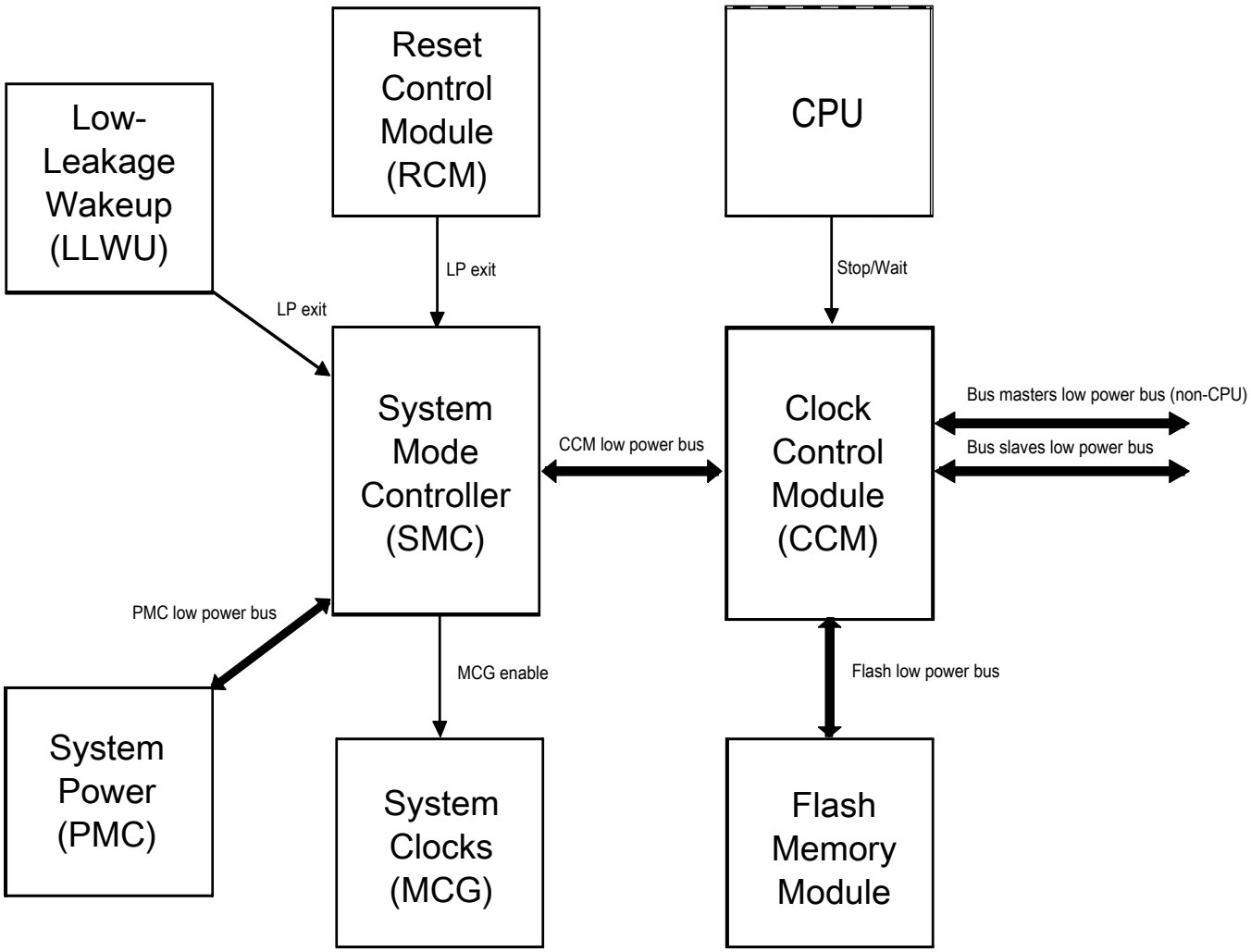


Figure 45-2. Low-power system components and connections

45.4.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS, LLS, VLLSx) is initiated by a CPU executing the WFI instruction. After the instruction is executed, the following sequence occurs:

1. The CPU clock is gated off immediately.
2. Requests are made to all non-CPU bus masters to enter Stop mode.
3. After all masters have acknowledged they are ready to enter Stop mode, requests are made to all bus slaves to enter Stop mode.
4. After all slaves have acknowledged they are ready to enter Stop mode, all system and bus clocks are gated off.
5. Clock generators are disabled in the SCG unless configured to be enabled in Stop mode. See the SCG module information for the programming options.

6. The on-chip regulator in the PMC and internal power switches are configured to meet the power consumption goals for the targeted low-power mode.

45.4.2.2 Stop mode exit sequence

Exit from a low-power stop mode is initiated either by a reset or an interrupt event. The following sequence then executes to restore the system to a run mode (RUN or VLPR):

1. The on-chip regulator in the PMC and internal power switches are restored.
2. Clock generators are enabled in the SCG.
3. System and bus clocks are enabled to all masters and slaves.
4. The CPU clock is enabled and the CPU begins servicing the reset or interrupt that initiated the exit from the low-power stop mode.

45.4.2.3 Aborted stop mode entry

If an interrupt occurs during a stop entry sequence, the SMC can abort the transition early and return to RUN mode without completely entering the stop mode. An aborted entry is possible only if the interrupt occurs before the PMC begins the transition to stop mode regulation. After this point, the interrupt is ignored until the PMC has completed its transition to stop mode regulation. When an aborted stop mode entry sequence occurs, SMC_PMCTRL[STOPA] is set to 1.

45.4.2.4 Transition to wait modes

For wait modes (WAIT and VLPW), the CPU clock is gated off while all other clocking continues, as in RUN and VLPR mode operation. Some modules that support stop-in-wait functionality have their clocks disabled in these configurations.

45.4.2.5 Transition from stop modes to Debug mode

The debugger module supports a transition from STOP, WAIT, VLPS, and VLPW back to a Halted state when the debugger has been enabled. As part of this transition, system clocking is re-established and is equivalent to the normal RUN and VLPR mode clocking configuration.

45.4.3 Run modes

The run modes supported by this device can be found here.

- Run (RUN)
- Very Low-Power Run (VLPR)
- High Speed Run (HSRUN)

45.4.3.1 RUN mode

This is the normal operating mode for the device.

This mode is selected after any reset. In order to reduce peak power consumption, however, the SMC begins transitioning the device into VLPR mode during the reset recovery sequence. When the ARM processor exits reset, it sets up the stack, program counter (PC), and link register (LR):

- The processor reads the start SP (SP_main) from vector-table offset 0x000
- The processor reads the start PC from vector-table offset 0x004
- LR is set to 0xFFFF_FFFF.

To reduce power in this mode, disable the clocks to unused modules.

45.4.3.2 Very-Low Power Run (VLPR) mode

In VLPR mode, the on-chip voltage regulator is put into a stop mode regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the PCC's registers.

Before entering this mode, the following conditions must be met:

- All clock monitors in the SCG must be disabled.
- The maximum frequencies of the system, bus, flash, and core are restricted. See the Power Management details about which frequencies are supported.
- Mode protection must be set to allow VLP modes, that is, PMPROT[AVLP] is 1.
- PMCTRL[RUNM] must be set to 10b to enter VLPR.
- Flash programming/erasing is not allowed.

NOTE

Do not increase the clock frequency while in VLPR mode, because the regulator is slow in responding and cannot manage fast load transitions. In addition, do not modify the clock source

in the SCG module or any clock divider registers. Module clock enables in the PCC can be set, but not cleared.however should only be cleared one at a time to limit load transitions.

To reenter Normal Run mode, clear PMCTRL[RUNM]. PMSTAT is a read-only status register that can be used to determine when the system has completed an exit to RUN mode. When PMSTAT=RUN, the system is in run regulation and the MCU can run at full speed in any clock mode. If a higher execution frequency is desired, poll PMSTAT until it is set to RUN when returning from VLPR mode.

Any reset always causes an exit from VLPR and returns the device to RUN mode after the MCU exits its reset flow.

Any reset will initially cause the device to exit to RUN mode, however Flash IFR settings can be used to return the part to VLPR during the MCU reset flow.

45.4.3.3 High Speed Run (HSRUN) mode

In HSRUN mode, the on-chip voltage regulator remains in a run regulation state, but with a slightly elevated voltage output. In this state, the MCU is able to operate at a faster frequency compared to normal RUN mode. For the maximum allowable frequencies, see the Power Management chapter.

While in this mode, the following restrictions must be adhered to:

- The maximum allowable change in frequency of the system, bus, flash or core clocks is restricted to 2x (double the frequency).
- Stop mode entry is not supported from HSRUN.
- Modifications to clock gating control bits are prohibited.
- Flash programming/erasing is not allowed.

To enter HSRUN mode, set PMPROT[AHSRUN] to allow HSRUN and then set PMCTRL[RUNM]=HSRUN. Before increasing clock frequencies, the PMSTAT register should be polled to determine when the system has completed entry into HSRUN mode. To reenter normal RUN mode, clear PMCTRL[RUNM]. Any reset also clears PMCTRL[RUNM] and causes the system to exit to normal RUN mode after the MCU exits its reset flow.

45.4.4 Wait modes

This device contains two different wait modes which are listed here.

- Wait
- Very-Low Power Wait (VLPW)

45.4.4.1 WAIT mode

WAIT mode is entered when the ARM core enters the Sleep-Now or Sleep-On-Exit modes while SLEEPDEEP is cleared. The ARM CPU enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled.

When an interrupt request occurs, the CPU exits WAIT mode and resumes processing in RUN mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset causes an exit from WAIT mode, returning the device to normal RUN mode.

A system reset causes an exit from WAIT mode, temporarily returning the device to RUN mode before transitioning into VLPR during the device reset flow.

45.4.4.2 Very-Low-Power Wait (VLPW) mode

VLPW mode is entered by entering the Sleep-Now or Sleep-On-Exit mode while SLEEPDEEP is cleared and the device is in VLPR mode.

In VLPW, the on-chip voltage regulator remains in its stop regulation state. In this state, the regulator is designed to supply enough current to the device at a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules.

VLPR mode restrictions also apply to VLPW.

When an interrupt from VLPW occurs, the device returns to VLPR mode to execute the interrupt service routine.

A system reset causes an exit from VLPW mode, returning the device to normal RUN mode.

A system reset causes an exit from VLPW mode, temporarily returning the device to RUN mode before transitioning into VLPR during the device reset flow.

45.4.5 Stop modes

This device contains a variety of stop modes to meet your application needs.

The stop modes range from:

- a stopped CPU, with all I/O, logic, and memory states retained, and certain asynchronous mode peripherals operating

to:

- a powered down CPU, with only I/O and a small register file retained, very few asynchronous mode peripherals operating, while the remainder of the MCU is powered down.

The choice of stop mode depends upon the user's application, and how power usage and state retention versus functional needs and recovery time may be traded off.

The various stop modes are selected by setting the appropriate fields in PMPROT and PMCTRL. The selected stop mode is entered during the sleep-now or sleep-on-exit entry with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The available stop modes are:

- Normal Stop (STOP)
- Very-Low Power Stop (VLPS)
- Low-Leakage Stop (LLS)
- Very-Low-Leakage Stop (VLLSx)

45.4.5.1 STOP mode

STOP mode is entered via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The SCG module can be configured to leave the reference clocks running.

A module capable of providing an asynchronous interrupt to the device takes the device out of STOP mode and returns the device to normal RUN mode. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode. When an interrupt request occurs, the CPU exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from STOP mode, returning the device to normal RUN mode via an MCU reset.

A system reset will cause an exit from STOP mode, temporarily returning the device to RUN mode before transitioning into VLPR during the MCU reset flow.

45.4.5.2 Very-Low-Power Stop (VLPS) mode

The two ways in which VLPS mode can be entered are listed here.

- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in VLPR mode and PMCTRL[STOPM] = 010 or 000.
- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in normal RUN mode and PMCTRL[STOPM] = 010. When VLPS is entered directly from RUN mode, exit to VLPR is disabled by hardware and the system will always exit back to RUN.

In VLPS, the on-chip voltage regulator remains in its stop regulation state as in VLPR.

A module capable of providing an asynchronous interrupt to the device takes the device out of VLPS and returns the device to VLPR mode.

A system reset will also cause a VLPS exit, returning the device to normal RUN mode.

A system reset will cause an exit from VLPS mode, temporarily returning the device to RUN mode before transitioning into VLPR during the MCU reset flow.

45.4.5.3 Low-Leakage Stop (LLSx) modes

This device contains two Low-Leakage Stop modes: LLS3 and LLS2. LLS or LLSx is often used in this document to refer to both modes. All LLS modes can be entered from normal RUN or VLPR modes.

The MCU enters LLS mode if:

- In Sleep-Now or Sleep-On-Exit mode, SLEEPDEEP is set in the System Control Register in the ARM core, and
- The device is configured as shown in Table 45-2.

In LLS, the on-chip voltage regulator is in stop regulation. Most of the peripherals are put in a state-retention mode that does not allow them to operate while in LLS.

Before entering LLS mode, the user should configure the Low-Leakage Wake-up (LLWU) module to enable the desired wake-up sources. The available wake-up sources in LLS are detailed in the chip configuration details for this device.

After wakeup from LLS, the device returns to the run mode from which LLS was entered (either normal RUN or VLPR) with a pending LLWU module interrupt. If LLS was entered from VLPR mode, then the MCU will begin VLPR entry shortly after wake-up. In the LLWU interrupt service routine (ISR), the user can poll the LLWU module wake-up flags to determine the source of the wakeup.

NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit Stop mode on an LLS recovery.

An asserted RESET pin will cause an exit from LLS mode, returning the device to normal RUN mode. When LLS is exiting via the RESET pin, RCM_SRS[PIN] and RCM_SRS[WAKEUP] are set.

An asserted RESET pin will cause an exit from LLS mode, temporarily returning the device to normal RUN mode before transitioning into VLPR during the MCU reset flow. When LLS is exiting via the RESET pin, RCM_SRS[PIN] and RCM_SRS[WAKEUP] are set.

45.4.5.4 Very-Low-Leakage Stop (VLLSx) modes

This device contains these very low leakage modes:

- VLLS3
- VLLS2
- VLLS1
- VLLS0

VLLSx is often used in this document to refer to all of these modes.

All VLLSx modes can be entered from normal RUN or VLPR modes.

The MCU enters the configured VLLS mode if:

- In Sleep-Now or Sleep-On-Exit mode, the SLEEPDEEP bit is set in the System Control Register in the ARM core, and
- The device is configured as shown in Table 45-2.

In VLLS, the on-chip voltage regulator is in its stop-regulation state while most digital logic is powered off.

Before entering VLLS mode, the user should configure the Low-Leakage Wake-up (LLWU) module to enable the desired wakeup sources. The available wake-up sources in VLLS are detailed in the chip configuration details for this device.

After wakeup from VLLS, the device returns to normal RUN mode with a pending LLWU interrupt. If VLLS was entered from VLPR mode, then the MCU will begin VLPR entry shortly after wakeup. In the LLWU interrupt service routine (ISR), the user can poll the LLWU module wake-up flags to determine the source of the wake-up.

When entering VLLS, each I/O pin is latched as configured before executing VLLS. Because all digital logic in the MCU is powered off, all port and peripheral data is lost during VLLS. This information must be restored before PMC_REGSC[ACKISO] is set.

An asserted $\overline{\text{RESET}}$ pin will cause an exit from any VLLS mode, returning the device to normal RUN mode. When exiting VLLS via the $\overline{\text{RESET}}$ pin, RCM_SRS[PIN] and RCM_SRS[WAKEUP] are set.

An asserted $\overline{\text{RESET}}$ pin will cause an exit from any VLLS mode, temporarily returning the device to normal RUN mode before transitioning into VLPR during the MCU reset flow. When exiting VLLS via the $\overline{\text{RESET}}$ pin, RCM_SRS[PIN] and RCM_SRS[WAKEUP] are set.

45.4.6 Debug in low power modes

When the MCU is secure, the device disables/limits debugger operation. When the MCU is unsecure, the ARM debugger can assert two power-up request signals:

- System power up, via SYSPWR in the Debug Port Control/Stat register
- Debug power up, via CDBGPWRUPREQ in the Debug Port Control/Stat register

When asserted while in RUN, WAIT, VLPR, or VLPR the mode controller drives a corresponding acknowledge for each signal, that is, both CDBGPWRUPACK and CSYSPWRUPACK. When both requests are asserted, the mode controller handles attempts to enter STOP and VLPS by entering an emulated stop state. In this emulated stop state:

- the regulator is in run regulation,
- the SCG-generated clock source is enabled,
- all system clocks, except the core clock, are disabled,
- the debug module has access to core registers, and
- access to the on-chip peripherals is blocked.

No debug is available while the MCU is in LLS or VLLS modes. LLS is a state-retention mode and all debug operation can continue after waking from LLS, even in cases where system wakeup is due to a system reset event.

Entering into a VLLS mode causes all of the debug controls and settings to be powered off. To give time to the debugger to sync with the MCU, the MDM AP Control Register includes a Very-Low-Leakage Debug Request (VLLDBGREQ) bit that is set to configure the Reset Controller logic to hold the system in reset after the next recovery from a VLLS mode. This bit allows the debugger time to reinitialize the debug module before the debug session continues.

The MDM AP Control Register also includes a Very Low Leakage Debug Acknowledge (VLLDBGACK) bit that is set to release the ARM core being held in reset following a VLLS recovery. The debugger reinitializes all debug IP, and then asserts the VLLDBGACK control bit to allow the RCM to release the ARM core from reset and allow CPU operation to begin.

The VLLDBGACK bit is cleared by the debugger (or can be left set as is) or clears automatically due to the reset generated as part of the next VLLS recovery.

Chapter 46
System Register File

46.1 System Register file

This device includes a 32-byte register file that is powered in all power modes. This register file retains its contents during low-voltage detect (LVD) events and is only reset during a power-on reset.

46.2 Memory Map and Registers

RFSYS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_C000	Register file register (RFSYS_REG0)	32	R/W	0000_0000h	46.2.1/1155
4007_C004	Register file register (RFSYS_REG1)	32	R/W	0000_0000h	46.2.1/1155
4007_C008	Register file register (RFSYS_REG2)	32	R/W	0000_0000h	46.2.1/1155
4007_C00C	Register file register (RFSYS_REG3)	32	R/W	0000_0000h	46.2.1/1155
4007_C010	Register file register (RFSYS_REG4)	32	R/W	0000_0000h	46.2.1/1155
4007_C014	Register file register (RFSYS_REG5)	32	R/W	0000_0000h	46.2.1/1155
4007_C018	Register file register (RFSYS_REG6)	32	R/W	0000_0000h	46.2.1/1155
4007_C01C	Register file register (RFSYS_REG7)	32	R/W	0000_0000h	46.2.1/1155

46.2.1 Register file register (RFSYS_REG*n*)

Each register can be accessed as 8-, 16-, or 32-bits.

Address: 4007_C000h base + 0h offset + (4d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HH								HL								LH								LL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RFSYS_REG*n* field descriptions

Field	Description
31–24 HH	High higher byte
23–16 HL	High lower byte
15–8 LH	Low higher byte
LL	Low lower byte

Chapter 47
Timer/PWM Module (TPM)

47.1 Introduction

The TPM (Timer/PWM Module) is a 2- to 8-channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications.

The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low power modes. An example of using the TPM with the asynchronous DMA is described in AN4631:Using the Asynchronous DMA features of the Kinetis L Series .

47.1.1 TPM Philosophy

The TPM is built upon a very simple timer (HCS08 Timer PWM Module – TPM) used for many years on NXP's 8-bit microcontrollers. The TPM extends the functionality to support operation in low power modes by clocking the counter, compare and capture registers from an asynchronous clock that can remain functional in low power modes.

47.1.2 Features

The TPM features include:

- TPM clock mode is selectable
  - Can increment on every edge of the asynchronous counter clock
  - Can increment on rising edge of an external clock input synchronized to the asynchronous counter clock
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128

- TPM includes a 16-bit counter
  - It can be a free-running counter or modulo counter
  - The counting can be up or up-down
- Includes 6 channels that can be configured for input capture, output compare, edge-aligned PWM mode, or center-aligned PWM mode
  - In input capture mode the capture can occur on rising edges, falling edges or both edges
  - In output compare mode the output signal can be set, cleared, pulsed, or toggled on match
  - All channels can be configured for edge-aligned PWM mode or center-aligned PWM mode
- Support the generation of an interrupt and/or DMA request per channel
- Support the generation of an interrupt and/or DMA request when the counter overflows
- Support selectable trigger input to optionally reset or cause the counter to start incrementing.
  - The counter can also optionally stop incrementing on counter overflow
- Support the generation of hardware triggers when the counter overflows and per channel

47.1.3 Modes of operation

During debug mode, the TPM can can be configured to temporarily pause all counting until the core returns to normal user operating mode or to operate normally. When the counter is paused, trigger inputs and input capture events are ignored.

During doze mode, the TPM can be configured to operate normally or to pause all counting for the duration of doze mode. When the counter is paused, trigger inputs and input capture events are ignored.

During stop mode, the TPM counter clock can remain functional and the TPM can generate an asynchronous interrupt to exit the MCU from stop mode.

47.1.4 Block diagram

The TPM uses one input/output (I/O) pin per channel, CHn (TPM channel (n)) where n is the channel number.

The following figure shows the TPM structure. The central component of the TPM is the 16-bit counter with programmable final value and its counting can be up or up-down.

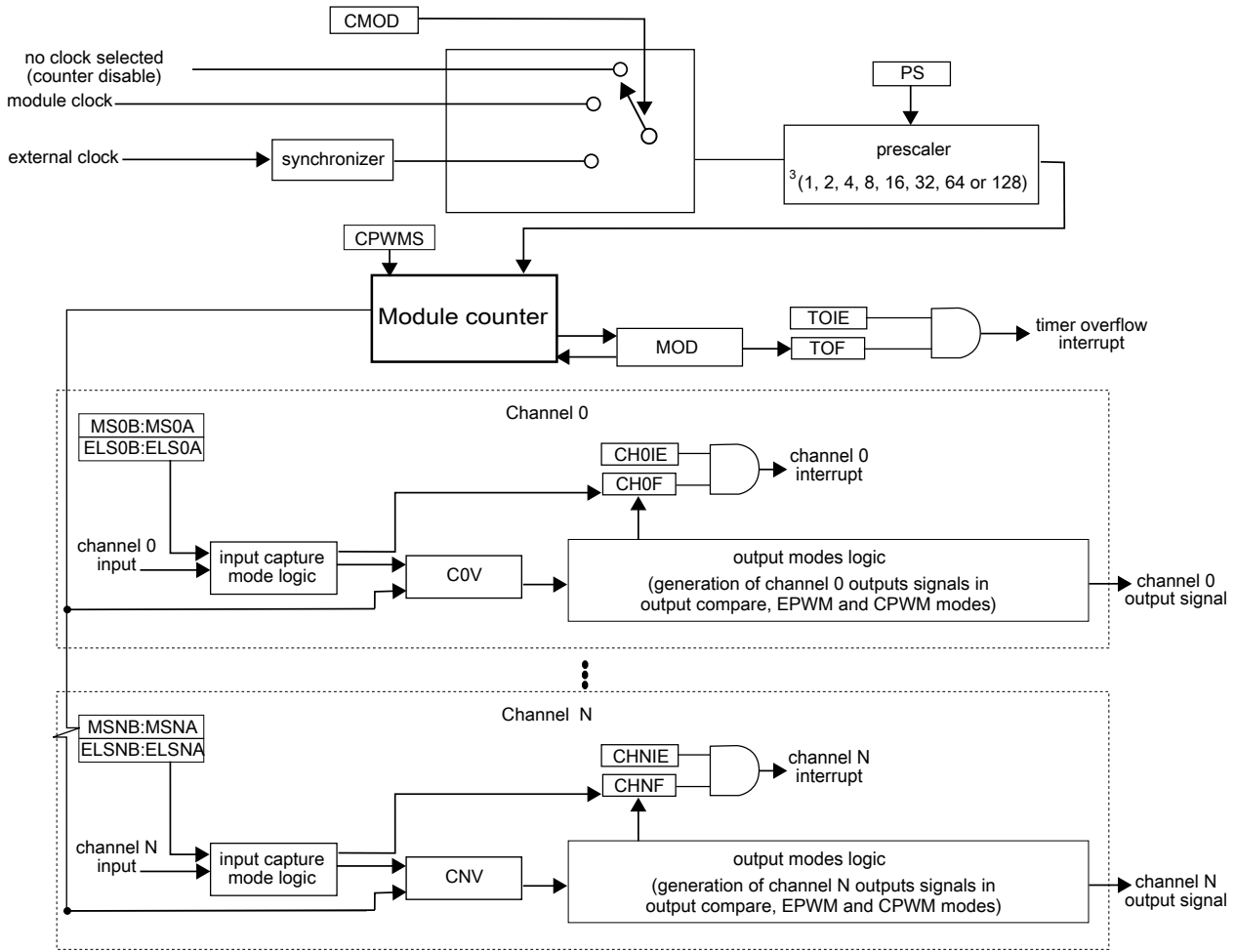


Figure 47-1. TPM block diagram

47.2 TPM Signal Descriptions

Table 47-1. TPM signal descriptions

Signal	Description	I/O
TPM_EXTCLK	External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock.	I
TPM_CHn	TPM channel (n = 5 to 0). A TPM channel pin is configured as output when configured in an output compare or PWM mode and the TPM counter is enabled, otherwise the TPM channel pin is an input.	I/O

47.2.1 TPM_EXTCLK — TPM External Clock

The rising edge of the external input signal is used to increment the TPM counter if selected by CMOD[1:0] bits in the SC register. This input signal must be less than half of the TPM counter clock frequency. The TPM counter prescaler selection and settings are also used when an external input is selected.

47.2.2 TPM_CHn — TPM Channel (n) I/O Pin

Each TPM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.

47.3 Memory Map and Register Definition

This section provides a detailed description of all TPM registers.

Attempting to access a reserved register location in the TPM memory map will generate a bus error.

TPM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Version ID Register (TPM_VERID)	32	R	0500_0007h	47.3.1/1161
4	Parameter Register (TPM_PARAM)	32	R	See section	47.3.2/1162
8	TPM Global Register (TPM_GLOBAL)	32	R/W	0000_0000h	47.3.3/1162
10	Status and Control (TPM_SC)	32	R/W	0000_0000h	47.3.4/1163
14	Counter (TPM_CNT)	32	R/W	0000_0000h	47.3.5/1165
18	Modulo (TPM_MOD)	32	R/W	0000_FFFFh	47.3.6/1165
1C	Capture and Compare Status (TPM_STATUS)	32	R/W	0000_0000h	47.3.7/1166
20	Channel (n) Status and Control (TPM_C0SC)	32	R/W	0000_0000h	47.3.8/1168
24	Channel (n) Value (TPM_C0V)	32	R/W	0000_0000h	47.3.9/1170
28	Channel (n) Status and Control (TPM_C1SC)	32	R/W	0000_0000h	47.3.8/1168
2C	Channel (n) Value (TPM_C1V)	32	R/W	0000_0000h	47.3.9/1170
30	Channel (n) Status and Control (TPM_C2SC)	32	R/W	0000_0000h	47.3.8/1168
34	Channel (n) Value (TPM_C2V)	32	R/W	0000_0000h	47.3.9/1170
38	Channel (n) Status and Control (TPM_C3SC)	32	R/W	0000_0000h	47.3.8/1168

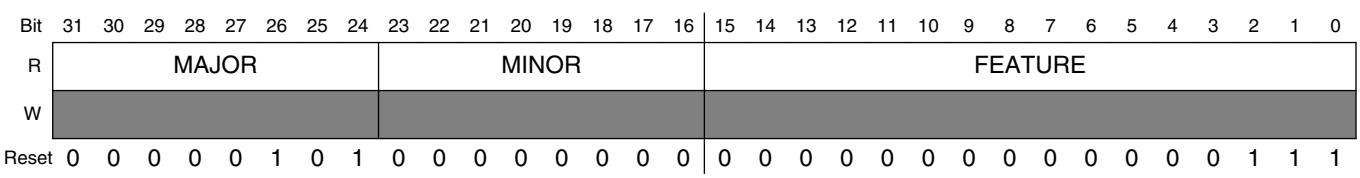
Table continues on the next page...

TPM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3C	Channel (n) Value (TPM_C3V)	32	R/W	0000_0000h	47.3.9/1170
40	Channel (n) Status and Control (TPM_C4SC)	32	R/W	0000_0000h	47.3.8/1168
44	Channel (n) Value (TPM_C4V)	32	R/W	0000_0000h	47.3.9/1170
48	Channel (n) Status and Control (TPM_C5SC)	32	R/W	0000_0000h	47.3.8/1168
4C	Channel (n) Value (TPM_C5V)	32	R/W	0000_0000h	47.3.9/1170
64	Combine Channel Register (TPM_COMBINE)	32	R/W	0000_0000h	47.3.10/ 1171
6C	Channel Trigger (TPM_TRIG)	32	R/W	0000_0000h	47.3.11/ 1173
70	Channel Polarity (TPM_POL)	32	R/W	0000_0000h	47.3.12/ 1174
78	Filter Control (TPM_FILTER)	32	R/W	0000_0000h	47.3.13/ 1175
80	Quadrature Decoder Control and Status (TPM_QDCTRL)	32	R/W	0000_0000h	47.3.14/ 1176
84	Configuration (TPM_CONF)	32	R/W	0000_0000h	47.3.15/ 1177

47.3.1 Version ID Register (TPM_VERID)

Address: 0h base + 0h offset = 0h

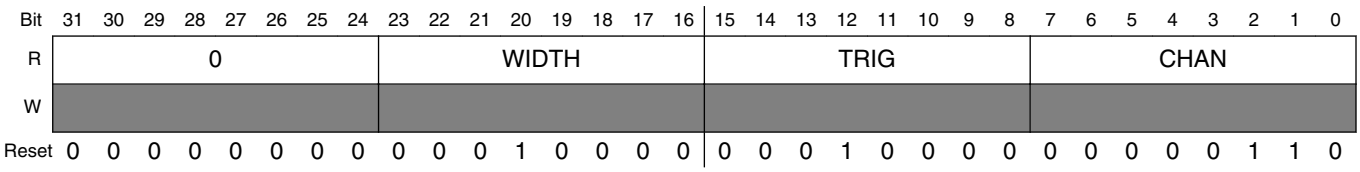


TPM_VERID field descriptions

Field	Description
31–24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23–16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
FEATURE	Feature Identification Number 0x0001 Standard feature set. 0x0003 Standard feature set with Filter and Combine registers implemented. 0x0007 Standard feature set with Filter, Combine and Quadrature registers implemented.

47.3.2 Parameter Register (TPM_PARAM)

Address: 0h base + 4h offset = 4h

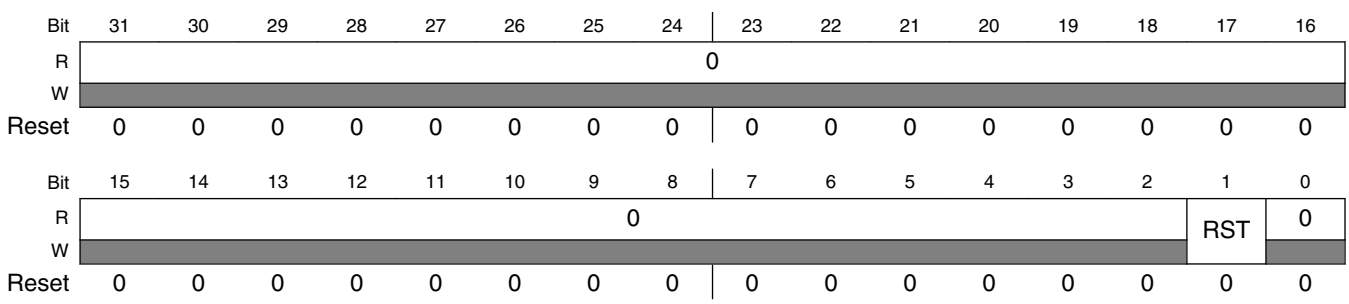


TPM_PARAM field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 WIDTH	Counter Width Width of the counter and timer channels.
15–8 TRIG	Trigger Count Number of trigger inputs implemented.
CHAN	Channel Count Number of timer channels implemented.

47.3.3 TPM Global Register (TPM_GLOBAL)

Address: 0h base + 8h offset = 8h



TPM_GLOBAL field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 RST	Software Reset Reset all internal logic and registers, except the Global Register. Remains set until cleared by software.

Table continues on the next page...

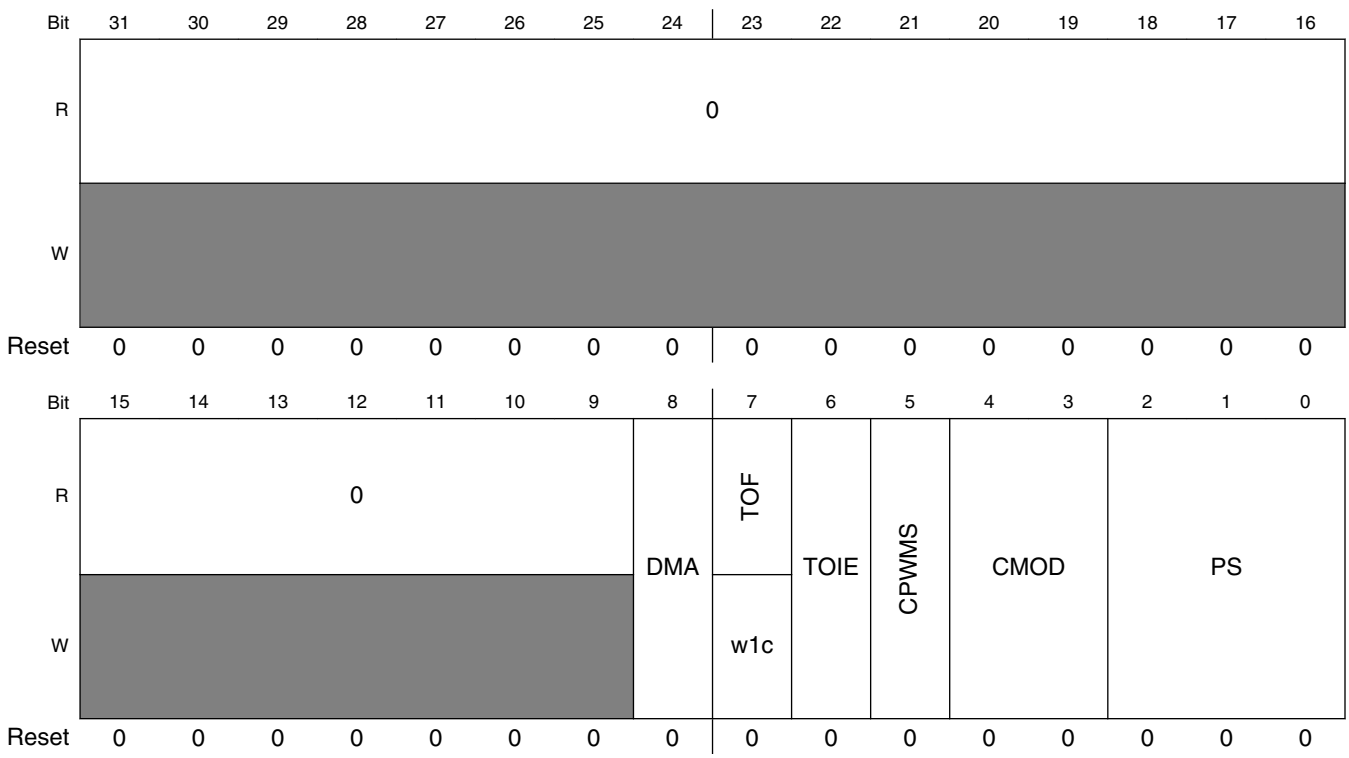
TPM_GLOBAL field descriptions (continued)

Field	Description
	0 Module is not reset. 1 Module is reset.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

47.3.4 Status and Control (TPM_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, module configuration and prescaler factor. These controls relate to all channels within this module.

Address: 0h base + 10h offset = 10h



TPM_SC field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 DMA	DMA Enable Enables DMA transfers for the overflow flag.

Table continues on the next page...

TPM_SC field descriptions (continued)

Field	Description
	0 Disables DMA transfers. 1 Enables DMA transfers.
7 TOF	Timer Overflow Flag Set by hardware when the TPM counter equals the value in the MOD register and increments. Writing a 1 to TOF clears it. Writing a 0 to TOF has no effect. If another TPM overflow occurs between the flag setting and the flag clearing, the write operation has no effect; therefore, TOF remains set indicating another overflow has occurred. In this case a TOF interrupt request is not lost due to a delay in clearing the previous TOF. 0 TPM counter has not overflowed. 1 TPM counter has overflowed.
6 TOIE	Timer Overflow Interrupt Enable Enables TPM overflow interrupts. 0 Disable TOF interrupts. Use software polling or DMA request. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the TPM to operate in up-down counting mode. This field is write protected. It can be written only when the counter is disabled. 0 TPM counter operates in up counting mode. 1 TPM counter operates in up-down counting mode.
4–3 CMOD	Clock Mode Selection Selects the TPM counter clock modes. When disabling the counter, this field remain set until acknolwedged in the TPM clock domain. 00 TPM counter is disabled 01 TPM counter increments on every TPM counter clock 10 TPM counter increments on rising edge of TPM_EXTCLK synchronized to the TPM counter clock 11 TPM counter increments on rising edge of the selected external input trigger.
PS	Prescale Factor Selection Selects one of 8 division factors for the clock mode selected by CMOD. This field is write protected. It can be written only when the counter is disabled. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128

47.3.5 Counter (TPM_CNT)

The CNT register contains the TPM counter value.

Reset clears the CNT register. Writing any value to COUNT also clears the counter.

When debug is active, the TPM counter does not increment unless configured otherwise.

Reading the CNT register adds two wait states to the register access due to synchronization delays.

Address: 0h base + 14h offset = 14h

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

TPM_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Counter value

47.3.6 Modulo (TPM_MOD)

The Modulo register contains the modulo value for the TPM counter. When the TPM counter reaches the modulo value and increments, the overflow flag (TOF) is set and the next value of TPM counter depends on the selected counting method (see Counter).

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to MOD Register Update . Additional writes to the MOD write buffer are ignored until the register has been updated.

It is recommended to initialize the TPM counter (write to CNT) before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Address: 0h base + 18h offset = 18h

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

TPM_MOD field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MOD	Modulo value This field must be written with single 16-bit or 32-bit access.

47.3.7 Capture and Compare Status (TPM_STATUS)

The STATUS register contains a copy of the status flag, CnSC[CHnF] for each TPM channel, as well as SC[TOF], for software convenience.

Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by writing all ones to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. Writing a 1 to CHF clears it. Writing a 0 to CHF has no effect.

If another event occurs between the flag setting and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

Address: 0h base + 1Ch offset = 1Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0							TOF	0		CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W								w1c			w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TPM_STATUS field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 TOF	Timer Overflow Flag See register description 0 TPM counter has not overflowed. 1 TPM counter has overflowed.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 CH5F	Channel 5 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
4 CH4F	Channel 4 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
3 CH3F	Channel 3 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
2 CH2F	Channel 2 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
1 CH1F	Channel 1 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.

47.3.8 Channel (n) Status and Control (TPM_C*n*SC)

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function. When switching from one channel mode to a different channel mode, the channel must first be disabled and this must be acknowledged in the TPM counter clock domain.

Table 47-2. Mode, Edge, and Level Selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	00	00	None	Channel disabled
X	01	00	Software compare	Pin not used for TPM
0	00	01	Input capture	Capture on Rising Edge Only
		10		Capture on Falling Edge Only
		11		Capture on Rising or Falling Edge
	01	01	Output compare	Toggle Output on match
		10		Clear Output on match
		11		Set Output on match
	10	10	Edge-aligned PWM	High-true pulses (clear Output on match, set Output on reload)
		X1		Low-true pulses (set Output on match, clear Output on reload)
	11	10	Output compare	Pulse Output low on match
		01		Pulse Output high on match
1	10	10	Center-aligned PWM	High-true pulses (clear Output on match-up, set Output on match-down)
		01		Low-true pulses (set Output on match-up, clear Output on match-down)

Address: 0h base + 20h offset + (8d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0								CHF	CHIE	MSB	MSA	ELSB	ELSA	0	DMA
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TPM_C*n*SC field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CHF	Channel Flag Set by hardware when an event occurs on the channel. CHF is cleared by writing a 1 to the CHF bit. Writing a 0 to CHF has no effect. If another event occurs between the CHF sets and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the delay in clearing the previous CHF. 0 No channel event has occurred. 1 A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0 Disable channel interrupts. 1 Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 DMA	DMA Enable Enables DMA transfers for the channel. 0 Disable DMA transfers. 1 Enable DMA transfers.

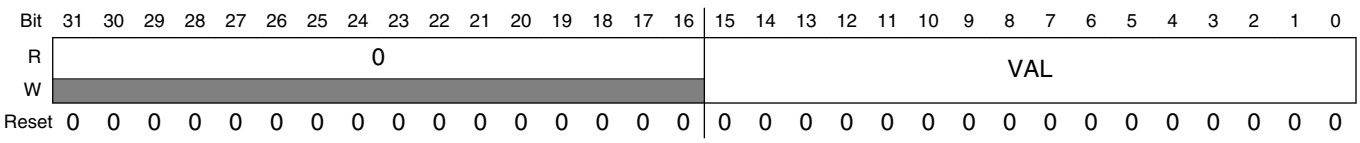
47.3.9 Channel (n) Value (TPM_C*n*V)

These registers contain the captured TPM counter value for the input modes or the match value for the output modes.

In input capture mode, any write to a CnV register is ignored.

In compare modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to CnV Register Update . Additional writes to the CnV write buffer are ignored until the register has been updated.

Address: 0h base + 24h offset + (8d × i), where i=0d to 5d



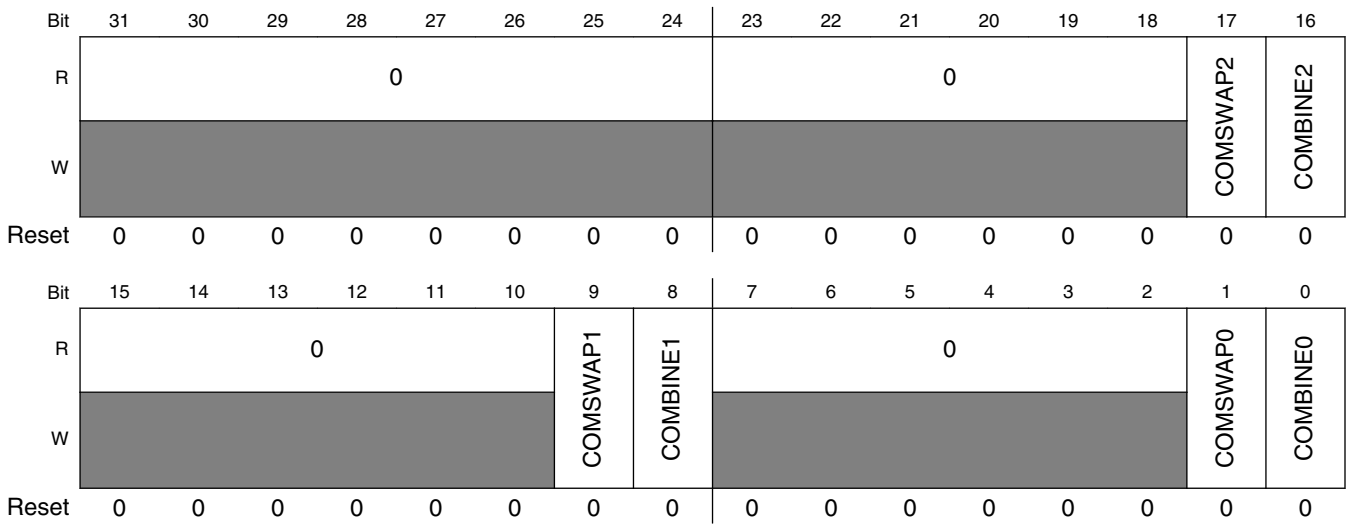
TPM_C*n*V field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
VAL	Channel Value Captured TPM counter value of the input modes or the match value for the output modes. This field must be written with single 16-bit or 32-bit access.

47.3.10 Combine Channel Register (TPM_COMBINE)

This register contains the control bits used to configure the combine channel modes for each pair of channels (n) and (n+1), where n is all the even numbered channels.

Address: 0h base + 64h offset = 64h



TPM_COMBINE field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 COMSWAP2	Combine Channels 4 and 5 Swap When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare. 0 Even channel is used for input capture and 1st compare. 1 Odd channel is used for input capture and 1st compare.
16 COMBINE2	Combine Channels 4 and 5 Enables the combine feature for channels 2 and 3. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0 Channels 4 and 5 are independent. 1 Channels 4 and 5 are combined.
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 COMSWAP1	Combine Channels 2 and 3 Swap

Table continues on the next page...

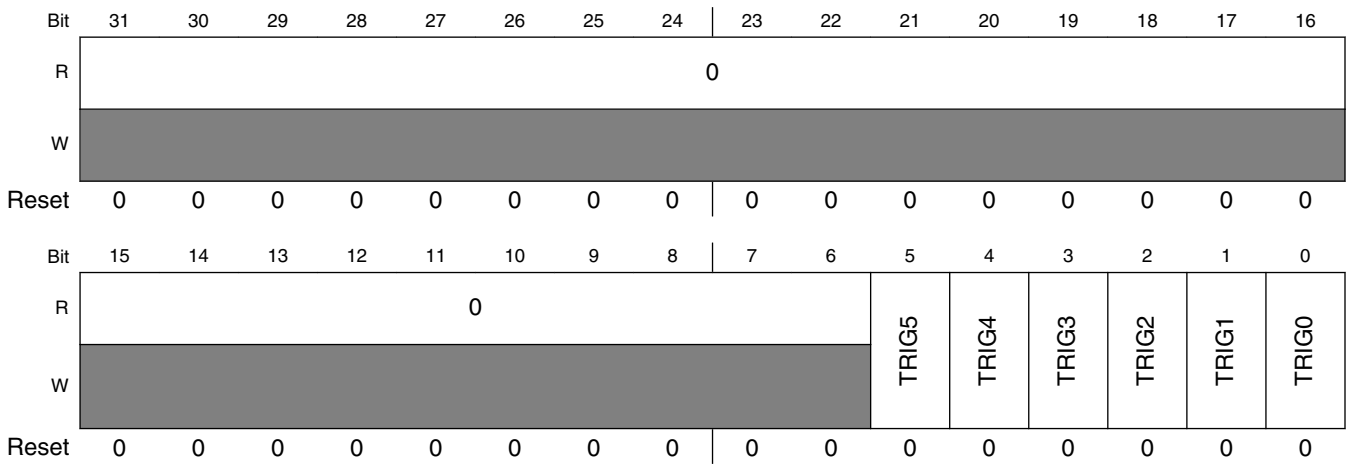
TPM_COMBINE field descriptions (continued)

Field	Description
	When set in combine mode, the odd channel is used for the input capture and 1st compare, the even channel is used for the 2nd compare. 0 Even channel is used for input capture and 1st compare. 1 Odd channel is used for input capture and 1st compare.
8 COMBINE1	Combine Channels 2 and 3 Enables the combine feature for channels 2 and 3. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0 Channels 2 and 3 are independent. 1 Channels 2 and 3 are combined.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 COMSWAP0	Combine Channel 0 and 1 Swap When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare. 0 Even channel is used for input capture and 1st compare. 1 Odd channel is used for input capture and 1st compare.
0 COMBINE0	Combine Channels 0 and 1 Enables the combine feature for channels 0 and 1. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0 Channels 0 and 1 are independent. 1 Channels 0 and 1 are combined.

47.3.11 Channel Trigger (TPM_TRIG)

This register configures the trigger input for each channel.

Address: 0h base + 6Ch offset = 6Ch



TPM_TRIG field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 TRIG5	Channel 5 Trigger 0 No effect. 1 The input trigger is used for input capture and modulates output (for output compare and PWM).
4 TRIG4	Channel 4 Trigger 0 No effect. 1 The input trigger is used for input capture and modulates output (for output compare and PWM).
3 TRIG3	Channel 3 Trigger 0 No effect. 1 The input trigger is used for input capture and modulates output (for output compare and PWM).
2 TRIG2	Channel 2 Trigger 0 No effect. 1 The input trigger is used for input capture and modulates output (for output compare and PWM).
1 TRIG1	Channel 1 Trigger 0 No effect. 1 The input trigger is used for input capture and modulates output (for output compare and PWM).
0 TRIG0	Channel 0 Trigger 0 No effect. 1 The input trigger is used for input capture and modulates output (for output compare and PWM).

47.3.12 Channel Polarity (TPM_POL)

This register defines the input and output polarity of each of the channels.

Address: 0h base + 70h offset = 70h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
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
R	0										POL5	POL4	POL3	POL2	POL1	POL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

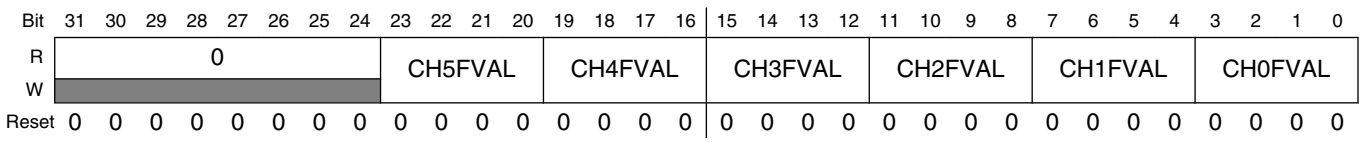
TPM_POL field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 POL5	Channel 5 Polarity 0 The channel polarity is active high. 1 The channel polarity is active low.
4 POL4	Channel 4 Polarity 0 The channel polarity is active high 1 The channel polarity is active low.
3 POL3	Channel 3 Polarity 0 The channel polarity is active high. 1 The channel polarity is active low.
2 POL2	Channel 2 Polarity 0 The channel polarity is active high. 1 The channel polarity is active low.
1 POL1	Channel 1 Polarity 0 The channel polarity is active high. 1 The channel polarity is active low.
0 POL0	Channel 0 Polarity 0 The channel polarity is active high. 1 The channel polarity is active low.

47.3.13 Filter Control (TPM_FILTER)

This register selects the filter value of the channel inputs, and an additional output delay value for the channel outputs. In PWM combine modes, the filter can effectively implements deadtime insertion.

Address: 0h base + 78h offset = 78h



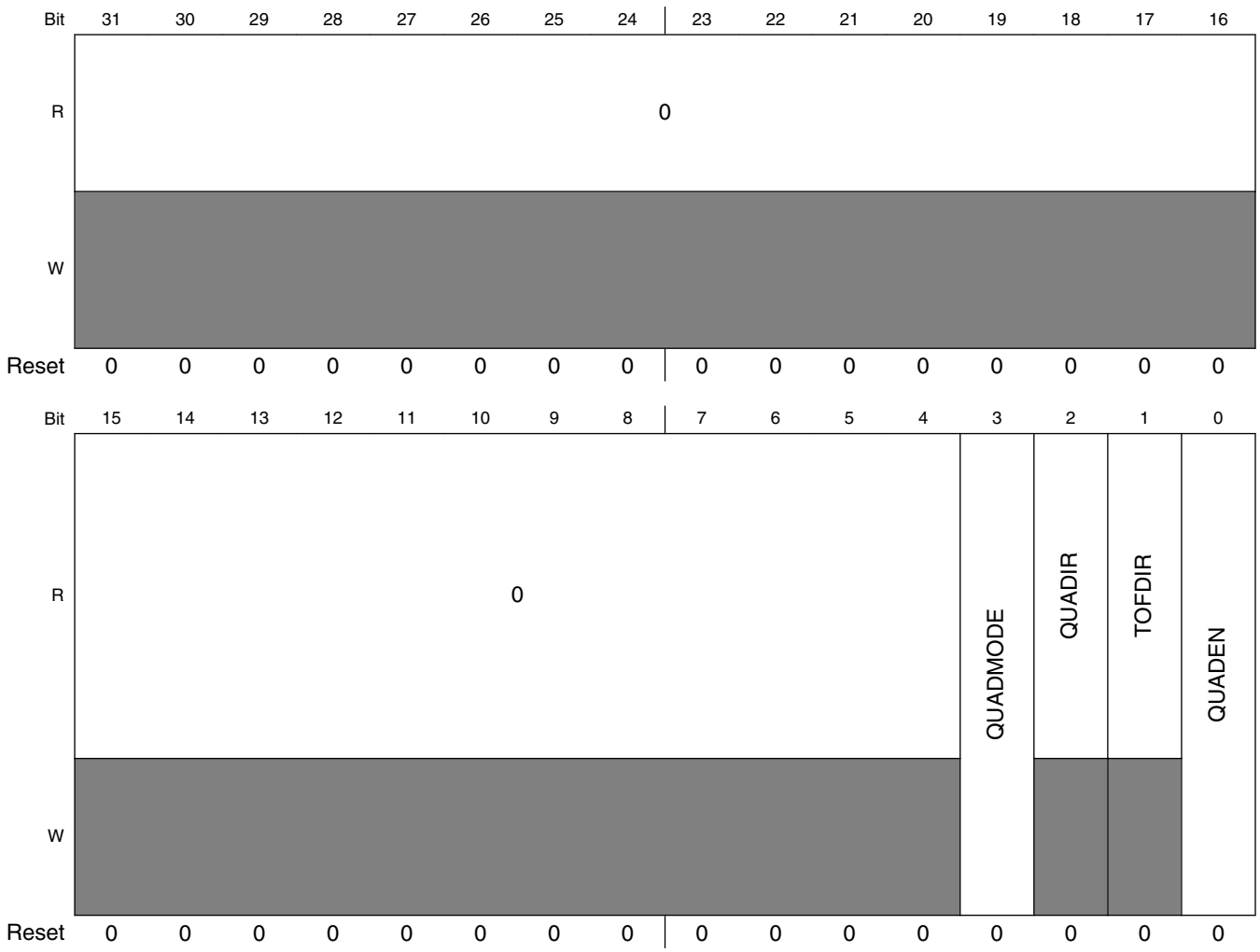
TPM_FILTER field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–20 CH5FVAL	Channel 5 Filter Value Selects the filter value for the channel input and the delay value for the channel output. The filter/delay is disabled when the value is zero, otherwise the filter/delay is configured as (CH5FVAL * 4) clock cycles.
19–16 CH4FVAL	Channel 4 Filter Value Selects the filter value for the channel input and the delay value for the channel output. The filter/delay is disabled when the value is zero, otherwise the filter/delay is configured as (CH4FVAL * 4) clock cycles.
15–12 CH3FVAL	Channel 3 Filter Value Selects the filter value for the channel input and the delay value for the channel output. The filter/delay is disabled when the value is zero, otherwise the filter/delay is configured as (CH3FVAL * 4) clock cycles.
11–8 CH2FVAL	Channel 2 Filter Value Selects the filter value for the channel input and the delay value for the channel output. The filter/delay is disabled when the value is zero, otherwise the filter/delay is configured as (CH2FVAL * 4) clock cycles.
7–4 CH1FVAL	Channel 1 Filter Value Selects the filter value for the channel input and the delay value for the channel output. The filter/delay is disabled when the value is zero, otherwise the filter/delay is configured as (CH1FVAL * 4) clock cycles.
CH0FVAL	Channel 0 Filter Value Selects the filter value for the channel input and the delay value for the channel output. The filter/delay is disabled when the value is zero, otherwise the filter/delay is configured as (CH0FVAL * 4) clock cycles.

47.3.14 Quadrature Decoder Control and Status (TPM_QDCTRL)

This register has the control and status bits for the quadrature decoder mode.

Address: 0h base + 80h offset = 80h



TPM_QDCTRL field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the quadrature decoder mode. 0 Phase encoding mode. 1 Count and direction encoding mode.
2 QUADIR	Counter Direction in Quadrature Decode Mode

Table continues on the next page...

TPM_QDCTRL field descriptions (continued)

Field	Description
	Indicates the counting direction. 0 Counter direction is decreasing (counter decrement). 1 Counter direction is increasing (counter increment).
1 TOFDIR	Indicates if the TOF bit was set on the top or the bottom of counting. 0 TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (zero) to its maximum value (MOD register). 1 TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (zero).
0 QUADEN	Enables the quadrature decoder mode. In this mode, the channel 0 and channel 1 inputs control the TPM counter direction and can only be used for software compare. The quadrature decoder mode has precedence over the other modes. 0 Quadrature decoder mode is disabled. 1 Quadrature decoder mode is enabled.

47.3.15 Configuration (TPM_CONF)

This register selects the behavior in debug and wait modes and the use of an external global time base.

Address: 0h base + 84h offset = 84h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0		TRGSEL		TRGSRC	TRGPOL	0		CPOT	CROT	CSOO	CSOT
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
R	0						GTBEEN	GTBSYNC	DBGMODE		DOZEEN	0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TPM_CONF field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

TPM_CONF field descriptions (continued)

Field	Description
27–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 TRGSEL	Trigger Select Selects the input trigger to use for starting, reloading and/or pausing the counter. The source of the trigger (external or internal to the TPM) is configured by the TRGSRC field. This field should only be changed when the TPM counter is disabled. See the chip-specific TPM information for available options.
23 TRGSRC	Trigger Source Selects between internal (channel pin input capture) or external trigger sources. When selecting an internal trigger, the channel selected should be configured for input capture. Only a rising edge input capture can be used to initially start the counter using the CSOT configuration; either rising edge or falling edge input capture can be used to reload the counter using the CROT configuration; and the state of the channel input pin is used to pause the counter using the CPOT configuration. The channel polarity register can be used to invert the polarity of the channel input pins. This field should only be changed when the TPM counter is disabled. 0 Trigger source selected by TRGSEL is external. 1 Trigger source selected by TRGSEL is internal (channel pin input capture).
22 TRGPOL	Trigger Polarity Selects the polarity of the external trigger source. This field should only be changed when the TPM counter is disabled. 0 Trigger is active high. 1 Trigger is active low.
21–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19 CPOT	Counter Pause On Trigger When enabled, the counter will pause incrementing while the trigger remains asserted (level sensitive). This field should only be changed when the TPM counter is disabled.
18 CROT	Counter Reload On Trigger When set, the TPM counter will reload with 0 (and initialize PWM outputs to their default value) when a rising edge is detected on the selected trigger input. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0 Counter is not reloaded due to a rising edge on the selected input trigger 1 Counter is reloaded when a rising edge is detected on the selected input trigger
17 CSOO	Counter Stop On Overflow When set, the TPM counter will stop incrementing once the counter equals the MOD value and incremented (this also sets the TOF). Reloading the counter with 0 due to writing to the counter register or due to a trigger input does not cause the counter to stop incrementing. Once the counter has stopped incrementing, the counter will not start incrementing unless it is disabled and then enabled again, or a rising edge on the selected trigger input is detected when CSOT set. This field should only be changed when the TPM counter is disabled.

Table continues on the next page...

TPM_CONF field descriptions (continued)

Field	Description
	0 TPM counter continues incrementing or decrementing after overflow 1 TPM counter stops incrementing or decrementing after overflow.
16 CSOT	Counter Start on Trigger When set, the TPM counter will not start incrementing after it is enabled until a rising edge on the selected trigger input is detected. If the TPM counter is stopped due to an overflow, a rising edge on the selected trigger input will also cause the TPM counter to start incrementing again. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0 TPM counter starts to increment immediately, once it is enabled. 1 TPM counter only starts to increment when it a rising edge on the selected input trigger is detected, after it has been enabled or after it has stopped due to overflow.
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 GTBEEN	Global time base enable Configures the TPM to use an externally generated global time base counter. When an externally generated timebase is used, the internal TPM counter is not used by the channels but can be used to generate a periodic interruptor DMA request using the Modulo register and timer overflow flag. 0 All channels use the internally generated TPM counter as their timebase 1 All channels use an externally generated global timebase as their timebase
8 GTBSYNC	Global Time Base Synchronization When enabled, the TPM counter is synchronized to the global time base. It uses the global timebase enable, trigger and overflow to ensure the TPM counter starts incrementing at the same time as the global timebase, stops incrementing at the same time as the global timebase and is reset at the same time as the global timebase. This field should only be changed when the TPM counter is disabled. 0 Global timebase synchronization disabled. 1 Global timebase synchronization enabled.
7–6 DBGMODE	Debug Mode Configures the TPM behavior in debug mode. All other configurations are reserved. 00 TPM counter is paused and does not increment during debug mode. Trigger inputs and input capture events are also ignored. 11 TPM counter continues in debug mode.
5 DOZEEN	Doze Enable Configures the TPM behavior in wait mode. 0 Internal TPM counter continues in Doze mode. 1 Internal TPM counter is paused and does not increment during Doze mode. Trigger inputs and input capture events are also ignored.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

47.4 Functional description

The following sections describe the TPM features.

47.4.1 Clock domains

The TPM module supports two clock domains.

The bus clock domain is used by the register interface and for synchronizing interrupts and DMA requests.

The TPM counter clock domain is used to clock the counter and prescaler along with the output compare and input capture logic. The TPM counter clock is considered asynchronous to the bus clock, can be a higher or lower frequency than the bus clock and can remain operational in Stop mode. Multiple TPM instances are all clocked by the same TPM counter clock in support of the external timebase feature.

47.4.1.1 Counter Clock Mode

The CMOD[1:0] bits in the SC register either disable the TPM counter or select one of two possible clock modes for the TPM counter. After any reset, CMOD[1:0] = 0:0 so the TPM counter is disabled.

The CMOD[1:0] bits may be read or written at any time. Disabling the TPM counter by writing zero to the CMOD[1:0] bits does not affect the TPM counter value or other registers, but must be acknowledged by the TPM counter clock domain before they read as zero.

The external clock input passes through a synchronizer clocked by the TPM counter clock to assure that counter transitions are properly aligned to counter clock transitions. Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must be less than half of the counter clock frequency.

47.4.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter.

The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and TPM counter.

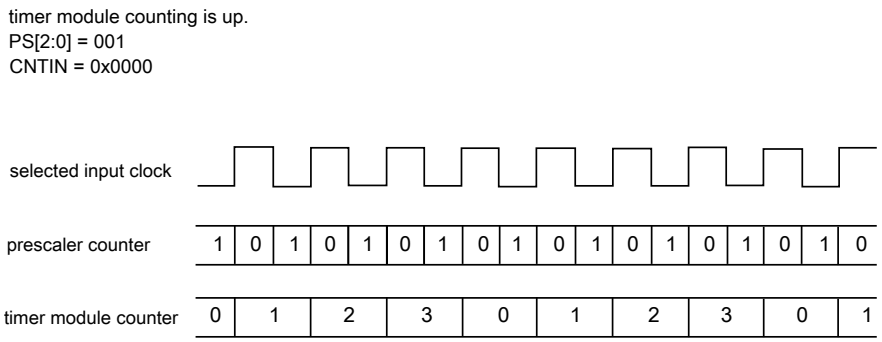


Figure 47-2. Example of the Prescaler Counter

47.4.3 Counter

The TPM has a 16-bit counter that is used by the channels either for input or output modes.

The counter updates from the selected clock divided by the prescaler.

The TPM counter has these modes of operation:

- up counting (see Up counting)
- up-down counting (see Up-down counting)

47.4.3.1 Up counting

Up counting is selected when SC[CPWMS] = 0.

The value of zero is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with zero.

The TPM period when using up counting is (MOD + 0x0001) × period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to zero.

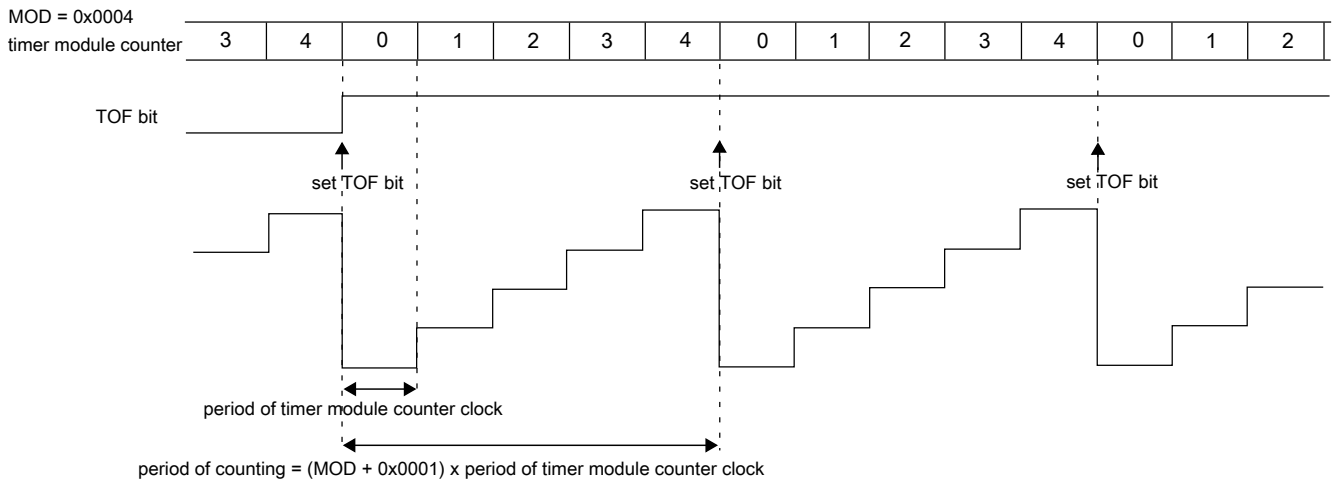


Figure 47-3. Example of TPM Up Counting

Note

- MOD = 0000 is a redundant condition. In this case, the TPM counter is always equal to MOD and the TOF bit is set in each rising edge of the TPM counter clock.

47.4.3.2 Up-down counting

Up-down counting is selected when SC[CPWMS] = 1. When configured for up-down counting, configuring CONF[MOD] to less than 2 is not supported.

The value of 0 is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to zero and the up-down counting restarts.

The TPM period when using up-down counting is 2 × MOD × period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to (MOD – 1).

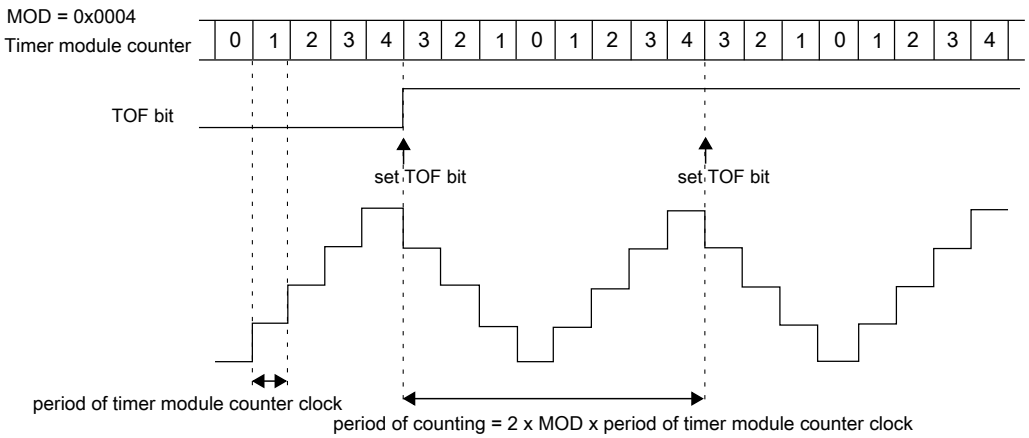


Figure 47-4. Example of up-down counting

47.4.3.3 Counter Reset

Any write to CNT resets the TPM counter and the channel outputs to their initial values (except for channels in output compare mode).

47.4.3.4 Global time base (GTB)

The global time base (GTB) is a TPM function that allows multiple TPM modules to share the same timebase. When the global time base is enabled (CONF[GTBEEN] = 1), the local TPM channels use the counter value, counter enable and overflow indication from the TPM generating the global time base. If the local TPM counter is not generating the global time base, then it can be used as an independent counter or pulse accumulator.

The local TPM counter can also be configured to synchronize to the global time base, by configuring (GTBSYNC = 1). When synchronized to the global time base, the local counter will use the counter enable and counter overflow indication from the TPM generating the global time base. This enables multiple TPM to be configured with the same phase, but with different periods (although the global time base must be configured with the longest period).

47.4.3.5 Counter trigger

The TPM counter can be configured to start, stop or reset in response to a hardware trigger input. The trigger input is synchronized to the asynchronous counter clock, so there is a 3 counter clock delay between the trigger assertion and the counter responding.

- When (CSOT = 1), the counter will not start incrementing until a rising edge is detected on the trigger input.
- When (CSOO= 1), the counter will stop incrementing whenever the TOF flag is set. The counter does not increment again unless it is disabled, or if CSOT = 1 and a rising edge is detected on the trigger input.
- When (CROT= 1), the counter will reset to zero as if an overflow occurred whenever a rising edge is detected on the trigger input.
- When (CPOT = 1), the counter will pause incrementing whenever the trigger input is asserted. The counter will continue incrementing when the trigger input negates.

The polarity of the external input trigger can be configured by the TRGPOL register bit.

When an internal trigger source is selected, the trigger input is selected from one or more channel input capture events. The input capture filters are used with the internal trigger sources and the POLn bits can be used to invert the polarity of the input channels. Note that following restrictions apply with input capture channel sources.

- When (CSOT = 1), the counter will only start incrementing on a rising edge on the channel input, provided ELSnA = 1.
- When (CROT= 1), the counter will reset to zero on either edge of the channel input, as configured by ELSnB:ELSnA.
- When (CPOT = 1), the counter will pause incrementing whenever the channel input is asserted.

47.4.4 Input Capture Mode

The input capture mode is selected when (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

When a selected edge occurs on the channel input, the current value of the TPM counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1 (see the following figure).

When a channel is configured for input capture, the TPM_CHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is counter clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register are ignored in input capture mode.

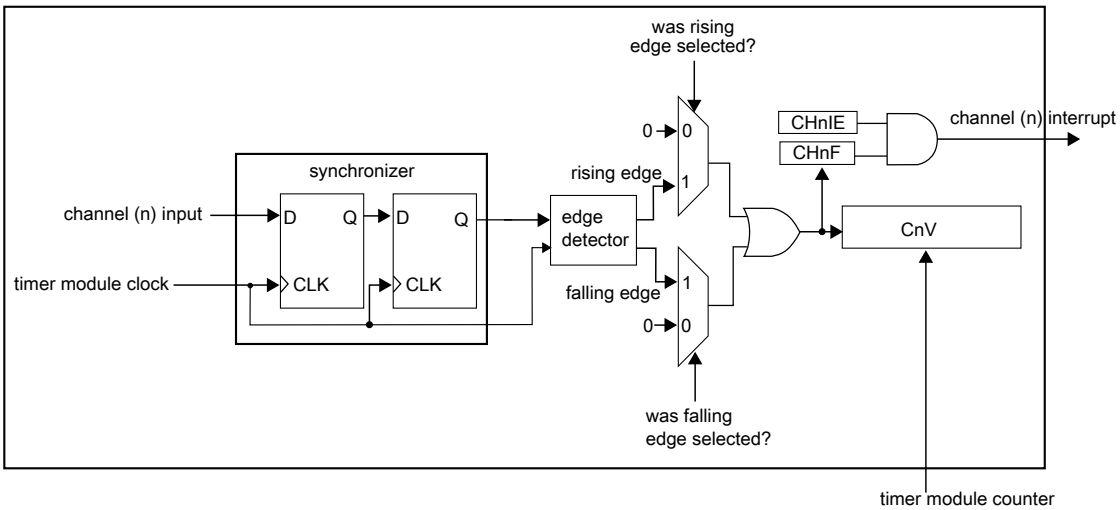


Figure 47-5. Input capture mode

The CHnF bit is set on the third rising edge of the counter clock after a valid edge occurs on the channel input.

47.4.5 Output Compare Mode

The output compare mode is selected when (CPWMS = 0), and (MSnB:MSnA = X:1).

In output compare mode, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared or toggled if MSnB is clear. If MSnB is set then the channel (n) output is pulsed high or low for as long as the counter matches the value in the CnV register.

When a channel is initially configured to output compare mode, the channel output updates with its negated value (logic 0 for set/toggle/pulse high and logic one for clear/ pulse low).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV).

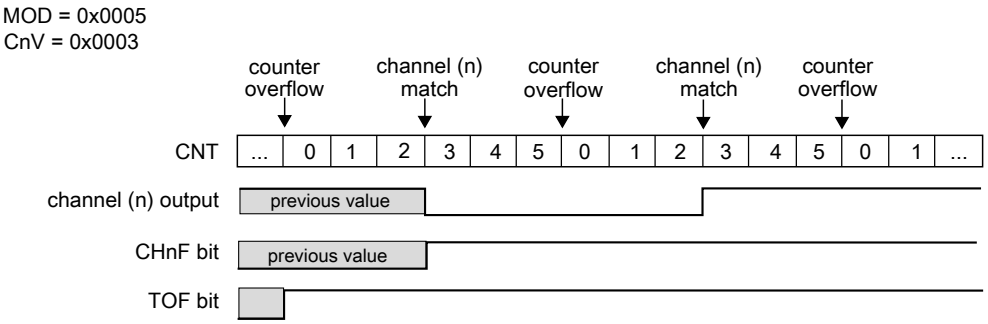


Figure 47-6. Example of the output compare mode when the match toggles the channel output

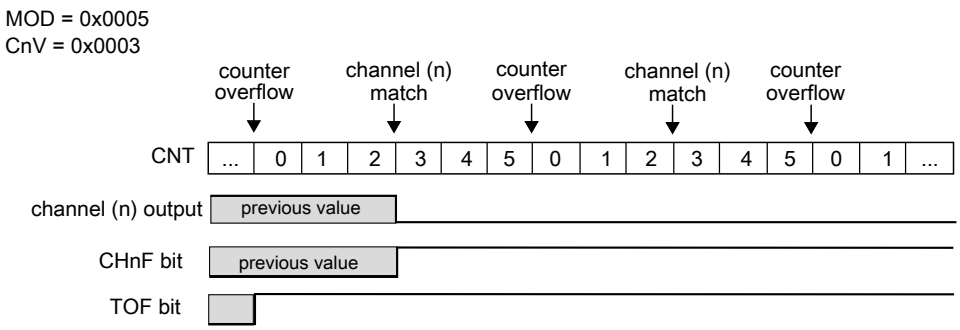


Figure 47-7. Example of the output compare mode when the match clears the channel output

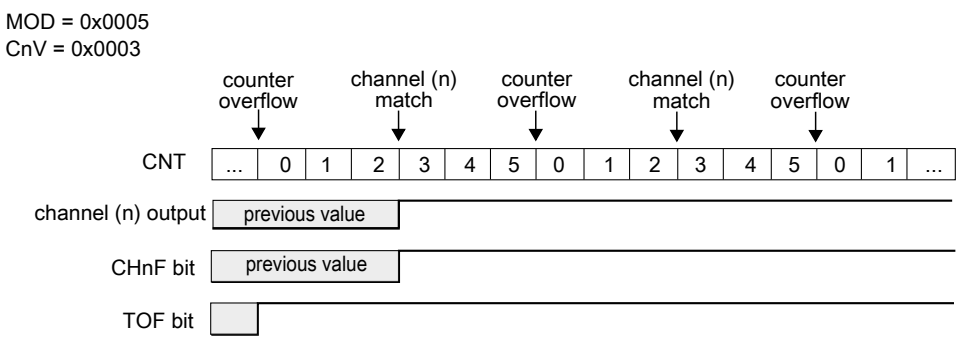


Figure 47-8. Example of the output compare mode when the match sets the channel output

It is possible to use the output compare mode with (ELSnB:ELSnA = 0:0). In this case, when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not modified and controlled by TPM.

47.4.6 Edge-Aligned PWM (EPWM) Mode

The edge-aligned mode is selected when (CPWMS = 0), and (MSnB:MSnA = 1:0).

The EPWM period is determined by (MOD + 0x0001) and the pulse width (duty cycle) is determined by CnV.

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an TPM.

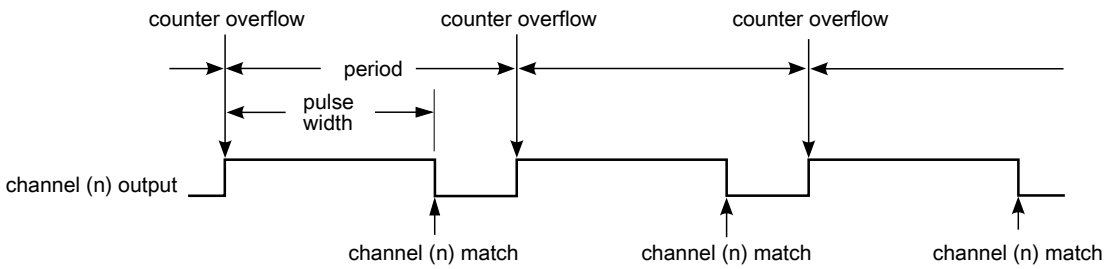


Figure 47-9. EPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not controlled by TPM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow (when the zero is loaded into the TPM counter), and it is forced low at the channel (n) match (TPM counter = CnV) (see the following figure).

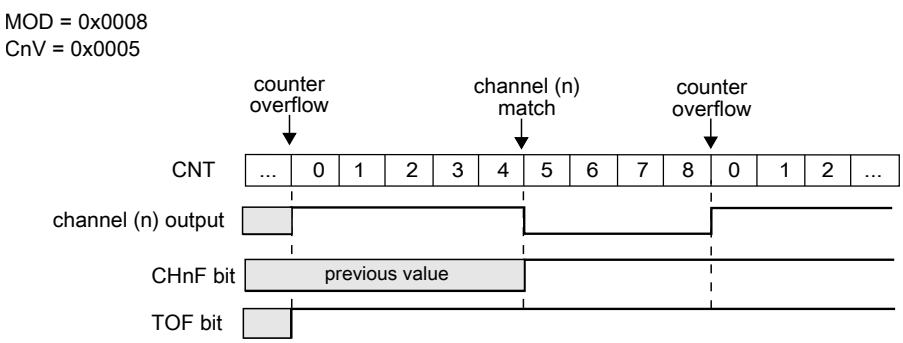


Figure 47-10. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow (when zero is loaded into the TPM counter), and it is forced high at the channel (n) match (TPM counter = CnV) (see the following figure).

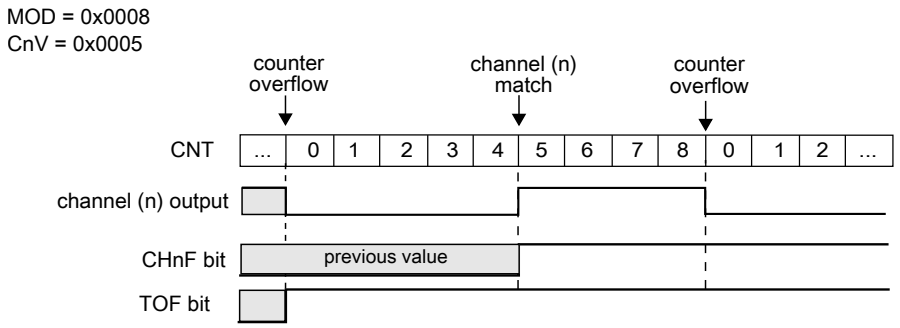


Figure 47-11. EPWM signal with ELSnB:ELSnA = X:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal. If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set since there is never a channel (n) match. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

47.4.7 Center-Aligned PWM (CPWM) Mode

The center-aligned mode is selected when (CPWMS = 1) and (MSnB:MSnA = 1:0).

The CPWM pulse width (duty cycle) is determined by 2 × CnV and the period is determined by 2 × MOD (see the following figure). MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the TPM counter counts up until it reaches MOD and then counts down until it reaches zero.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV) when the TPM counting is down (at the begin of the pulse width) and when the TPM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are when the TPM counter is zero.

The other channel modes are not designed to be used with the up-down counter (CPWMS = 1). Therefore, all TPM channels should be used in CPWM mode when (CPWMS = 1).

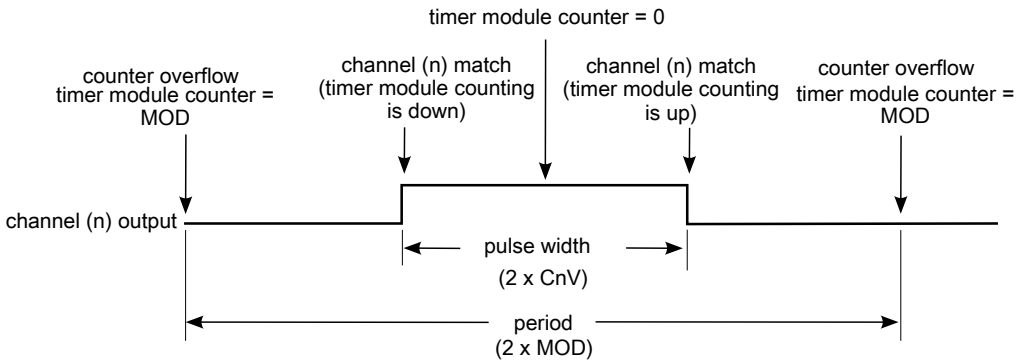


Figure 47-12. CPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the TPM counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not controlled by TPM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (TPM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up (see the following figure).

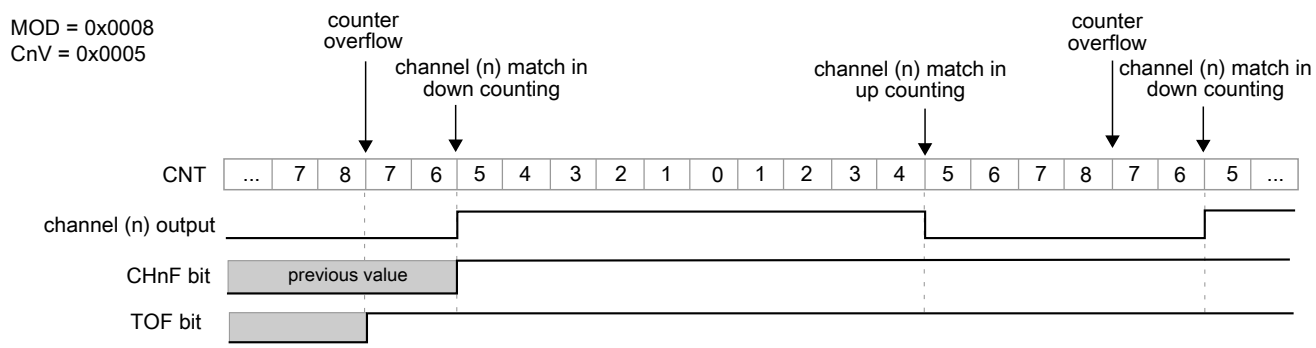


Figure 47-13. CPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the channel (n) match (TPM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up (see the following figure).

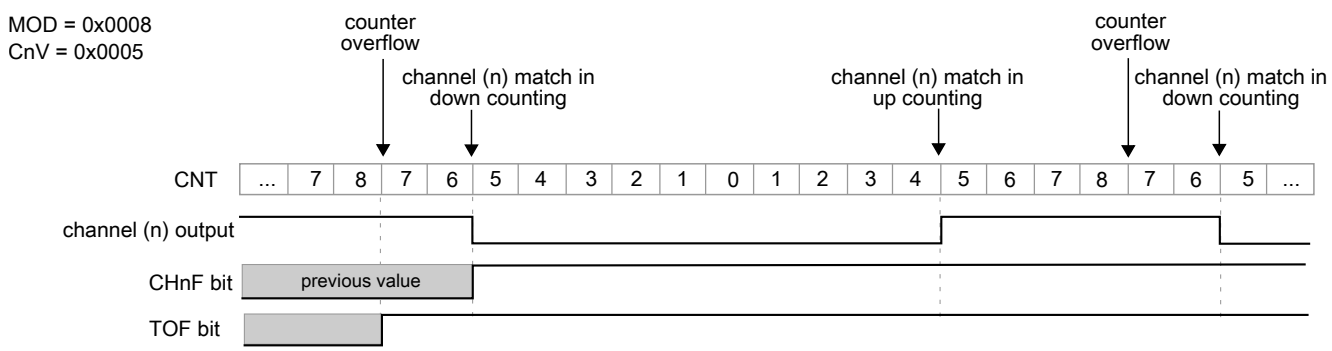


Figure 47-14. CPWM signal with ELSnB:ELSnA = X:1

If (CnV = 0x0000) then the channel (n) output is a 0% duty cycle CPWM signal.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle CPWM signal, although the CHnF bit is set when the counter changes from incrementing to decrementing. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle CPWM signal.

47.4.8 Combine PWM mode

The Combine PWM mode is selected when:

- MSnB:MSnA = 10
- COMBINEn = 1
- QUADEN = 0, and
- CPWMS = 0

In Combine PWM mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

In the Combine mode, the PWM period is determined by (MOD + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = C(n)V). The CH(n+1)F bit is set and the channel (n +1) interrupt is generated, if CH(n+1)IE = 1, at the channel (n+1) match (TPM counter = C(n+1)V).

If channel (n) (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the beginning of the period (TPM counter is zero) and at the channel (n+1) match (TPM counter = C(n+1)V). It is forced high at the channel (n) match (TPM counter = C(n)V).

If channel (n) (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the beginning of the period (TPM counter is zero) and at the channel (n+1) match (TPM counter = C(n+1)V). It is forced low at the channel (n) match (TPM counter = C(n)V).

When (COMSWAPn = 1), then the channel (n) output is forced low or high at the beginning of the period (TPM counter is zero) and at the channel (n) match (TPM counter = C(n)V). It is forced high or low at the channel (n+1) match (TPM counter = C(n+1)V).

The channel (n+1) output is generated the same as the channel (n) output, but the output polarity is controlled by the channel (n+1) ELSnB:ELSnA configuration.

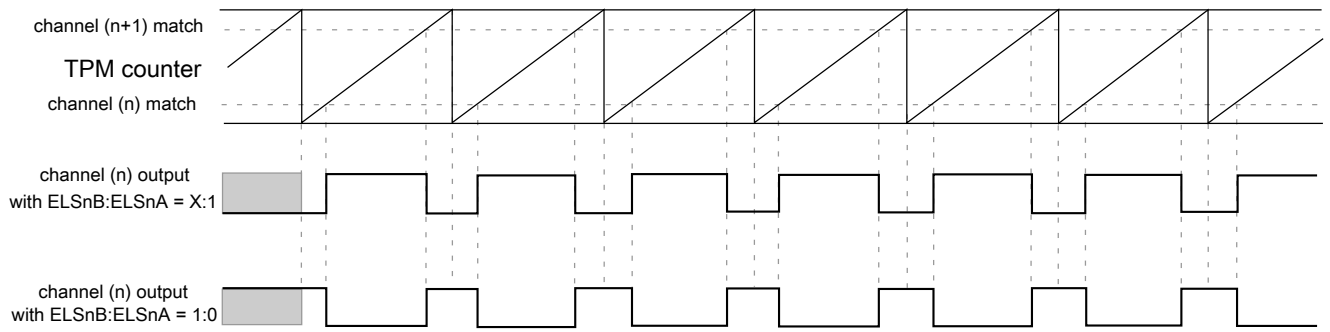


Figure 47-15. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

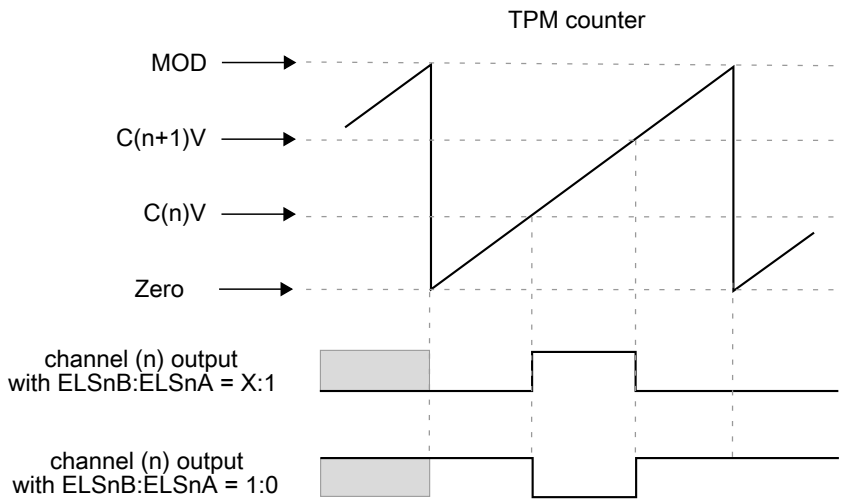


Figure 47-16. Channel (n) output if (C(n)V < MOD) and (C(n+1)V < MOD) and (C(n)V < C(n+1)V)

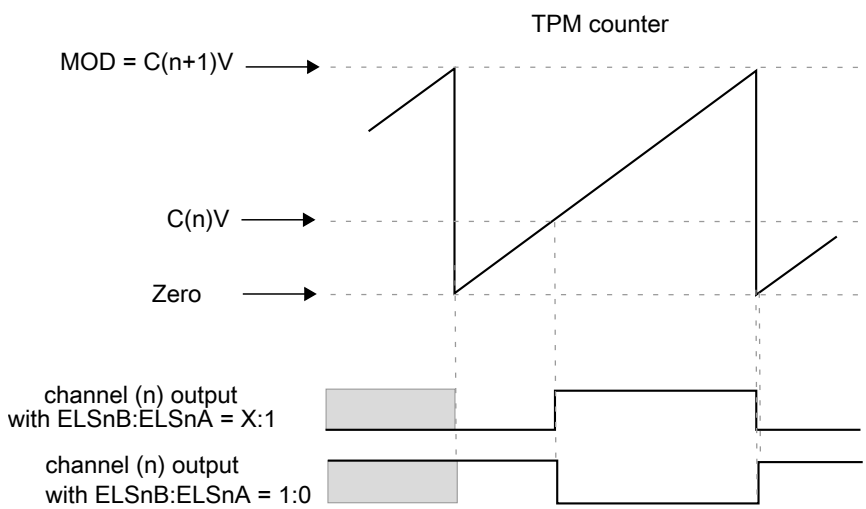


Figure 47-17. Channel (n) output if (C(n)V < MOD) and (C(n+1)V = MOD)

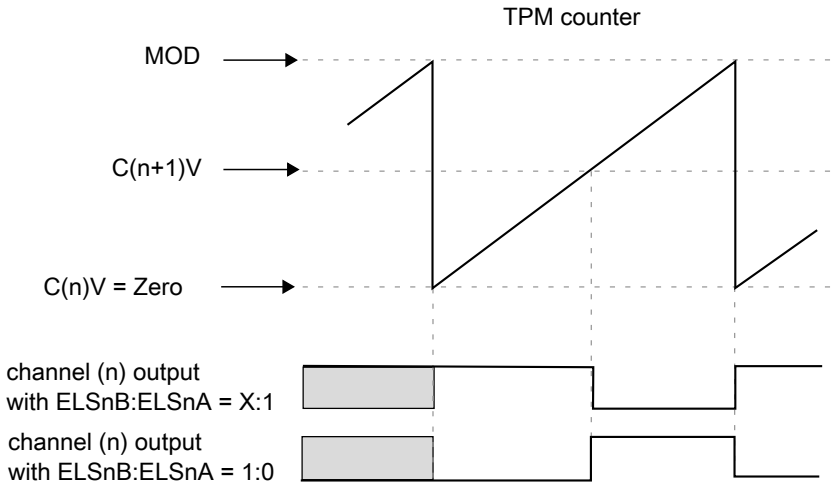


Figure 47-18. Channel (n) output if (C(n)V = zero) and (C(n+1)V < MOD)

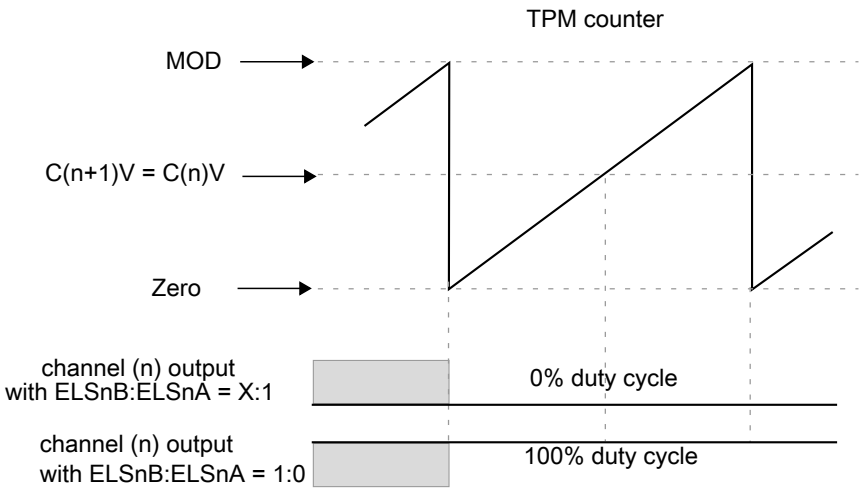


Figure 47-19. Channel (n) output if (C(n)V < MOD) and (C(n+1)V < MOD) and (C(n)V = C(n +1)V)

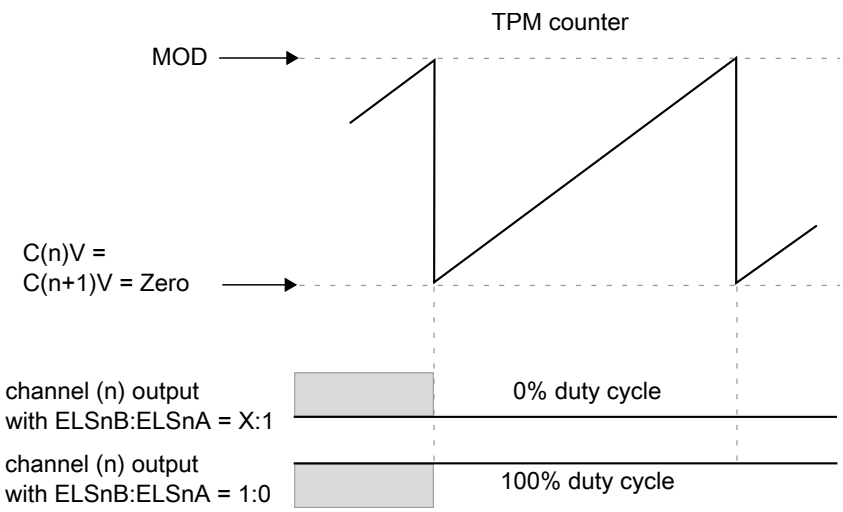


Figure 47-20. Channel (n) output if (C(n)V = C(n+1)V = zero)

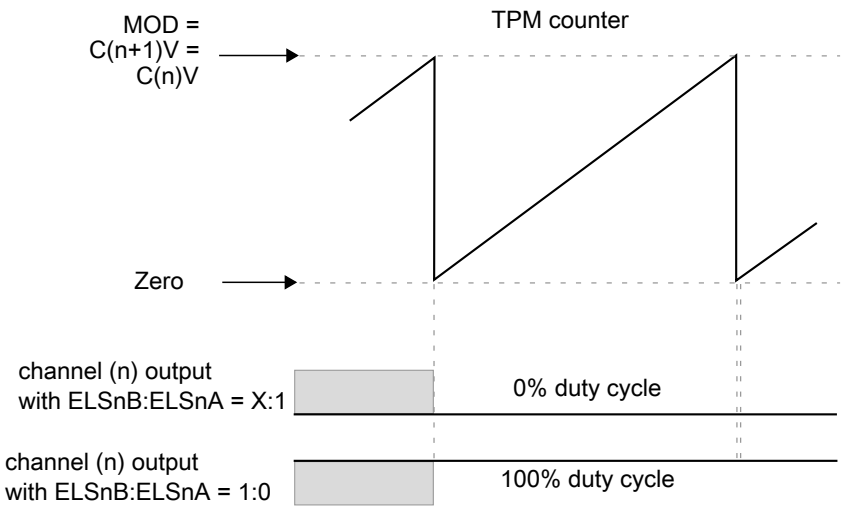


Figure 47-21. Channel (n) output if (C(n)V = C(n+1)V = MOD)

47.4.9 Combine Input Capture mode

The Combine Input Capture mode is selected if COMBINEn= 1 and MSnB:MSnA = 00 and ELSnB:ELSnA != 00. This mode allows to measure a pulse width of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode.

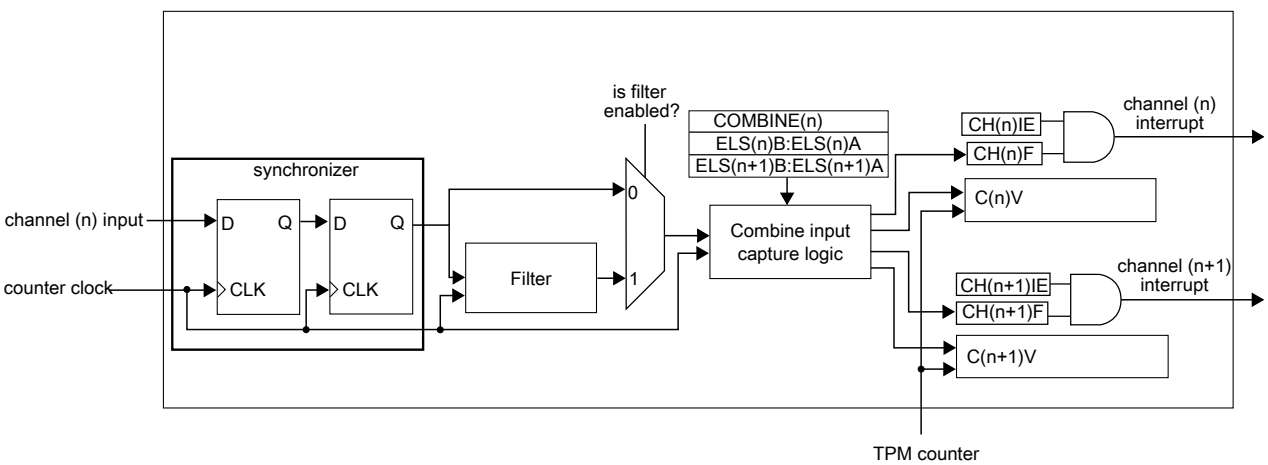


Figure 47-22. Combine Input Capture mode block diagram

The ELS(n)B:ELS(n)A bits select the edge that is captured by channel (n), and ELS(n +1)B:ELS(n+1)A bits select the edge that is captured by channel (n+1).

In the Combine Input Capture mode, only channel (n) input is used and channel (n+1) input is ignored, when COMSWAPn=1 then only channel (n+1) input is used and channel (n) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then CH(n)F bit is set and the channel (n) interrupt is generated (if CH(n)IE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input, then CH(n+1)F bit is set and the channel (n+1) interrupt is generated (if CH(n+1)IE = 1).

The C(n)V register stores the value of TPM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of TPM counter when the selected edge by channel (n+1) is detected at channel (n) input.

Note

- The CH(n)F, CH(n)IE, MS(n)A, ELS(n)B, and ELS(n)A bits are channel (n) bits.
- The CH(n+1)F, CH(n+1)IE, MS(n+1)A, ELS(n+1)B, and ELS(n+1)A bits are channel (n+1) bits.
- The Combine Input Capture mode must be used with ELS(n)B:ELS(n)A = 0:1 or 1:0, ELS(n+1)B:ELS(n+1)A = 0:1 or 1:0.

47.4.10 Input Capture Filter

The input capture filter function is only in input capture mode, or in software compare mode when quadrature decoder mode is enabled.

First, the input signal is synchronized by the counter clock. Following synchronization, the input signal enters the filter block. See the following figure.

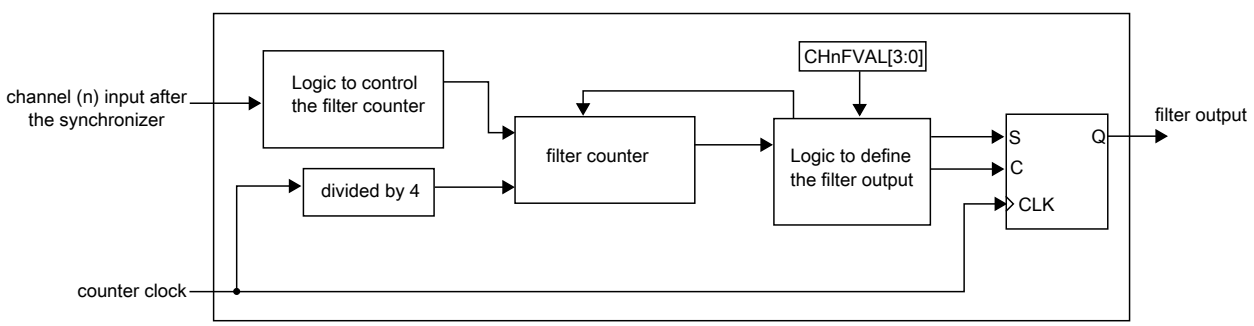


Figure 47-23. Channel input filter

When there is a state change in the input signal, the counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to (CHnFVAL[3:0] × 4), the state change of the input signal is validated.

If the opposite edge appears on the input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by (CHnFVAL[3:0] × 4 counter clocks) is regarded as a glitch and is not passed through the filter. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed by 2 rising edges of the counter clock. If (CHnFVAL[3:0] ≠ 0000), then the input signal is delayed by the minimum pulse width (CHnFVAL[3:0] × 4 system clocks) plus a further 3 rising edges of the system clock: two rising edges to the synchronizer, plus one more to the edge detector. In other words, CHnF is set (3 + 4 × CHnFVAL[3:0]) counter clock periods after a valid edge occurs on the channel input.

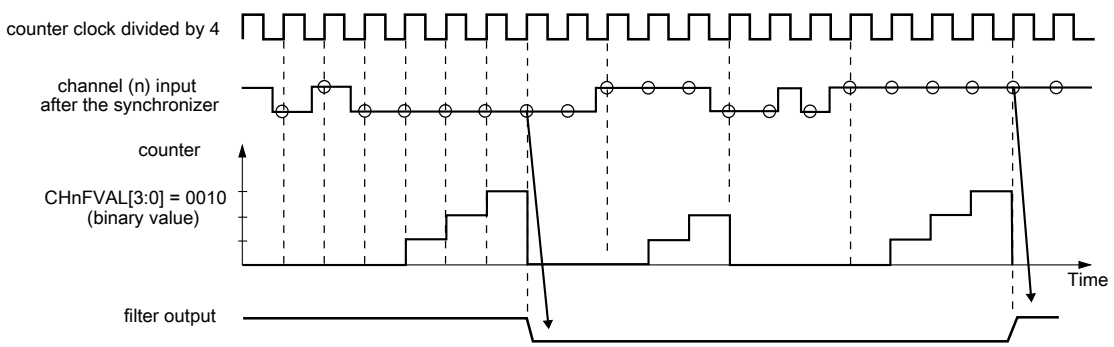


Figure 47-24. Channel input filter example

47.4.11 Deadtime insertion

The deadtime insertion is enabled in PWM combine modes when CHnFVAL is non-zero. The deadtime delay that is used for each TPM channel is defined as (CHnFVAL[3:0] x 4).

The deadtime delay insertion ensures that no two complementary signals (channels (n) and (n+1)) drive the active state at the same time.

If POL(n) = 0, POL(n+1) = 1, and the deadtime is enabled, then when the channel (n) match (TPM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (TPM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If POL(n) = 1, POL(n+1) = 0, and the deadtime is enabled, then when the channel (n) match (TPM counter = C(n)V) occurs, the channel (n) output remains at the high value until the end of the deadtime delay when the channel (n) output is cleared. Similarly,

when the channel (n+1) match (TPM counter = C(n+1)V) occurs, the channel (n+1) output remains at the high value until the end of the deadtime delay when the channel (n +1) output is cleared.

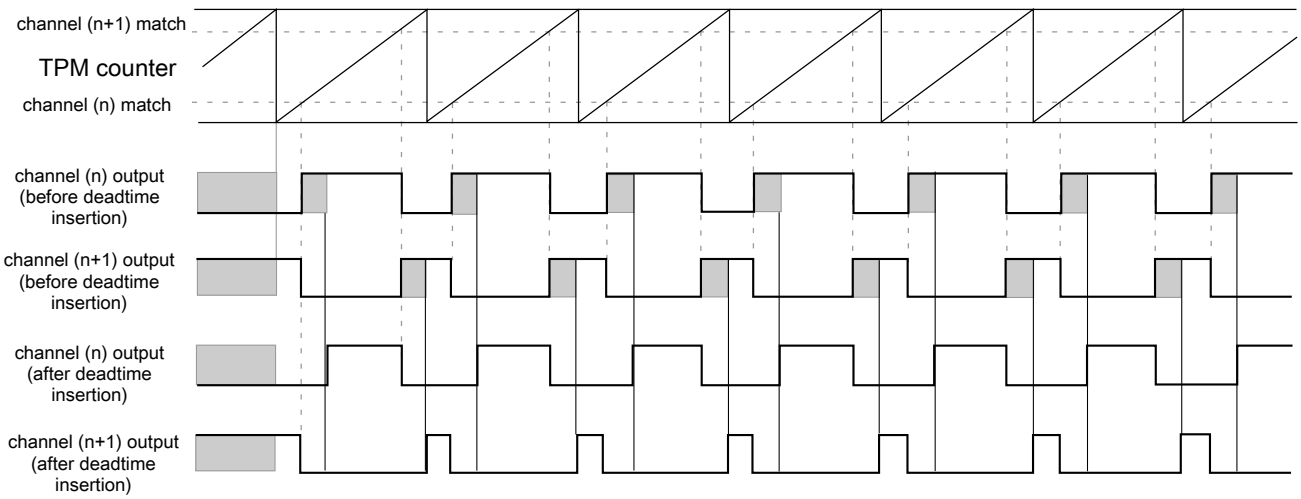


Figure 47-25. Deadtime insertion with ELSnB:ELSnA = X:1, POL(n) = 0, and POL(n+1) = 1

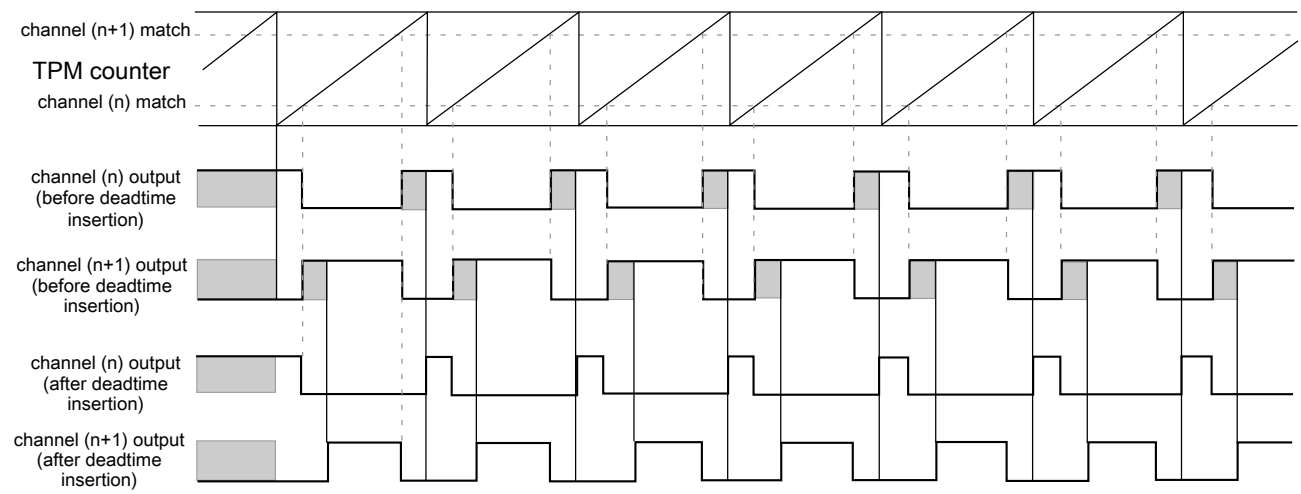


Figure 47-26. Deadtime insertion with ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 1

47.4.12 Quadrature Decoder mode

The Quadrature Decoder mode is selected if (QUADEN = 1). The Quadrature Decoder mode uses the channel 0 (phase A) and channel 1 (phase B) input signals to control the TPM counter increment and decrement. The following figure shows the quadrature decoder block diagram.

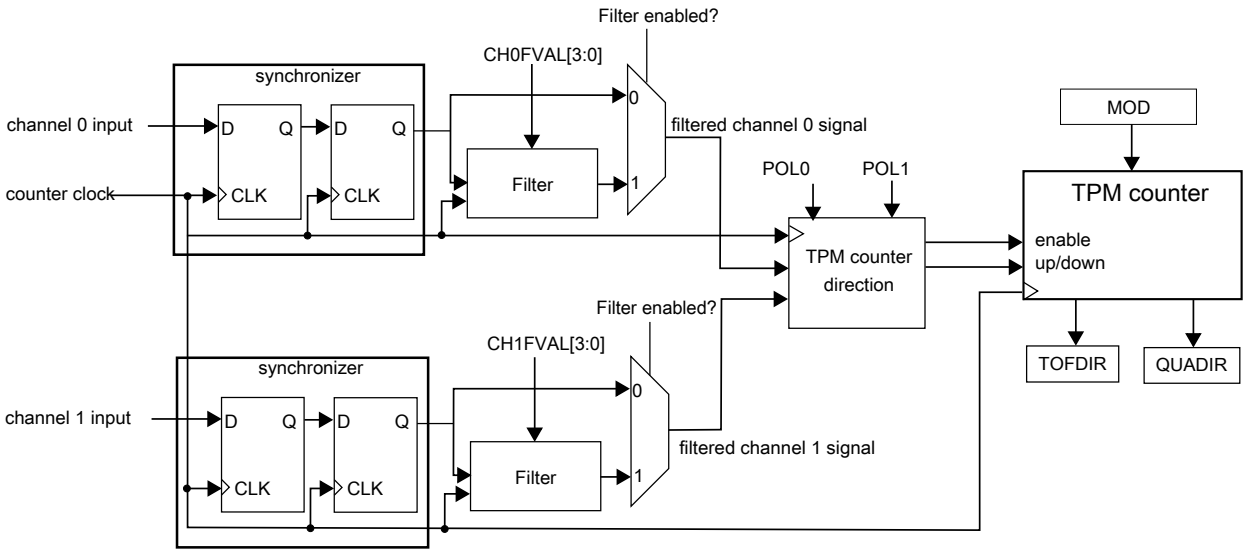


Figure 47-27. Quadrature Decoder block diagram

The input capture filter and channel polarity registers are used to configure the input filter and polarity for the channel 0 and channel 1 inputs in quadrature decode mode.

Note

Notice that the TPM counter is clocked by the channel 0 and channel 1 input signals when quadrature decoder mode is selected. Therefore In quadrature decoder mode, channel 0 and channel 1 can only be used in software compare mode and other TPM channels can only be used in input capture or output compare modes.

The QUADMODE selects the encoding mode used in the Quadrature Decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the channel 1 input value indicates the counting direction, and the channel 0 input defines the counting rate. The TPM counter is updated when there is a rising edge at channel 0 input signal.

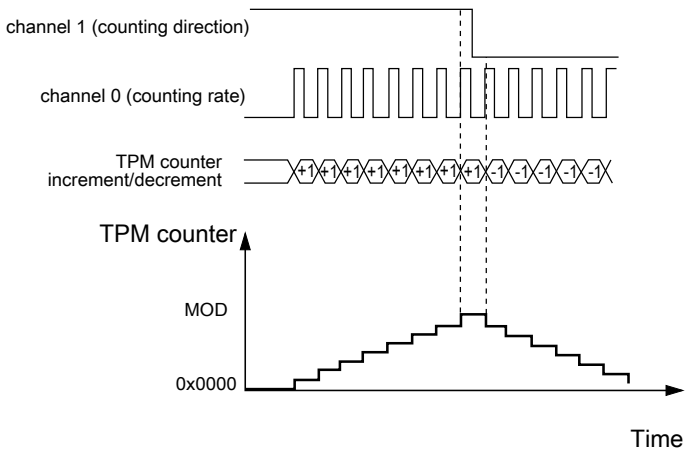


Figure 47-28. Quadrature Decoder – Count and Direction Encoding mode

If QUADMODE = 0, then the Phase Encoding mode is enabled; see the following figure. In this mode, the relationship between channel 0 and channel 1 signals indicates the counting direction, and channel 0 and channel 1 signals define the counting rate. The TPM counter is updated when there is an edge either at the channel 0 or channel 1 signals.

If CH0POL= 0 and CH1POL = 0, then the TPM counter increment happens when:

- there is a rising edge at channel 0 signal and channel 1 signal is at logic zero;
- there is a rising edge at channel 1 signal and channel 0 signal is at logic one;
- there is a falling edge at channel 1 signal and channel 0 signal is at logic zero;
- there is a falling edge at channel 0 signal and channel 1 signal is at logic one;

and the TPM counter decrement happens when:

- there is a falling edge at channel 0 signal and channel 1 signal is at logic zero;
- there is a falling edge at channel 1 signal and channel 0 signal is at logic one;
- there is a rising edge at channel 1 signal and channel 0 signal is at logic zero;
- there is a rising edge at channel 0 signal and channel 1 signal is at logic one.

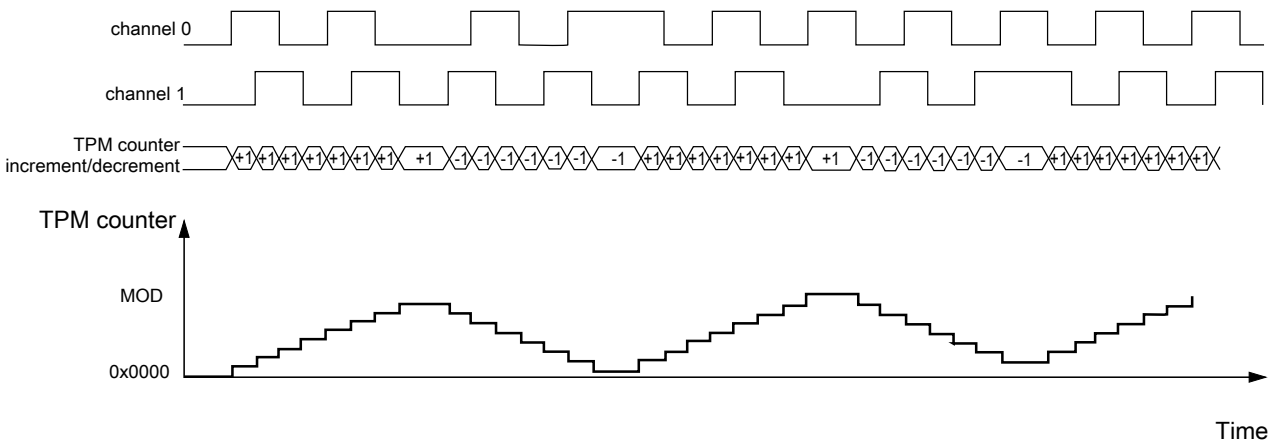


Figure 47-29. Quadrature Decoder – Phase Encoding mode

The following figure shows the TPM counter overflow in up counting. In this case, when the TPM counter changes from MOD to zero, TOF and TOFDIR bits are set. TOF bit indicates the TPM counter overflow occurred. TOFDIR indicates the counting was up when the TPM counter overflow occurred.

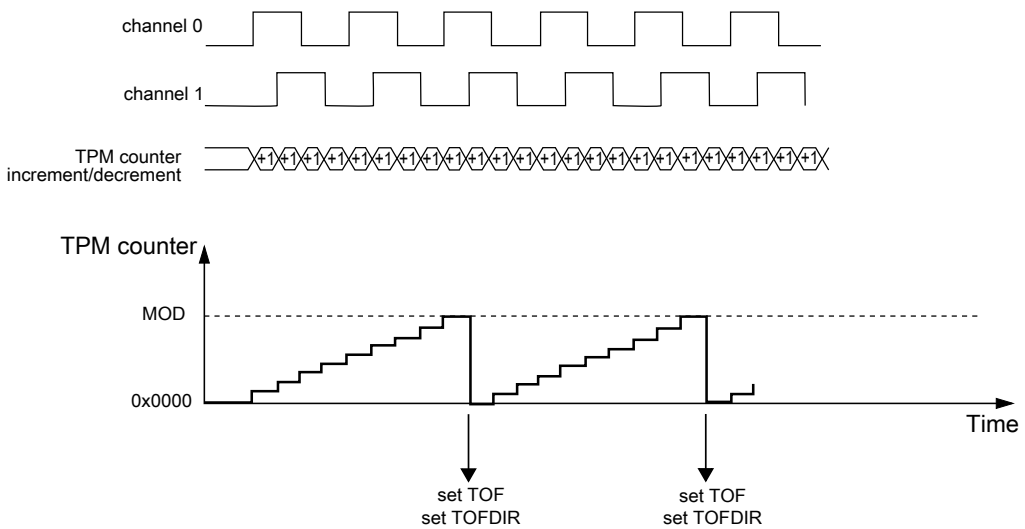


Figure 47-30. TPM Counter overflow in up counting for Quadrature Decoder mode

The following figure shows the TPM counter overflow in down counting. In this case, when the TPM counter changes from zero to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the TPM counter overflow occurred. TOFDIR indicates the counting was down when the TPM counter overflow occurred.

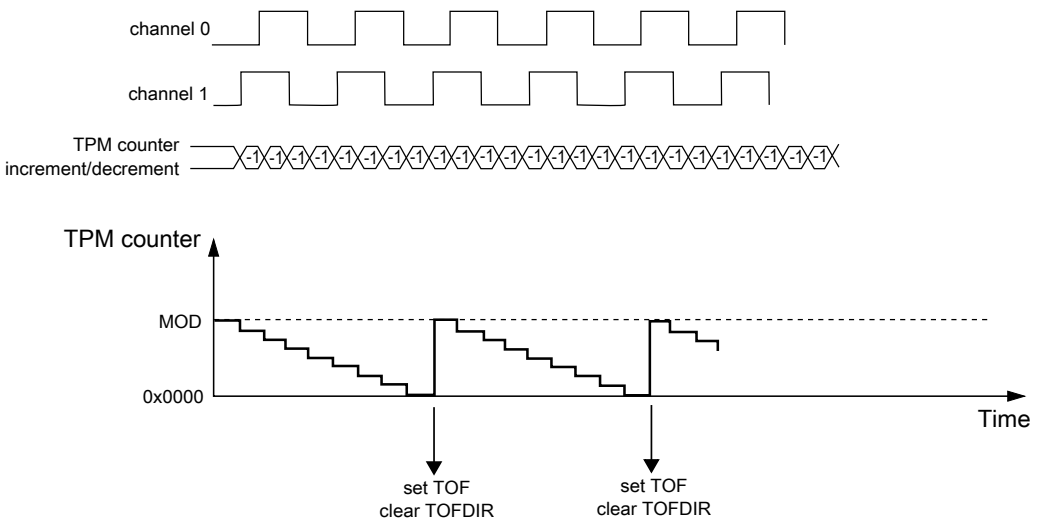


Figure 47-31. TPM counter overflow in down counting for Quadrature Decoder mode

47.4.13 Registers Updated from Write Buffers

47.4.13.1 MOD Register Update

If (CMOD[1:0] = 0:0) then MOD register is updated when MOD register is written.

If (CMOD[1:0] ≠ 0:0), then MOD register is updated according to the CPWMS bit, that is:

- If the selected mode is not CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to (MOD – 1).

47.4.13.2 CnV Register Update

If (CMOD[1:0] = 0:0) then CnV register is updated when CnV register is written.

If (CMOD[1:0] ≠ 0:0), then CnV register is updated according to the selected mode, that is:

- If the selected mode is output compare then CnV register is updated on the next TPM counter increment (end of the prescaler counting) after CnV register was written.

- If the selected mode is EPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to (MOD – 1).

47.4.14 DMA

The channel and overflow flags generate a DMA transfer request according to DMA and CHnIE/TOIE bits.

See the following table for more information.

Table 47-3. DMA Transfer Request

DMA	CHnIE/ TOIE	Channel/Overflow DMA Transfer Request	Channel/Overflow Interrupt
0	0	The channel/overflow DMA transfer request is not generated.	The channel/overflow interrupt is not generated.
0	1	The channel/overflow DMA transfer request is not generated.	The channel/overflow interrupt is generated if (CHnF/TOF = 1).
1	0	The channel/overflow DMA transfer request is generated if (CHnF/TOF = 1).	The channel/overflow interrupt is not generated.
1	1	The channel/overflow DMA transfer request is generated if (CHnF/TOF = 1).	The channel/overflow interrupt is generated if (CHnF/TOF = 1).

If DMA = 1, the CHnF/TOF bit can be cleared either by DMA transfer done or writing a one to CHnF/TOF bit (see the following table).

Table 47-4. Clear CHnF/TOF Bit

DMA	How CHnF/TOF Bit Can Be Cleared
0	CHnF/TOF bit is cleared by writing a 1 to CHnF/TOF bit.
1	CHnF/TOF bit is cleared either when the DMA transfer is done or by writing a 1 to CHnF/TOF bit.

47.4.15 Output triggers

The TPM generates output triggers for the counter and each channel that can be used to trigger events in other peripherals.

The counter trigger asserts whenever the TOF is set and remains asserted until the next increment.

Each TPM channel generates both a pre-trigger output and a trigger output. The pre-trigger output asserts whenever the CHnF is set, the trigger output asserts on the first counter increment after the pre-trigger asserts, and then both the trigger and pre-trigger negate on the first counter increment after the trigger asserts.

When (COMBINEn = 1) in output compare modes, the pre-trigger output for both channel (n) and channel (n+1) will assert when CH(n)F is set and will negate when CH(n +1)F is set. The trigger continues to assert on the first counter increment after the pre-trigger asserts and negates at the same time as the pre-trigger negation.

47.4.16 Reset Overview

The TPM is reset whenever any chip reset occurs.

When the TPM exits from reset:

- the TPM counter and the prescaler counter are zero and are stopped (CMOD[1:0] = 0:0);
- the timer overflow interrupt is zero;
- the channels interrupts are zero;
- the channels are in input capture mode;
- the channels outputs are zero;
- the channels pins are not controlled by TPM (ELS(n)B:ELS(n)A = 0:0).

47.4.17 TPM Interrupts

This section describes TPM interrupts.

47.4.17.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

47.4.17.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

Chapter 48
Trigger MUX Control (TRGMUX)

48.1 Introduction

The Trigger MUX module (TRGMUX) allows software to configure the trigger inputs for various peripherals.

48.1.1 Features

The Trigger MUX module allows software to configure the trigger inputs for various peripherals.

- Trigger MUX select

48.1.1.1 Select Bit Fields

Field	Function
5-0 SEL	This read/write bit field is used to configure the MUX select for the peripheral trigger inputs. 000000 - (0x00) Trigger function is disabled. 000001 - (0x01) Port pin trigger input is selected. 000010 - (0x02) FlexIO Timer 0 input is selected. 000011 - (0x03) FlexIO Timer 1 input is selected. 000100 - (0x04) FlexIO Timer 2 input is selected. 000101 - (0x05) FlexIO Timer 3 input is selected. 000110 - (0x06) FlexIO Timer 4 input is selected. 000111 - (0x07) FlexIO Timer 5 input is selected. 001000 - (0x08) FlexIO Timer 6 input is selected. 001001 - (0x09) FlexIO Timer 7 input is selected. 001010 - (0x0A) TPM0 Overflow is selected. 001011 - (0x0B) TPM0 Channel 0 is selected.

Field	Function
	001100 - (0x0C) TPM0 Channel 1 is selected. 001101 - (0x0D) TPM1 Overflow is selected. 001110 - (0x0E) TPM1 Channel 0 is selected. 001111 - (0x0F) TPM1 Channel 1 is selected. 010000 - (0x10) Reserved 010001 - (0x11) Reserved 010010 - (0x12) Reserved 010011 - (0x13) Reserved 010100 - (0x14) LPUART0 RX Data is selected. 010101 - (0x15) LPUART0 TX Data is selected. 010110 - (0x16) LPUART0 RX Idle is selected. 010111 - (0x17) LPUART1 RX Data is selected. 011000 - (0x18) LPUART1 TX Data is selected. 011001 - (0x19) LPUART1 RX Idle is selected. 011010 - (0x1A) LPI2C0 Master STOP is selected. 011011 - (0x1B) LPI2C0 Slave STOP is selected. 011100 - (0x1C) LPI2C1 Master STOP is selected. 011101 - (0x1D) LPI2C1 Slave STOP is selected. 011110 - (0x1E) LPSPI0 Frame is selected. 011111 - (0x1F) LPSPI0 RX data is selected. 100000 - (0x20) LPSPI1 Frame is selected. 100001 - (0x21) LPSPI1 RX data is selected. 100010 - (0x22) RTC Seconds Counter is selected. 100011 - (0x23) RTC Alarm is selected. 100100 - (0x24) LPTMR0 Trigger is selected. 100101 - (0x25) LPTMR1 Trigger is selected. 100110 - (0x26) CMP0 Output is selected. 100111 - (0x27) CMP1 Output is selected. 101000 - (0x28) ADC0 Conversion A Complete is selected. 101001 - (0x29) ADC0 Conversion B Complete is selected. 101010 - (0x2A) Port A Pin Trigger is selected. 101011 - (0x2B) Port B Pin Trigger is selected. 101100 - (0x2C) Port C Pin Trigger is selected. 101101 - (0x2D) Port D Pin Trigger is selected. 101110 - (0x2E) Port E Pin Trigger is selected. 101111 - (0x2F) TPM2 Overflow is selected. 110000 - (0x30) TPM2 Channel 0 is selected. 110001 - (0x31) TPM2 Channel 1 is selected. 110010 - (0x32) LPIT0 Channel 0 is selected.

Field	Function
	110011 - (0x33) LPIT0 Channel 1 is selected. 110100 - (0x34) LPIT0 Channel 2 is selected. 110101 - (0x35) LPIT0 Channel 3 is selected. 110110 - (0x36) USB Start-of-Frame is selected. 110111 - (0x37) LPUART2 RX Data is selected. 111000 - (0x38) LPUART2 TX Data is selected. 111001 - (0x39) LPUART2 RX Idle is selected. 111010 - (0x3A) LPI2C2 Master STOP is selected. 111011 - (0x3B) LPI2C2 Slave STOP is selected. 111100 - (0x3C) LPSPI2 Frame is selected. 111101 - (0x3D) LPSPI2 RX Data is selected. 111110 - (0x3E) SAI TX Frame Sync is selected. 111111 - (0x3F) SAI RX Frame Sync is selected.

48.2 Memory map and register definition

The TRGMUX module contains register fields for selecting the trigger input for peripheral modules.

48.2.1 TRGMUX Register Descriptions

These register may not be applicable to all instances of TRGMUX. For more details on the registers supported on each module instance, please refer to "The TRGMUX as implemented on the chip."

Table 48-1. TRGMUX Memory Map

Offset	Register	Width (In bits)	Access	Reset value
40027000h	TRGMUX DMAMUX0 (TRGMUX_DMAMUX0)	32	RW	00000000h
40027004h	TRGMUX LPIT0 (TRGMUX_LPIT0)	32	RW	00000000h
40027008h	TRGMUX TPM2 (TRGMUX_TPM2)	32	RW	00000000h
40027010h	TRGMUX ADC0 (TRGMUX_ADC0)	32	RW	00000000h
40027014h	TRGMUX LPUART2 (TRGMUX_LPUART2)	32	RW	00000000h
4002701Ch	TRGMUX LPI2C2 (TRGMUX_LPI2C2)	32	RW	00000000h
40027024h	TRGMUX LPSPI2 (TRGMUX_LPSPI2)	32	RW	00000000h
4002702Ch	TRGMUX CMP0 (TRGMUX_CMP0)	32	RW	00000000h

Table continues on the next page...

Table 48-1. TRGMUX Memory Map (continued)

Offset	Register	Width (In bits)	Access	Reset value
40027030h	TRGMUX CMP1 (TRGMUX_CMP1)	32	RW	00000000h
40027034h	TRGMUX DAC0 (TRGMUX_DAC0)	32	RW	00000000h

48.2.1.1 TRGMUX Register Descriptions

48.2.1.2 TRGMUX DMAMUX0 (TRGMUX_DMAMUX0)

48.2.1.2.1 Address

Register	Offset
TRGMUX_DMAMUX0	40027000h

48.2.1.2.2 Function

TRGMUX Register

48.2.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	SEL3						Reserved		SEL2					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.2.4 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written.

Table continues on the next page...

Field	Function
	1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 SEL3	This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.3 TRGMUX LPIT0 (TRGMUX_LPIT0)

48.2.1.3.1 Address

Register	Offset
TRGMUX_LPIT0	40027004h

TRGMUX Register

48.2.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	SEL3						Reserved		SEL2					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.3.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 SEL3	This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.4 TRGMUX TPM2 (TRGMUX_TPM2)

48.2.1.4.1 Address

Register	Offset
TRGMUX_TPM2	40027008h

TRGMUX Register

48.2.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		SEL2					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.4.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.5 TRGMUX ADC0 (TRGMUX_ADC0)

48.2.1.5.1 Address

Register	Offset
TRGMUX_ADC0	40027010h

TRGMUX Register

48.2.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.5.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.6 TRGMUX LPUART2 (TRGMUX_LPUART2)

48.2.1.6.1 Address

Register	Offset
TRGMUX_LPUART2	40027014h

TRGMUX Register

48.2.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.6.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.7 TRGMUX LPI2C2 (TRGMUX_LPI2C2)

48.2.1.7.1 Address

Register	Offset
TRGMUX_LPI2C2	4002701Ch

TRGMUX Register

48.2.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.7.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.8 TRGMUX LPSPI2 (TRGMUX_LPSPI2)

48.2.1.8.1 Address

Register	Offset
TRGMUX_LPSPI2	40027024h

TRGMUX Register

48.2.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.8.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.9 TRGMUX CMP0 (TRGMUX_CMP0)

48.2.1.9.1 Address

Register	Offset
TRGMUX_CMP0	4002702Ch

TRGMUX Register

48.2.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.9.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.10 TRGMUX CMP1 (TRGMUX_CMP1)

48.2.1.10.1 Address

Register	Offset
TRGMUX_CMP1	40027030h

TRGMUX Register

48.2.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.10.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.1.11 TRGMUX DAC0 (TRGMUX_DAC0)

48.2.1.11.1 Address

Register	Offset
TRGMUX_DAC0	40027034h

TRGMUX Register

48.2.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.1.11.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2 TRGMUX Register Descriptions

These register may not be applicable to all instances of TRGMUX. For more details on the registers supported on each module instance, please refer to "The TRGMUX as implemented on the chip."

Table 48-2. TRGMUX Memory Map

Offset	Register	Width (In bits)	Access	Reset value
400A7008h	TRGMUX TPM0 (TRGMUX_TPM0)	32	RW	00000000h
400A700Ch	TRGMUX TPM1 (TRGMUX_TPM1)	32	RW	00000000h
400A7010h	TRGMUX FLEXIO0 (TRGMUX_FLEXIO0)	32	RW	00000000h
400A7014h	TRGMUX LPUART0 (TRGMUX_LPUART0)	32	RW	00000000h
400A7018h	TRGMUX LPUART1 (TRGMUX_LPUART1)	32	RW	00000000h
400A701Ch	TRGMUX LPI2C0 (TRGMUX_LPI2C0)	32	RW	00000000h
400A7020h	TRGMUX LPI2C1 (TRGMUX_LPI2C1)	32	RW	00000000h

Table continues on the next page...

Table 48-2. TRGMUX Memory Map (continued)

Offset	Register	Width (In bits)	Access	Reset value
400A7024h	TRGMUX LPSPI0 (TRGMUX_LPSPI0)	32	RW	00000000h
400A7028h	TRGMUX LPSPI1 (TRGMUX_LPSPI1)	32	RW	00000000h

48.2.2.1 TRGMUX Register Descriptions

48.2.2.2 TRGMUX TPM0 (TRGMUX_TPM0)

48.2.2.2.1 Address

Register	Offset
TRGMUX_TPM0	400A7008h

48.2.2.2.2 Function

TRGMUX Register

48.2.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		SEL2					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.2.4 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written.

Table continues on the next page...

Field	Function
	1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.3 TRGMUX TPM1 (TRGMUX_TPM1)

48.2.2.3.1 Address

Register	Offset
TRGMUX_TPM1	400A700Ch

TRGMUX Register

48.2.2.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		SEL2					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.3.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.4 TRGMUX FLEXIO0 (TRGMUX_FLEXIO0)

48.2.2.4.1 Address

Register	Offset
TRGMUX_FLEXIO0	400A7010h

TRGMUX Register

48.2.2.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	SEL3						Reserved		SEL2					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		SEL1						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.4.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 SEL3	This read/write bit field is used to configure the MUX select for peripheral trigger input 3. Refer to the Select Bit Fields table in the Features section for bit field information.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	This read/write bit field is used to configure the MUX select for peripheral trigger input 2. Refer to the Select Bit Fields table in the Features section for bit field information.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. Refer to the Select Bit Fields table in the Features section for bit field information.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.5 TRGMUX LPUART0 (TRGMUX_LPUART0)

48.2.2.5.1 Address

Register	Offset
TRGMUX_LPUART0	400A7014h

TRGMUX Register

48.2.2.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.5.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.6 TRGMUX LPUART1 (TRGMUX_LPUART1)

48.2.2.6.1 Address

Register	Offset
TRGMUX_LPUART1	400A7018h

TRGMUX Register

48.2.2.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.6.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.7 TRGMUX LPI2C0 (TRGMUX_LPI2C0)

48.2.2.7.1 Address

Register	Offset
TRGMUX_LPI2C0	400A701Ch

TRGMUX Register

48.2.2.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.7.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.8 TRGMUX LPI2C1 (TRGMUX_LPI2C1)

48.2.2.8.1 Address

Register	Offset
TRGMUX_LPI2C1	400A7020h

TRGMUX Register

48.2.2.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.8.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.9 TRGMUX LPSPI0 (TRGMUX_LPSPI0)

48.2.2.9.1 Address

Register	Offset
TRGMUX_LPSPI0	400A7024h

TRGMUX Register

48.2.2.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.9.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

48.2.2.10 TRGMUX LPSPI1 (TRGMUX_LPSPI1)

48.2.2.10.1 Address

Register	Offset
TRGMUX_LPSPI1	400A7028h

TRGMUX Register

48.2.2.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LK	Reserved	Reserved						Reserved		Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		Reserved						Reserved		SEL0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

48.2.2.10.3 Fields

Field	Function
31 LK	This bit shows whether the register can be written or not. 0 - Register can be written. 1 - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 —	This read-only bit field is reserved and always has the value 0.
23-22	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
21-16 —	This read-only bit field is reserved and always has the value 0.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	This read/write bit field is used to configure the MUX select for peripheral trigger input 0. Refer to the Select Bit Fields table in the Features section for bit field information.

Chapter 49
SA-TRNG Standalone

49.1 Standalone True Random Number Generator (SA-TRNG).

The Standalone True Random Number Generator (SA-TRNG) is a hardware accelerator module that generates a 512-bit entropy as needed by an entropy-consuming module or by other post-processing functions. A typical entropy consumer is a pseudo random-number generator (PRNG) that can be implemented to achieve both true randomness and cryptographic-strength random numbers using the TRNG output as its entropy seed. The PRNG is not part of this module.

The entropy generated by a TRNG is intended for direct use by functions that generate secret keys, per-message secrets, random challenges, and other similar quantities used in cryptographic algorithms. In each of these cases, it is important that a random number be difficult to guess or predict. It is important that a random number is at least as difficult to predict as it is difficult to break the cryptographic algorithm with which it is being used. This stringent requirement is particularly difficult to fulfill if the entropy source from a TRNG contains bias and/or correlation. To increase the trustworthiness/quality of the generated random data, PRNGs are often used to post process the output of a TRNG.

This document describes only the TRNG design functionality and usage.

Note that before entropy can be obtained from the TRNG, it must be initialized and instantiated in a particular mode by setting the appropriate TRNG registers.

The TRNG contains the following sub modules: IP Slave bus (SkyBlue bus) interface, the TRNG Core and the free running oscillator (OSC).

49.1.1 Standalone True Random Number Generator Block Diagram

The following figure is a top-level diagram of the True Random Number Generator.

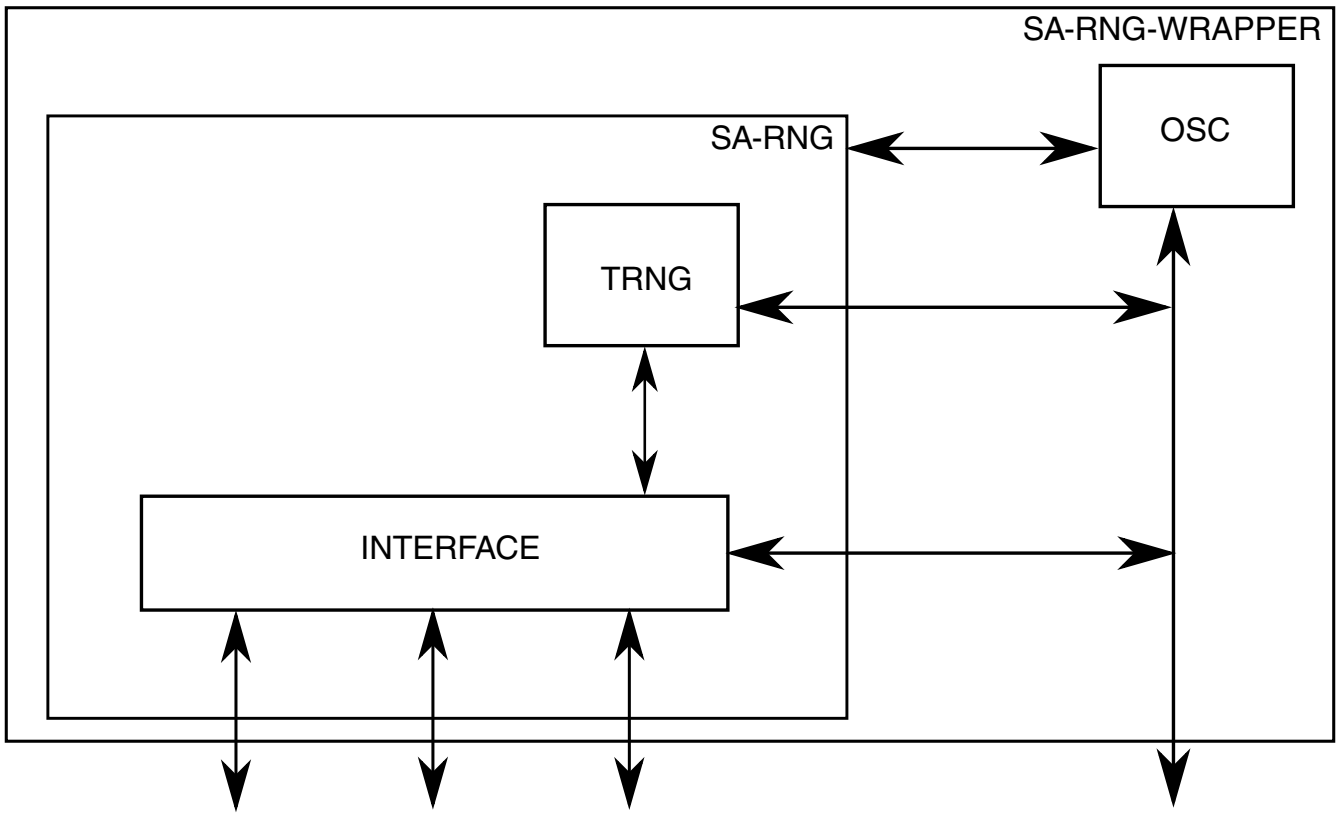


Figure 49-1. SA TRNG Block Diagram

49.1.2 TRNG Functional Description.

The TRNG consists of several functional sub-modules. Its overall functionality can be easily described from the top level in terms of generating entropy for seed generation. The functionality of each sub-module is briefly described in the following subsections.

TRNG is based on collecting bits from a random noise source. This random noise source is a ring oscillator that is sensitive to random noise (temperature variations, voltage variations, cross-talk and other random noise) within the device in which the TRNG is used. This noise causes various small changes in the period of the oscillator. Therefore, if the count of the ring oscillator clock cycles is sampled after a known period of time, this count will vary each time the sample is taken. By using the variance in this count over a large number of samples, random bits can be derived.

The TRNG comprises two entropy sources, each of which provides a single bit of output. Concatenated together, these 2 output bits are expected to provide 1 bit of entropy every 100 clock cycles. In addition to generating entropy, the TRNG also performs several statistical tests on its output.

49.1.3 SA-TRNG hardware functional description.

SA-TRNG functionality consists of several major subcomponents. This table describes these subcomponents.

Table 49-1. SA-TRNG subcomponents.

Description	Cross-reference(s)
Interfaces	
Register interface • Used for access to configuration, control, status and debugging registers	Register interface (IP Slave bus)
True Random Number Generator (TRNG)	Standalone True Random Number Generator (SA-TRNG).

49.1.3.1 Software Use Cases for the Stand Alone TRNG.

There are four things that a user (programmer/integrator) will want to do with a TRNG.

- Initialization.

  Set up the parameters to proper values, and start generation of the first block of entropy. This is done once.

- Read entropy from the TRNG, and start generation of the next block of entropy.

  This is done many times and is the normal flow of operation.

- Run a self-test on the TRNG, to assure proper continued operation.

  This involves taking TRNG off-line, setting some self-test parameters, running TRNG, and then reading the statistical test registers, to see that they are within proper operation values. This may not be needed, as TRNG has built-in self-test.

- Off-line determination and checking of TRNG parameter values.

  This is done in development in order to determine the proper initialization and self-test parameters. The TRNG is taken off-line. Test parameter values are written and entropy generation is started. If the statistical tests indicate poor operation (i.e., failing statistical tests), the entropy_delay value should be increased and entropy generation should be re-started. Every case is a variation of setting TRNG parameter values, starting or re-starting entropy generation and reading out the entropy. This process requires pausing or stopping and re-starting the TRNG.

The TRNG is designed to operate as a slave module on the standard IP Slave Bus. By understanding the TRNG register descriptions in "TRNG Register Descriptions" section below, the TRNG module can be controlled via the IP slave bus. In order to write to most TRNG registers, the MCTL register must be initialized in programming mode as described in the "TRNG Register Descriptions" section. At Power On Reset (POR), the TRNG resets to programming mode. And the it will not generate entropy until it is out of programming mode (in run mode) and access to Entropy Registers have been enabled.

Here is an example program flow of using the TRNG.

- After POR the TRNG will be reset into programming mode with the OK to stop bit set(MCTL[TSTOP_OK]=1). The TRNG must be put into Run Mode for Entropy Generation to begin (MCTL[PRGM]=0). Additionally, in order to have access to the Entropy registers and other critical TRNG registers, the TRNG access bit must be set (MCTL[TRNG_ACC]=1). Using the default self test limits that exist after bootup, the entropy valid bit can be polled until asserted (MCTL[ENT_VAL]=1). Alternatively, if using the interrupt, and the interrupts are enabled via the INT_MASK register and the ipi_rng_int_b is asserted when MCTL[ENT_VAL]=1.
- After the polling completes, the 512-bit entropy generated by the TRNG can be read. The values can be read in any order from entropy register 0 to register 15 (ENT0 to ENT 15). After reading ENT 15, the old entropy value is reset and a new entropy value is generated.

NOTE

Reading ENT 15always resets the entropy, so should always be read last.

- You can poll again for the new entropy value or you can use the Interrupt Status Register to handle reading the entropy values when the entropy valid interrupt is triggered.
- The interrupt can be masked or cleared as needed. See the Interrupt Status Register description.
- To change the self-test limits,the seed counters, how fast the entropy is generated, and how entropy is sampled, see the register description section. In particular, see the the TRNG Frequency Count Minimum Limit Register (FRQMIN), the seed control register(SCML), the statistical run length registers, and other parameter registers.
- Once in Run Mode, the entropy is re-generated automatically after ENT 15 is read. To stop the TRNG or access to TRNG registers at any point while in running mode, you can always set MCTL[TRNG_ACC]=0. Setting the TRNG back to programming mode (MCTL[PRGM]=1) also achieves the purpose of stopping entropy generation.

49.1.3.2 Register interface (IP Slave bus)

The TRNG's register interface (32-bit IP bus) is used to read and write registers within TRNG for the following purposes:

Table 49-2. Summary of register interface uses

Purpose	For more information, see
During chip initialization time	
To configure TRNG including initialization of the • Registers • TRNG Register Interface	
During hardware and software debugging	
Read status registers	• RNG TRNG Status Register •

NOTE

Accesses to registers must use full-word (32-bit) reads or writes.

49.1.3.3 TRNG0 Register Descriptions

All accesses of undefined addresses always return zero and assert IPS transfer error. Writes to undefined and read-only addresses are ignored. Undefined addresses are those undocumented, protected or reserved addresses within and outside the range of the addresses defined in the memory map below. Although many of the TRNG0 registers hold more than 32 bits, the register addresses shown in the Memory Map below represent how these registers are accessed over the register bus as 32-bit words.

The format and fields in each TRNG0 register are defined below. Some of the register format figures apply to several different registers. In such cases a different register name will be associated with each of the register offset addresses that appear at the top of the register format figure. Although these registers share the same format, they are independent registers. In addition, many registers can be accessed at multiple addresses. In these cases there will be a single register name and the list of addresses at which that register is accessible will be indicated as aliases. Unless noted in the individual register descriptions, registers are reset only at Power-On Reset (POR).

49.1.3.3.1 TRNG0 Memory Map

Offset	Register	Width (In bits)	Access	Reset value
400A5000h	TRNG0 Miscellaneous Control (TRNG0_MCTL)	32	RW	00012001h
400A5004h	TRNG0 Statistical Check Miscellaneous (TRNG0_SCMISC)	32	RW	00010022h[1]
400A5008h	TRNG0 Poker Range (TRNG0_PKRRNG)	32	RW	000009A3h
400A500Ch	TRNG0 Poker Maximum Limit (TRNG0_PKRMAX)	32	RW	00006920h
400A500Ch	TRNG0 Poker Square Calculation Result (TRNG0_PKRSQ)	32	RO	00000000h
400A5010h	TRNG0 Seed Control (TRNG0_SDCTL)	32	RW	0C8009C4h
400A5014h	TRNG0 Sparse Bit Limit (TRNG0_SBLIM)	32	RW	0000003Fh
400A5014h	TRNG0 Total Samples (TRNG0_TOTSAM)	32	RO	00000000h
400A5018h	TRNG0 Frequency Count Minimum Limit (TRNG0_FRQMIN)	32	RW	00000640h
400A501Ch	TRNG0 Frequency Count Maximum Limit (TRNG0_FRQMAX)	32	RW	00006400h
400A501Ch	TRNG0 Frequency Count (TRNG0_FRQCNT)	32	RO	00000000h
400A5020h	TRNG0 Statistical Check Monobit Count (TRNG0_SCMC)	32	RO	00000000h
400A5020h	TRNG0 Statistical Check Monobit Limit (TRNG0_SCML)	32	RW	010C0568h
400A5024h	TRNG0 Statistical Check Run Length 1 Limit (TRNG0_SCR1L)	32	RW	00B20195h
400A5024h	TRNG0 Statistical Check Run Length 1 Count (TRNG0_SCR1C)	32	RO	00000000h
400A5028h	TRNG0 Statistical Check Run Length 2 Limit (TRNG0_SCR2L)	32	RW	007A00DCh
400A5028h	TRNG0 Statistical Check Run Length 2 Count (TRNG0_SCR2C)	32	RO	00000000h
400A502Ch	TRNG0 Statistical Check Run Length 3 Count (TRNG0_SCR3C)	32	RO	00000000h
400A502Ch	TRNG0 Statistical Check Run Length 3 Limit (TRNG0_SCR3L)	32	RW	0058007Dh
400A5030h	TRNG0 Statistical Check Run Length 4 Count (TRNG0_SCR4C)	32	RO	00000000h
400A5030h	TRNG0 Statistical Check Run Length 4 Limit (TRNG0_SCR4L)	32	RW	0040004Bh
400A5034h	TRNG0 Statistical Check Run Length 5 Limit (TRNG0_SCR5L)	32	RW	002E002Fh
400A5034h	TRNG0 Statistical Check Run Length 5 Count (TRNG0_SCR5C)	32	RO	00000000h
400A5038h	TRNG0 Statistical Check Run Length 6+ Limit (TRNG0_SCR6PL)	32	RW	002E002Fh
400A5038h	TRNG0 Statistical Check Run Length 6+ Count (TRNG0_SCR6PC)	32	RO	00000000h
400A503Ch	TRNG0 Status (TRNG0_STATUS)	32	RO	00000000h
400A5040h - 400A507Ch	TRNG0 Entropy Read (TRNG0_ENT0 - TRNG0_ENT15)	32	RO	00000000h
400A5080h	TRNG0 Statistical Check Poker Count 1 and 0 (TRNG0_PKRCNT10)	32	RO	00000000h
400A5084h	TRNG0 Statistical Check Poker Count 3 and 2 (TRNG0_PKRCNT32)	32	RO	00000000h
400A5088h	TRNG0 Statistical Check Poker Count 5 and 4 (TRNG0_PKRCNT54)	32	RO	00000000h
400A508Ch	TRNG0 Statistical Check Poker Count 7 and 6 (TRNG0_PKRCNT76)	32	RO	00000000h
400A5090h	TRNG0 Statistical Check Poker Count 9 and 8 (TRNG0_PKRCNT98)	32	RO	00000000h
400A5094h	TRNG0 Statistical Check Poker Count B and A (TRNG0_PKRCNT BA)	32	RO	00000000h
400A5098h	TRNG0 Statistical Check Poker Count D and C (TRNG0_PKRCNT DC)	32	RO	00000000h
400A509Ch	TRNG0 Statistical Check Poker Count F and E (TRNG0_PKRCNT FE)	32	RO	00000000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
400A50A0h	TRNG0 Security Configuration (TRNG0_SEC_CFG)	32	RW	00000000h
400A50A4h	TRNG0 Interrupt Control (TRNG0_INT_CTRL)	32	RW	FFFFFFFFh
400A50A8h	TRNG0 Mask (TRNG0_INT_MASK)	32	RW	00000000h
400A50ACh	TRNG0 Interrupt Status (TRNG0_INT_STATUS)	32	RW	00000000h
400A50F0h	TRNG0 Version ID (MS) (TRNG0_VID1)	32	RO	00300100h
400A50F4h	TRNG0 Version ID (LS) (TRNG0_VID2)	32	RO	00000000h

1. Reset occurs at POR, and when TRNG0_MCTL[RST_DEF] is written to 1.

49.1.3.3.2 TRNG0 Miscellaneous Control (TRNG0_MCTL)

49.1.3.3.2.1 Address

Register	Offset
TRNG0_MCTL	400A5000h

49.1.3.3.2.2 Function

This register is intended to be used for programming, configuring and testing the RNG. It is the main register to read/write, in order to enable Entropy generation, to stop entropy generation and to block access to entropy registers. This is done via the special TRNG_ACC and PRGM bits below.

The TRNG0 Miscellaneous Control Register is a read/write register used to control the RNG's True Random Number Generator (TRNG) access, operation and test.

NOTE

Note that in many cases two RNG registers share the same address, and a particular register at the shared address is selected based upon the value in the PRGM field of the TRNG0_MCTL register.

49.1.3.3.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															PRGM
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		TSTOP_OK	ERR	TST_OUT	ENT_VAL	FCT_VAL	FCT_FAIL	FOR_SCLK		TRNG_ACC	UNUSED	OSC_DIV		SAMP_MODE	
W				w1c						RST_DEF						
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1

49.1.3.3.2.4 Fields

Field	Function
31-17 —	Reserved.
16 PRGM	Programming Mode Select. When this bit is 1, the TRNG is in Program Mode, otherwise it is in Run Mode. No Entropy value will be generated while the TRNG is in Program Mode. Note that different RNG registers are accessible at the same address depending on whether PRGM is set to 1 or 0. This is noted in the RNG register descriptions.
15-14 —	Reserved.
13 TSTOP_OK	TRNG_OK_TO_STOP. Software should check that this bit is a 1 before transitioning TRNG0 to low power mode (TRNG0 clock stopped). TRNG0 turns on the TRNG free-running ring oscillator whenever new entropy is being generated and turns off the ring oscillator when entropy generation is complete. If the TRNG0 clock is stopped while the TRNG ring oscillator is running, the oscillator will continue running even though the TRNG0 clock is stopped. TSTOP_OK is asserted when the TRNG ring oscillator is not running. and therefore it is ok to stop the TRNG0 clock.
12 ERR	Read: Error status. 1 = error detected. 0 = no error. Write: Write 1 to clear errors. Writing 0 has no effect.
11 TST_OUT	Read only: Test point inside ring oscillator.
10 ENT_VAL	Read only: Entropy Valid. Will assert only if TRNG ACC bit is set, and then after an entropy value is generated. Will be cleared at most one (1) bus clock cycle after reading the TRNG0_ENT15 register. (TRNG0_ENT0 through TRNG0_ENT14 should be read before reading TRNG0_ENT15).
9 FCT_VAL	Read only: Frequency Count Valid. Indicates that a valid frequency count may be read from TRNG0_FRQCNT.
8 FCT_FAIL	Read only: Frequency Count Fail. The frequency counter has detected a failure. This may be due to improper programming of the TRNG0_FRQMAX and/or TRNG0_FRQMIN registers, or a hardware failure in the ring oscillator. This error may be cleared by writing a 1 to the ERR bit.

Table continues on the next page...

Field	Function
7 FOR_SCLK	Force System Clock. If set, the system clock is used to operate the TRNG, instead of the ring oscillator. This is for test use only, and indeterminate results may occur. This bit is writable only if PRGM bit is 1, or PRGM bit is being written to 1 simultaneously to writing this bit. This bit is cleared by writing the RST_DEF bit to 1.
6 RST_DEF	Reset Defaults. Writing a 1 to this bit clears various TRNG registers, and bits within registers, to their default state. This bit is writable only if PRGM bit is 1, or PRGM bit is being written to 1 simultaneously to writing this bit. Reading this bit always produces a 0.
5 TRNG_ACC	TRNG Access Mode. If this bit is set to 1, the TRNG will generate an Entropy value that can be read via the TRNG0_ENT0-TRNG0_ENT15 registers. The Entropy value may be read once the ENT VAL bit is asserted. Also see TRNG0_ENTa register descriptions (For a = 0 to 15).
4 UNUSED	This bit is unused but write-able. Must be left as zero.
3-2 OSC_DIV	Oscillator Divide. Determines the amount of dividing done to the ring oscillator before it is used by the TRNG. This field is writable only if PRGM bit is 1, or PRGM bit is being written to 1 simultaneously to writing this field. This field is cleared to the default POR value by writing the RST_DEF bit to 1. 00 - use ring oscillator with no divide 01 - use ring oscillator divided-by-2 10 - use ring oscillator divided-by-4 11 - use ring oscillator divided-by-8
1-0 SAMP_MODE	Sample Mode. Determines the method of sampling the ring oscillator while generating the Entropy value: This field is writable only if PRGM bit is 1, or PRGM bit is being written to 1 simultaneously with writing this field. This field is cleared to the POR default value by writing the RST_DEF bit to 1. 00 - use Von Neumann data into both Entropy shifter and Statistical Checker 01 - use raw data into both Entropy shifter and Statistical Checker 10 - use Von Neumann data into Entropy shifter. Use raw data into Statistical Checker 11 - undefined/reserved.

49.1.3.3.3 TRNG0 Statistical Check Miscellaneous (TRNG0_SCMISC)

49.1.3.3.3.1 Address

Register	Offset
TRNG0_SCMISC	400A5004h

49.1.3.3.3.2 Function

The TRNG0 Statistical Check Miscellaneous Register contains the Long Run Maximum Limit value and the Retry Count value. This register is accessible only when the TRNG0_MCTL[PRGM] bit is 1, otherwise this register will read zeroes, and cannot be written.

49.1.3.3.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												RTY_CT			
Reset	0[1]	0	0	0	0	0	0	0	0	0	0	0	0[1]	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								LRUN_MAX							
Reset	0[1]	0	0	0	0	0	0	0	0[1]	0	1	0	0	0	1	0

1. Reset occurs at POR, and when TRNG0_MCTL[RST_DEF] is written to 1.

49.1.3.3.3.4 Fields

Field	Function
31-20 —	Reserved.
19-16 RTY_CT	RETRY COUNT. If a statistical check fails during the TRNG Entropy Generation, the RTY_CT value indicates the number of times a retry should occur before generating an error. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field will read zeroes if TRNG0_MCTL[PRGM] = 0. This field is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15-8 —	Reserved.
7-0 LRUN_MAX	LONG RUN MAX LIMIT. This value is the largest allowable number of consecutive samples of all 1, or all 0, that is allowed during the Entropy generation. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field will read zeroes if TRNG0_MCTL[PRGM] = 0. This field is cleared to the POR reset value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.4 TRNG0 Poker Range (TRNG0_PKRRNG)

49.1.3.3.4.1 Address

Register	Offset
TRNG0_PKRRNG	400A5008h

49.1.3.3.4.2 Function

The TRNG0 Poker Range Register defines the difference between the TRNG Poker Maximum Limit and the minimum limit. These limits are used during the TRNG Statistical Check Poker Test.

49.1.3.3.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PKR_RNG															
Reset	0	0	0	0	1	0	0	1	1	0	1	0	0	0	1	1

49.1.3.3.4.4 Fields

Field	Function
31-16 —	Reserved. Always 0.
15-0 PKR_RNG	Poker Range. During the TRNG Statistical Checks, a "Poker Test" is run which requires a maximum and minimum limit. The maximum is programmed in the PKRMAX[PKR_MAX] register, and the minimum is derived by subtracting the PKR_RNG value from the programmed maximum value. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field will read zeroes if TRNG0_MCTL[PRGM] = 0. This field is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1. Note that the minimum allowable Poker result is PKR_MAX - PKR_RNG + 1.

49.1.3.3.5 TRNG0 Poker Maximum Limit (TRNG0_PKRMAX)

49.1.3.3.5.1 Address

Register	Offset	Description
TRNG0_PKRMAX	400A500Ch	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.5.2 Function

The TRNG0 Poker Maximum Limit Register defines Maximum Limit allowable during the TRNG Statistical Check Poker Test. Note that this offset (0x0C) is used as TRNG0_PKRMAX only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as the TRNG0_PKRSQ readback register.

49.1.3.3.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								PKR_MAX							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PKR_MAX															
Reset	0	1	1	0	1	0	0	1	0	0	1	0	0	0	0	0

49.1.3.3.5.4 Fields

Field	Function
31-24 —	
23-0 PKR_MAX	Poker Maximum Limit. During the TRNG Statistical Checks, a "Poker Test" is run which requires a maximum and minimum limit. The maximum allowable result is programmed in the TRNG0_PKRMAX[PKR_MAX] register. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1. Note that the TRNG0_PKRMAX and TRNG0_PKRRNG registers combined are used to define the minimum allowable Poker result, which is PKR_MAX - PKR_RNG + 1. Note that if TRNG0_MCTL[PRGM] bit is 0, this register address is used to read the Poker Test Square Calculation result in register TRNG0_PKRSQ, as defined in the following section.

49.1.3.3.6 TRNG0 Poker Square Calculation Result (TRNG0_PKRSQ)

49.1.3.3.6.1 Address

Register	Offset	Description
TRNG0_PKRSQ	400A500Ch	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.6.2 Function

The TRNG0 Poker Square Calculation Result Register is a read-only register used to read the result of the TRNG Statistical Check Poker Test's Square Calculation. This test starts with the TRNG0_PKRMAX value and decreases towards a final result, which is read here. For the Poker Test to pass, this final result must be less than the programmed TRNG0_PKRRNG value. Note that this offset (0x0C) is used as TRNG0_PKRMAX if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_PKRSQ readback register, as described here.

49.1.3.3.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								PKR_SQ							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_SQ															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.6.4 Fields

Field	Function
31-24 —	
23-0 PKR_SQ	Poker Square Calculation Result. During the TRNG Statistical Checks, a "Poker Test" is run which starts with the value TRNG0_PKRMAX[PKR_MAX]. This value decreases according to a "sum of squares" algorithm, and must remain greater than zero, but less than the TRNG0_PKRRNG[PKR_RNG] limit. The resulting value may be read through this register, if TRNG0_MCTL[PRGM] bit is 0. Note that if TRNG0_MCTL[PRGM] bit is 1, this register address is used to access the Poker Test Maximum Limit in register TRNG0_PKRMAX, as defined in the previous section.

49.1.3.3.7 TRNG0 Seed Control (TRNG0_SDCTL)

49.1.3.3.7.1 Address

Register	Offset
TRNG0_SDCTL	400A5010h

49.1.3.3.7.2 Function

The TRNG0 Seed Control Register contains two fields. One field defines the length (in system clocks) of each Entropy sample (ENT_DLY), and the other field indicates the number of samples that will taken during each TRNG Entropy generation (SAMP_SIZE).

49.1.3.3.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ENT_DLY															
Reset	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SAMP_SIZE															
Reset	0	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0

49.1.3.3.7.4 Fields

Field	Function
31-16 ENT_DLY	Entropy Delay. Defines the length (in system clocks) of each Entropy sample taken. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field will read zeroes if TRNG0_MCTL[PRGM] = 0. This field is cleared to its reset value at POR.
15-0 SAMP_SIZE	Sample Size. Defines the total number of Entropy samples that will be taken during Entropy generation. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field will read zeroes if TRNG0_MCTL[PRGM] = 0. This field is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.8 TRNG0 Sparse Bit Limit (TRNG0_SBLIM)

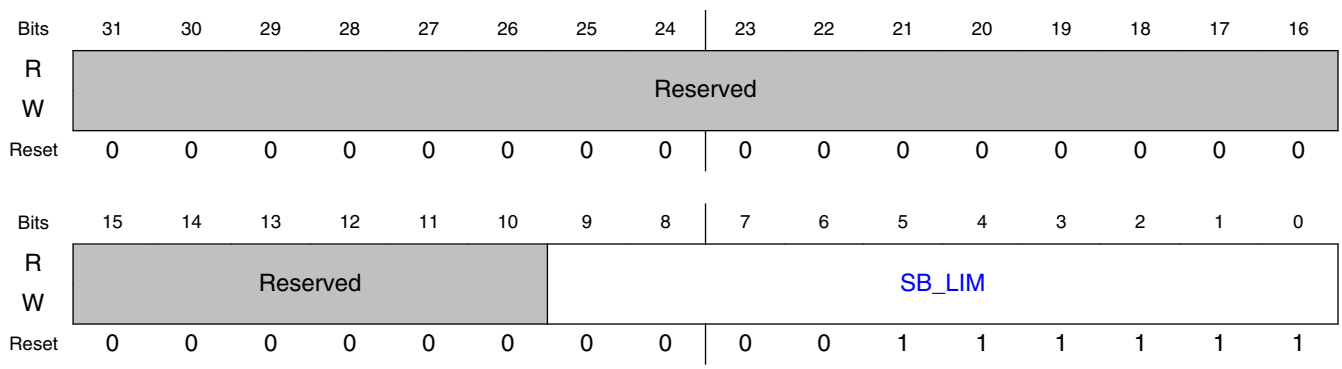
49.1.3.3.8.1 Address

Register	Offset	Description
TRNG0_SBLIM	400A5014h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.8.2 Function

The TRNG0 Sparse Bit Limit Register is used when Von Neumann sampling is selected during Entropy Generation. It defines the maximum number of consecutive Von Neumann samples which may be discarded before an error is generated. Note that this address (0x14) is used as TRNG0_SBLIM only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_TOTSAM readback register.

49.1.3.3.8.3 Diagram



49.1.3.3.8.4 Fields

Field	Function
31-10 —	Reserved. Always 0.
9-0 SB_LIM	Sparse Bit Limit. During Von Neumann sampling (if enabled by TRNG0_MCTL[SAMP_MODE], samples are discarded if two consecutive raw samples are both 0 or both 1. If this discarding occurs for a long period of time, it indicates that there is insufficient Entropy. The Sparse Bit Limit defines the maximum number of consecutive samples that may be discarded before an error is generated. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1. Note that if TRNG0_MCTL[PRGM] bit is 0, this register address is used to read the Total Samples count in register TRNG0_TOTSAM, as defined in the following section.

49.1.3.3.9 TRNG0 Total Samples (TRNG0_TOTSAM)

49.1.3.3.9.1 Address

Register	Offset	Description
TRNG0_TOTSAM	400A5014h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.9.2 Function

The TRNG0 Total Samples Register is a read-only register used to read the total number of samples taken during Entropy generation. It is used to give an indication of how often a sample is actually used during Von Neumann sampling. Note that this offset (0x14) is used as TRNG0_SBLIM if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_TOTSAM readback register, as described here.

49.1.3.3.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												TOT_SAM			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOT_SAM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.9.4 Fields

Field	Function
31-20 —	Reserved. Always 0.
19-0 TOT_SAM	Total Samples. During Entropy generation, the total number of raw samples is counted. This count is useful in determining how often a sample is used during Von Neumann sampling. The count may be read through this register, if TRNG0_MCTL[PRGM] bit is 0. Note that if TRNG0_MCTL[PRGM] bit is 1, this register address is used to access the Sparse Bit Limit in register TRNG0_SBLIM, as defined in the previous section.

49.1.3.3.10 TRNG0 Frequency Count Minimum Limit (TRNG0_FRQMIN)

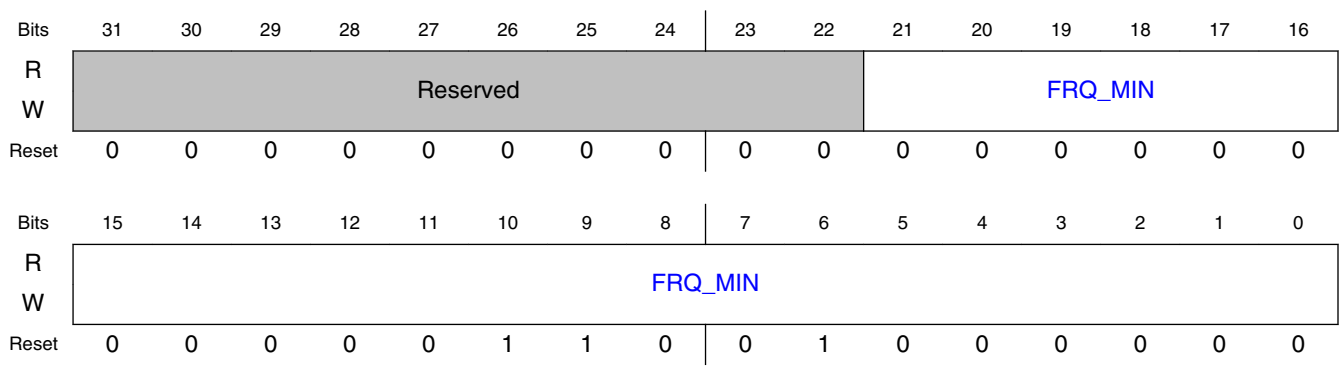
49.1.3.3.10.1 Address

Register	Offset
TRNG0_FRQMIN	400A5018h

49.1.3.3.10.2 Function

The TRNG0 Frequency Count Minimum Limit Register defines the minimum allowable count taken by the Entropy sample counter during each Entropy sample. During any sample period, if the count is less than this programmed minimum, a Frequency Count Fail is flagged in TRNG0_MCTL[FCT_FAIL] and an error is generated.

49.1.3.3.10.3 Diagram



49.1.3.3.10.4 Fields

Field	Function
31-22 —	Reserved. Always 0.
21-0 FRQ_MIN	Frequency Count Minimum Limit. Defines the minimum allowable count taken during each entropy sample. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field will read zeroes if TRNG0_MCTL[PRGM] = 0. This field is cleared to its reset value at POR.

49.1.3.3.11 TRNG0 Frequency Count (TRNG0_FRQCNT)

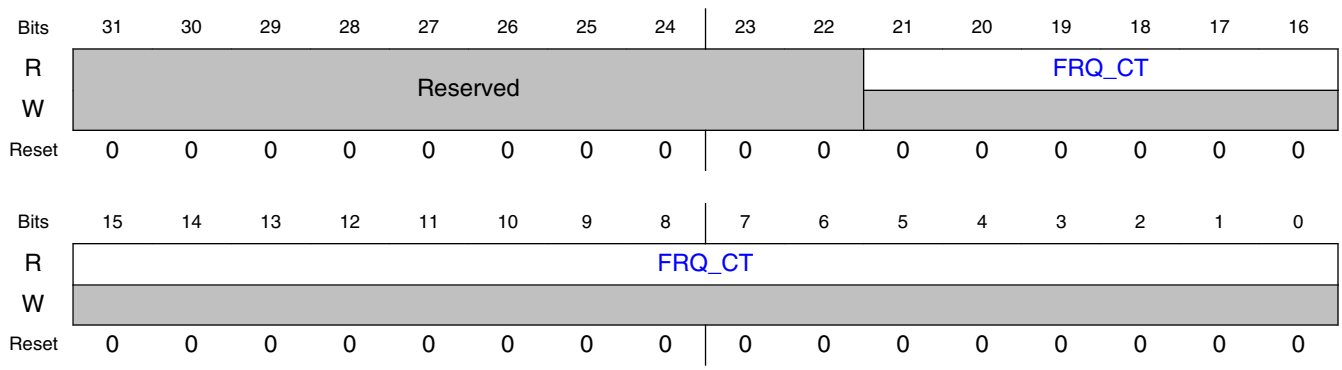
49.1.3.3.11.1 Address

Register	Offset	Description
TRNG0_FRQCNT	400A501Ch	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.11.2 Function

The TRNG0 Frequency Count Register is a read-only register used to read the frequency counter within the TRNG entropy generator. It will read all zeroes unless TRNG0_MCTL[TRNG_ACC] = 1. Note that this offset (0x1C) is used as TRNG0_FRQMAX if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_FRQCNT readback register, as described here.

49.1.3.3.11.3 Diagram



49.1.3.3.11.4 Fields

Field	Function
31-22 —	Reserved. Always 0.
21-0 FRQ_CT	Frequency Count. If TRNG0_MCTL[TRNG_ACC] = 1, reads a sample frequency count taken during entropy generation. Requires TRNG0_MCTL[PRGM] = 0. Note that if TRNG0_MCTL[PRGM] bit is 1, this register address is used to access the Poker Test Maximum Limit in register TRNG0_PKRMAX, as defined in the previous section.

49.1.3.3.12 TRNG0 Frequency Count Maximum Limit (TRNG0_FRQMAX)

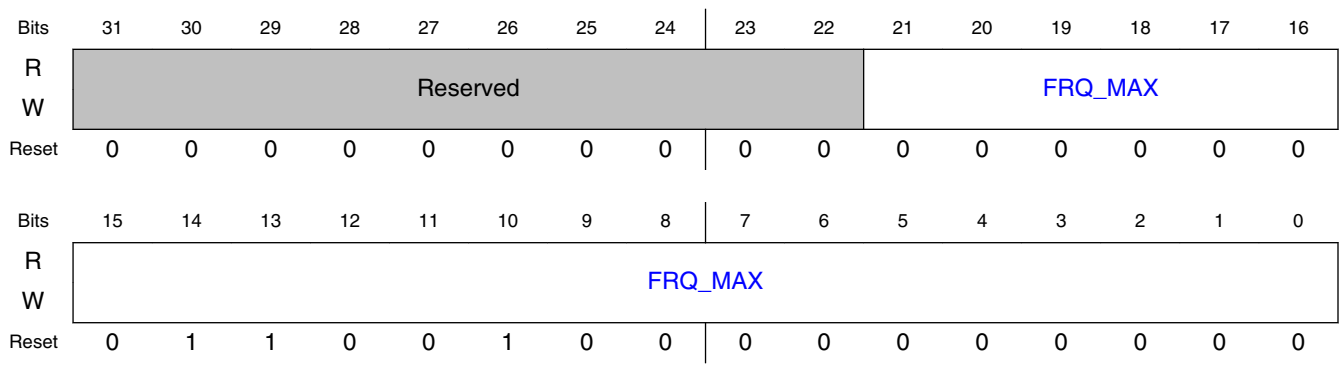
49.1.3.3.12.1 Address

Register	Offset	Description
TRNG0_FRQMAX	400A501Ch	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.12.2 Function

The TRNG0 Frequency Count Maximum Limit Register defines the maximum allowable count taken by the Entropy sample counter during each Entropy sample. During any sample period, if the count is greater than this programmed maximum, a Frequency Count Fail is flagged in TRNG0_MCTL[FCT_FAIL] and an error is generated. Note that this address (001C) is used as TRNG0_FRQMAX only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_FRQCNT readback register.

49.1.3.3.12.3 Diagram



49.1.3.3.12.4 Fields

Field	Function
31-22 —	Reserved. Always 0.
21-0 FRQ_MAX	Frequency Counter Maximum Limit. Defines the maximum allowable count taken during each entropy sample. This field is writable only if TRNG0_MCTL[PRGM] bit is 1. This field is cleared to its reset value at POR. Note that if TRNG0_MCTL[PRGM] bit is 0, this register address is used to read the Frequency Count result in register TRNG0_FRQCNT, as defined in the following section.

49.1.3.3.13 TRNG0 Statistical Check Monobit Count (TRNG0_SCMC)

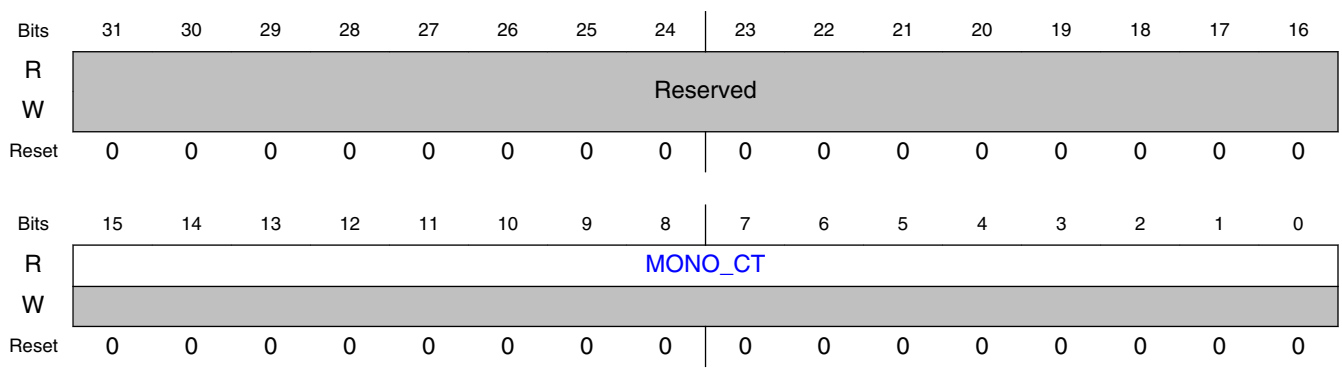
49.1.3.3.13.1 Address

Register	Offset	Description
TRNG0_SCMC	400A5020h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.13.2 Function

The TRNG0 Statistical Check Monobit Count Register is a read-only register used to read the final monobit count after entropy generation. This counter starts with the value in TRNG0_SCML[MONO_MAX], and is decremented each time a one is sampled. Note that this offset (0x20) is used as TRNG0_SCML if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCMC readback register, as described here.

49.1.3.3.13.3 Diagram



49.1.3.3.13.4 Fields

Field	Function
31-16 —	Reserved. Always 0.
15-0 MONO_CT	Monobit Count. Reads the final Monobit count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0. Note that if TRNG0_MCTL[PRGM] bit is 1, this register address is used to access the Statistical Check Monobit Limit in register TRNG0_SCML, as defined in the previous section.

49.1.3.3.14 TRNG0 Statistical Check Monobit Limit (TRNG0_SCML)

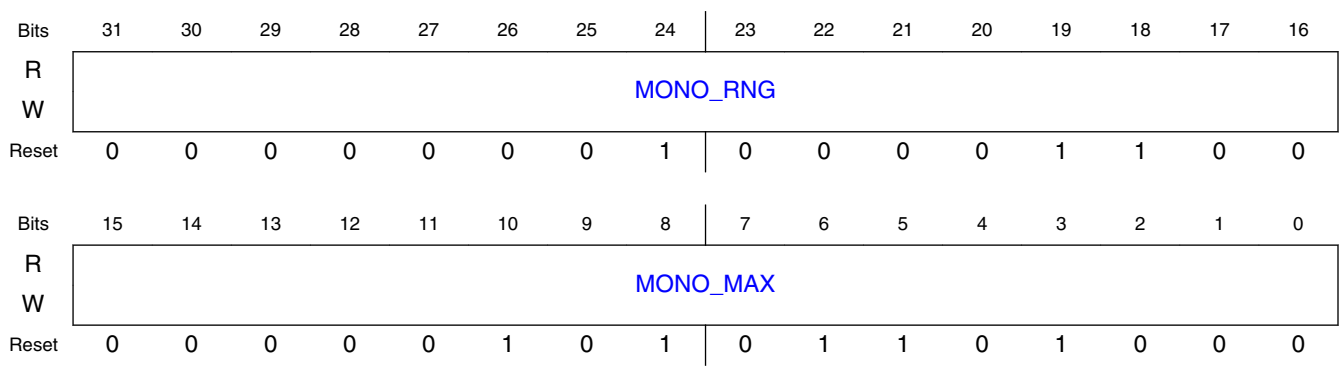
49.1.3.3.14.1 Address

Register	Offset	Description
TRNG0_SCML	400A5020h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.14.2 Function

The TRNG0 Statistical Check Monobit Limit Register defines the allowable maximum and minimum number of ones/zero detected during entropy generation. To pass the test, the number of ones/zeroes generated must be less than the programmed maximum value, and the number of ones/zeroes generated must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this offset (0x20) is used as TRNG0_SCML only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCMC readback register.

49.1.3.3.14.3 Diagram



49.1.3.3.14.4 Fields

Field	Function
31-16 MONO_RNG	Monobit Range. The number of ones/zeroes detected during entropy generation must be greater than MONO_MAX - MONO_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

Table continues on the next page...

Field	Function
15-0 MONO_MAX	Monobit Maximum Limit. Defines the maximum allowable count taken during entropy generation. The number of ones/zeroes detected during entropy generation must be less than MONO_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.15 TRNG0 Statistical Check Run Length 1 Count (TRNG0_SCR1C)

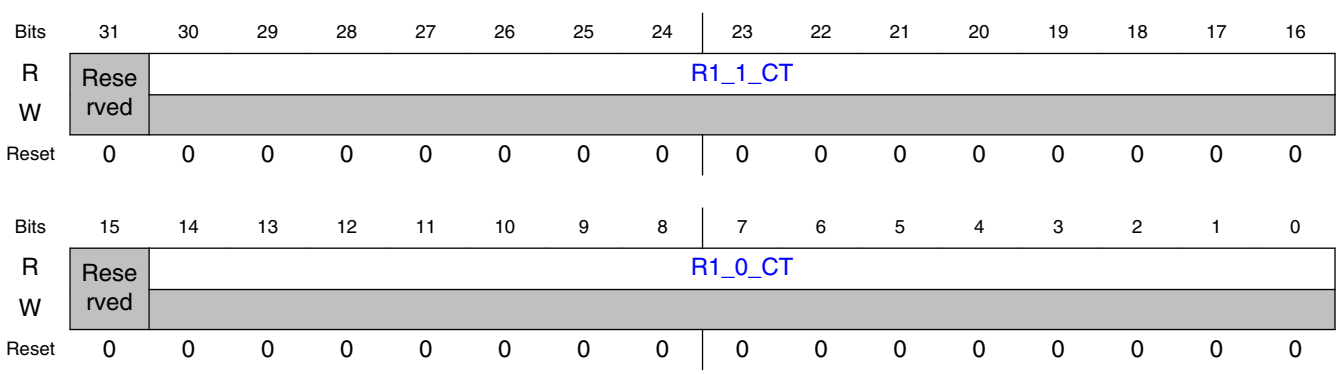
49.1.3.3.15.1 Address

Register	Offset	Description
TRNG0_SCR1C	400A5024h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.15.2 Function

The TRNG0 Statistical Check Run Length 1 Counters Register is a read-only register used to read the final Run Length 1 counts after entropy generation. These counters start with the value in TRNG0_SCRxC1L[RUN1_MAX]. The R1_1_CT decrements each time a single one is sampled (preceded by a zero and followed by a zero). The R1_0_CT decrements each time a single zero is sampled (preceded by a one and followed by a one). Note that this offset (0x24) is used as TRNG0_SCRxC1L if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC1C readback register, as described here.

49.1.3.3.15.3 Diagram



49.1.3.3.15.4 Fields

Field	Function
31 —	Reserved. Always 0.
30-16 R1_1_CT	Runs of One, Length 1 Count. Reads the final Runs of Ones, length 1 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.
15 —	Reserved. Always 0.
14-0 R1_0_CT	Runs of Zero, Length 1 Count. Reads the final Runs of Zeroes, length 1 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.16 TRNG0 Statistical Check Run Length 1 Limit (TRNG0_SCR1L)

49.1.3.3.16.1 Address

Register	Offset	Description
TRNG0_SCR1L	400A5024h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.16.2 Function

The TRNG0 Statistical Check Run Length 1 Limit Register defines the allowable maximum and minimum number of runs of length 1 detected during entropy generation. To pass the test, the number of runs of length 1 (for samples of both 0 and 1) must be less than the programmed maximum value, and the number of runs of length 1 must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this address (0x24) is used as TRNG0_SCRxC1L only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_SCRxC1C readback register.

49.1.3.3.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RUN1_RNG							
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RUN1_MAX							
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	1

49.1.3.3.16.4 Fields

Field	Function
31 —	Reserved. Always 0.
30-16 RUN1_RNG	Run Length 1 Range. The number of runs of length 1 (for both 0 and 1) detected during entropy generation must be greater than RUN1_MAX - RUN1_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15 —	Reserved. Always 0.
14-0 RUN1_MAX	Run Length 1 Maximum Limit. Defines the maximum allowable runs of length 1 (for both 0 and 1) detected during entropy generation. The number of runs of length 1 detected during entropy generation must be less than RUN1_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.17 TRNG0 Statistical Check Run Length 2 Count (TRNG0_SCR2C)

49.1.3.3.17.1 Address

Register	Offset	Description
TRNG0_SCR2C	400A5028h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.17.2 Function

The TRNG0 Statistical Check Run Length 2 Counters Register is a read-only register used to read the final Run Length 2 counts after entropy generation. These counters start with the value in TRNG0_SCRxC2L[RUN2_MAX]. The R2_1_CT decrements each time two consecutive ones are sampled (preceded by a zero and followed by a zero). The

R2_0_CT decrements each time two consecutive zeroes are sampled (preceded by a one and followed by a one). Note that this offset (0x28) is used as TRNG0_SCRxC2L if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC2C readback register, as described here.

49.1.3.3.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		R2_1_CT													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		R2_0_CT													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.17.4 Fields

Field	Function
31-30 —	Reserved. Always 0.
29-16 R2_1_CT	Runs of One, Length 2 Count. Reads the final Runs of Ones, length 2 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.
15-14 —	Reserved. Always 0.
13-0 R2_0_CT	Runs of Zero, Length 2 Count. Reads the final Runs of Zeroes, length 2 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.18 TRNG0 Statistical Check Run Length 2 Limit (TRNG0_SCR2L)

49.1.3.3.18.1 Address

Register	Offset	Description
TRNG0_SCR2L	400A5028h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.18.2 Function

The TRNG0 Statistical Check Run Length 2 Limit Register defines the allowable maximum and minimum number of runs of length 2 detected during entropy generation. To pass the test, the number of runs of length 2 (for samples of both 0 and 1) must be less than the programmed maximum value, and the number of runs of length 2 must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this address (0x28) is used as TRNG0_SCRxC2L only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_SCRxC2C readback register.

49.1.3.3.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		RUN2_RNG													
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		RUN2_MAX													
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	1	1	1	0	0

49.1.3.3.18.4 Fields

Field	Function
31-30 —	Reserved. Always 0.
29-16 RUN2_RNG	Run Length 2 Range. The number of runs of length 2 (for both 0 and 1) detected during entropy generation must be greater than RUN2_MAX - RUN2_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15-14 —	Reserved. Always 0.
13-0 RUN2_MAX	Run Length 2 Maximum Limit. Defines the maximum allowable runs of length 2 (for both 0 and 1) detected during entropy generation. The number of runs of length 2 detected during entropy generation must be less than RUN2_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.19 TRNG0 Statistical Check Run Length 3 Count (TRNG0_SCR3C)

49.1.3.3.19.1 Address

Register	Offset	Description
TRNG0_SCR3C	400A502Ch	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.19.2 Function

The TRNG0 Statistical Check Run Length 3 Counters Register is a read-only register used to read the final Run Length 3 counts after entropy generation. These counters start with the value in TRNG0_SCRxC3L[RUN3_MAX]. The R3_1_CT decrements each time three consecutive ones are sampled (preceded by a zero and followed by a zero). The R3_0_CT decrements each time three consecutive zeroes are sampled (preceded by a one and followed by a one). Note that this offset (0x2C) is used as TRNG0_SCRxC3L if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC3C readback register, as described here.

49.1.3.3.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			R3_1_CT												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			R3_0_CT												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.19.4 Fields

Field	Function
31-29 —	Reserved. Always 0.
28-16 R3_1_CT	Runs of Ones, Length 3 Count. Reads the final Runs of Ones, length 3 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.
15-13	Reserved. Always 0.

Table continues on the next page...

Field	Function
—	
12-0 R3_0_CT	Runs of Zeroes, Length 3 Count. Reads the final Runs of Zeroes, length 3 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.20 TRNG0 Statistical Check Run Length 3 Limit (TRNG0_SCR3L)

49.1.3.3.20.1 Address

Register	Offset	Description
TRNG0_SCR3L	400A502Ch	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.20.2 Function

The TRNG0 Statistical Check Run Length 3 Limit Register defines the allowable maximum and minimum number of runs of length 3 detected during entropy generation. To pass the test, the number of runs of length 3 (for samples of both 0 and 1) must be less than the programmed maximum value, and the number of runs of length 3 must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this address (0x2C) is used as TRNG0_SCRxC3L only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_SCRxC3C readback register.

49.1.3.3.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			RUN3_RNG												
Reset	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			RUN3_MAX												
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1

49.1.3.3.20.4 Fields

Field	Function
31-29 —	Reserved. Always 0.
28-16 RUN3_RNG	Run Length 3 Range. The number of runs of length 3 (for both 0 and 1) detected during entropy generation must be greater than RUN3_MAX - RUN3_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15-13 —	Reserved. Always 0.
12-0 RUN3_MAX	Run Length 3 Maximum Limit. Defines the maximum allowable runs of length 3 (for both 0 and 1) detected during entropy generation. The number of runs of length 3 detected during entropy generation must be less than RUN3_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.21 TRNG0 Statistical Check Run Length 4 Count (TRNG0_SCR4C)

49.1.3.3.21.1 Address

Register	Offset	Description
TRNG0_SCR4C	400A5030h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.21.2 Function

The TRNG0 Statistical Check Run Length 4 Counters Register is a read-only register used to read the final Run Length 4 counts after entropy generation. These counters start with the value in TRNG0_SCRxC4L[RUN4_MAX]. The R4_1_CT decrements each time four consecutive ones are sampled (preceded by a zero and followed by a zero). The R4_0_CT decrements each time four consecutive zeroes are sampled (preceded by a one and followed by a one). Note that this offset (0x30) is used as TRNG0_SCRxC4L if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC4C readback register, as described here.

49.1.3.3.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				R4_1_CT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				R4_0_CT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.21.4 Fields

Field	Function
31-28 —	Reserved. Always 0.
27-16 R4_1_CT	Runs of One, Length 4 Count. Reads the final Runs of Ones, length 4 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.
15-12 —	Reserved. Always 0.
11-0 R4_0_CT	Runs of Zero, Length 4 Count. Reads the final Runs of Ones, length 4 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.22 TRNG0 Statistical Check Run Length 4 Limit (TRNG0_SCR4L)

49.1.3.3.22.1 Address

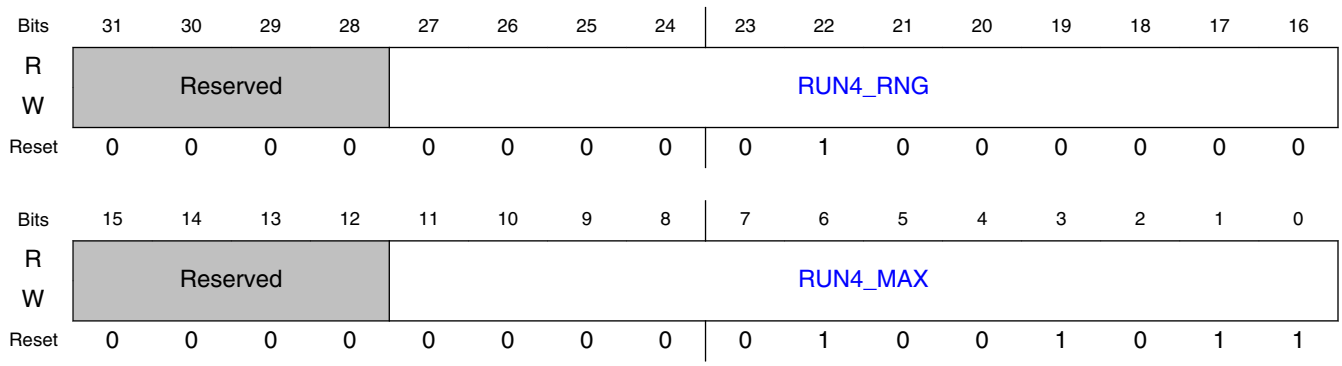
Register	Offset	Description
TRNG0_SCR4L	400A5030h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.22.2 Function

The TRNG0 Statistical Check Run Length 4 Limit Register defines the allowable maximum and minimum number of runs of length 4 detected during entropy generation. To pass the test, the number of runs of length 4 (for samples of both 0 and 1) must be less than the programmed maximum value, and the number of runs of length 4 must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC

will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this address (0x30) is used as TRNG0_SCRxC4L only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_SCRxC4C readback register.

49.1.3.3.22.3 Diagram



49.1.3.3.22.4 Fields

Field	Function
31-28 —	Reserved. Always 0.
27-16 RUN4_RNG	Run Length 4 Range. The number of runs of length 4 (for both 0 and 1) detected during entropy generation must be greater than RUN4_MAX - RUN4_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15-12 —	Reserved. Always 0.
11-0 RUN4_MAX	Run Length 4 Maximum Limit. Defines the maximum allowable runs of length 4 (for both 0 and 1) detected during entropy generation. The number of runs of length 4 detected during entropy generation must be less than RUN4_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.23 TRNG0 Statistical Check Run Length 5 Count (TRNG0_SCR5C)

49.1.3.3.23.1 Address

Register	Offset	Description
TRNG0_SCR5C	400A5034h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.23.2 Function

The TRNG0 Statistical Check Run Length 5 Counters Register is a read-only register used to read the final Run Length 5 counts after entropy generation. These counters start with the value in TRNG0_SCRxC5L[RUN5_MAX]. The R5_1_CT decrements each time five consecutive ones are sampled (preceded by a zero and followed by a zero). The R5_0_CT decrements each time five consecutive zeroes are sampled (preceded by a one and followed by a one). Note that this offset (0x34) is used as TRNG0_SCRxC5L if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC5C readback register, as described here.

49.1.3.3.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					R5_1_CT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					R5_0_CT										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.23.4 Fields

Field	Function
31-27 —	Reserved. Always 0.
26-16 R5_1_CT	Runs of One, Length 5 Count. Reads the final Runs of Ones, length 5 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.
15-11 —	Reserved. Always 0.
10-0 R5_0_CT	Runs of Zero, Length 5 Count. Reads the final Runs of Ones, length 5 count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.24 TRNG0 Statistical Check Run Length 5 Limit (TRNG0_SCR5L)

49.1.3.3.24.1 Address

Register	Offset	Description
TRNG0_SCR5L	400A5034h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.24.2 Function

The TRNG0 Statistical Check Run Length 5 Limit Register defines the allowable maximum and minimum number of runs of length 5 detected during entropy generation. To pass the test, the number of runs of length 5 (for samples of both 0 and 1) must be less than the programmed maximum value, and the number of runs of length 5 must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this address (0x34) is used as TRNG0_SCRxC5L only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this address is used as TRNG0_SCRxC5C readback register.

49.1.3.3.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					RUN5_RNG										
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					RUN5_MAX										
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1

49.1.3.3.24.4 Fields

Field	Function
31-27 —	Reserved. Always 0.
26-16 RUN5_RNG	Run Length 5 Range. The number of runs of length 5 (for both 0 and 1) detected during entropy generation must be greater than RUN5_MAX - RUN5_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15-11	Reserved. Always 0.

Table continues on the next page...

Field	Function
—	
10-0 RUN5_MAX	Run Length 5 Maximum Limit. Defines the maximum allowable runs of length 5 (for both 0 and 1) detected during entropy generation. The number of runs of length 5 detected during entropy generation must be less than RUN5_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.25 TRNG0 Statistical Check Run Length 6+ Count (TRNG0_SCR6PC)

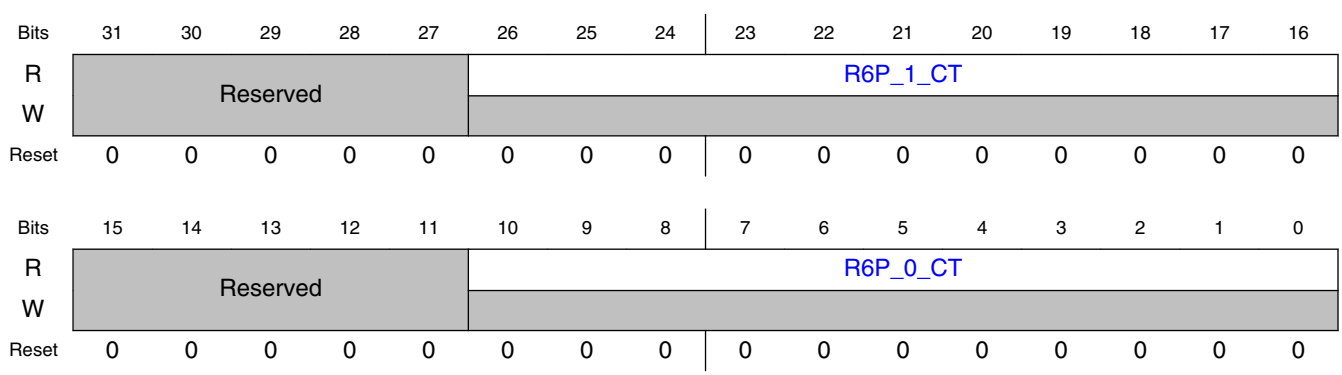
49.1.3.3.25.1 Address

Register	Offset	Description
TRNG0_SCR6PC	400A5038h	Accessible at this address when TRNG0_MCTL[PRGM] = 0]

49.1.3.3.25.2 Function

The TRNG0 Statistical Check Run Length 6+ Counters Register is a read-only register used to read the final Run Length 6+ counts after entropy generation. These counters start with the value in TRNG0_SCRxC6PL[RUN6P_MAX]. The R6P_1_CT decrements each time six or more consecutive ones are sampled (preceded by a zero and followed by a zero). The R6P_0_CT decrements each time six or more consecutive zeroes are sampled (preceded by a one and followed by a one). Note that this offset (0x38) is used as TRNG0_SCRxC6PL if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC6PC readback register, as described here.

49.1.3.3.25.3 Diagram



49.1.3.3.25.4 Fields

Field	Function
31-27 —	Reserved. Always 0.
26-16 R6P_1_CT	Runs of One, Length 6+ Count. Reads the final Runs of Ones, length 6+ count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.
15-11 —	Reserved. Always 0.
10-0 R6P_0_CT	Runs of Zero, Length 6+ Count. Reads the final Runs of Ones, length 6+ count after entropy generation. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.26 TRNG0 Statistical Check Run Length 6+ Limit (TRNG0_SCR6PL)

49.1.3.3.26.1 Address

Register	Offset	Description
TRNG0_SCR6PL	400A5038h	Accessible at this address when TRNG0_MCTL[PRGM] = 1]

49.1.3.3.26.2 Function

The TRNG0 Statistical Check Run Length 6+ Limit Register defines the allowable maximum and minimum number of runs of length 6 or more detected during entropy generation. To pass the test, the number of runs of length 6 or more (for samples of both 0 and 1) must be less than the programmed maximum value, and the number of runs of length 6 or more must be greater than (maximum - range). If this test fails, the Retry Counter in TRNG0_SCMISC will be decremented, and a retry will occur if the Retry Count has not reached zero. If the Retry Count has reached zero, an error will be generated. Note that this offset (0x38) is used as TRNG0_SCRxC6PL only if TRNG0_MCTL[PRGM] is 1. If TRNG0_MCTL[PRGM] is 0, this offset is used as TRNG0_SCRxC6PC readback register.

49.1.3.3.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					RUN6P_RNG										
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					RUN6P_MAX										
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1

49.1.3.3.26.4 Fields

Field	Function
31-27 —	Reserved. Always 0.
26-16 RUN6P_RNG	Run Length 6+ Range. The number of runs of length 6 or more (for both 0 and 1) detected during entropy generation must be greater than RUN6P_MAX - RUN6P_RNG, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.
15-11 —	Reserved. Always 0.
10-0 RUN6P_MAX	Run Length 6+ Maximum Limit. Defines the maximum allowable runs of length 6 or more (for both 0 and 1) detected during entropy generation. The number of runs of length 6 or more detected during entropy generation must be less than RUN6P_MAX, else a retry or error will occur. This register is cleared to the default POR value by writing the TRNG0_MCTL[RST_DEF] bit to 1.

49.1.3.3.27 TRNG0 Status (TRNG0_STATUS)

49.1.3.3.27.1 Address

Register	Offset
TRNG0_STATUS	400A503Ch

49.1.3.3.27.2 Function

Various statistical tests are run as a normal part of the TRNG's entropy generation process. The least-significant 16 bits of the TRNG0_STATUS register reflect the result of each of these tests. The status of these bits will be valid when the TRNG has finished its entropy generation process. Software can determine when this occurs by polling the ENT_VAL bit in the TRNG0 Miscellaneous Control Register.

Note that there is a very small probability that a statistical test will fail even though the TRNG is operating properly. If this happens the TRNG will automatically retry the entire entropy generation process, including running all the statistical tests. The value in RETRY_CT is decremented each time an entropy generation retry occurs. If a statistical check fails when the retry count is nonzero, a retry is initiated. But if a statistical check fails when the retry count is zero, an error is generated by the RNG. By default RETRY_CT is initialized to 1, but software can increase the retry count by writing to the RTY_CT field in the TRNG0_SCMISC register.

All 0s will be returned if this register address is read while the RNG is in Program Mode (see PRGM field in TRNG0_MCTL register. If this register is read while the RNG is in Run Mode the value returned will be formatted as follows.

49.1.3.3.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												RETRY_CT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TFMB	TFP	TFLR	TFSB	TF6PBR1	TF6PBR0	TF5BR1	TF5BR0	TF4BR1	TF4BR0	TF3BR1	TF3BR0	TF2BR1	TF2BR0	TF1BR1	TF1BR0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.27.4 Fields

Field	Function
31-20 —	Reserved. Always 0.
19-16 RETRY_CT	RETRY COUNT. This represents the current number of entropy generation retries left before a statistical text failure will cause the RNG to generate an error condition.
15	Test Fail, Mono Bit. If TFMB=1, the Mono Bit Test has failed.

Table continues on the next page...

Field	Function
TFMB	
14 TFP	Test Fail, Poker. If TFP=1, the Poker Test has failed.
13 TFLR	Test Fail, Long Run. If TFLR=1, the Long Run Test has failed.
12 TFSB	Test Fail, Sparse Bit. If TFSB=1, the Sparse Bit Test has failed.
11 TF6PBR1	Test Fail, 6 Plus Bit Run, Sampling 1s. If TF6PBR1=1, the 6 Plus Bit Run, Sampling 1s Test has failed.
10 TF6PBR0	Test Fail, 6 Plus Bit Run, Sampling 0s. If TF6PBR0=1, the 6 Plus Bit Run, Sampling 0s Test has failed.
9 TF5BR1	Test Fail, 5-Bit Run, Sampling 1s. If TF5BR1=1, the 5-Bit Run, Sampling 1s Test has failed.
8 TF5BR0	Test Fail, 5-Bit Run, Sampling 0s. If TF5BR0=1, the 5-Bit Run, Sampling 0s Test has failed.
7 TF4BR1	Test Fail, 4-Bit Run, Sampling 1s. If TF4BR1=1, the 4-Bit Run, Sampling 1s Test has failed.
6 TF4BR0	Test Fail, 4-Bit Run, Sampling 0s. If TF4BR0=1, the 4-Bit Run, Sampling 0s Test has failed.
5 TF3BR1	Test Fail, 3-Bit Run, Sampling 1s. If TF3BR1=1, the 3-Bit Run, Sampling 1s Test has failed.
4 TF3BR0	Test Fail, 3-Bit Run, Sampling 0s. If TF3BR0=1, the 3-Bit Run, Sampling 0s Test has failed.
3 TF2BR1	Test Fail, 2-Bit Run, Sampling 1s. If TF2BR1=1, the 2-Bit Run, Sampling 1s Test has failed.
2 TF2BR0	Test Fail, 2-Bit Run, Sampling 0s. If TF2BR0=1, the 2-Bit Run, Sampling 0s Test has failed.
1 TF1BR1	Test Fail, 1-Bit Run, Sampling 1s. If TF1BR1=1, the 1-Bit Run, Sampling 1s Test has failed.
0 TF1BR0	Test Fail, 1-Bit Run, Sampling 0s. If TF1BR0=1, the 1-Bit Run, Sampling 0s Test has failed.

49.1.3.3.28 TRNG0 Entropy Read (TRNG0_ENTa)

49.1.3.3.28.1 Address

For a = 0 to 15:

Register	Offset	Description
TRNG0_ENTa	40h + (a × 4h)	Word a

49.1.3.3.28.2 Function

The RNG TRNG can be programmed to generate an entropy value that is readable via the SkyBlue bus. To do this, set the TRNG0_MCTL[TRNG_ACC] bit to 1. Once the entropy value has been generated, the TRNG0_MCTL[ENT_VAL] bit will be set to 1. At this point, TRNG0_ENT0 through TRNG0_ENT15 may be read to retrieve the 512-bit entropy value. Note that once TRNG0_ENT15 is read, the entropy value will be cleared and a new value will begin generation, so it is important that TRNG0_ENT15 be read last. These registers are readable only when TRNG0_MCTL[PRGM] = 0 (Run Mode), TRNG0_MCTL[TRNG_ACC] = 1 (TRNG access mode) and TRNG0_MCTL[ENT_VAL] = 1. After at most one (1) bus clock cycle of reading a valid TRNG0_ENT15 register value, reading any TRNG0_ENT0 through TRNG0_ENT15 register would return zeroes.

49.1.3.3.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.28.4 Fields

Field	Function
31-0 ENT	Entropy Value. Will be non-zero only if TRNG0_MCTL[PRGM] = 0 Run Mode) and TRNG0_MCTL[ENT_VAL] = 1 (Entropy Valid). The most significant bits of the entropy are read from the lowest offset, and the least significant bits are read from the highest offset. Note that reading the highest offset also clears the entire entropy value, and starts a new entropy generation.

49.1.3.3.29 TRNG0 Statistical Check Poker Count 1 and 0 (TRNG0_PKRCNT10)

49.1.3.3.29.1 Address

Register	Offset
TRNG0_PKRCNT10	400A5080h

49.1.3.3.29.2 Function

The TRNG0 Statistical Check Poker Count 1 and 0 Register is a read-only register used to read the final Poker test counts of 1h and 0h patterns. The Poker 0h Count increments each time a nibble of sample data is found to be 0h. The Poker 1h Count increments each time a nibble of sample data is found to be 1h. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.29.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_1_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_0_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.29.4 Fields

Field	Function
31-16 PKR_1_CT	Poker 1h Count. Total number of nibbles of sample data which were found to be 1h. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_0_CT	Poker 0h Count. Total number of nibbles of sample data which were found to be 0h. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.30 TRNG0 Statistical Check Poker Count 3 and 2 (TRNG0_PKRCNT32)

49.1.3.3.30.1 Address

Register	Offset
TRNG0_PKRCNT32	400A5084h

49.1.3.3.30.2 Function

The TRNG0 Statistical Check Poker Count 3 and 2 Register is a read-only register used to read the final Poker test counts of 3h and 2h patterns. The Poker 2h Count increments each time a nibble of sample data is found to be 2h. The Poker 3h Count increments each time a nibble of sample data is found to be 3h. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.30.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_3_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_2_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.30.4 Fields

Field	Function
31-16 PKR_3_CT	Poker 3h Count. Total number of nibbles of sample data which were found to be 3h. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_2_CT	Poker 2h Count. Total number of nibbles of sample data which were found to be 2h. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.31 TRNG0 Statistical Check Poker Count 5 and 4 (TRNG0_PKRCNT54)

49.1.3.3.31.1 Address

Register	Offset
TRNG0_PKRCNT54	400A5088h

49.1.3.3.31.2 Function

The TRNG0 Statistical Check Poker Count 5 and 4 Register is a read-only register used to read the final Poker test counts of 5h and 4h patterns. The Poker 4h Count increments each time a nibble of sample data is found to be 4h. The Poker 5h Count increments each time a nibble of sample data is found to be 5h. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.31.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_5_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_4_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.31.4 Fields

Field	Function
31-16 PKR_5_CT	Poker 5h Count. Total number of nibbles of sample data which were found to be 5h. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_4_CT	Poker 4h Count. Total number of nibbles of sample data which were found to be 4h. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.32 TRNG0 Statistical Check Poker Count 7 and 6 (TRNG0_PKRCNT76)

49.1.3.3.32.1 Address

Register	Offset
TRNG0_PKRCNT76	400A508Ch

49.1.3.3.32.2 Function

The TRNG0 Statistical Check Poker Count 7 and 6 Register is a read-only register used to read the final Poker test counts of 7h and 6h patterns. The Poker 6h Count increments each time a nibble of sample data is found to be 6h. The Poker 7h Count increments each time a nibble of sample data is found to be 7h. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.32.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_7_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_6_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.32.4 Fields

Field	Function
31-16 PKR_7_CT	Poker 7h Count. Total number of nibbles of sample data which were found to be 7h. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_6_CT	Poker 6h Count. Total number of nibbles of sample data which were found to be 6h. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.33 TRNG0 Statistical Check Poker Count 9 and 8 (TRNG0_PKRCNT98)

49.1.3.3.33.1 Address

Register	Offset
TRNG0_PKRCNT98	400A5090h

49.1.3.3.33.2 Function

The TRNG0 Statistical Check Poker Count 9 and 8 Register is a read-only register used to read the final Poker test counts of 9h and 8h patterns. The Poker 8h Count increments each time a nibble of sample data is found to be 8h. The Poker 9h Count increments each time a nibble of sample data is found to be 9h. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_9_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_8_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.33.4 Fields

Field	Function
31-16 PKR_9_CT	Poker 9h Count. Total number of nibbles of sample data which were found to be 9h. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_8_CT	Poker 8h Count. Total number of nibbles of sample data which were found to be 8h. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.34 TRNG0 Statistical Check Poker Count B and A (TRNG0_PKRCNTBA)

49.1.3.3.34.1 Address

Register	Offset
TRNG0_PKRCNTBA	400A5094h

49.1.3.3.34.2 Function

The TRNG0 Statistical Check Poker Count B and A Register is a read-only register used to read the final Poker test counts of Bh and Ah patterns. The Poker Ah Count increments each time a nibble of sample data is found to be Ah. The Poker Bh Count increments each time a nibble of sample data is found to be Bh. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.34.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_B_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_A_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.34.4 Fields

Field	Function
31-16 PKR_B_CT	Poker Bh Count. Total number of nibbles of sample data which were found to be Bh. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_A_CT	Poker Ah Count. Total number of nibbles of sample data which were found to be Ah. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.35 TRNG0 Statistical Check Poker Count D and C (TRNG0_PKRCNTDC)

49.1.3.3.35.1 Address

Register	Offset
TRNG0_PKRCNTDC	400A5098h

49.1.3.3.35.2 Function

The TRNG0 Statistical Check Poker Count D and C Register is a read-only register used to read the final Poker test counts of Dh and Ch patterns. The Poker Ch Count increments each time a nibble of sample data is found to be Ch. The Poker Dh Count increments each time a nibble of sample data is found to be Dh. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.35.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_D_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_C_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.35.4 Fields

Field	Function
31-16 PKR_D_CT	Poker Dh Count. Total number of nibbles of sample data which were found to be Dh. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_C_CT	Poker Ch Count. Total number of nibbles of sample data which were found to be Ch. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.36 TRNG0 Statistical Check Poker Count F and E (TRNG0_PKRCNTFE)

49.1.3.3.36.1 Address

Register	Offset
TRNG0_PKRCNTFE	400A509Ch

49.1.3.3.36.2 Function

The TRNG0 Statistical Check Poker Count F and E Register is a read-only register used to read the final Poker test counts of Fh and Eh patterns. The Poker Eh Count increments each time a nibble of sample data is found to be Eh. The Poker Fh Count increments each time a nibble of sample data is found to be Fh. Note that this register is readable only if TRNG0_MCTL[PRGM] is 0, otherwise zeroes will be read.

49.1.3.3.36.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PKR_F_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PKR_E_CT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.36.4 Fields

Field	Function
31-16 PKR_F_CT	Poker Fh Count. Total number of nibbles of sample data which were found to be Fh. Requires TRNG0_MCTL[PRGM] = 0.
15-0 PKR_E_CT	Poker Eh Count. Total number of nibbles of sample data which were found to be Eh. Requires TRNG0_MCTL[PRGM] = 0.

49.1.3.3.37 TRNG0 Security Configuration (TRNG0_SEC_CFG)

49.1.3.3.37.1 Address

Register	Offset
TRNG0_SEC_CFG	400A50A0h

49.1.3.3.37.2 Function

The TRNG0 Security Configuration Register is a read/write register used to control the test mode, programmability and state modes of the TRNG0. Many bits are place holders for this version. More configurability will be added here. Clears on asynchronous reset. For TRNG0 releases before 2014/July/01, offsets 0xA0 to 0xAC used to be 0xB0 to 0xBC respectively. So, update newer tests that use these registers, if hard coded.

49.1.3.3.37.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													SK_VAL	NO_PRGM	SH0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.37.4 Fields

Field	Function
31-3 —	Reserved.
2 SK_VAL	Reserved. DRNG-specific, not applicable to this version. 0 - See DRNG version. 1 - See DRNG version.
1 NO_PRGM	If set, the TRNG registers cannot be programmed. That is, regardless of the TRNG access mode in the TRNG0 Miscellaneous Control Register. 0 - Programability of registers controlled only by the TRNG0 Miscellaneous Control Register's access mode bit. 1 - Overides TRNG0 Miscellaneous Control Register access mode and prevents TRNG register programming.
0 SH0	Reserved. DRNG specific, not applicable to this version. 0 - See DRNG version. 1 - See DRNG version.

49.1.3.3.38 TRNG0 Interrupt Control (TRNG0_INT_CTRL)

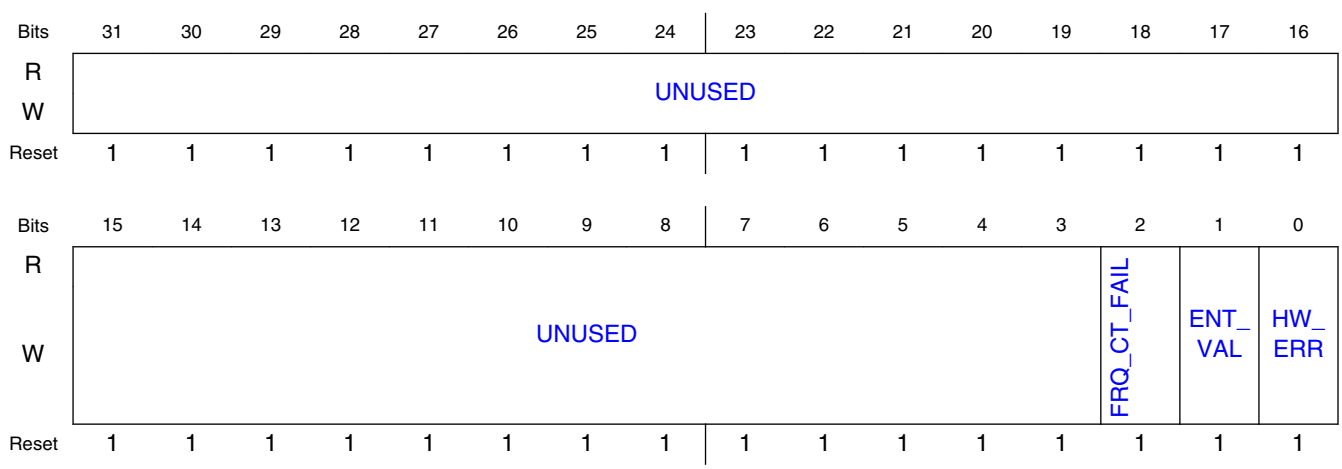
49.1.3.3.38.1 Address

Register	Offset
TRNG0_INT_CTRL	400A50A4h

49.1.3.3.38.2 Function

The TRNG0 Interrupt Control Register is a read/write register used to control the status for the (currently) three important interrupts that are generated by the TRNG. See TRNG0_INT_STATUS register description above. Each interrupt can be cleared by de-asserting the corresponding bit in the TRNG0_INT_CTRL register. Only a new interrupt will reassert the corresponding bit in the status register. Even if the interrupt is cleared or masked, interrupt status information can be read from the TRNG0_MCTL register.

49.1.3.3.38.3 Diagram



49.1.3.3.38.4 Fields

Field	Function
31-3 UNUSED	Reserved but writeable.
2 FRQ_CT_FAIL	Same behavior as bit 0 above. 0 - Same behavior as bit 0 above.

Table continues on the next page...

Field	Function
	1 - Same behavior as bit 0 above.
1 ENT_VAL	Same behavior as bit 0 above. 0 - Same behavior as bit 0 above. 1 - Same behavior as bit 0 above.
0 HW_ERR	Bit position that can be cleared if corresponding bit of TRNG0_INT_STATUS has been asserted. 0 - Corresponding bit of TRNG0_INT_STATUS cleared. 1 - Corresponding bit of TRNG0_INT_STATUS active.

49.1.3.3.39 TRNG0 Mask (TRNG0_INT_MASK)

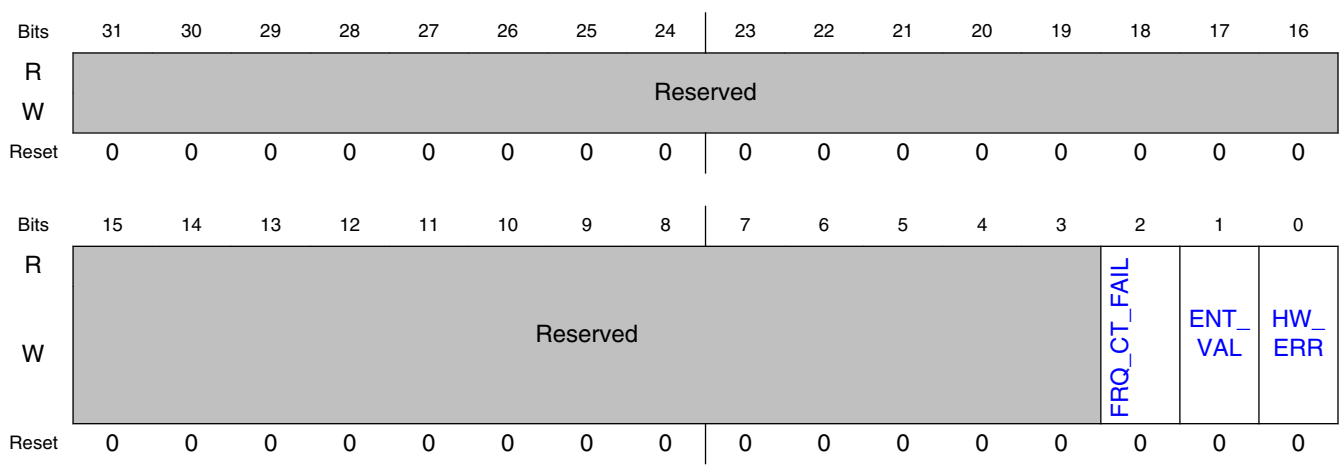
49.1.3.3.39.1 Address

Register	Offset
TRNG0_INT_MASK	400A50A8h

49.1.3.3.39.2 Function

The TRNG0 Interrupt Mask Register is a read/write register used to disable/mask the status reporting of the (currently) three important interrupts that are generated by the TRNG. See TRNG0_INT_STATUS register description above. Each interrupt can be masked/disabled by de-asserting the corresponding bit in the TRNG0_INT_MASK register. Only setting this bit high will re-enable the interrupt in the status register. Even if the interrupt is cleared or masked, interrupt status information can be read from the TRNG0_MCTL register.

49.1.3.3.39.3 Diagram



49.1.3.3.39.4 Fields

Field	Function
31-3 —	Reserved.
2 FRQ_CT_FAIL	Same behavior as bit 0 above. 0 - Same behavior as bit 0 above. 1 - Same behavior as bit 0 above.
1 ENT_VAL	Same behavior as bit 0 above. 0 - Same behavior as bit 0 above. 1 - Same behavior as bit 0 above.
0 HW_ERR	Bit position that can be cleared if corresponding bit of TRNG0_INT_STATUS has been asserted. 0 - Corresponding interrupt of TRNG0_INT_STATUS is masked. 1 - Corresponding bit of TRNG0_INT_STATUS is active.

49.1.3.3.40 TRNG0 Interrupt Status (TRNG0_INT_STATUS)

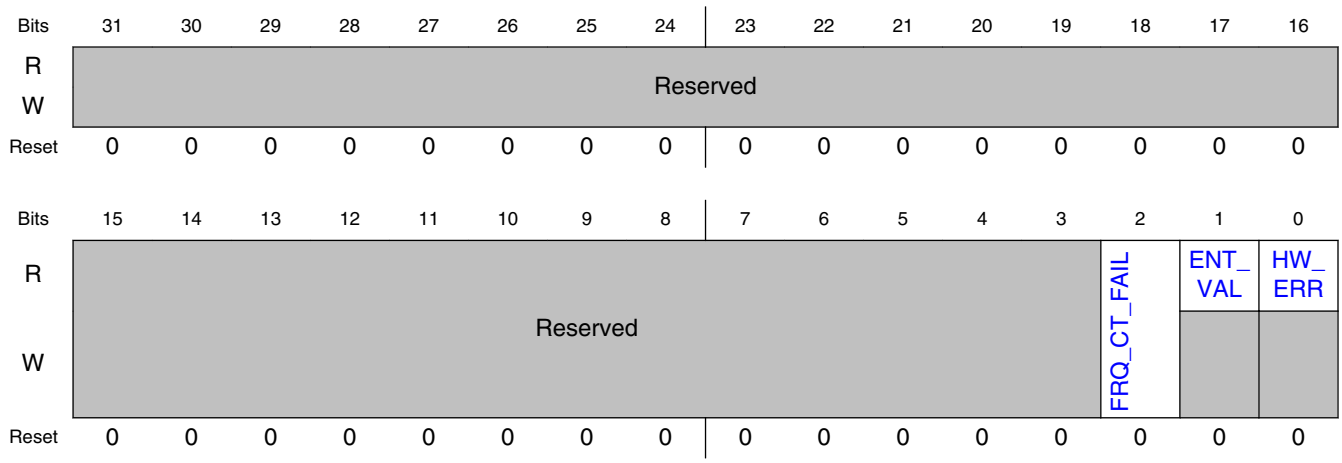
49.1.3.3.40.1 Address

Register	Offset
TRNG0_INT_STATUS	400A50ACh

49.1.3.3.40.2 Function

The TRNG0 Interrupt Status Register is a read register used to control and provide status for the (currently) three important interrupts that are generated by the TRNG. The ipi_rng_int_b interrupt signals that TRNG0 has either generated a Frequency Count Fail, Entropy Valid or Error Interrupt. The cause of the interrupt can be decoded by checking the least significant bits of the TRNG0_INT_STATUS register. Each interrupt can be temporarily cleared by de-asserting the corresponding bit in the TRNG0_INT_CTRL register. To mask the interrupts, clear the corresponding bits in the TRNG0_INT_MASK register. The description of each of the 3 interrupts is defined in the Block Guide under the TRNG0_MCTL register description. Even if the interrupt is cleared or masked, interrupt status information can be read from the TRNG0_MCTL register.

49.1.3.3.40.3 Diagram



49.1.3.3.40.4 Fields

Field	Function
31-3 —	Reserved.
2 FRQ_CT_FAIL	Read only: Frequency Count Fail. The frequency counter has detected a failure. This may be due to improper programming of the TRNG0_FRQMAX and/or TRNG0_FRQMIN registers, or a hardware failure in the ring oscillator. 0 - No hardware nor self test frequency errors. 1 - The frequency counter has detected a failure.
1 ENT_VAL	Read only: Entropy Valid. Will assert only if TRNG ACC bit is set, and then after an entropy value is generated. Will be cleared when TRNG0_ENT15 is read. (TRNG0_ENT0 through TRNG0_ENT14 should be read before reading TRNG0_ENT15). 0 - Busy generation entropy. Any value read is invalid. 1 - TRNG can be stopped and entropy is valid if read.
0 HW_ERR	Read: Error status. 1 = error detected. 0 = no error. Any HW error in the TRNG will trigger this interrupt. 0 - no error 1 - error detected.

49.1.3.3.41 TRNG0 Version ID (MS) (TRNG0_VID1)

49.1.3.3.41.1 Address

Register	Offset
TRNG0_VID1	400A50F0h

49.1.3.3.41.2 Function

The TRNG0 Version ID Register is a read only register used to identify the version of the TRNG in use. This register as well as TRNG0_VID2 should both be read to verify the expected version.

49.1.3.3.41.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRNG0_IP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRNG0_MAJ_REV								TRNG0_MIN_REV							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

49.1.3.3.41.4 Fields

Field	Function
31-16 TRNG0_IP_ID	Shows the IP ID. 0000000000110000 - ID for TRNG.
15-8 TRNG0_MAJ_REV	Shows the IP's Major revision of the TRNG. 00000001 - Major revision number for TRNG.
7-0 TRNG0_MIN_REV	Shows the IP's Minor revision of the TRNG. 00000000 - Minor revision number for TRNG.

49.1.3.3.42 TRNG0 Version ID (LS) (TRNG0_VID2)

49.1.3.3.42.1 Address

Register	Offset
TRNG0_VID2	400A50F4h

49.1.3.3.42.2 Function

The TRNG0 Version ID Register LSB is a read only register used to identify the architecture of the TRNG in use. This register as well as TRNG0_VID1 should both be read to verify the expected version.

49.1.3.3.42.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRNG0_ERA								TRNG0_INTG_OPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRNG0_ECO_REV								TRNG0_CONFIG_OPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.1.3.3.42.4 Fields

Field	Function
31-24 TRNG0_ERA	Shows the compile options for the TRNG. 00000000 - COMPILE_OPT for TRNG.
23-16 TRNG0_INTG_OPT	Shows the integration options for the TRNG. 00000000 - INTG_OPT for TRNG.
15-8 TRNG0_ECO_REV	Shows the IP's ECO revision of the TRNG. 00000000 - TRNG_ECO_REV for TRNG.
7-0 TRNG0_CONFIG_OPT	Shows the IP's Configuaration options for the TRNG. 00000000 - TRNG_CONFIG_OPT for TRNG.

49.1.4 Another TRNG usage example.

The TRNG can be used by a post processing pseudo-random number generator function. For example, TRNG can be used to seed a hardware or software based implementation of a DRBG defined by SP800-90.

Chapter 50
Touch Sensing Input (TSI)

50.1 Introduction

The touch sensing input (TSI) module provides capacitive touch sensing detection with high sensitivity and enhanced robustness.

Each TSI pin implements the capacitive measurement by a current source scan, charging and discharging the electrode, once or several times. A reference oscillator ticks the scan time and stores the result in a 16-bit register when the scan completes. Meanwhile, an interrupt request is submitted to CPU for post-processing if TSI interrupt is enabled and DMA function is not selected.The TSI module can be periodically triggered to work in low power mode with ultra-low current adder and wake CPU at the end of scan or the conversion result is out of the range specified by TSI threshold. It provides a solid capacitive measurement module to the implementation of touch keyboard, rotaries and sliders.

50.1.1 Features

TSI features includes:

- Support up to 16 external electrodes
- Automatic detection of electrode capacitance across all operational power modes
- Internal reference oscillator for high-accuracy measurement
- Configurable software or hardware scan trigger
- Fully support NXP touch sensing software (TSS) library, see www.nxp.com/touchsensing.
- Capability to wake MCU from low power modes
- Compensate for temperature and supply voltage variations
- High sensitivity change with 16-bit resolution register
- Configurable up to 4096 scan times.
- Support DMA data transfer
- The auxiliary noise detection mode supplies improved EMC immunity.

For electrode design recommendations, refer to AN3863: Designing Touch Sensing Electrodes

50.1.2 Modes of operation

This module supports the following operation modes.

Table 50-1. Operating modes

Mode	Description
Stop and low power stop	TSI module is fully functional in all of the stop modes as long as TSI_GENCS[STPE] is set. The channel specified by TSI_DATA[TSICH] will be scanned upon the trigger. After scan finishes, either end-of-scan or out-of-range interrupt can be selected to bring MCU out of low power modes.
Wait	TSI module is fully functional in this mode. When a scan completes, TSI submits an interrupt request to CPU if the interrupt is enabled.
Run	TSI module is fully functional in this mode. When a scan completes, TSI submits an interrupt request to CPU if the interrupt is enabled.

50.1.3 Block diagram

The following figure is a block diagram of the TSI module.

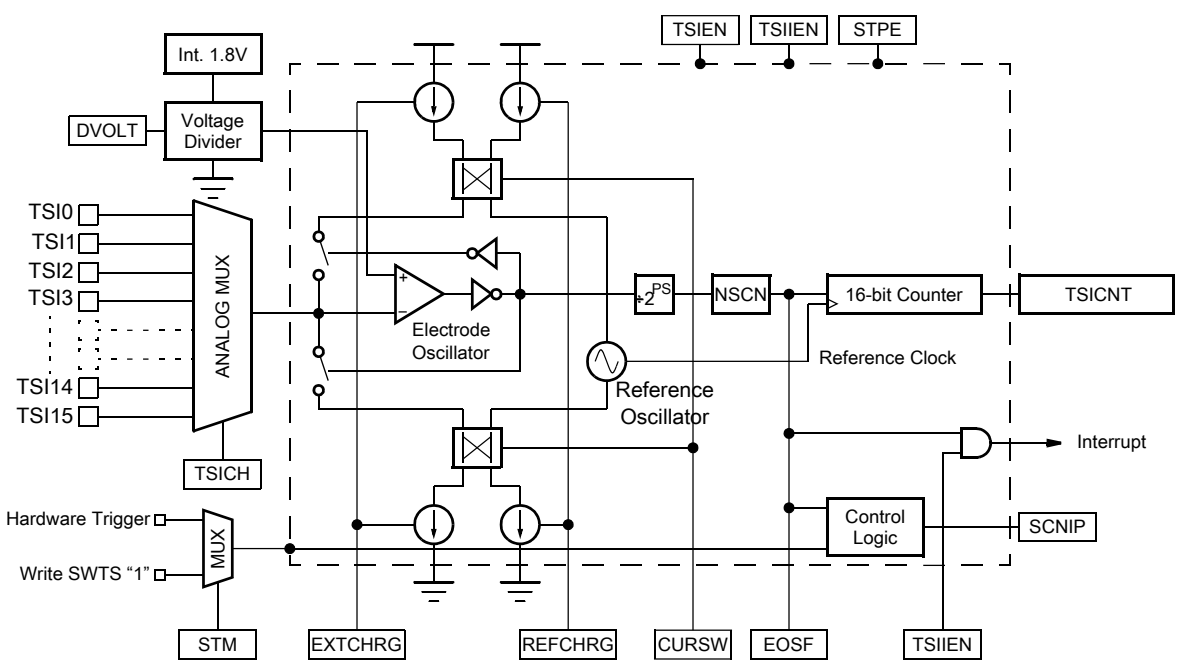


Figure 50-1. TSI module block diagram

50.2 External signal description

The TSI module contains up to 16 external pins for touch sensing. The table found here describes each of the TSI external pins.

Table 50-2. TSI signal description

Name	Port	Direction	Function	Reset state
TSI[15:0]	TSI	I/O	TSI capacitive pins. Switches driver that connects directly to the electrode pins TSI[15:0] can operate as GPIO pins.	I/O

50.2.1 TSI[15:0]

When TSI functionality is enabled , the TSI analog portion uses the corresponding channel to connect external on-board touch capacitors. The PCB connection between the pin and the touch pad must be kept as short as possible to reduce distribution capacity on board.

50.3 Register definition

This section describes the memory map and control/status registers for the TSI module.

TSI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_2000	TSI General Control and Status Register (TSI0_GENCS)	32	R/W	0000_0000h	50.3.1/1287
4006_2004	TSI DATA Register (TSI0_DATA)	32	R/W	0000_0000h	50.3.2/1292
4006_2008	TSI Threshold Register (TSI0_TSHD)	32	R/W	0000_0000h	50.3.3/1293

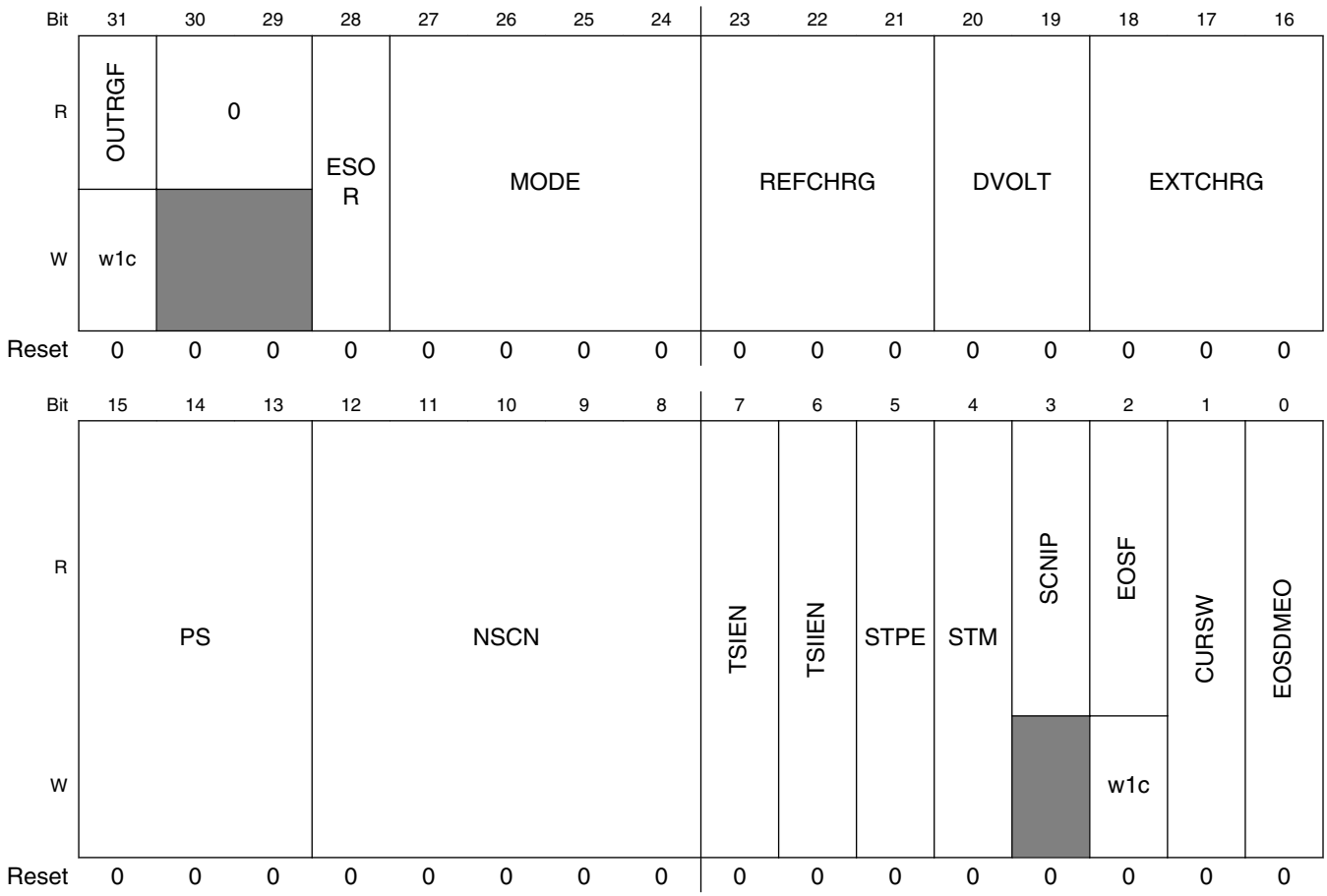
50.3.1 TSI General Control and Status Register (TSIx_GENCS)

This control register provides various control and configuration information for the TSI module.

NOTE

When TSI is working, the configuration bits (GENCS[TSIEN], GENCS[TSIIEN], and GENCS[STM]) must not be changed. The EOSF flag is kept until the software acknowledge it.

Address: 4006_2000h base + 0h offset = 4006_2000h



TSIx_GENCS field descriptions

Field	Description
31 OUTRGF	Out of Range Flag. This flag is set if the result register of the enabled electrode is out of the range defined by the TSI_THRESHOLD register. This flag is set only when TSI is configured in non-noise detection mode. It can be read once the CPU wakes. Write "1" , when this flag is set, to clear it.
30–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 ESOR	End-of-scan or Out-of-Range Interrupt Selection This bit is used to select out-of-range or end-of-scan event to generate an interrupt. 0 Out-of-range interrupt is allowed. 1 End-of-scan interrupt is allowed.

Table continues on the next page...

TSIx_GENCS field descriptions (continued)

Field	Description
27–24 MODE	TSI analog modes setup and status bits. Set up TSI analog modes, especially, setting MODE[3:2] to not 2'b00 will configure TSI to noise detection modes. MODE[1:0] take no effect on TSI operation mode and should always write to 2'b00 for setting up. When reading this field will return the analog status. Refer to chapter "Noise detection mode" for details. 0000 Set TSI in capacitive sensing(non-noise detection) mode. 0100 Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is disabled. 1000 Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is enabled to work in higher frequencies operations. 1100 Set TSI analog to work in automatic noise detection mode.
23–21 REFCHRG	REFCHRG These bits indicate the reference oscillator charge and discharge current value. 000 500 nA. 001 1 µA. 010 2 µA. 011 4 µA. 100 8 µA. 101 16 µA. 110 32 µA. 111 64 µA.
20–19 DVOLT	DVOLT These bits indicate the oscillator's voltage rails as below. 00 DV = 1.026 V; V_P = 1.328 V; V_m = 0.302 V. 01 DV = 0.592 V; V_P = 1.111 V; V_m = 0.519 V. 10 DV = 0.342 V; V_P = 0.986 V; V_m = 0.644 V. 11 DV = 0.197 V; V_P = 0.914 V; V_m = 0.716 V.
18–16 EXTCHRG	EXTCHRG These bits indicate the electrode oscillator charge and discharge current value. 000 500 nA. 001 1 µA. 010 2 µA. 011 4 µA. 100 8 µA. 101 16 µA. 110 32 µA. 111 64 µA.
15–13 PS	PS These bits indicate the prescaler of the output of electrode oscillator. 000 Electrode Oscillator Frequency divided by 1 001 Electrode Oscillator Frequency divided by 2

Table continues on the next page...

TSIx_GENCS field descriptions (continued)

Field	Description
	010 Electrode Oscillator Frequency divided by 4 011 Electrode Oscillator Frequency divided by 8 100 Electrode Oscillator Frequency divided by 16 101 Electrode Oscillator Frequency divided by 32 110 Electrode Oscillator Frequency divided by 64 111 Electrode Oscillator Frequency divided by 128
12–8 NSCN	NSCN These bits indicate the scan number for each electrode. The scan number is equal to NSCN + 1, which allows the scan time ranges from 1 to 32. By default, NSCN is configured as 0, which asserts the TSI scans once on the selected eletrode channel. 00000 Once per electrode 00001 Twice per electrode 00010 3 times per electrode 00011 4 times per electrode 00100 5 times per electrode 00101 6 times per electrode 00110 7 times per electrode 00111 8 times per electrode 01000 9 times per electrode 01001 10 times per electrode 01010 11 times per electrode 01011 12 times per electrode 01100 13 times per electrode 01101 14 times per electrode 01110 15 times per electrode 01111 16 times per electrode 10000 17 times per electrode 10001 18 times per electrode 10010 19 times per electrode 10011 20 times per electrode 10100 21 times per electrode 10101 22 times per electrode 10110 23 times per electrode 10111 24 times per electrode 11000 25 times per electrode 11001 26 times per electrode 11010 27 times per electrode 11011 28 times per electrode 11100 29 times per electrode 11101 30 times per electrode 11110 31 times per electrode 11111 32 times per electrode
7 TSIEN	Touch Sensing Input Module Enable This bit enables TSI module.

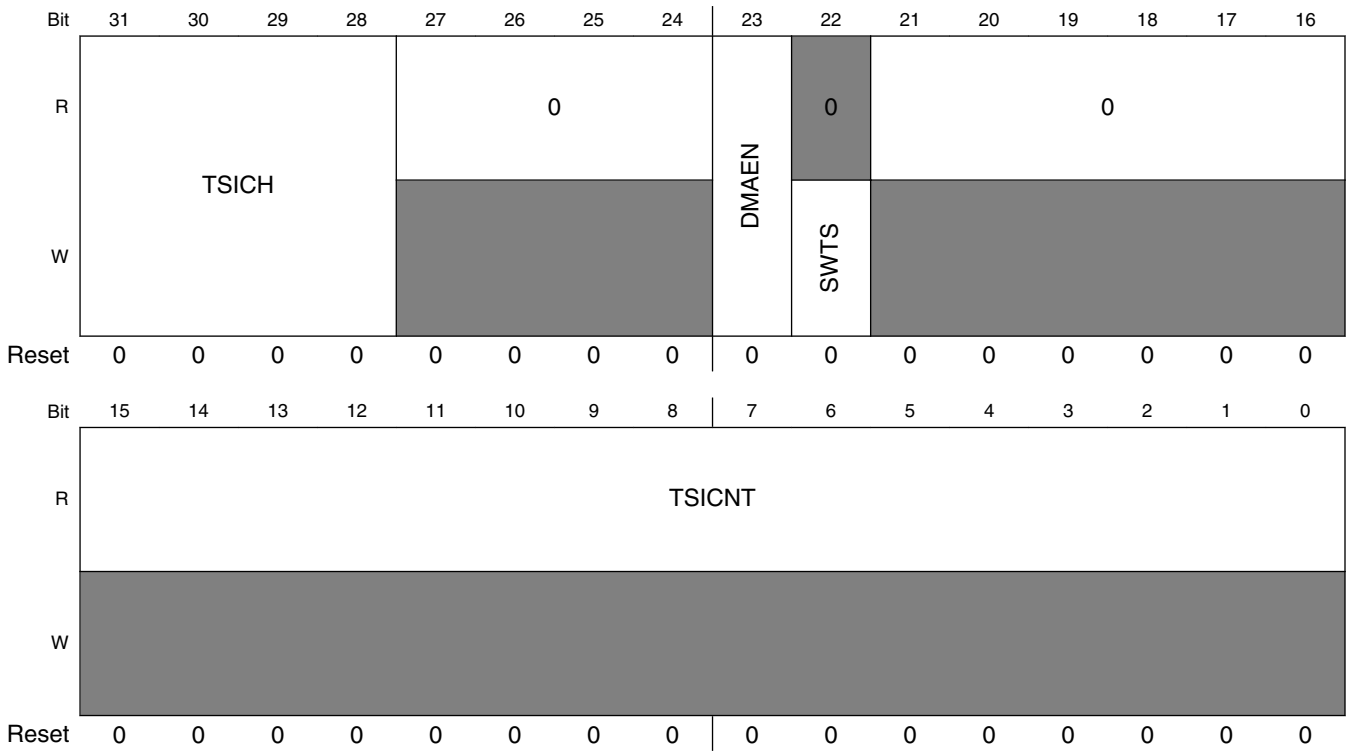
Table continues on the next page...

TSIx_GENCS field descriptions (continued)

Field	Description
	0 TSI module disabled. 1 TSI module enabled.
6 TSIIEN	Touch Sensing Input Interrupt Enable This bit enables TSI module interrupt request to CPU when the scan completes. The interrupt will wake MCU from low power mode if this interrupt is enabled. 0 TSI interrupt is disabled. 1 TSI interrupt is enabled.
5 STPE	TSI STOP Enable This bit enables TSI module function in low power modes (stop, VLPS, LLS and VLLS{3,2,1}). 0 TSI is disabled when MCU goes into low power mode. 1 Allows TSI to continue running in all low power modes.
4 STM	Scan Trigger Mode This bit specifies the trigger mode. User is allowed to change this bit when TSI is not working in progress. 0 Software trigger scan. 1 Hardware trigger scan.
3 SCNIP	Scan In Progress Status This read-only bit indicates if scan is in progress. This bit will get asserted after the analog bias circuit is stable after a trigger and it changes automatically by the TSI. 0 No scan in progress. 1 Scan in progress.
2 EOSF	End of Scan Flag This flag is set when all active electrodes are finished scanning after a scan trigger. Write "1" , when this flag is set, to clear it. 0 Scan not complete. 1 Scan complete.
1 CURSW	CURSW This bit specifies if the current sources of electrode oscillator and reference oscillator are swapped. 0 The current source pair are not swapped. 1 The current source pair are swapped.
0 EOSDMEO	End-of-Scan DMA Transfer Request Enable Only This bit makes simultaneous DMA request at End-of-Scan and Interrupt at Out-of-Range possible. EOSDMEO has precedence to ESOR when trying to set this bit and ESOR bit. When EOSDMEO = 1, End-of-Scan will generate DMA request and Out-of-Range will generate interrupt. 0 Do not enable the End-of-Scan DMA transfer request only. Depending on ESOR state, either Out-of-Range or End-of-Scan can trigger a DMA transfer request and interrupt. 1 Only the End-of-Scan event can trigger a DMA transfer request. The Out-of-Range event only and always triggers an interrupt if TSIIE is set.

50.3.2 TSI DATA Register (TSIx_DATA)

Address: 4006_2000h base + 4h offset = 4006_2004h



TSIx_DATA field descriptions

Field	Description
31–28 TSICH	TSICH These bits specify current channel to be measured. In hardware trigger mode (TSI_GENCS[STM] = 1), the scan will not start until the hardware trigger occurs. In software trigger mode (TSI_GENCS[STM] = 0), the scan starts immediately when TSI_DATA[SWTS] bit is written by 1. 0000 Channel 0. 0001 Channel 1. 0010 Channel 2. 0011 Channel 3. 0100 Channel 4. 0101 Channel 5. 0110 Channel 6. 0111 Channel 7. 1000 Channel 8. 1001 Channel 9. 1010 Channel 10. 1011 Channel 11. 1100 Channel 12. 1101 Channel 13.

Table continues on the next page...

TSIx_DATA field descriptions (continued)

Field	Description
	1110 Channel 14. 1111 Channel 15.
27–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 DMAEN	DMA Transfer Enabled This bit is used together with the TSI interrupt enable bits(TSIIE, ESOR) to generate a DMA transfer request instead of an interrupt. 0 Interrupt is selected when the interrupt enable bit is set and the corresponding TSI events assert. 1 DMA transfer request is selected when the interrupt enable bit is set and the corresponding TSI events assert.
22 SWTS	Software Trigger Start This write-only bit is a software start trigger. When STM bit is clear, write "1" to this bit will start a scan. The electrode channel to be scanned is determinated by TSI_DATA[TSICH] bits. 0 No effect. 1 Start a scan to determine which channel is specified by TSI_DATA[TSICH].
21–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TSICNT	TSI Conversion Counter Value These read-only bits record the accumulated scan counter value ticked by the reference oscillator.

50.3.3 TSI Threshold Register (TSIx_TSHD)

Address: 4006_2000h base + 8h offset = 4006_2008h

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

TSIx_TSHD field descriptions

Field	Description
31–16 THRESH	TSI Wakeup Channel High-threshold This half-word specifies the high threshold of the wakeup channel.
THRESL	TSI Wakeup Channel Low-threshold This half-word specifies the low threshold of the wakeup channel.

50.4 Functional description

50.4.1 Capacitance measurement

The electrode pin capacitance measurement uses a dual oscillator approach. The frequency of the TSI electrode oscillator depends on the external electrode capacitance and the TSI module configuration. After going to a configurable prescaler, the TSI electrode oscillator signal goes to the input of the module counter. The time for the module counter to reach its module value is measured using the TSI reference oscillator. The measured electrode capacitance is directly proportional to the time.

50.4.1.1 TSI electrode oscillator

The TSI electrode oscillator circuit is illustrated in the following figure. A configurable constant current source is used to charge and discharge the external electrode capacitance. A buffer hysteresis defines the oscillator delta voltage. The delta voltage defines the margin of high and low voltage which are the reference input of the comparator in different time.

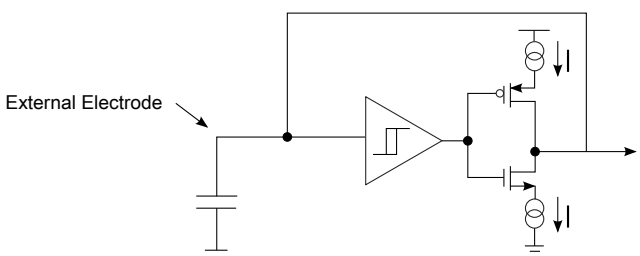


Figure 50-2. TSI electrode oscillator circuit

The current source applied to the pad capacitance is controlled by the GENCS[EXTCHRG]. The hysteresis delta voltage is defined in the module electrical specifications present in the device Data Sheet. The figure below shows the voltage amplitude waveform of the electrode capacitance charging and discharging with a programmable current.

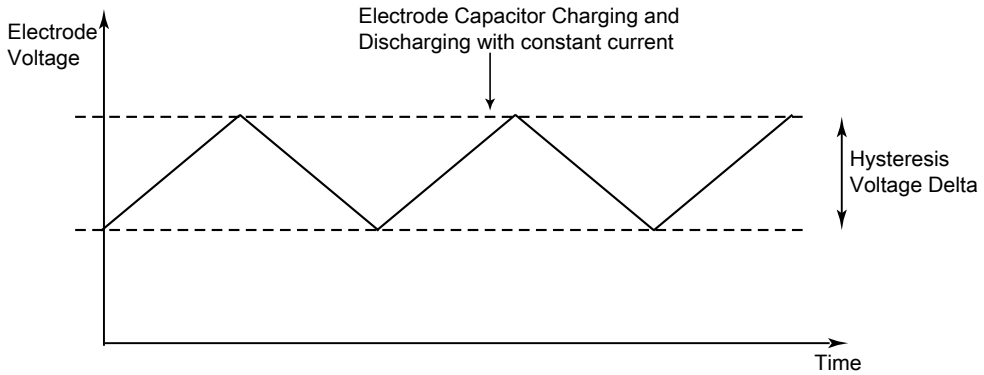


Figure 50-3. TSI electrode oscillator chart

The oscillator frequency is give by the following equation

$$F_{elec} = \frac{I}{2 * C_{elec} * \Delta V}$$

Equation 4. TSI electrode oscillator frequency

Where:

I: constant current

C_{elec}: electrode capacitance

ΔV: Hysteresis delta voltage

So by this equation, for example, an electrode with C_{elec}= 20 pF, with a current source of I = 16 μA and ΔV = 600 m V have the following oscillation frequency:

$$F_{elec} = \frac{16\ \mu A}{2 * 20pF * 600mV} = 0.67MHz$$

Equation 5. TSI electrode oscillator frequency

The current source is used to accommodate the TSI electrode oscillator frequency with different electrode capacitance sizes.

50.4.1.2 Electrode oscillator and counter module control

The TSI oscillator frequency signal goes through a prescaler defined by the GENCS[PS] and then enters in a modulus counter. GENCS[NSCN] defines the maximum count value of the modulus counter.

The pin capacitance sampling time is given by the time the module counter takes to go from 0 to its maximum value, defined by NSCN. The electrode sample time is expressed by the following equation:

$$T_{cap\_samp} = \frac{PS * NSCN}{F_{elec}}$$

Using Equation 1

$$T_{cap\_samp} = \frac{2 * PS * NSCN * C_{elec} * \Delta V}{I}$$

Equation 6. Electrode sampling time

Where:

PS: prescaler value

NSCN: module counter maximum value

I: constant current

C_{elec}: electrode capacitance

ΔV: Hysteresis delta voltage

By this equation we have that an electrode with C = 20 pF, with a current source of I = 16 µA and ΔV = 600 mV, PS = 2 and NSCN = 16 have the following sampling time:

$$T_{cap\_samp} = \frac{2*2*16*20pF*600mV}{16\mu A} = 48\mu s$$

50.4.1.3 TSI reference oscillator

The TSI reference oscillator has the same topology of the TSI electrode oscillator. The TSI reference oscillator instead of using an external capacitor for the electrode oscillator has an internal reference capacitor.

The TSI reference oscillator has an independent programmable current source controlled by GENCS[REFCHRG].

The reference oscillator frequency is given by the following equation:

$$F_{ref\_osc} = \frac{I_{ref}}{2 * C_{ref} * \Delta V}$$

Equation 7. TSI reference oscillator frequency

Where:

C_{ref}: Internal reference capacitor

I_{ref}: Reference oscillator current source

ΔV : Hysteresis delta voltage

Considering C_{ref} = 1.0 pF, I_{ref} = 12 µA and ΔV = 600 mV, follows

$$F_{ref\_osc} = \frac{12\mu A}{2 * 1.0pF * 600mV} = 10.0MHz$$

50.4.2 TSI measurement result

The capacitance measurement result is defined by the number of TSI reference oscillator periods during the sample time and is stored in the TSICHnCNT register.

TSICHnCNT = T_{cap_samp} * F_{ref_osc}

Using Equation 2 and Equation 1 follows:

$$TSICHnCNT = \frac{I_{ref} * PS * NSCN}{C_{ref} * I_{elec}} * C_{elec}$$

Equation 8. Capacitance result value

In the example where F_{ref_osc} = 10.0 MHz and T_{cap_samp} = 48 µs, TSICHnCNT = 480

50.4.3 Enable TSI module

The TSI module can be fully functional in run, wait and low power modes. The TSI_GENCS[TSIEN] bit must be set to enable the TSI module in run and wait mode. When TSI_GENCS[STPE] bit is set, it allows the TSI module to work in low power mode.

50.4.4 Software and hardware trigger

The TSI module allows a software or hardware trigger to start a scan. When a software trigger is applied (TSI_GENCS[STM] bit clear), the TSI_GENCS[SWTS] bit must be written "1" to start the scan electrode channel that is identified by TSI_DATA[TSICH]. When a hardware trigger is applied (TSI_GENCS[STM] bit set), the TSI will not start scanning until the hardware trigger arrives. The hardware trigger is different depending on the MCU configuration. Generally, it could be an event that RTC overflows. See chip configuration section for details.

50.4.5 Scan times

The TSI provides multi-scan function. The number of scans is indicated by TSI_GENCS[NSCN] that allow the scan number from 1 to 32. When TSI_GENCS[NSCN] is set to 0 (only once), the single scan is engaged. The 16-bit counter accumulates all scan results until the NSCN time scan completes, and users can read TSI_DATA[TSICNT] to get this accumulation. When DMA transfer is enabled, the counter values can also be read out by DMA engine.

50.4.6 Clock setting

TSI is built with dual oscillator architecture. In normal sensing application, the reference oscillator clock is the only clock source for operations. The reference clock is used to measure the electrode oscillator by ticking a 16-bit counter. The reference oscillator frequency depends on the current source setting. Please refer to the Current source for more details.

The output of electrode oscillator has several prescalers up to 128 indicated by TSI_GENCS[PS]. This allows a flexible counter configuration for different electrode oscillator frequency.

50.4.7 Reference voltage

The TSI module offers a internal reference voltage for both electrode oscillator and reference oscillator. The internal reference voltage can work in low power modes even when the MCU regulator is partially powered down, which is ideally for low-power touch detection.

The charge and discharge difference voltage is configurable upon the setting of TSI_GENCS[DVOLT]. The following table shows the all the delta voltage configurations.

NOTE

This table doesn't apply to noise mode, see noise mode sections for its configuration.

Table 50-3. Delta voltage configuration

DVOLT	V_p (V)	V_m (V)	ΔV (V)
00	1.328	0.302	1.026

Table continues on the next page...

Table 50-3. Delta voltage configuration (continued)

DVOLT	V_p (V)	V_m (V)	ΔV (V)
01	1.111	0.519	0.592
10	0.986	0.644	0.342
11	0.914	0.716	0.198

50.4.8 Current source

The TSI module supports eight different current source power to increment from 500 nA to 64 μA. TSI_GENCS[EXTCHRG] determines the current of electrode oscillator that charges and discharges external electrodes. The TSI_GENCS[REFCHRG] determines the current of reference oscillator on which the internal reference clock depends. The lower current source takes more time for charge and discharge, which is useful to detect high-accuracy change. The higher current source takes less time, which can be used to charge a big electrode by less power consumption.

TSI_GENCS[CURSW] allows the current source to swap, so that the reference oscillator and electrode oscillator use the opposite current sources. When TSI_GENCS[CURSW] is set and the current sources are swapped, TSI_GENCS[EXTCHRG] and TSI_GENCS[REFCHRG] still control the corresponding current sources, that is, TSI_GENCS[EXTCHRG] controls the reference oscillator current and TSI_GENCS[REFCHRG] controls the electrode oscillator current.

50.4.9 End of scan

As a scan starts, [SCNIP] bit is set to indicate scan is in progress. When the scan completes, the [EOSF] bit is set. Before clearing the [EOSF] bit, the value in TSI_DATA[TSICNT] must be read. If the TSI_GENCS[TSIIEN] and TSI_GENCS[ESOR] are set andTSI_GENCS[DMAEN]is not set, an interrupt is submitted to CPU for post-processing immediately. The interrupt is also optional to wake MCU to execute ISR if it is in low power mode. When DMA function is enabled by setting TSI_GENCS[TSIIEN] and TSI_GENCS[ESOR], as soon as scan completes, a DMA transfer request is asserted to DMA controller for data movement, generally, DMA engine will fetch TSI conversion result from TSI_DATA register,store it to other memory space and then refresh the TSI scan channel index(TSI_DATA[TSICH]) for next loop. When DMA transfer is done, TSI_GENCS[EOSF] is cleared automatically.

50.4.10 Out-of-range interrupt

If enabled, TSI will scan the electrode specified by TSI_DATA[TSICH] as soon as the trigger arrives. The TSI_GENCS[OUTRGF] flag generates a TSI interrupt request if the TSI_GENCS[TSIIE] bit is set and GENCS[ESOR] bit is cleared. With this configuration, after the end-of-electrode scan, the electrode capacitance will be converted and stored to the result register TSI_DATA[TSICNT], the out-of-range interrupt is only requested if there is a considerable capacitance change defined by the TSI_TSHD. For instance, if in low power mode the electrode capacitance does not vary, the out-of-range interrupt does not interrupt the CPU. This interrupt will not happen in noise detection mode. It is worthy to note that when the counter value reaches 0xFFFF is treated as an extreme case the out-of-range will not happen. Also in noise detection mode, the out-of-range will not assert either.

50.4.11 Wake up MCU from low power modes

In low power modes, once enabled by TSI_GENCS[STPE] and TSI_GENCS[TSIIE], TSI can bring MCU out of its low power modes(STOP, VLPS, VLLS,etc) by either end of scan or out of range interrupt, that is, if TSI_GENCS[ESOR] is set, end of scan interrupt is selected and otherwise, out of range is selected.

50.4.12 DMA function support

Transmit by DMA is supported only when TSI_DATA[DMAEN] is set. A DMA transfer request is asserted when all the flags based on TSI_GENCS[ESOR] settings and TSI_GENCS[TSIIE] are set. Then the on-chip DMA controller detects this request and transfers data between memory space and TSI register space. After the data tranfer, DMA DONE is asserted to clear TSI_GENCS[EOSF] automatically. This function is normally used by DMA controller to get the conversion result from TSI_DATA[TSICNT] upon a end-of-scan event and then refresh the channel index(TSI_DATA[TSICH]) for next trigger.

50.4.13 Noise detection mode

The noise detection mode change the circuit configuration as shown in the following figure. With this configuration, it is possible to detect touch with high levels of EMC noise present. To enter the test mode, see TSI_GENCS[MODE] register field.

In noise detection mode the reference oscillator has the same configuration except the output goes to Counter2 and this counter will have its maximum count set by NSCNx2(PS). This means this oscillator will setup the noise detection mode sense duration.

The blocks of external oscillator is changed and instead of an oscillator the circuit implements an RF amplitude detection. The threshold for this amplitude detection is set by DVOLT register bits.

Also the external voltage is biased by vmid voltage with a Rs series resistance.

The vmid voltage is defined as V(vmid) = (V(vp) + V(vm))/2.

The Rs value is defined by TSI_GENCS[EXTCHRG] register bits.

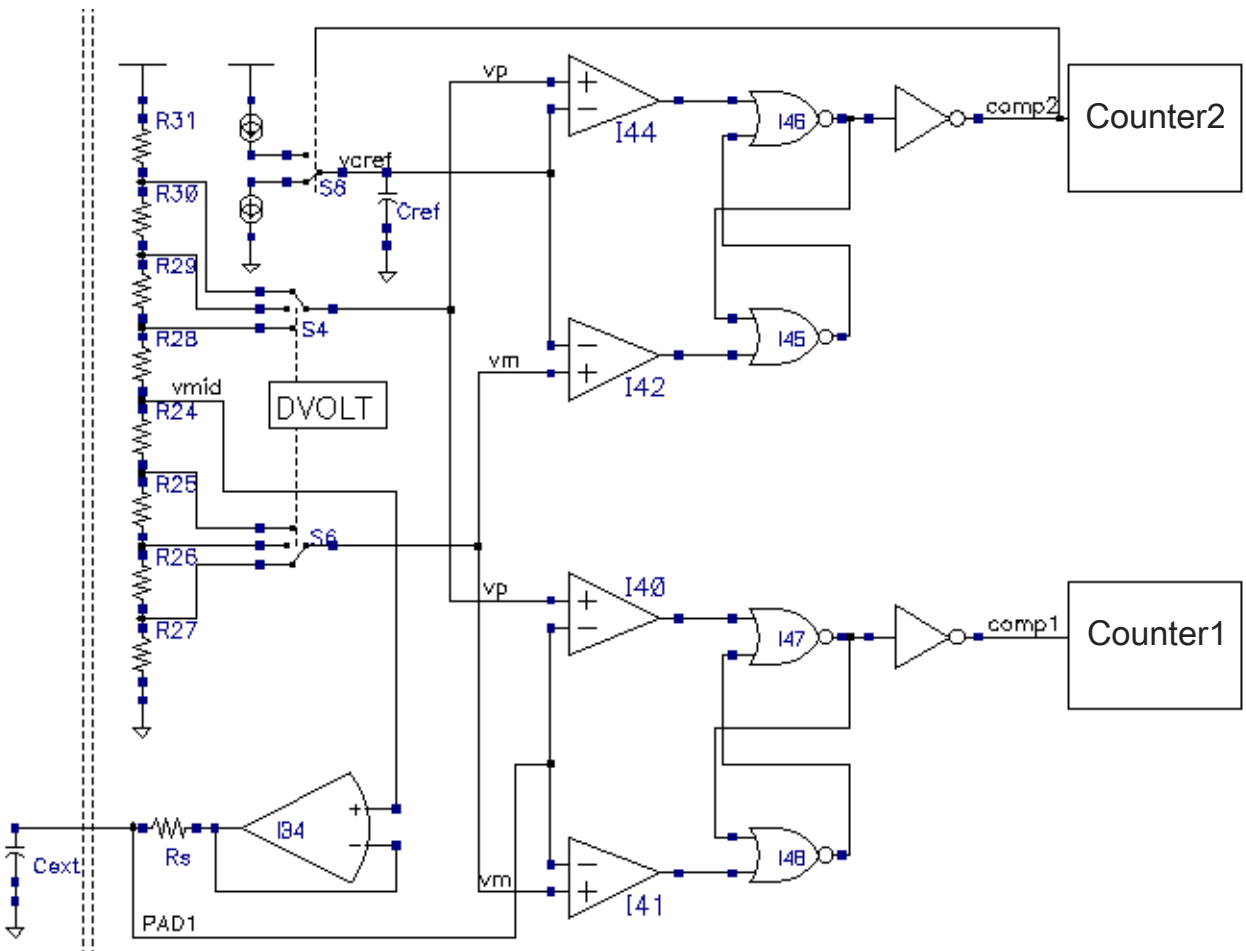


Figure 50-4. TSI circuit in noise detection mode

To determine the noise level the below algorithm can be used:

1. Initialize Rs = maxrs; Dvolt = minDv (set other configurations also)

2. Perform a noise cycle.

3. If TSIcounter < 3, go to step 8

4. If Rs = minrs, go to step 6.

5. Reduce value of Rs. go to step 2

6. If Dvolt = maxDv, go to END

7. Increase value of Dvolt. Set Rs = maxrs. go to step 2

8. If Rs > minrs, (Reduce value of Rs, go to END)

9. Rs = maxrs, reduce value of Dvolt.

10. END Get value of Rs and Dvolt.

One example of noise detection mode is shown in the following figure. in this figure the TSI is working in capacitive mode until 28 μs (T1) when it is changed to noise detection mode. In noise detection mode the selected pad is biased with 0.815V and all AC waveform in this pad is caused by a noise source external to the MCU.

It is possible to observe in the following figure that, in noise detection mode, the clkref output has the peak detection and the number of detected peaks can be counted or used by digital block. The clkext output has the internal oscillator output and can be used to set the maximum noise detection time window.

The waveform of the following figure shows two operations during noise detection mode:

- The V(vp) and V(vm) thresholds are changed in 34.4 μs (T2).
- The Rs series resistance value is changed between 184 kΩ (TSI_GENCS[EXTCHRG]=011b), 77 kΩ (TSI_GENCS[EXTCHRG]=100) and 32 kΩ(TSI_GENCS[EXTCHRG] = 101). Because of this Rs change the amplitude of noise waveform change also.

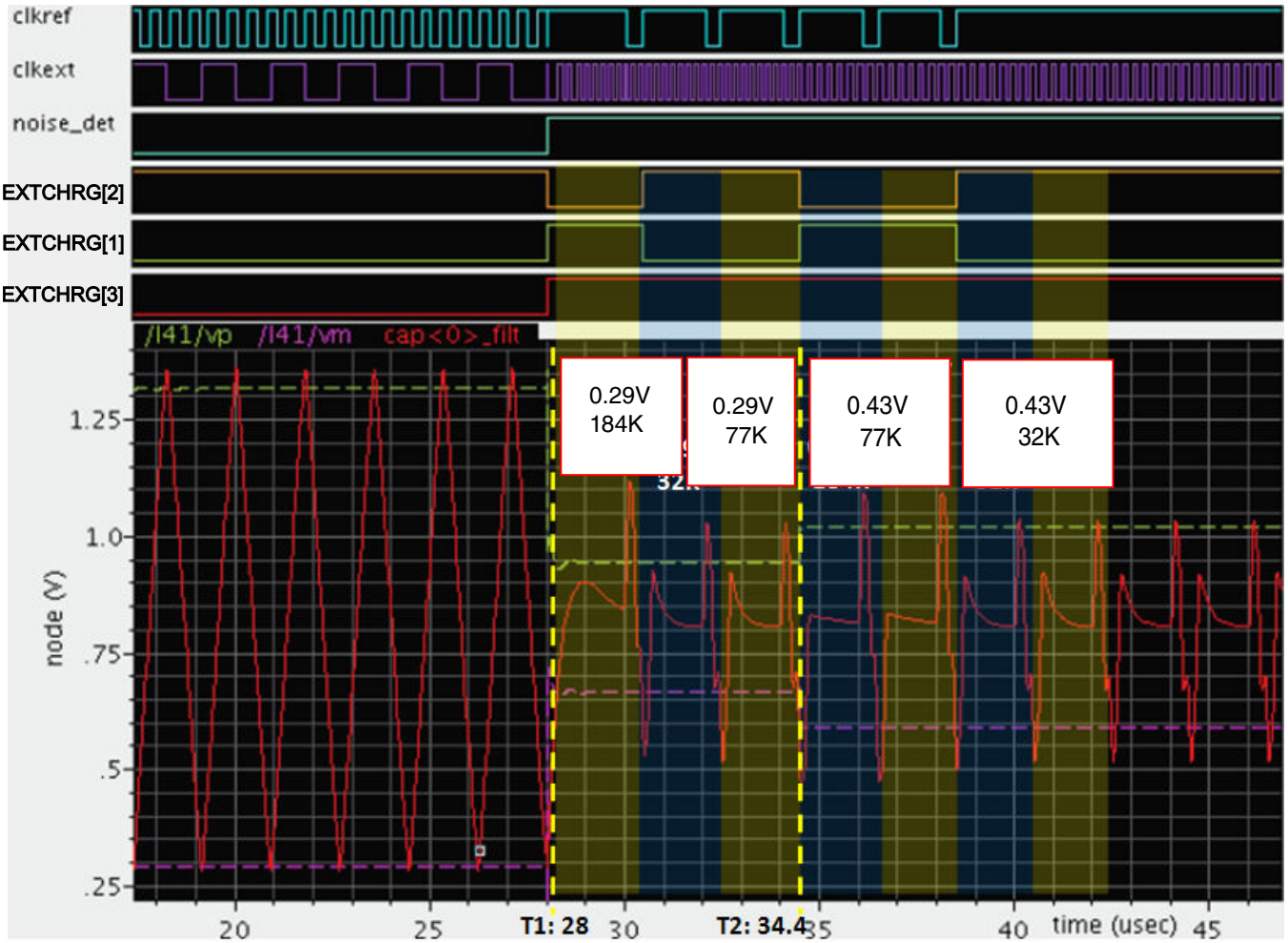


Figure 50-5. TSI noise detection mode waveform

50.4.13.1 Automatic noise detection modes

This mode is setup by MODE[3:2] to 2'b11 (noise mode 3). In this mode the thresholds are incremented internally by the module until the point that there is no noise voltage trepassing the threshold.

The following diagram shows how it is done. The threshold comparator output goes to a counter and as the DVOLT control bits are increased the DVOLT thresholds are increased as well. The four bits are counted until 1111 (=15) and the counter is stop with this maximum value.

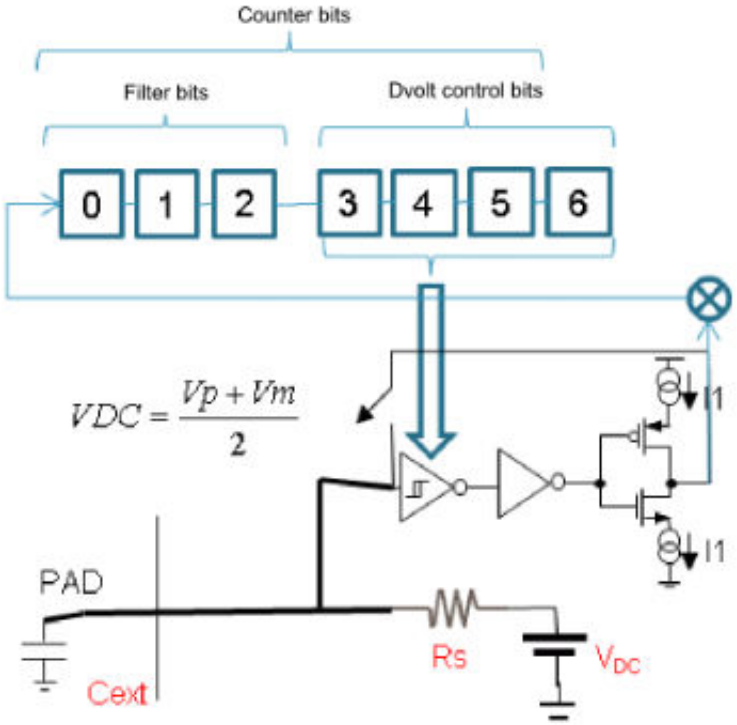


Figure 50-6. Block diagram automatic noise threshold operation

The signals that have different behavior in this noise mode (wrt capacitive mode) are shown in the following table.

Table 50-4. Signal properties in automatic noise operation mode

Name	Function	I/O type	Power Up/Reset state
EXTCHRG[0]	In this operation mode this bits selects the series resistance. 0 - uses Rs=32 kΩ 1 - uses Rs=187 kΩ Independent of this bit selection the threshold	I	0
DVOLT[1:0]	Select voltage rails of the internal oscillator	I	00
MODE[3:0]	DVOLT counter bits output. This register content is reset always the MODE[3:2] is not 11, and after beginning of automatic noise mode operatio	O	0000

50.4.13.2 Single threshold noise modes

These modes reset by MODE[3:2]=01 and 10. The difference between these two modes is that in mode 2 (MODE[3:2]=10) there is a frequency limitation circuit that enables the circuit to operate in higher frequencies. In mode 1 (MODE[3:2]=01) this frequency limitation circuit is not enabled.

In this mode the thresholds are set by user via register bits as described in the following table.

During these modes the internal oscillator rails are set to the maximum (equivalent to DVOLT[1:0]=00).

Table 50-5. Signal properties in single noise modes (1,2)

Name	Function	I/O type	Power up / reset
MODE[3:2]	01 or 10- Single thrshold noise mode operation.	I	00
DVOLT[1:0], EXTCHRG[2:1]	In this operation mode these 4 bits are used select the noise threshold. 0000 - DVpm = 0.038 V, Vp = 0.834 V, Vm = 0.796 V 0001 - DVpm = 0.050 V, Vp = 0.830 V, Vm = 0.790 V 0010 - DVpm = 0.066 V, Vp = 0.848 V, Vm = 0.782 V 0011 - DVpm = 0.087 V, Vp = 0.858 V, Vm = 0.772 V 0100 - DVpm = 0.114 V, Vp = 0.872 V, Vm = 0.758 V 0101 - DVpm = 0.150 V, Vp = 0.890 V, Vm = 0.740 V 0110 - DVpm = 0.197 V, Vp = 0.914 V, Vm = 0.716 V 0111 - DVpm = 0.260 V, Vp = 0.945 V, Vm = 0.685 V 1000 - DVpm = 0.342 V, Vp = 0.986 V, Vm = 0.644 V 1001 - DVpm = 0.450 V, Vp = 1.040 V, Vm = 0.590 V 1010 - DVpm = 0.592 V, Vp = 1.111 V, Vm = 0.519 V 1011 - DVpm = 0.780 V, Vp = 1.205 V, Vm = 0.425 V 1100 - DVpm = 1.026 V, Vp = 1.328 V, Vm = 0.302 V	I	XXXX

Table continues on the next page...

Table 50-5. Signal properties in single noise modes (1,2) (continued)

Name	Function	I/O type	Power up / reset
	1101 - DVpm = 1.350 V, Vp = 1.490 V, Vm = 0.140 V 1110 - DVpm = 1.630 V, Vp = 1.630 V, Vm = 0 V 1111 - DVpm = 1.630 V, Vp = 1.630 V, Vm = 0 V		
EXTCHRG[0]	In this operation mode this bits selects the series resistance. 0 - uses Rs = 32 kΩ. 1- uses Rs = 187 kΩ. Independent of this bit selection the threshold 15 is done with Rs = 5.5 kΩ.	I	XX

Chapter 51
Time Stamp Timer Module (TSTMR)

51.1 Introduction

The Time Stamp Timer (TSTMR) is a 56-bit clock cycle counter, reset by system reset.

51.1.1 Features

- Free-running Time Stamp Timer

51.2 Memory map and register definition

The TSTMR module contains a 56-bit clock cycle counter, which is reset by system reset.

NOTE

The TSTMR registers can be read with 32-bit accesses only.

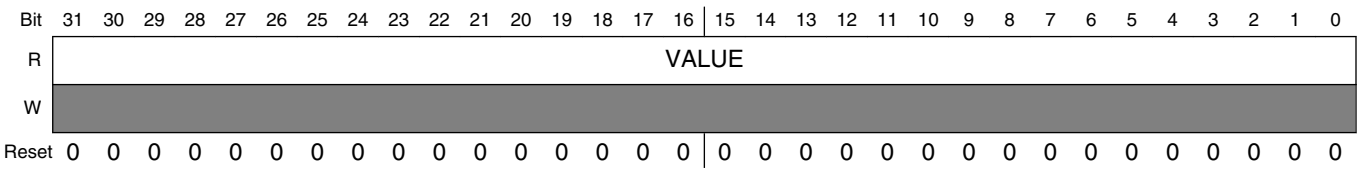
TSTMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_50F0	Time Stamp Timer Register Low (TSTMR0_L)	32	R	0000_0000h	51.2.1/1308
4007_50F4	Time Stamp Timer Register High (TSTMR0_H)	32	R	0000_0000h	51.2.2/1308

51.2.1 Time Stamp Timer Register Low (TSTMRx_L)

The Time Stamp Timer is a 56-bit clock cycle counter, reset by system reset. It is readable by way of the TSTMRH and TSTMRL registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Address: 4007_50F0h base + 0h offset = 4007_50F0h



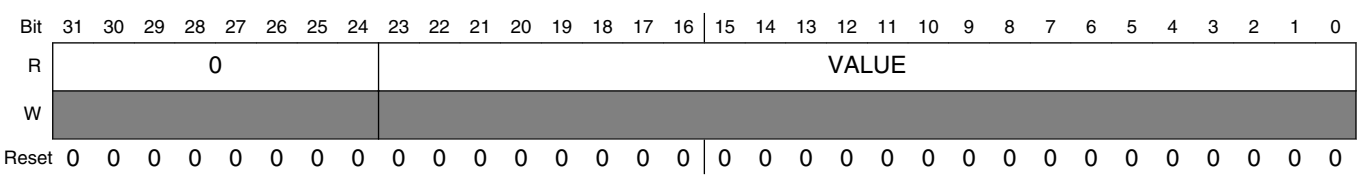
TSTMRx_L field descriptions

Field	Description
VALUE	Time Stamp Timer Low Lower 32 bits of the 56-bit time stamp value. The software must read the TSTMR_L first, followed by a read to TSTMR_H to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

51.2.2 Time Stamp Timer Register High (TSTMRx_H)

The Time Stamp Timer is a 56-bit counters clock cycle counter, reset by system reset. It is readable by way of the TSTMR_H and TSTMR_L registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Address: 4007_50F0h base + 4h offset = 4007_50F4h



TSTMRx_H field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
VALUE	Time Stamp Timer High

Table continues on the next page...

TSTMRx_H field descriptions (continued)

Field	Description
	Upper 24 bits of the 56-bit time stamp value.. The software must read the TSTMR_L first, followed by a read to TSTMR_H to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

51.3 Functional description

The TSTMR module is a free running incrementing counter that starts running after system reset de-assertion and can be read at any time by the software for determining the software ticks. However, the software must follow the read sequence as mentioned in the TSTMR register descriptions for correctly reading the TSTMR value. The TSTMR is a 56-bit counter and hence requires two 32-bit reads to read the full value. The TSTMR runs off the 1 MHz clock and resets on every system reset. The counter only stops when the clock to the TSTMR is disabled.

For the chip-specific implementation details of this module's instances, see the chip configuration chapter.

Chapter 52
Universal Serial Bus Full Speed OTG Controller (USBFSOTG)

52.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

This chapter describes the USB full speed OTG controller. The OTG implementation in this module provides limited host functionality and device solutions for implementing a USB 2.0 full-speed/low-speed compliant peripheral. The OTG logic implements features required by the *On-The-Go and Embedded Host Supplement to the USB 2.0 Specification* (usb.org, 2008). The USB full speed controller interfaces to a USBFS/LS transceiver.

NOTE

This chapter describes the following registers that have similar names: USB_OTGCTL, USB_CTL, USB_CTRL, and USB_CONTROL. These are all separate registers.

52.1.1 References

The following publications are referenced in this document. For copies or updates to these specifications, see the USB Implementers Forum, Inc. website at http://www.usb.org.

- *Universal Serial Bus Specification, Revision 2.0* , 2000, with amendments including those listed below
- *Errata for "USB Revision 2.0 April 27, 2000" as of 12/7/2000*
- *Errata for "USB Revision 2.0 April 27, 2000" as of May 28, 2002*

- *Pull-up / Pull-down Resistors* (USB Engineering Change Notice)
- *Suspend Current Limit Changes* (USB Engineering Change Notice)
- *Device Capacitance* (USB Engineering Change Notice)
- *USB 2.0 Connect Timing Update* (USB Engineering Change Notice as of April 4, 2013)
- *USB 2.0 VBUS Max Limit* (USB Engineering Change Notice)
- *On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification*, Revision 2.0 version 1.1a, July 27, 2012
- *Maximum VBUS Voltage* (USB OTGEH Engineering Change Notice)
- *Universal Serial Bus Micro-USB Cables and Connectors Specification*, Revision 1.01, 2007

52.1.2 USB—Overview

The USB is a cable bus that supports data exchange between a host computer and a wide range of simultaneously accessible peripherals. The attached peripherals share USB bandwidth through a host-scheduled, token-based protocol. The bus allows peripherals to be attached, configured, used, and detached while the host and other peripherals are in operation.

USB software provides a uniform view of the system for all application software, hiding implementation details to make application software more portable. It manages the dynamic attach and detach of peripherals.

There is only one host in any USB system. The USB interface to the host computer system is referred to as the Host Controller.

There may be multiple USB devices in any system such as human interface devices, speakers, printers, etc. USB devices present a standard USB interface in terms of comprehension, response, and standard capability.

The host initiates transactions to specific peripherals, whereas the device responds to control transactions. The device sends and receives data to and from the host using a standard USB data format. USB 2.0 full-speed /low-speed peripherals operate at 12Mbit/s or 1.5 Mbit/s.

For additional information, see the USB 2.0 specification.

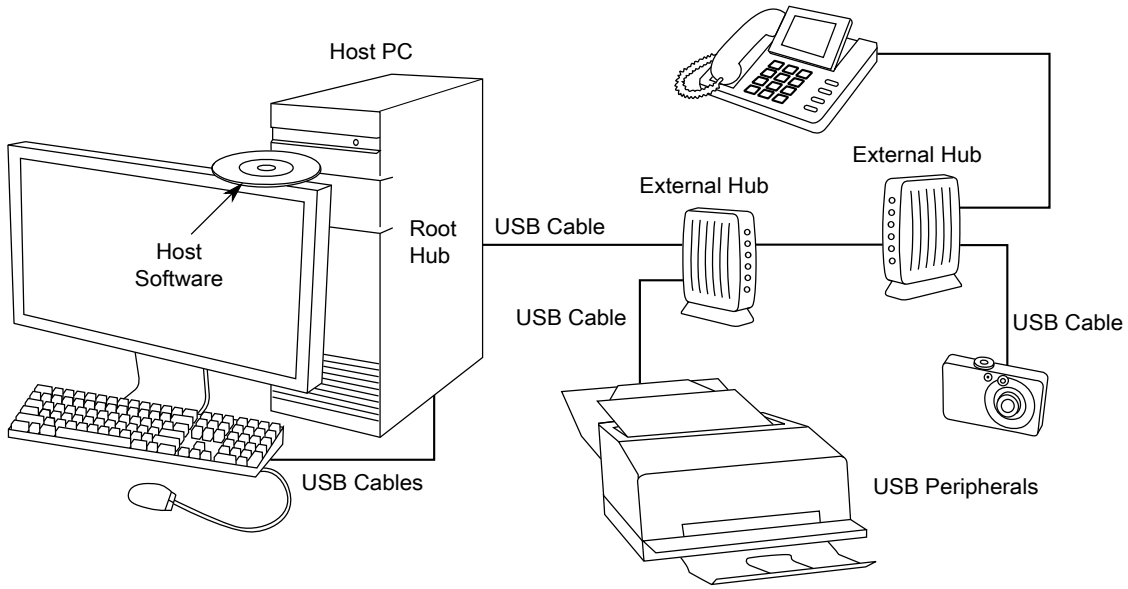


Figure 52-1. Example USB 2.0 system configuration

52.1.3 USB On-The-Go

USB is a popular standard for connecting peripherals and portable consumer electronic devices such as digital cameras and tablets to host PCs. The On-The-Go (OTG) Supplement to the USB Specification extends USB to peer-to-peer application. Using USB OTG technology, consumer electronics, peripherals, and portable devices can connect to each other to exchange data. For example, a digital camera can connect directly to a printer, or a keyboard can connect to a tablet to exchange data.

With the USB On-The-Go product, you can develop a fully USB-compliant peripheral device that can also assume the role of a USB host. Software determines the role of the device based on hardware signals, and then initializes the device in the appropriate mode of operation (host or peripheral) based on how it is connected. After connecting, the devices can negotiate using the OTG protocols to assume the role of host or peripheral based on the task to be accomplished.

For additional information, see the *On-The-Go and Embedded Host Supplement to the USB 2.0 Specification*.

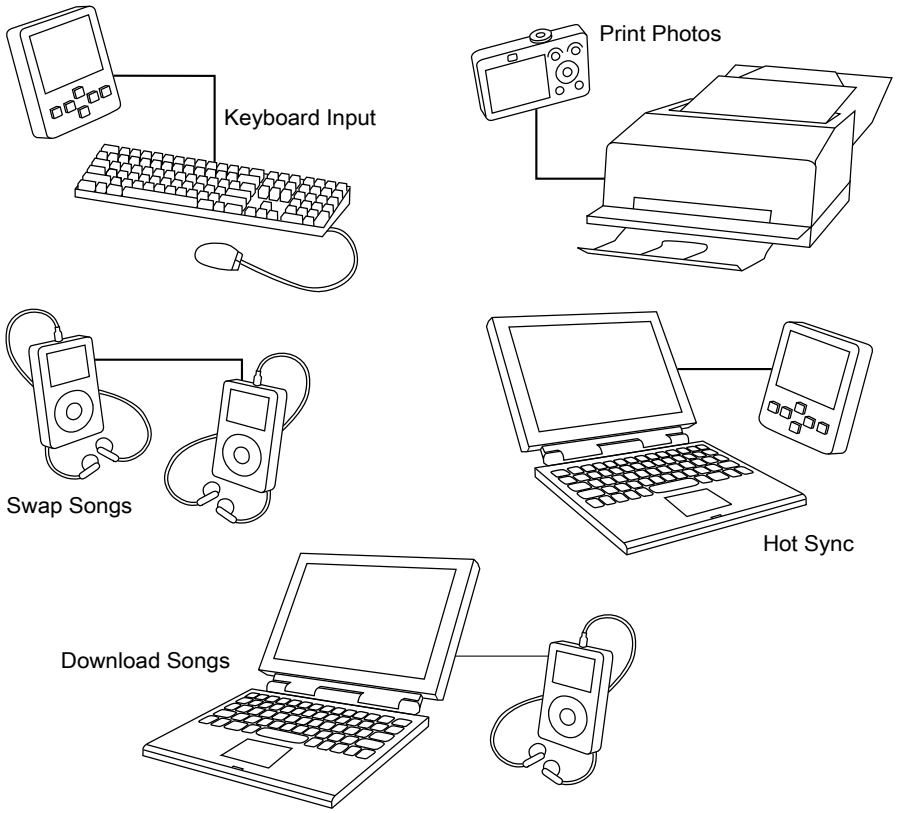


Figure 52-2. Example USB 2.0 On-The-Go configurations

52.1.4 USBFS Features

- USB 1.1 and 2.0 compatible FS device and FS/LS host controller with On-The-Go protocol logic
- 16 bidirectional endpoints
- DMA or FIFO data stream interfaces
- Low-power consumption
- FIRC with clock-recovery is supported to eliminate the 48 MHz crystal. It is used for USB device-only implementation.

52.2 Functional description

USBOTG communicates with the processor core through status registers, control registers, and data structures in memory.

52.2.1 Data Structures

To efficiently manage USB endpoint communications, USBFS implements a Buffer Descriptor Table (BDT) in system memory. See Figure 52-6.

52.2.2 On-chip transceiver required external components

USB system operation requires external components to ensure that driver output impedance, eye diagram, and VBUS cable fault tolerance requirements are met. DM and DP I/O pads must connect through series resistors (approximately 33 Ω each) to the USB connector on the application printed circuit board (PCB). Additionally, signal quality optimizes when these 33 Ω resistors are mounted closer to the processor than to the USB board-level connector. The USB transceiver includes:

- An internal 1.5 kΩ pullup resistor on the USB_DP line for full-speed device (controlled by USB_CONTROL[DPPULLUPNONOTG] or USB_OTGCTL[DPHIGH])
- Internal 15 kΩ pulldown resistors on the USB_DP and USB_DM signals, which are primarily intended for Host mode operation but are also useful in Device mode as explained below.

NOTE

For device operation, the internal 15 kΩ pulldowns should be enabled to keep the DP and DM ports in a known quiescent state when the VBUS detection software determines that the USB connection is not activated, including the cases when no cable to a host is present or when the USB port is not used. The internal 15 kΩ pulldowns should be controlled by USB_CTRL[PDE] in this case.

For host operation, the internal 15 kΩ pulldowns are enabled automatically, as required by the USB 2.0 specification, when USB_CTL[HOSTMODEEN] is asserted high.

The following diagrams provide overviews of host-only, device-only, and dual-role connections, respectively. The VDD pin connections in the following figures are shown as examples only. The VDD supply voltage can be chosen from the range shown for that pin in the device datasheet for this SOC, while the USBVDD supply must be 3.3v nominal for USB compliance. For more details, see the *Kinetis Peripheral Module Quick Reference*(KQRUG).

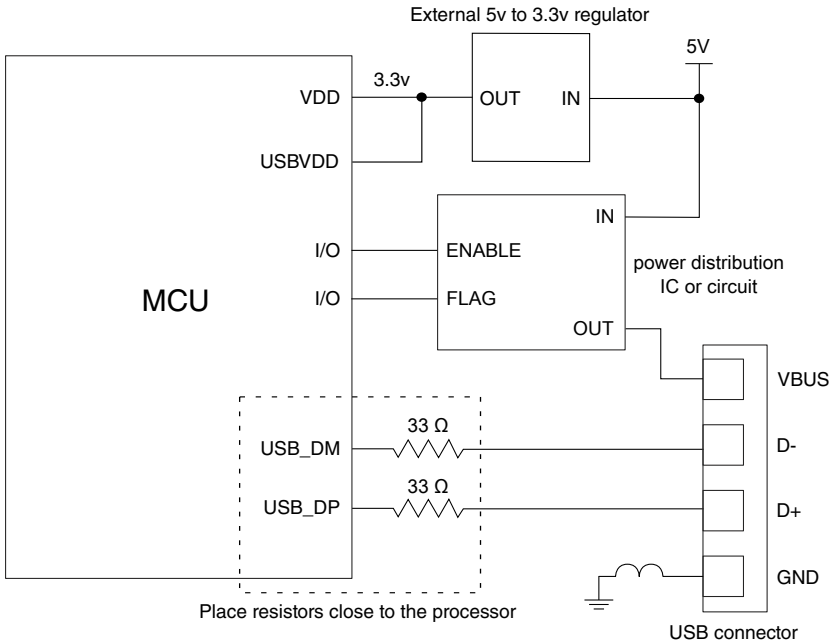


Figure 52-3. Host-only diagram

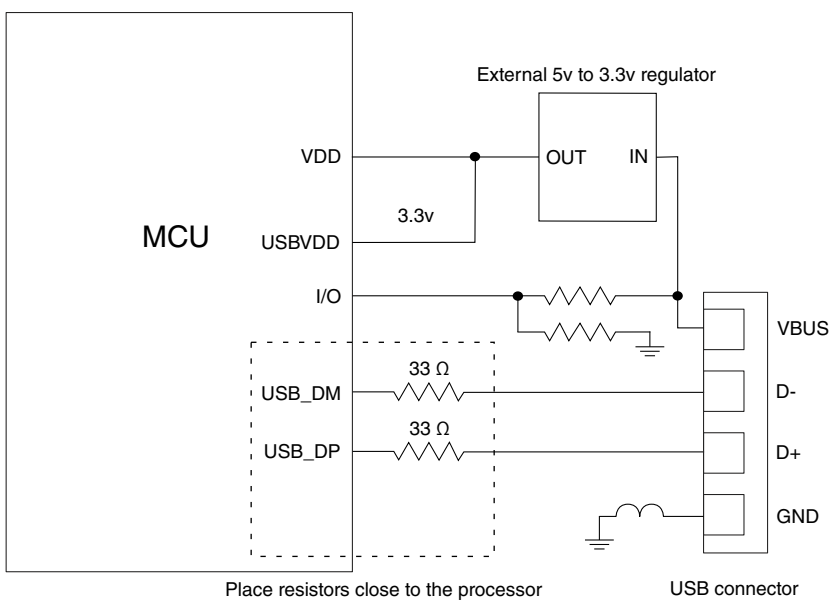


Figure 52-4. Typical Device-only diagram (bus-powered with external regulator)

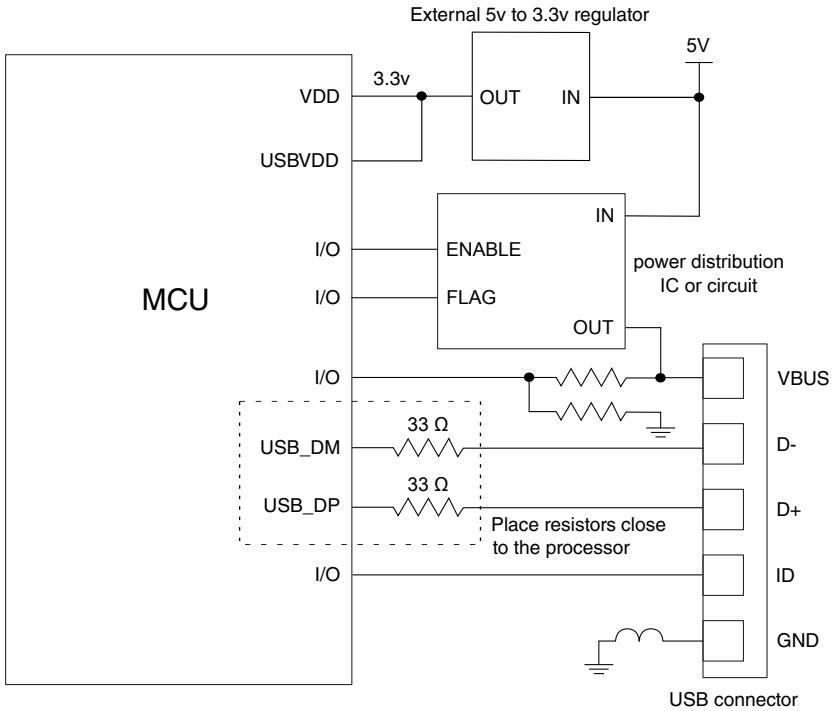


Figure 52-5. Dual-role diagram

52.3 Programmers interface

This section discusses the major components of the programming model for the USB module.

52.3.1 Buffer Descriptor Table

To efficiently manage USB endpoint communications USBFS implements a Buffer Descriptor Table (BDT) in system memory. The BDT resides on a 512-byte boundary in system memory and is pointed to by the BDT Page Registers. Every endpoint direction requires two 8-byte Buffer Descriptor (BD) entries. Therefore, a system with 16 fully bidirectional endpoints would require 512 bytes of system memory to implement the BDT. The two BD entries allows for an EVEN BD and ODD BD entry for each endpoint direction. This allows the microprocessor to process one BD while USBFS is processing the other BD. Double buffering BDs in this way allows USBFS to transfer data easily at the maximum throughput provided by USB.

Software should manage buffers for USBFS by updating the BDT when needed. This allows USBFS to efficiently manage data transmission and reception, while the microprocessor performs communication overhead processing and other function

dependent applications. Because the buffers are shared between the microprocessor and USBFS, a simple semaphore mechanism is used to distinguish who is allowed to update the BDT and buffers in system memory. A semaphore, the OWN bit, is cleared to 0 when the BD entry is owned by the microprocessor. The microprocessor is allowed read and write access to the BD entry and the buffer in system memory when the OWN bit is 0. When the OWN bit is set to 1, the BD entry and the buffer in system memory are owned by USBFS. USBFS now has full read and write access and the microprocessor must not modify the BD or its corresponding data buffer. The BD also contains indirect address pointers to where the actual buffer resides in system memory. This indirect address mechanism is shown in the following diagram.

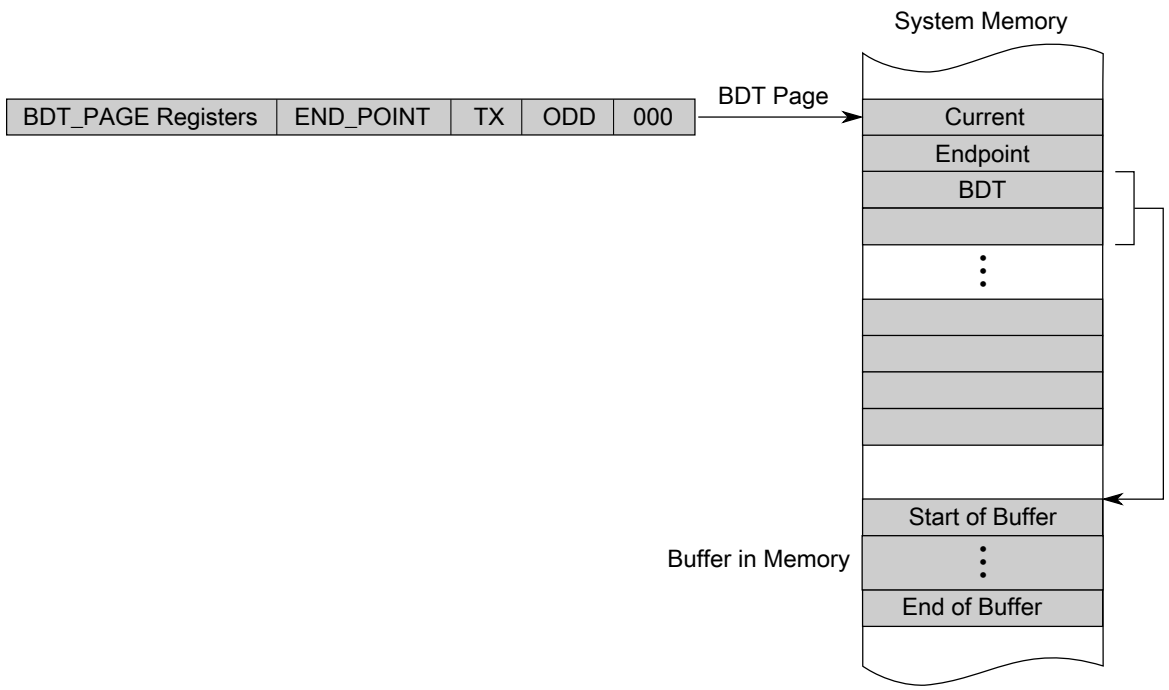


Figure 52-6. Buffer descriptor table

52.3.2 RX vs. TX as a USB peripheral device or USB host

The USBFS core uses software control to switch between two modes of operation:

- USB peripheral device
- USB hosts

In either mode, USB host or USB peripheral device, the same data paths and buffer descriptors are used for the transmission and reception of data. For this reason, a USBFS core-centric nomenclature is used to describe the direction of the data transfer between the USBFS core and USB:

- "RX" (or "receive") describes transfers that move data from USB to memory.
- "TX" (or "transmit") describes transfers that move data from memory to USB.

The following table shows how the data direction corresponds to the USB token type in host and peripheral device applications.

Table 52-1. Data direction for USB host or USB peripheral device

	RX	TX
Device	OUT or SETUP	IN
Host	IN	OUT or SETUP

52.3.3 Addressing BDT entries

An understanding of the addressing mechanism of the Buffer Descriptor Table is useful when accessing endpoint data via USBFS or microprocessor. Some points of interest are:

- The BDT occupies up to 512 bytes of system memory.
- 16 bidirectional endpoints can be supported with a full BDT of 512 bytes.
- 16 bytes are needed for each USB endpoint direction.
- Applications with fewer than 16 endpoints require less RAM to implement the BDT.
- The BDT Page Registers (BDT_PAGE) point to the starting location of the BDT.
- The BDT must be located on a 512-byte boundary in system memory.
- All enabled TX and RX endpoint BD entries are indexed into the BDT to allow easy access via USBFS or MCU core.

When a USB token on an enabled endpoint is received, USBFS uses its integrated DMA controller to interrogate the BDT. USBFS reads the corresponding endpoint BD entry to determine whether it owns the BD and corresponding buffer in system memory.

To compute the entry point into the BDT, the BDT_PAGE registers are concatenated with the current endpoint and the TX and ODD fields to form a 32-bit address. This address mechanism is shown in the following tables:

Table 52-2. BDT Address Calculation

31:24	23:16	15:9	8:5	4	3	2:0
BDT_PAGE_03	BDT_PAGE_02	BDT_PAGE_01[7:1]	Endpoint	TX	ODD	000

Table 52-3. BDT address calculation fields

Field	Description
BDT_PAGE	BDT_PAGE registers in the Control Register Block

Table continues on the next page...

Table 52-3. BDT address calculation fields (continued)

Field	Description
ENDPOINT	ENDPOINT field from the USB TOKEN
TX	1 for transmit transfers and 0 for receive transfers
ODD	Maintained within the USBFS SIE. It corresponds to the buffer currently in use. The buffers are used in a ping-pong fashion.

52.3.4 Buffer Descriptors (BDs)

A buffer descriptor provides endpoint buffer control information for USBFS and the processor. The Buffer Descriptors have different meaning based on whether it is USBFS or the processor reading the BD in memory.

The USBFS Controller uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- Whether to release ownership upon packet completion
- No address increment (FIFO mode)
- Whether data toggle synchronization is enabled
- How much data is to be transmitted or received
- Where the buffer resides in system memory

While the processor uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- The received TOKEN PID
- How much data was transmitted or received
- Where the buffer resides in system memory

The format for the BD is shown in the following figure.

Table 52-4. Buffer descriptor format

31:26	25:16	15:8	7	6	5	4	3	2	1	0

Table continues on the next page...

Table 52-4. Buffer descriptor format (continued)

RSVD	BC (10 bits)	RSVD	OWN	DATA0/1	KEEP/ TOK_PID[3]	NINC/ TOK_PID[2]	DTS/ TOK_PID[1]	BDT_STALL/ TOK_PID[0]	0	0
Buffer Address (32-Bits)										

Table 52-5. Buffer descriptor fields

Field	Description
31–26 RSVD	Reserved
25–16 BC	Byte Count Represents the 10-bit byte count. The USBFS SIE changes this field upon the completion of a RX transfer with the byte count of the data received.
15–8 RSVD	Reserved
7 OWN	Determines whether the processor or USBFS currently owns the buffer. Except when KEEP=1, the SIE hands ownership back to the processor after completing the token by clearing this bit. This must always be the last byte of the BD that the processor updates when it initializes a BD. 0 The processor has access to the BD. USBFS ignores all other fields in the BD. 1 USBFS has access to the BD. While USBFS owns the BD, the processor should not modify any other fields in the BD.
6 DATA0/1	Defines whether a DATA0 field (DATA0/1=0) or a DATA1 (DATA0/1=1) field was transmitted or received. It is unchanged by USBFS.
5 KEEP/ TOK_PID[3]	This bit has two functions: • KEEP bit—When written by the processor, it serves as the KEEP bit. Typically, this bit is 1 with ISO endpoints feeding a FIFO. The microprocessor is not informed that a token has been processed, the data is simply transferred to or from the FIFO. When KEEP is set, normally the NINC bit is also set to prevent address increment. 0 Allows USBFS to release the BD when a token has been processed. 1 This bit is unchanged by USBFS. Bit 3 of the current token PID is written back to the BD by USBFS. • TOK_PID[3]—If the OWN bit is also set, the BD remains owned by USBFS indefinitely; when written by USB, it serves as the TOK_PID[3] bit. 0 or 1 Bit 3 of the current token PID is written back to the BD by USBFS. Typically, this bit is 1 with ISO endpoints feeding a FIFO. The microprocessor is not informed that a token has been processed, the data is simply transferred to or from the FIFO. When KEEP is set, normally the NINC bit is also set to prevent address increment.
4 NINC/ TOK_PID[2]	No Increment (NINC) Disables the DMA engine address increment. This forces the DMA engine to read or write from the same address. This is useful for endpoints when data needs to be read from or written to a single location such as a FIFO. Typically this bit is set with the KEEP bit for ISO endpoints that are interfacing to a FIFO. 0 USBFS writes bit 2 of the current token PID to the BD. 1 This bit is unchanged by USBFS.

Table continues on the next page...

Table 52-5. Buffer descriptor fields (continued)

Field	Description
3 DTS/ TOK_PID[1]	Setting this bit enables USBFS to perform Data Toggle Synchronization. • If KEEP=0, bit 1 of the current token PID is written back to the BD. • If KEEP=1, this bit is unchanged by USBFS. 0 Data Toggle Synchronization is disabled. 1 Enables USBFS to perform Data Toggle Synchronization.
2 BDT_STALL TOK_PID[0]	Setting this bit causes USBFS to issue a STALL handshake if a token is received by the SIE that would use the BDT in this location. The BDT is not consumed by the SIE (the owns bit remains set and the rest of the BDT is unchanged) when a BDT_STALL bit is set. • If KEEP=0, bit 0 of the current token PID is written back to the BD. • If KEEP=1, this bit is unchanged by USBFS. 0 No stall issued. 1 The BDT is not consumed by the SIE (the OWN bit remains set and the rest of the BDT is unchanged). Setting BDT_STALL also causes the corresponding USB_ENDPT*n*[EPSTALL] bit to set. This causes USBOTG to issue a STALL handshake for both directions of the associated endpoint. To clear the stall condition: 1. Clear the associated USB_ENDPT*n*[EPSTALL] bit. 2. Write the BDT to clear OWN and BDT_STALL.
TOK_PID[n]	Bits [5:2] can also represent the current token PID. The current token PID is written back in to the BD by USBFS when a transfer completes. The values written back are the token PID values from the USB specification: • 0x1h for an OUT token. • 0x9h for an IN token. • 0xDh for a SETUP token. In host mode, this field is used to report the last returned PID or a transfer status indication. The possible values returned are: • 0x3h DATA0 • 0xBh DATA1 • 0x2h ACK • 0xEh STALL • 0xAh NAK • 0x0h Bus Timeout • 0xFh Data Error
1–0 Reserved	Reserved, should read as zeroes.
ADDR[31:0]	Address Represents the 32-bit buffer address in system memory; this address must also be 32-bit aligned. These bits are unchanged by USBFS.

52.3.5 USB transaction

When USBFS transmits or receives data, it computes the BDT address using the address generation shown in "Addressing Buffer Descriptor Entries" table.

If OWN =1, the following process occurs:

1. USBFS reads the BDT.
2. The SIE transfers the data via the DMA to or from the buffer pointed to by the ADDR field of the BD.
3. When the TOKEN is complete, USBFS updates the BDT and, if KEEP=0, changes the OWN bit to 0.
4. The STAT register is updated and the TOK_DNE interrupt is set.
5. When the processor processes the TOK_DNE interrupt, it reads from the status register all the information needed to process the endpoint.
6. At this point, the processor allocates a new BD so that additional USB data can be transmitted or received for that endpoint, and then processes the last BD.

The following figure shows a timeline of how a typical USB token is processed after the BDT is read and OWN=1.

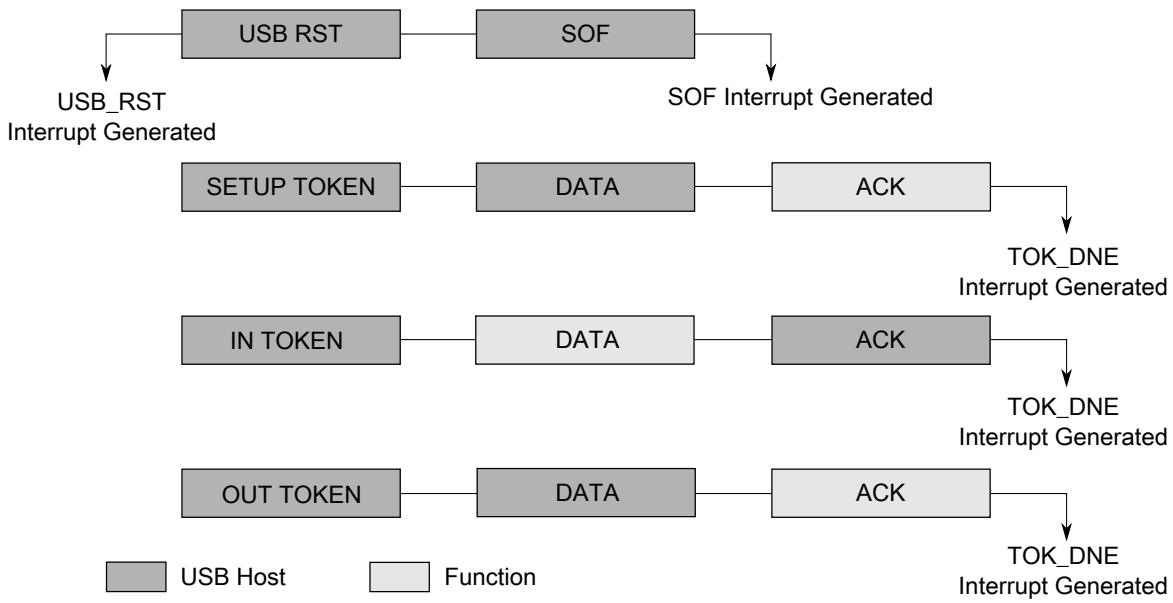


Figure 52-7. USB token transaction

The USB has two sources for the DMA overrun error:

Memory Latency

The memory latency may be too high and cause the receive FIFO to overflow. This is predominantly a hardware performance issue, usually caused by transient memory access issues.

Oversized Packets

The packet received may be larger than the negotiated *MaxPacket* size. Typically, this is caused by a software bug. For DMA overrun errors due to oversized data packets, the USB specification is ambiguous. It assumes correct software drivers on both sides. NAKing the packet can result in retransmission of the already oversized packet data. Therefore, in response to oversized packets, the USB core continues ACKing the packet for non-isochronous transfers.

Table 52-6. USB responses to DMA overrun errors

Errors due to Memory Latency	Errors due to Oversized Packets
Non-Acknowledgment (NAK) or Bus Timeout (BTO) — See bit 4 in "Error Interrupt Status Register (ERRSTAT)" as appropriate for the class of transaction.	Continues acknowledging (ACKing) the packet for non-isochronous transfers.
—	The data written to memory is clipped to the MaxPacket size so as not to corrupt system memory.
The DMAERR bit is set in the ERRSTAT register for host and device modes of operation. Depending on the values of the INTENB and ERRENB register, the core may assert an interrupt to notify the processor of the DMA error.	Asserts ERRSTAT[DMAERR] ,which can trigger an interrupt and TOKDNE interrupt fires. Note: The TOK_PID field of the BDT is not 1111 because the DMAERR is not due to latency.
• For host mode, the TOKDNE interrupt is generated and the TOK_PID field of the BDT is 1111 to indicate the DMA latency error. Host mode software can decide to retry or move to next scheduled item. • In device mode, the BDT is not written back nor is the TOKDNE interrupt triggered because it is assumed that a second attempt is queued and will succeed in the future.	The packet length field written back to the BDT is the MaxPacket value that represents the length of the clipped data actually written to memory.
From here, the software can decide an appropriate course of action for future transactions such as stalling the endpoint, canceling the transfer, disabling the endpoint, etc.	

52.4 Memory map/Register definitions

This section provides the memory map and detailed descriptions of all USB interface registers.

USB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_5000	Peripheral ID register (USB0_PERID)	8	R	04h	52.4.1/1327
4005_5004	Peripheral ID Complement register (USB0_IDCOMP)	8	R	FBh	52.4.2/1327
4005_5008	Peripheral Revision register (USB0_REV)	8	R	33h	52.4.3/1328
4005_500C	Peripheral Additional Info register (USB0_ADDINFO)	8	R	01h	52.4.4/1328
4005_5010	OTG Interrupt Status register (USB0_OTGISTAT)	8	R/W	00h	52.4.5/1329
4005_5014	OTG Interrupt Control register (USB0_OTGICR)	8	R/W	00h	52.4.6/1330

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_5018	OTG Status register (USB0_OTGSTAT)	8	R/W	00h	52.4.7/1331
4005_501C	OTG Control register (USB0_OTGCTL)	8	R/W	00h	52.4.8/1332
4005_5080	Interrupt Status register (USB0_ISTAT)	8	R/W	00h	52.4.9/1333
4005_5084	Interrupt Enable register (USB0_INTEN)	8	R/W	00h	52.4.10/ 1334
4005_5088	Error Interrupt Status register (USB0_ERRSTAT)	8	R/W	00h	52.4.11/ 1335
4005_508C	Error Interrupt Enable register (USB0_ERREN)	8	R/W	00h	52.4.12/ 1336
4005_5090	Status register (USB0_STAT)	8	R	00h	52.4.13/ 1337
4005_5094	Control register (USB0_CTL)	8	R/W	00h	52.4.14/ 1338
4005_5098	Address register (USB0_ADDR)	8	R/W	00h	52.4.15/ 1339
4005_509C	BDT Page register 1 (USB0_BDTPAGE1)	8	R/W	00h	52.4.16/ 1340
4005_50A0	Frame Number register Low (USB0_FRMNUML)	8	R/W	00h	52.4.17/ 1340
4005_50A4	Frame Number register High (USB0_FRMNUMH)	8	R/W	00h	52.4.18/ 1341
4005_50A8	Token register (USB0_TOKEN)	8	R/W	00h	52.4.19/ 1341
4005_50AC	SOF Threshold register (USB0_SOFTHLD)	8	R/W	00h	52.4.20/ 1342
4005_50B0	BDT Page Register 2 (USB0_BDTPAGE2)	8	R/W	00h	52.4.21/ 1343
4005_50B4	BDT Page Register 3 (USB0_BDTPAGE3)	8	R/W	00h	52.4.22/ 1343
4005_50C0	Endpoint Control register (USB0_ENDPT0)	8	R/W	00h	52.4.23/ 1344
4005_50C4	Endpoint Control register (USB0_ENDPT1)	8	R/W	00h	52.4.23/ 1344
4005_50C8	Endpoint Control register (USB0_ENDPT2)	8	R/W	00h	52.4.23/ 1344
4005_50CC	Endpoint Control register (USB0_ENDPT3)	8	R/W	00h	52.4.23/ 1344
4005_50D0	Endpoint Control register (USB0_ENDPT4)	8	R/W	00h	52.4.23/ 1344
4005_50D4	Endpoint Control register (USB0_ENDPT5)	8	R/W	00h	52.4.23/ 1344
4005_50D8	Endpoint Control register (USB0_ENDPT6)	8	R/W	00h	52.4.23/ 1344

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_50DC	Endpoint Control register (USB0_ENDPT7)	8	R/W	00h	52.4.23/ 1344
4005_50E0	Endpoint Control register (USB0_ENDPT8)	8	R/W	00h	52.4.23/ 1344
4005_50E4	Endpoint Control register (USB0_ENDPT9)	8	R/W	00h	52.4.23/ 1344
4005_50E8	Endpoint Control register (USB0_ENDPT10)	8	R/W	00h	52.4.23/ 1344
4005_50EC	Endpoint Control register (USB0_ENDPT11)	8	R/W	00h	52.4.23/ 1344
4005_50F0	Endpoint Control register (USB0_ENDPT12)	8	R/W	00h	52.4.23/ 1344
4005_50F4	Endpoint Control register (USB0_ENDPT13)	8	R/W	00h	52.4.23/ 1344
4005_50F8	Endpoint Control register (USB0_ENDPT14)	8	R/W	00h	52.4.23/ 1344
4005_50FC	Endpoint Control register (USB0_ENDPT15)	8	R/W	00h	52.4.23/ 1344
4005_5100	USB Control register (USB0_USBCTRL)	8	R/W	C0h	52.4.24/ 1345
4005_5104	USB OTG Observe register (USB0_OBSERVE)	8	R	50h	52.4.25/ 1346
4005_5108	USB OTG Control register (USB0_CONTROL)	8	R/W	00h	52.4.26/ 1347
4005_510C	USB Transceiver Control register 0 (USB0_USBTRC0)	8	R/W	00h	52.4.27/ 1347
4005_5114	Frame Adjust Register (USB0_USBFRMADJUST)	8	R/W	00h	52.4.28/ 1349
4005_512C	Miscellaneous Control register (USB0_MISCCTRL)	8	R/W	00h	52.4.29/ 1349
4005_5140	USB Clock recovery control (USB0_CLK_RECOVER_CTRL)	8	R/W	00h	52.4.30/ 1350
4005_5144	FIRC oscillator enable register (USB0_CLK_RECOVER_IRC_EN)	8	R (reads 0)	00h	52.4.31/ 1351
4005_5154	Clock recovery combined interrupt enable (USB0_CLK_RECOVER_INT_EN)	8	R/W	10h	52.4.31/ 1352
4005_515C	Clock recovery separated interrupt status (USB0_CLK_RECOVER_INT_STATUS)	8	w1c	00h	52.4.32/ 1352

52.4.1 Peripheral ID register (USB*x*_PERID)

Reads back the value of 0x04. This value is defined for the USB peripheral.

Address: 4005_5000h base + 0h offset = 4005_5000h

Bit	7	6	5	4	3	2	1	0
Read	0		ID					
Write								
Reset	0	0	0	0	0	1	0	0

USB*x*_PERID field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ID	Peripheral Identification This field always reads 0x4h.

52.4.2 Peripheral ID Complement register (USB*x*_IDCOMP)

Reads back the complement of the Peripheral ID register. For the USB peripheral, the value is 0xFB.

Address: 4005_5000h base + 4h offset = 4005_5004h

Bit	7	6	5	4	3	2	1	0
Read	1		NID					
Write								
Reset	1	1	1	1	1	0	1	1

USB*x*_IDCOMP field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
NID	Ones' complement of PERID[ID] bits.

52.4.3 Peripheral Revision register (USB*x*_REV)

Contains the revision number of the USB module.

Address: 4005_5000h base + 8h offset = 4005_5008h

Bit	7	6	5	4	3	2	1	0
Read	REV							
Write								
Reset	0	0	1	1	0	0	1	1

USB*x*_REV field descriptions

Field	Description
REV	Revision Indicates the revision number of the USB Core.

52.4.4 Peripheral Additional Info register (USB*x*_ADDINFO)

Reads back the value of the Host Enable bit.

Address: 4005_5000h base + Ch offset = 4005_500Ch

Bit	7	6	5	4	3	2	1	0
Read	0					0		IEHOST
Write								
Reset	0	0	0	0	0	0	0	1

USB*x*_ADDINFO field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 IEHOST	This bit is set if host mode is enabled.

52.4.5 OTG Interrupt Status register (USBx_OTGISTAT)

Records changes to the state of the one-millisecond timer and line state stable logic. Software can read this register to determine the event that triggers an interrupt. Only bits that have changed since the last software read are set. Writing a one to a bit clears the associated interrupt.

Address: 4005_5000h base + 10h offset = 4005_5010h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	ONEMSEC	LINE_ STATE_ CHG	0	Reserved	Reserved	0	Reserved
Write	w1c	w1c	w1c		w1c	w1c		w1c
Reset	0	0	0	0	0	0	0	0

USBx_OTGISTAT field descriptions

Field	Description
7 Reserved	This field is reserved. Software must not change the value of this bitfield.
6 ONEMSEC	This bit is set when the 1 millisecond timer expires. This bit stays asserted until cleared by software. The interrupt must be serviced every millisecond to avoid losing 1msec counts.
5 LINE_STATE_ CHG	This interrupt is set when the USB line state (CTL[SE0] and CTL[JSTATE] bits) are stable without change for 1 millisecond, and the value of the line state is different from the last time when the line state was stable. It is set on transitions between SE0 and J-state, SE0 and K-state, and J-state and K-state. Changes in J-state while SE0 is true do not cause an interrupt. This interrupt can be used in detecting Reset, Resume, Connect, and Data Line Pulse signaling.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. Software must not change the value of this bitfield.
2 Reserved	This field is reserved. Software must not change the value of this bitfield.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. Software must not change the value of this bitfield.

52.4.6 OTG Interrupt Control register (USB*x*_OTGICR)

Enables the corresponding interrupt status bits defined in the OTG Interrupt Status Register.

Address: 4005_5000h base + 14h offset = 4005_5014h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	ONEMSECEN	LINESTATEEN	0	Reserved	Reserved	0	Reserved
Write								
Reset	0	0	0	0	0	0	0	0

USB*x*_OTGICR field descriptions

Field	Description
7 Reserved	Software must not change the value of this bitfield. This field is reserved.
6 ONEMSECEN	One Millisecond Interrupt Enable 0 Diables the 1ms timer interrupt. 1 Enables the 1ms timer interrupt.
5 LINESTATEEN	Line State Change Interrupt Enable 0 Disables the LINE_STAT_CHG interrupt. 1 Enables the LINE_STAT_CHG interrupt.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	Software must not change the value of this bitfield. This field is reserved.
2 Reserved	Software must not change the value of this bitfield. This field is reserved.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	Software must not change the value of this bitfield. This field is reserved.

52.4.7 OTG Status register (USB*x*_OTGSTAT)

Displays the actual value from the one-millisecond timer and line state stable logic.

Address: 4005_5000h base + 18h offset = 4005_5018h

Bit	7	6	5	4
Read	0	ONEMSECEN	LINESTATESTABLE	0
Write				
Reset	0	0	0	0

Bit	3	2	1	0
Read	0	0	0	0
Write				
Reset	0	0	0	0

USB*x*_OTGSTAT field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 ONEMSECEN	This bit is reserved for the 1ms count, but it is not useful to software.
5 LINESTATESTABLE	Indicates that the internal signals that control the LINE_STATE_CHG field of OTGISTAT are stable for at least 1 ms. This bit is used to provide a hardware debounce of the linestate in detection of Connect, Disconnect and Resume signaling. First read LINE_STATE_CHG field and then read this field. If this field reads as 1, then the value of LINE_STATE_CHG can be considered stable. 0 The LINE_STAT_CHG bit is not yet stable. 1 The LINE_STAT_CHG bit has been debounced and is stable.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.8 OTG Control register (USBx_OTGCTL)

Controls the operation of VBUS and Data Line termination resistors.

Address: 4005_5000h base + 1Ch offset = 4005_501Ch

Bit	7	6	5	4	3	2	1	0
Read	DPHIGH	0	DPLOW	DMLOW	0	OTGEN	0	
Write								
Reset	0	0	0	0	0	0	0	0

USBx_OTGCTL field descriptions

Field	Description
7 DPHIGH	D+ Data Line pullup resistor enable 0 D+ pullup resistor is not enabled 1 D+ pullup resistor is enabled
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 DPLOW	D+ Data Line pull-down resistor enable This bit should always be enabled together with bit 4 (DMLOW) 0 D+ pulldown resistor is not enabled. 1 D+ pulldown resistor is enabled.
4 DMLOW	D– Data Line pull-down resistor enable 0 D- pulldown resistor is not enabled. 1 D- pulldown resistor is enabled.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 OTGEN	On-The-Go pullup/pulldown resistor enable 0 If USB_EN is 1 and HOST_MODE is 0 in the Control Register (CTL), then the D+ Data Line pull-up resistors are enabled. If HOST_MODE is 1 the D+ and D– Data Line pull-down resistors are engaged. 1 The pull-up and pull-down controls in this register are used.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.9 Interrupt Status register (USBx_ISTAT)

Contains fields for each of the interrupt sources within the USB Module. Each of these fields are qualified with their respective interrupt enable bits. All fields of this register are logically OR'd together along with the OTG Interrupt Status Register (OTGSTAT) to form a single interrupt source for the processor's interrupt controller. After an interrupt bit has been set it may only be cleared by writing a one to the respective interrupt bit. This register contains the value of 0x00 after a reset.

Address: 4005_5000h base + 80h offset = 4005_5080h

Bit	7	6	5	4	3	2	1	0
Read	STALL	ATTACH	RESUME	SLEEP	TOKDNE	SOFTOK	ERROR	USBRST
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

USBx_ISTAT field descriptions

Field	Description
7 STALL	Stall Interrupt In Device mode this bit is asserted when a STALL handshake is sent by the SIE. In Host mode this bit is set when the USB Module detects a STALL acknowledge during the handshake phase of a USB transaction.This interrupt can be used to determine whether the last USB transaction was completed successfully or stalled.
6 ATTACH	Attach Interrupt This field is set when the USB Module detects an attach of a USB device. This field is only valid if CTL[HOSTMODEEN]=1. This interrupt signifies that a peripheral is now present and must be configured; it is asserted if there have been no transitions on the USB for 2.5 µs and the current bus state is not SE0." 0 No Attach is detected since the last time the ATTACH bit was cleared. 1 A peripheral is now present and must be configured (a stable non-SE0 state is detected for more than 2.5 µs).
5 RESUME	This bit is set when a K-state is observed on the DP/DM signals for 2.5 µs. When not in suspend mode this interrupt must be disabled.
4 SLEEP	This bit is set when the USB Module detects a constant idle on the USB bus for 3 ms. The sleep timer is reset by activity on the USB bus.
3 TOKDNE	This bit is set when the current token being processed has completed. The processor must immediately read the STATUS (STAT) register to determine the EndPoint and BD used for this token. Clearing this bit (by writing a one) causes STAT to be cleared or the STAT holding register to be loaded into the STAT register.
2 SOFTOK	This bit is set when the USB Module receives a Start Of Frame (SOF) token. In Host mode this field is set when the SOF threshold is reached (MISCCTRL[SOFBUSSET]=0), or when the SOF counter reaches 0 (MISCCTRL[SOFBUSSET]=1), so that software can prepare for the next SOF.

Table continues on the next page...

USBx_ISTAT field descriptions (continued)

Field	Description
1 ERROR	This bit is set when any of the error conditions within Error Interrupt Status (ERRSTAT) register occur. The processor must then read the ERRSTAT register to determine the source of the error.
0 USBRST	This bit is set when the USB Module has decoded a valid USB reset. This informs the processor that it should write 0x00 into the address register and enable endpoint 0. USBRST is set after a USB reset has been detected for 2.5 microseconds. It is not asserted again until the USB reset condition has been removed and then reasserted.

52.4.10 Interrupt Enable register (USBx_INTEN)

Contains enable fields for each of the interrupt sources within the USB Module. Setting any of these bits enables the respective interrupt source in the ISTAT register. This register contains the value of 0x00 after a reset.

Address: 4005_5000h base + 84h offset = 4005_5084h

Bit	7	6	5	4	3	2	1	0
Read Write	STALLEN	ATTACHEN	RESUMEEN	SLEEPEN	TOKDNEEN	SOFTOKEN	ERROREN	USBRSTEN
Reset	0	0	0	0	0	0	0	0

USBx_INTEN field descriptions

Field	Description
7 STALLEN	STALL Interrupt Enable 0 Diasbles the STALL interrupt. 1 Enables the STALL interrupt.
6 ATTACHEN	ATTACH Interrupt Enable 0 Disables the ATTACH interrupt. 1 Enables the ATTACH interrupt.
5 RESUMEEN	RESUME Interrupt Enable 0 Disables the RESUME interrupt. 1 Enables the RESUME interrupt.
4 SLEEPEN	SLEEP Interrupt Enable 0 Disables the SLEEP interrupt. 1 Enables the SLEEP interrupt.
3 TOKDNEEN	TOKDNE Interrupt Enable 0 Disables the TOKDNE interrupt. 1 Enables the TOKDNE interrupt.
2 SOFTOKEN	SOFTOK Interrupt Enable 0 Disbles the SOFTOK interrupt. 1 Enables the SOFTOK interrupt.

Table continues on the next page...

USBx_INTEN field descriptions (continued)

Field	Description
1 ERROREN	ERROR Interrupt Enable 0 Disables the ERROR interrupt. 1 Enables the ERROR interrupt.
0 USBRSTEN	USBRST Interrupt Enable 0 Disables the USBRST interrupt. 1 Enables the USBRST interrupt.

52.4.11 Error Interrupt Status register (USBx_ERRSTAT)

Contains enable bits for each of the error sources within the USB Module. Each of these bits are qualified with their respective error enable bits. All bits of this register are logically OR'd together and the result placed in the ERROR bit of the ISTAT register. After an interrupt bit has been set it may only be cleared by writing a one to the respective interrupt bit. Each bit is set as soon as the error condition is detected. Therefore, the interrupt does not typically correspond with the end of a token being processed. This register contains the value of 0x00 after a reset.

Address: 4005_5000h base + 88h offset = 4005_5088h

Bit	7	6	5	4	3	2	1	0
Read	BTSERR	OWNERR	DMAERR	BTOERR	DFN8	CRC16	CRC5EOF	PIDERR
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

USBx_ERRSTAT field descriptions

Field	Description
7 BTSERR	This bit is set when a bit stuff error is detected. If set, the corresponding packet is rejected due to the error.
6 OWNERR	This field is valid when the USB Module is operating in peripheral mode (CTL[HOSTMODEEN]=0). It is set if the USB Module requires a new BD for SETUP, ISO IN, or ISO OUT transfer while a new BD is not available.
5 DMAERR	This bit is set if the USB Module has requested a DMA access to read a new BDT but has not been given the bus before it needs to receive or transmit data. If processing a TX transfer this would cause a transmit data underflow condition. If processing a RX transfer this would cause a receive data overflow condition. This interrupt is useful when developing device arbitration hardware for the microprocessor and the USB module to minimize bus request and bus grant latency. This bit is also set if a data packet to or from the host is larger than the buffer size allocated in the BDT. In this case the data packet is truncated as it is put in buffer memory.
4 BTOERR	This bit is set when a bus turnaround timeout error occurs. The USB module contains a bus turnaround timer that keeps track of the amount of time elapsed between the token and data phases of a SETUP or OUT TOKEN or the data and handshake phases of a IN TOKEN. If more than 16 bit times are counted from the previous EOP before a transition from IDLE, a bus turnaround timeout error occurs.

Table continues on the next page...

USBx_ERRSTAT field descriptions (continued)

Field	Description
3 DFN8	This bit is set if the data field received was not 8 bits in length. USB Specification 1.0 requires that data fields be an integral number of bytes. If the data field was not an integral number of bytes, this bit is set.
2 CRC16	This bit is set when a data packet is rejected due to a CRC16 error.
1 CRC5EOF	This error interrupt has two functions. When the USB Module is operating in peripheral mode (CTL[HOSTMODEEN]=0), this interrupt detects CRC5 errors in the token packets generated by the host. If set the token packet was rejected due to a CRC5 error. When the USB Module is operating in host mode (CTL[HOSTMODEEN]=1), this interrupt detects End Of Frame (EOF) error conditions. This occurs when the USB Module is transmitting or receiving data and the SOF counter reaches zero. This interrupt is useful when developing USB packet scheduling software to ensure that no USB transactions cross the start of the next frame.
0 PIDERR	This bit is set when the PID check field fails.

52.4.12 Error Interrupt Enable register (USBx_ERREN)

Contains enable bits for each of the error interrupt sources within the USB module. Setting any of these bits enables the respective interrupt source in ERRSTAT. Each bit is set as soon as the error condition is detected. Therefore, the interrupt does not typically correspond with the end of a token being processed. This register contains the value of 0x00 after a reset.

Address: 4005_5000h base + 8Ch offset = 4005_508Ch

Bit	7	6	5	4	3	2	1	0
Read Write	BTSERREN	OWNERREN	DMAERREN	BTOERREN	DFN8EN	CRC16EN	CRC5EOFEN	PIDERREN
Reset	0	0	0	0	0	0	0	0

USBx_ERREN field descriptions

Field	Description
7 BTSERREN	BTSERR Interrupt Enable 0 Disables the BTSERR interrupt. 1 Enables the BTSERR interrupt.
6 OWNERREN	OWNERR Interrupt Enable This field is valid when the USB module is operating in peripheral mode (CTL[HOSTMODEEN]=0). 0 Disables the OWNERR interrupt. 1 Enables the OWNERR interrupt.
5 DMAERREN	DMAERR Interrupt Enable 0 Disables the DMAERR interrupt. 1 Enables the DMAERR interrupt.

Table continues on the next page...

USBx_ERREN field descriptions (continued)

Field	Description
4 BTOERREN	BTOERR Interrupt Enable 0 Disables the BTOERR interrupt. 1 Enables the BTOERR interrupt.
3 DFN8EN	DFN8 Interrupt Enable 0 Disables the DFN8 interrupt. 1 Enables the DFN8 interrupt.
2 CRC16EN	CRC16 Interrupt Enable 0 Disables the CRC16 interrupt. 1 Enables the CRC16 interrupt.
1 CRC5EOFEN	CRC5/EOF Interrupt Enable 0 Disables the CRC5/EOF interrupt. 1 Enables the CRC5/EOF interrupt.
0 PIDERREN	PIDERR Interrupt Enable 0 Disables the PIDERR interrupt. 1 Enters the PIDERR interrupt.

52.4.13 Status register (USBx_STAT)

Reports the transaction status within the USB module. When the processor's interrupt controller has received a TOKDNE, interrupt the Status Register must be read to determine the status of the previous endpoint communication. The data in the status register is valid when TOKDNE interrupt is asserted. The Status register is actually a read window into a status FIFO maintained by the USB module. When the USB module uses a BD, it updates the Status register. If another USB transaction is performed before the TOKDNE interrupt is serviced, the USB module stores the status of the next transaction in the STAT FIFO. Thus STAT is actually a four byte FIFO that allows the processor core to process one transaction while the SIE is processing the next transaction. Clearing the TOKDNE bit in the ISTAT register causes the SIE to update STAT with the contents of the next STAT value. If the data in the STAT holding register is valid, the SIE immediately reasserts to TOKDNE interrupt.

Address: 4005_5000h base + 90h offset = 4005_5090h

Bit	7	6	5	4	3	2	1	0
Read	ENDP				TX	ODD	0	
Write								
Reset	0	0	0	0	0	0	0	0

USBx_STAT field descriptions

Field	Description
7–4 ENDP	This four-bit field encodes the endpoint address that received or transmitted the previous token. This allows the processor core to determine the BDT entry that was updated by the last USB transaction.
3 TX	Transmit Indicator 0 The most recent transaction was a receive operation. 1 The most recent transaction was a transmit operation.
2 ODD	This bit is set if the last buffer descriptor updated was in the odd bank of the BDT.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.14 Control register (USBx_CTL)

Provides various control and configuration information for the USB module.

Address: 4005_5000h base + 94h offset = 4005_5094h

Bit	7	6	5	4
Read Write	JSTATE	SE0	TXSUSPENDTOKENBUSY	RESET
Reset	0	0	0	0

Bit	3	2	1	0
Read Write	HOSTMODEEN	RESUME	ODDRST	USBENSOFEN
Reset	0	0	0	0

USBx_CTL field descriptions

Field	Description
7 JSTATE	Live USB differential receiver JSTATE signal The polarity of this signal is affected by the current state of LSEN .
6 SE0	Live USB Single Ended Zero signal
5 TXSUSPENDTOKENBUSY	In Host mode, TOKEN_BUSY is set when the USB module is busy executing a USB token. Software must not write more token commands to the Token Register when TOKEN_BUSY is set. Software should check this field before writing any tokens to the Token Register to ensure that token commands are not lost. In Device mode, TXD_SUSPEND is set when the SIE has disabled packet transmission and reception. Clearing this bit allows the SIE to continue token processing. This bit is set by the SIE when a SETUP Token is received allowing software to dequeue any pending packet transactions in the BDT before resuming token processing.
4 RESET	Setting this bit enables the USB Module to generate USB reset signaling. This allows the USB Module to reset USB peripherals. This control signal is only valid in Host mode (CTL[HOSTMODEEN]=1). Software must set RESET to 1 for the required amount of time and then clear it to 0 to end reset signaling.

Table continues on the next page...

USBx_CTL field descriptions (continued)

Field	Description
3 HOSTMODEEN	When set to 1, this bit enables the USB Module to operate in Host mode. In host mode, the USB module performs USB transactions under the programmed control of the host processor.
2 RESUME	When set to 1 this bit enables the USB Module to execute resume signaling. This allows the USB Module to perform remote wake-up. Software must set RESUME to 1 for the required amount of time and then clear it to 0. If HOSTMODEENis set, the USB module appends a Low Speed End of Packet to the Resume signaling when the RESUME bit is cleared.
1 ODDRST	Setting this bit to 1 resets all the BDT ODD ping/pong fields to 0, which then specifies the EVEN BDT bank.
0 USBENSOFEN	USB Enable Setting this bit enables the USB-FS to operate; clearing it disables the USB-FS. Setting the bit causes the SIE to reset all of its ODD bits to the BDTs. Therefore, setting this bit resets much of the logic in the SIE. When host mode is enabled, clearing this bit causes the SIE to stop sending SOF tokens. 0 Disables the USB Module. 1 Enables the USB Module.

52.4.15 Address register (USBx_ADDR)

Holds the unique USB address that the USB module decodes when in Peripheral mode (CTL[HOSTMODEEN]=0). When operating in Host mode (CTL[HOSTMODEEN]=1) the USB module transmits this address with a TOKEN packet. This enables the USB module to uniquely address any USB peripheral. In either mode, CTL[USBENSOFEN] must be 1. The Address register is reset to 0x00 after the reset input becomes active or the USB module decodes a USB reset signal. This action initializes the Address register to decode address 0x00 as required by the USB specification.

Address: 4005_5000h base + 98h offset = 4005_5098h

Bit	7	6	5	4	3	2	1	0
Read Write	LSEN	ADDR						
Reset	0	0	0	0	0	0	0	0

USBx_ADDR field descriptions

Field	Description
7 LSEN	Low Speed Enable bit Informs the USB module that the next token command written to the token register must be performed at low speed. This enables the USB module to perform the necessary preamble required for low-speed data transmissions.
ADDR	USB Address

Table continues on the next page...

USBx_ADDR field descriptions (continued)

Field	Description
	Defines the USB address that the USB module decodes in peripheral mode, or transmits when in host mode.

52.4.16 BDT Page register 1 (USBx_BDTPAGE1)

Provides address bits 15 through 9 of the base address where the current Buffer Descriptor Table (BDT) resides in system memory. See Buffer Descriptor Table. The 32-bit BDT Base Address is always aligned on 512-byte boundaries, so bits 8 through 0 of the base address are always zero.

Address: 4005_5000h base + 9Ch offset = 4005_509Ch

Bit	7	6	5	4	3	2	1	0
Read	BDTBA							0
Write								
Reset	0	0	0	0	0	0	0	0

USBx_BDTPAGE1 field descriptions

Field	Description
7–1 BDTBA	Provides address bits 15 through 9 of the BDT base address.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.17 Frame Number register Low (USBx_FRMNUML)

The Frame Number registers (low and high) contain the 11-bit frame number. These registers are updated with the current frame number whenever a SOF TOKEN is received.

Address: 4005_5000h base + A0h offset = 4005_50A0h

Bit	7	6	5	4	3	2	1	0
Read	FRM[7:0]							
Write								
Reset	0	0	0	0	0	0	0	0

USBx_FRMNUML field descriptions

Field	Description
FRM[7:0]	This 8-bit field and the 3-bit field in the Frame Number Register High are used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

52.4.18 Frame Number register High (USBx_FRMNUMH)

The Frame Number registers (low and high) contain the 11-bit frame number. These registers are updated with the current frame number whenever a SOF TOKEN is received.

Address: 4005_5000h base + A4h offset = 4005_50A4h

Bit	7	6	5	4	3	2	1	0
Read	0					FRM[10:8]		
Write								
Reset	0	0	0	0	0	0	0	0

USBx_FRMNUMH field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FRM[10:8]	This 3-bit field and the 8-bit field in the Frame Number Register Low are used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

52.4.19 Token register (USBx_TOKEN)

Used to initiate USB transactions when in host mode (CTL[HOSTMODEEN]=1). When the software needs to execute a USB transaction to a peripheral, it writes the TOKEN type and endpoint to this register. After this register has been written, the USB module begins the specified USB transaction to the address contained in the address register. The processor core must always check that the TOKEN_BUSY bit in the control register is not 1 before writing to the Token Register. This ensures that the token commands are not overwritten before they can be executed. The address register and endpoint control register 0 are also used when performing a token command and therefore must also be written before the Token Register. The address register is used to select the USB peripheral address transmitted by the token command. The endpoint control register determines the handshake and retry policies used during the transfer.

Address: 4005_5000h base + A8h offset = 4005_50A8h

Bit	7	6	5	4	3	2	1	0
Read	TOKENPID				TOKENENDPT			
Write								
Reset	0	0	0	0	0	0	0	0

USBx_TOKEN field descriptions

Field	Description
7–4 TOKENPID	Contains the token type executed by the USB module. 0001 OUT Token. USB Module performs an OUT (TX) transaction. 1001 IN Token. USB Module performs an In (RX) transaction. 1101 SETUP Token. USB Module performs a SETUP (TX) transaction
TOKENENDPT	Holds the Endpoint address for the token command. The four bit value written must be a valid endpoint.

52.4.20 SOF Threshold register (USBx_SOFTHLD)

The SOF Threshold Register is used only in Host mode (CTL[HOSTMODEEN]=1). When in Host mode, the 14-bit SOF counter counts the interval between SOF frames. The SOF must be transmitted every 1ms so therefore the SOF counter is loaded with a value of 12000. When the SOF counter reaches zero, a Start Of Frame (SOF) token is transmitted.

The SOF threshold register is used to program the number of USB byte times when SOFDYNTHLD=0, or 8 byte times when SOFDYNTHLD=1, before the SOF stops initiating token packet transactions. This register must be set to a value that ensures that other packets are not actively being transmitted when the SOF time counts to zero. When the SOF counter reaches the threshold value, no more tokens are transmitted until after the SOF has been transmitted.

The value programmed into the threshold register must reserve enough time to ensure the worst case transaction completes. In general the worst case transaction is an IN token followed by a data packet from the peripheral device followed by the response from the host. The actual time required is a function of the maximum packet size on the bus.

Typical values for the SOF threshold are:

- 64-byte packets=74;
- 32-byte packets=42;
- 16-byte packets=26;
- 8-byte packets=18.

Address: 4005_5000h base + ACh offset = 4005_50ACh

Bit	7	6	5	4	3	2	1	0
Read Write	CNT							
Reset	0	0	0	0	0	0	0	0

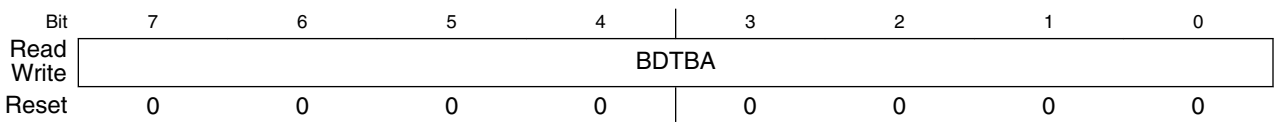
USBx_SOFTHLD field descriptions

Field	Description
CNT	Represents the SOF count threshold in byte times when SOFDYNTHLD=0 or 8 byte times when SOFDYNTHLD=1.

52.4.21 BDT Page Register 2 (USBx_BDTPAGE2)

Contains an 8-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory. See Buffer Descriptor Table.

Address: 4005_5000h base + B0h offset = 4005_50B0h



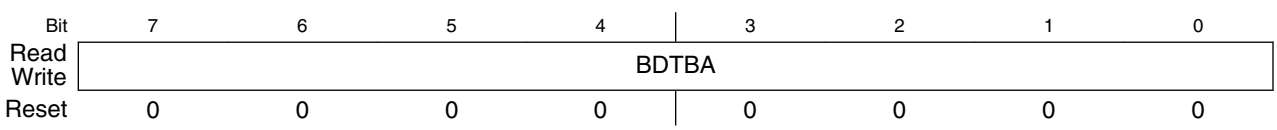
USBx_BDTPAGE2 field descriptions

Field	Description
BDTBA	Provides address bits 23 through 16 of the BDT base address that defines the location of Buffer Descriptor Table resides in system memory.

52.4.22 BDT Page Register 3 (USBx_BDTPAGE3)

Contains an 8-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory. See Buffer Descriptor Table.

Address: 4005_5000h base + B4h offset = 4005_50B4h



USBx_BDTPAGE3 field descriptions

Field	Description
BDTBA	Provides address bits 31 through 24 of the BDT base address that defines the location of Buffer Descriptor Table resides in system memory.

52.4.23 Endpoint Control register (USB*x*_ENDPT*n*)

Contains the endpoint control bits for each of the 16 endpoints available within the USB module for a decoded address. The format for these registers is shown in the following figure. Endpoint 0 (ENDPT0) is associated with control pipe 0, which is required for all USB functions. Therefore, after a USBRST interrupt occurs the processor core should set ENDPT0 to contain 0x0D.

In Host mode ENDPT0 is used to determine the handshake, retry and low speed characteristics of the host transfer. For Control, Bulk and Interrupt transfers, the EPHSHK bit should be 1. For Isochronous transfers it should be 0. Common values to use for ENDPT0 in host mode are 0x4D for Control, Bulk, and Interrupt transfers, and 0x4C for Isochronous transfers.

The three bits EPCTLDIS, EPRXEN, and EPTXEN define if an endpoint is enabled and define the direction of the endpoint. The endpoint enable/direction control is defined in the following table.

Table 52-7. Endpoint enable and direction control

EPCTLDIS	EPRXEN	EPTXEN	Endpoint enable/direction control
X	0	0	Disable endpoint
X	0	1	Enable endpoint for Tx transfers only
X	1	0	Enable endpoint for Rx transfers only
1	1	1	Enable endpoint for Rx and Tx transfers
0	1	1	Enable Endpoint for RX and TX as well as control (SETUP) transfers.

Address: 4005_5000h base + C0h offset + (4d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	HOSTWOHUB	RETRYDIS	0	EPCTLDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK
Write								
Reset	0	0	0	0	0	0	0	0

USB*x*_ENDPT*n* field descriptions

Field	Description
7 HOSTWOHUB	Host without a hub This is a Host mode only field and is present in the control register for endpoint 0 (ENDPT0) only.

Table continues on the next page...

USBx_ENDPTn field descriptions (continued)

Field	Description
	0 Low-speed device connected to Host through a hub. PRE_PID will be generated as required. 1 Low-speed device directly connected. No hub, or no low-speed device attached.
6 RETRYDIS	This is a Host mode only bit and is present in the control register for endpoint 0 (ENDPT0) only. When set this bit causes the host to not retry NAK'ed (Negative Acknowledgement) transactions. When a transaction is NAKed, the BDT PID field is updated with the NAK PID, and the TOKEN_DNE interrupt is set. When this bit is cleared, NAKed transactions are retried in hardware. This bit must be set when the host is attempting to poll an interrupt endpoint.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 EPCTLDIS	This bit, when set, disables control (SETUP) transfers. When cleared, control transfers are enabled. This applies if and only if the EPRXEN and EPTXEN bits are also set. See Table 52-7
3 EPRXEN	This bit, when set, enables the endpoint for RX transfers. See Table 52-7
2 EPTXEN	This bit, when set, enables the endpoint for TX transfers. See Table 52-7
1 EPSTALL	When set, this bit indicates that the endpoint is stalled. This bit has priority over all other control bits in this register, but it is only valid if EPTXEN=1 or EPRXEN=1. Any access to this endpoint causes the USB Module to return a STALL handshake. After an endpoint is stalled it requires intervention from the Host Controller.
0 EPHSHK	When set this bit enables an endpoint to perform handshaking during a transaction to this endpoint. This bit is generally 1 unless the endpoint is Isochronous.

52.4.24 USB Control register (USBx_USBCTRL)

Address: 4005_5000h base + 100h offset = 4005_5100h

Bit	7	6	5	4	3	2	1	0
Read	SUSP	PDE	UARTCHLS	UARTSEL	0			
Write								
Reset	1	1	0	0	0	0	0	0

USBx_USBCTRL field descriptions

Field	Description
7 SUSP	Places the USB transceiver into the suspend state. 0 USB transceiver is not in suspend state. 1 USB transceiver is in suspend state.
6 PDE	Enables the weak pulldowns on the USB transceiver. 0 Weak pulldowns are disabled on D+ and D–. 1 Weak pulldowns are enabled on D+ and D–.
5 UARTCHLS	UART Signal Channel Select This field is valid only when USB signals are selected to be used as UART signals.

Table continues on the next page...

USBx_USBCTRL field descriptions (continued)

Field	Description
	0 USB DP/DM signals used as UART TX/RX. 1 USB DP/DM signals used as UART RX/TX.
4 UARTSEL	Selects USB signals to be used as UART signals. 0 USB signals not used as UART signals. 1 USB signals used as UART signals.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.25 USB OTG Observe register (USBx_OBSERVE)

Provides visibility on the state of the pull-ups and pull-downs at the transceiver. Useful when interfacing to an external OTG control module via a serial interface.

Address: 4005_5000h base + 104h offset = 4005_5104h

Bit	7	6	5	4	3	2	1	0
Read	DPPU	DPPD	0	DMPD	0			
Write								
Reset	0	1	0	1	0	0	0	0

USBx_OBSERVE field descriptions

Field	Description
7 DPPU	Provides observability of the D+ Pullup enable at the USB transceiver. 0 D+ pullup disabled. 1 D+ pullup enabled.
6 DPPD	Provides observability of the D+ Pulldown enable at the USB transceiver. 0 D+ pulldown disabled. 1 D+ pulldown enabled.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 DMPD	Provides observability of the D- Pulldown enable at the USB transceiver. 0 D– pulldown disabled. 1 D– pulldown enabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.26 USB OTG Control register (USB*x*_CONTROL)

Address: 4005_5000h base + 108h offset = 4005_5108h

Bit	7	6	5	4
Read	0			DPPULLUPNONOTG
Write				
Reset	0	0	0	0

Bit	3	2	1	0
Read	0			
Write				
Reset	0	0	0	0

USB*x*_CONTROL field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 DPPULLUPNONOTG	Provides control of the DP Pullup in USBOTG, if USB is configured in non-OTG device mode. 0 DP Pullup in non-OTG device mode is not enabled. 1 DP Pullup in non-OTG device mode is enabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.27 USB Transceiver Control register 0 (USB*x*_USBTRC0)

Includes signals for basic operation of the on-chip USB Full Speed transceiver and configuration of the USB data connection that are not otherwise included in the USB Full Speed controller registers. VREGIN interrupt detection using the VFEDG_DET and VREDG_DET bitfields may be used for VBUS detection in bus-powered device use cases when the USB receptacle VBUS pin is connected to VREGIN.

Address: 4005_5000h base + 10Ch offset = 4005_510Ch

Bit	7	6	5	4
Read		0	USBRESMEN	VFEDG_DET
Write	USBRESET			
Reset	0	0	0	0

Bit	3	2	1	0
Read	VREDG_DET	USB_CLK_ RECOVERY_INT	SYNC_DET	USB_RESUME_INT
Write				
Reset	0	0	0	0

USB*x*_USBTRC0 field descriptions

Field	Description
7 USBRESET	USB Reset Generates a hard reset to USBOTG. After this bit is set and the reset occurs, this bit is automatically cleared. **NOTE: This bit is always read as zero. Wait two USB clock cycles after setting this bit before accessing other USB register bit fields.** 0 Normal USB module operation. 1 Returns the USB module to its reset state.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 USBRESMEN	Asynchronous Resume Interrupt Enable This bit, when set, allows the USB module to send an asynchronous wakeup event to the MCU upon detection of resume signaling on the USB bus. The MCU then re-enables clocks to the USB module. It is used for low-power suspend mode when USB module clocks are stopped or the USB transceiver is in Suspend mode. Async wakeup only works in device mode. 0 USB asynchronous wakeup from suspend mode disabled. 1 USB asynchronous wakeup from suspend mode enabled. The asynchronous resume interrupt differs from the synchronous resume interrupt in that it asynchronously detects K-state using the unfiltered state of the D+ and D– pins. This interrupt should only be enabled when the Transceiver is suspended.
4 VFEDG_DET	VREGIN Falling Edge Interrupt Detect Use USB*x*_MISCCTRL[VFEDG_EN] to enable this bitfield. 0 VREGIN falling edge interrupt has not been detected. 1 VREGIN falling edge interrupt has been detected.
3 VREDG_DET	VREGIN Rising Edge Interrupt Detect Use USB*x*_MISCCTRL[VREDG_EN] to enable this bitfield. 0 VREGIN rising edge interrupt has not been detected. 1 VREGIN rising edge interrupt has been detected.
2 USB_CLK_ RECOVERY_INT	Combined USB Clock Recovery interrupt status This read-only field will be set to value high at 1'b1 when any of USB clock recovery interrupt conditions are detected and those interrupts are unmasked. For customer use the only unmasked USB clock recovery interrupt condition results from an overflow of the frequency trim setting values indicating that the frequency trim calculated is out of the adjustment range of the FIRC output clock. To clear this bit after it has been set, Write 0xFF to register USB_CLK_RECOVER_INT_STATUS.
1 SYNC_DET	Synchronous USB Interrupt Detect 0 Synchronous interrupt has not been detected. 1 Synchronous interrupt has been detected.
0 USB_RESUME_ INT	USB Asynchronous Interrupt 0 No interrupt was generated. 1 Interrupt was generated because of the USB asynchronous interrupt.

52.4.28 Frame Adjust Register (USBx_USBFRMADJUST)

Address: 4005_5000h base + 114h offset = 4005_5114h

Bit	7	6	5	4	3	2	1	0
Read Write	ADJ							
Reset	0	0	0	0	0	0	0	0

USBx_USBFRMADJUST field descriptions

Field	Description
ADJ	Frame Adjustment In Host mode, the frame adjustment is a twos complement number that adjusts the period of each USB frame in 12-MHz clock periods. A SOF is normally generated every 12,000 12-MHz clock cycles. The Frame Adjust Register can adjust this by -128 to +127 to compensate for inaccuracies in the USB 48-MHz clock. Changes to the ADJ bit take effect at the start of the next frame.

52.4.29 Miscellaneous Control register (USBx_MISCCTRL)

Address: 4005_5000h base + 12Ch offset = 4005_512Ch

Bit	7	6	5	4
Read	0			VFEDG_EN
Write				
Reset	0	0	0	0

Bit	3	2	1	0
Read Write	VREDG_EN	OWNERRISODIS	SOFBUSSET	SOFDYNTHLD
Reset	0	0	0	0

USBx_MISCCTRL field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 VFEDG_EN	VREGIN Falling Edge Interrupt Enable 0 VREGIN falling edge interrupt disabled. 1 VREGIN falling edge interrupt enabled.
3 VREDG_EN	VREGIN Rising Edge Interrupt Enable 0 VREGIN rising edge interrupt disabled. 1 VREGIN rising edge interrupt enabled.
2 OWNERRISODIS	OWN Error Detect for ISO IN / ISO OUT Disable This field is only valid for Peripheral mode, that is, CTL[HOSTMODEEN]=0.

Table continues on the next page...

USBx_MISCCTRL field descriptions (continued)

Field	Description
	0 OWN error detect for ISO IN / ISO OUT is not disabled. 1 OWN error detect for ISO IN / ISO OUT is disabled.
1 SOFBUSSET	SOF_TOK Interrupt Generation Mode Select This field is only valid for Host mode, that is, CTL[HOSTMODEEN]=1. 0 SOF_TOK interrupt is set according to SOF threshold value. 1 SOF_TOK interrupt is set when SOF counter reaches 0.
0 SOFDYNTHLD	Dynamic SOF Threshold Compare mode This field is only valid for Host mode, that is, CTL[HOSTMODEEN]=1. 0 SOF_TOK interrupt is set when byte times SOF threshold is reached. 1 SOF_TOK interrupt is set when 8 byte times SOF threshold is reached or overstepped.

52.4.30 USB Clock recovery control (USBx_CLK_RECOVER_CTRL)

Signals in this register control the crystal-less USB clock mode in which the internal FIRC oscillator is tuned to match the clock extracted from the incoming USB data stream.

The FIRC must be enabled in the SCG module.

Address: 4005_5000h base + 140h offset = 4005_5140h

Bit	7	6	5	4	3	2	1	0
Read Write	CLOCK_ RECOVER_ EN	RESET_ RESUME_ ROUGH_EN	RESTART_ IFRTRIM_ EN	Reserved		Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0

USBx_CLK_RECOVER_CTRL field descriptions

Field	Description
7 CLOCK_ RECOVER_EN	Crystal-less USB enable This bit must be enabled if user wants to use the crystal-less USB mode for the Full Speed USB controller and transceiver. **NOTE: This bit should not be set for USB host mode or OTG.** 0 Disable clock recovery block (default) 1 Enable clock recovery block
6 RESET_ RESUME_ ROUGH_EN	Reset/resume to rough phase enable The clock recovery block tracks the FIRC to get an accurate 48Mhz clock. It has two phases after user enables clock_recover_en bit, rough phase and tracking phase. The step to fine tune the FIRC by adjusting the trim fine value is different during these 2 phases. The step in rough phase is larger than that in tracking phase. Switch back to rough stage whenever USB bus reset or bus resume occurs.

Table continues on the next page...

USBx_CLK_RECOVER_CTRL field descriptions (continued)

Field	Description
	0 Always works in tracking phase after the first time rough to track transition (default) 1 Go back to rough stage whenever bus reset or bus resume occurs
5 RESTART_ IFRTRIM_EN	Restart from IFR trim value FIRC has a default trim fine value whose default value is factory trimmed (the IFR trim value). Clock recover block tracks the accuracy of the clock 48Mhz and keeps updating the trim fine value accordingly 0 Trim fine adjustment always works based on the previous updated trim fine value (default) 1 Trim fine restarts from the IFR trim value whenever bus_reset/bus_resume is detected or module enable is desasserted
4–3 Reserved	This field is reserved.
2 Reserved	This field is reserved. This bit is for NXP use only. Customers should not change this bit from its default state.
1 Reserved	This field is reserved. This bit is for NXP use only. Customers should not change this bit from its default state.
0 Reserved	This field is reserved. Default should not be changed

52.4.31 FIRC oscillator enable register (USBx_CLK_RECOVER_IRC_EN)

This register is reserved.

Address: 4005_5000h base + 144h offset = 4005_5144h

Bit	7	6	5	4	3	2	1	0
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

USBx_CLK_RECOVER_IRC_EN field descriptions

Field	Description
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

52.4.31 Clock recovery combined interrupt enable (USB*x*_CLK_RECOVER_INT_EN)

Enables or masks the individual interrupt flags which are logically OR'ed together to produce the combined interrupt indication on the USB_CLK_RECOVERY_INT bit in the USB_USBTRC0 register if the indicated conditions have been detected in the USB clock recovery algorithm operation.

Address: 4005_5000h base + 154h offset = 4005_5154h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved			OVF_ ERROR_EN	Reserved			
Reset	0	0	0	1	0	0	0	0

USB*x*_CLK_RECOVER_INT_EN field descriptions

Field	Description
7–5 Reserved	This field is reserved. Should always be written as 0.
4 OVF_ERROR_ EN	Determines whether OVF_ERROR condition signal is used in generation of USB_CLK_RECOVERY_INT. 0 The interrupt will be masked 1 The interrupt will be enabled (default)
Reserved	This field is reserved. Should always be written as 0.

52.4.32 Clock recovery separated interrupt status (USB*x*_CLK_RECOVER_INT_STATUS)

A Write operation with value high at 1'b1 on any combination of individual bits will clear those bits.

Address: 4005_5000h base + 15Ch offset = 4005_515Ch

Bit	7	6	5	4	3	2	1	0
Read	Reserved			OVF_ ERROR	Reserved			
Write	w1c			w1c	w1c			
Reset	0	0	0	0	0	0	0	0

USBx_CLK_RECOVER_INT_STATUS field descriptions

Field	Description
7–5 Reserved	This field is reserved. Should always be written as 0.
4 OVF_ERROR	Indicates that the USB clock recovery algorithm has detected that the frequency trim adjustment needed for the FIRC output clock is outside the available TRIM_FINE adjustment range for the FIRC module. 0 No interrupt is reported 1 Unmasked interrupt has been generated
Reserved	This field is reserved. Should always be written as 0.

52.5 OTG and Host mode operation

The Host mode logic allows portable, mobile, and other devices using this SOC to function as a USB Embedded Host (EH) Controller. The OTG logic adds an interface to allow the OTG Host Negotiation and Session Request Protocols (HNP and SRP) to be implemented in software.

Host mode is intended for use in handheld-portable devices to allow easy connection to simple HID class devices such as printers and keyboards. It is not intended to perform the functions of a full OHCI or UHCI compatible host controller found on PC motherboards. Host mode allows bulk, isochronous, interrupt and control transfers. Bulk data transfers are performed at nearly the full USB interface bandwidth. Support is provided for ISO transfers, but the number of ISO streams that can be practically supported is affected by the interrupt latency of the processor servicing the Token Done interrupts from the SIE. Custom drivers must be written to support Host mode operation.

Setting the HOST_MODE_EN bit in the CTL register enables Host mode. The USB-FS core can only operate as a peripheral device or in Host mode. It cannot operate in both modes simultaneously. When HOST_MODE is enabled, only endpoint zero is used. All other endpoints should be disabled by software.

52.6 Host Mode Operation Examples

The following sections illustrate the steps required to perform USB host functions using the USB-FS core. For more information about these procedures, see the *Universal Serial Bus Specification, Revision 2.0*, "Chapter 9 USB Device Framework."

To enable host mode and discover a connected device:

1. Enable Host Mode (CTL[HOST_MODE_EN]=1). The pull-down resistors are enabled, and pull-up disabled. Start of Frame (SOF) generation begins. SOF counter loaded with 12,000. Disable SOF packet generation to eliminate noise on the USB by writing the USB enable bit to 0 (CTL[USB_EN]=0).
2. Enable the ATTACH interrupt (INTEN[ATTACHEN]=1).
3. Wait for ATTACH interrupt (ISTAT[ATTACH]). Signaled by USB peripheral device pull-up resistor changing the state of DPLUS or DMINUS from 0 to 1 (SE0 to J or K state).
4. Check the state of the JSTATE and SE0 bits in the control register. If the connecting device is low speed (JSTATE bit is 0), set the low-speed bit in the address registers (ADDR[LS_EN]=1) and the Host Without Hub bit in endpoint 0 register control (ENDPT0[HOSTWOHUB]=1).
5. Enable RESET (CTL[RESET]=1) for 10 ms.
6. Enable SOF packet to keep the connected device from going to suspend (CTL[USB_EN=1]).
7. Enumerate the attached device by sending the appropriate commands to the default control pipe of the connected device.

To complete a control transaction to a connected device:

1. Complete all the steps to discover a connected device
2. Set up the endpoint control register for bidirectional control transfers ENDPT0[4:0] = 0x0d.
3. Place a copy of the device framework setup command in a memory buffer.
4. Initialize current even or odd TX EP0 BDT to transfer the 8 bytes of command data for a device framework command (for example, a GET DEVICE DESCRIPTOR).
   - Set the BDT command word to 0x00080080 –Byte count to 8, OWN bit to 1.
   - Set the BDT buffer address field to the start address of the 8 byte command buffer.
5. Set the USB device address of the peripheral device in the address register (ADDR[6:0]). After the USB bus reset, the device USB address is zero. It is set to some other value usually 1 by the Set Address device framework command.

6. Write the TOKEN register with a SETUP to Endpoint 0, the peripheral device default control pipe (TOKEN=0xD0). This initiates a setup token on the bus followed by a data packet. The device handshake is returned in the BDT PID field after the packets complete. When the BDT is written, a Token Done (ISTAT[TOKDNE]) interrupt is asserted. This completes the setup phase of the setup transaction.
7. To initiate the data phase of the setup transaction (that is, get the data for the GET DEVICE DESCRIPTOR command), set up a buffer in memory for the data to be transferred.
8. Initialize the current even or odd TX EP0 BDT to transfer the data.
    - Set the BDT command word to 0x004000C0 – BC to 64 (the byte count of the data buffer in this case), OWN bit to 1, Data toggle to Data1.
    - Set the BDT buffer address field to the start address of the data buffer
9. Write the TOKEN register with an IN or OUT token to Endpoint 0, the peripheral device default control pipe, an IN token for a GET DEVICE DESCRIPTOR command (TOKEN=0x90). This initiates an IN token on the bus followed by a data packet from the device to the host. When the data packet completes, the BDT is written and a Token Done (ISTAT[DNE]) interrupt is asserted. For control transfers with a single packet data phase this completes the data phase of the setup transaction.
10. To initiate the status phase of the setup transaction, set up a buffer in memory to receive or send the zero length status phase data packet.
11. Initialize the current even or odd TX EP0 BDT to transfer the status data.
    - Set the BDT command word to 0x00000080 – BC to 0 (the byte count of the data buffer in this case), OWN bit to 1, Data toggle to Data1.
    - Set the BDT buffer address field to the start address of the data buffer
12. Write the TOKEN register with a IN or OUT token to Endpoint 0, the peripheral device default control pipe, an OUT token for a GET DEVICE DESCRIPTOR command (TOKEN=0x10). This initiates an OUT token on the bus followed by a zero length data packet from the host to the device. When the data packet completes, the BDT is written with the handshake from the device and a Token Done (ISTAT[TOKDNE]) interrupt is asserted. This completes the data phase of the setup transaction.

To send a full speed bulk data transfer to a peripheral device:

1. Complete all steps to discover a connected device and to configure a connected device. Write the ADDR register with the address of the peripheral device. Typically, there is only one other device on the USB bus in host mode so it is expected that the address is 0x01 and should remain constant.
2. Write 0x1D to ENDPT0 register to enable transmit and receive transfers with handshaking enabled.
3. Setup the even TX EP0 BDT to transfer up to 64 bytes.
4. Set the USB device address of the peripheral device in the address register (ADDR[6:0]).
5. Write the TOKEN register with an OUT token to the desired endpoint. The write to this register triggers the USB-FS transmit state machines to begin transmitting the token and the data.
6. Setup the odd TX EP0 BDT to transfer up to 64 bytes.
7. Write the TOKEN register with an OUT token as in step 4. Two tokens can be queued at a time to allow the packets to be double buffered to achieve maximum throughput.
8. Wait for the TOKDNE interrupt. This indicates that one of the BDTs has been released back to the processor and the transfer has completed. If the peripheral device asserts NAKs, the USB-FS continues to retry the transfer indefinitely without processor intervention unless the ENDPT0[RETRYDIS] is 1. If the retry disable field is set, the handshake (ACK, NAK, STALL, or ERROR (0xf)) is returned in the BDT PID field. If a stall interrupt occurs, the pending packet must be dequeued and the error condition in the peripheral device cleared. If a Reset interrupt occurs (SE0 for more than 2.5 μs), the peripheral device has detached.
9. After the TOK_DNE interrupt occurs, the BDTs can be examined and the next data packet queued by returning to step 2.

52.7 On-The-Go operation

The USB-OTG core provides sensors and controls to enable On-The-Go (OTG) operation. These sensors are used by the OTG driver software to implement the Host Negotiation Protocol (HNP) and Session Request Protocol (SRP) and give access to the OTG protocol control signals. The following state machines show the OTG operations involved with HNP and SRP protocols from either end of the USB cable.

52.7.1 OTG dual role A device operation

A device is considered the A device because of the type of cable attached. If the USB Type Standard-A or Micro-A plug is plugged into the device, it is considered the A device.

A dual role A device operates as the following flow diagram and state description table illustrates.

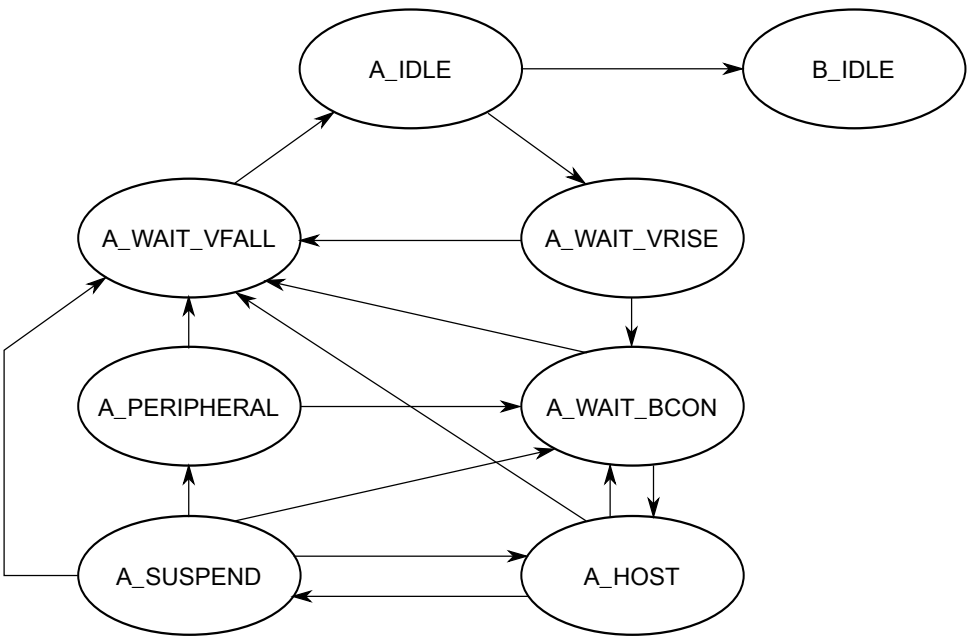


Figure 52-8. Dual role A device flow diagram

Table 52-8. State descriptions for the dual role A device flow

State	Action	Response
A_IDLE	If ID Interrupt. The cable has been unplugged or a Type B cable has been attached. The device now acts as a Type B device.	Go to B_IDLE
	If the A application wants to use the bus or if the B device is doing an SRP as indicated by an A_SESS_VLD Interrupt or Attach or Port Status Change Interrupt check data line for 5 –10 msec pulsing.	Go to A_WAIT_VRISE Turn on DRV_VBUS
A_WAIT_VRISE	If ID Interrupt or if A_VBUS_VLD is false after 100 msec The cable has been changed or the A device cannot support the current required from the B device.	Go to A_WAIT_VFALL Turn off DRV_VBUS
	If A_VBUS_VLD interrupt	Go to A_WAIT_BCON
A_WAIT_BCON	After 200 ms without Attach or ID Interrupt. (This could wait forever if desired.)	Go to A_WAIT_FALL Turn off DRV_VBUS
	A_VBUS_VLD Interrupt and B device attaches	Go to A_HOST

Table continues on the next page...

Table 52-8. State descriptions for the dual role A device flow (continued)

State	Action	Response
		Turn on Host mode
A_HOST	Enumerate Device determine OTG Support.	
	If A_VBUS_VLD/ Interrupt or A device is done and does not think it wants to do something soon or the B device disconnects	Go to A_WAIT_VFALL Turn off Host mode Turn off DRV_VBUS
	If the A device is finished with session or if the A device wants to allow the B device to take bus.	Go to A_SUSPEND
	ID Interrupt or the B device disconnects	Go to A_WAIT_BCON
A_SUSPEND	If ID Interrupt, or if 150 ms B disconnect timeout (This timeout value could be longer) or if A_VBUS_VLD\ Interrupt	Go to A_WAIT_VFALL Turn off DRV_VBUS
	If HNP enabled, and B disconnects in 150 ms then B device is becoming the host.	Go to A_PERIPHERAL Turn off Host mode
	If A wants to start another session	Go to A_HOST
A_PERIPHERAL	If ID Interrupt or if A_VBUS_VLD interrupt	Go to A_WAIT_VFALL Turn off DRV_VBUS.
	If 3 –200 ms of Bus Idle	Go to A_WAIT_BCON Turn on Host mode
A_WAIT_VFALL	If ID Interrupt or (A_SESS_VLD/ & b_conn/)	Go to A_IDLE

52.7.2 OTG dual role B device operation

A device is considered a B device if it is connected to the bus with a USB Type Standard-B, Mini-B, or Micro-B plug inserted into the local USB receptacle. The Type Mini-B plug and receptacle are now only allowed for dedicated peripheral devices, not dual role/OTG devices.

A dual role B device operates as the following flow diagram and state description table illustrates.

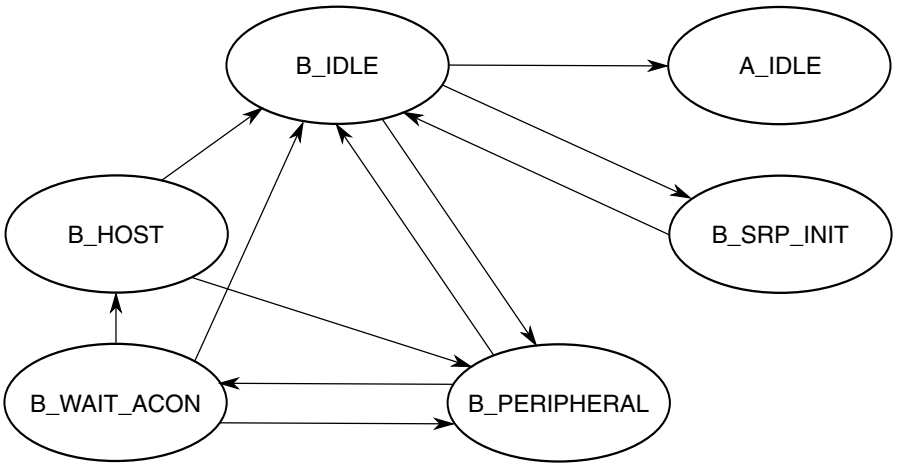


Figure 52-9. Dual role B device flow diagram

Table 52-9. State descriptions for the dual role B device flow

State	Action	Response
B_IDLE	If ID\ Interrupt. A Type A cable has been plugged in and the device should now respond as a Type A device.	Go to A_IDLE
	If B_SESS_VLD Interrupt. The A device has turned on VBUS and begins a session.	Go to B_PERIPHERAL Turn on DP_HIGH
	If B application wants the bus and Bus is Idle for 2 ms and the B_SESS_END bit is set, the B device can perform an SRP.	Go to B_SRP_INIT Pulse CHRG_VBUS Pulse DP_HIGH 5-10 ms
B_SRP_INIT	If ID\ Interrupt or SRP Done (SRP must be done in less than 100 ms.)	Go to B_IDLE
B_PERIPHERAL	If HNP enabled and the bus is suspended and B wants the bus, the B device can become the host.	Go to B_WAIT_ACON Turn off DP_HIGH
B_WAIT_ACON	If A connects, an attach interrupt is received	Go to B_HOST Turn on Host Mode
	If ID\ Interrupt or B_SESS_VLD/ Interrupt If the cable changes or if VBUS goes away, the host doesn't support us. Go to B_IDLE	Go to B_IDLE
	If 3.125 ms expires or if a Resume occurs	Go to B_PERIPHERAL
B_HOST	If ID\ Interrupt or B_SESS_VLD\ Interrupt If the cable changes or if VBUS goes away, the host doesn't support us.	Go to B_IDLE
	If B application is done or A disconnects	Go to B_PERIPHERAL

52.8 Device mode FIRC operation

The following are the FIRC initialization code steps:

1. Enable the FIRC in the SCG module
2. Enable the USB clock recovery tuning:
   USB_CLK_RECOVER_CTRL[CLOCK_RECOVER_EN] = 1b
3. Choose the clock source of USB by configuring the muxes .
4. The USB clock source must choose the output of the divided clock.

For chip-specific details, see the USB FS OTG controller clocking information in the "Clock Distribution" chapter.

Chapter 53
USB Voltage Regulator (VREG)

53.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The USB Voltage Regulator module is a LDO linear voltage regulator to provide 3.3V power from an input power supply varying from 2.7 V to 5.5 V. It consists of one 3.3 V power channel. When the input power supply is below 3.6 V, the regulator goes to pass-through mode. The following figure shows the ideal relation between the regulator output and input power supply.

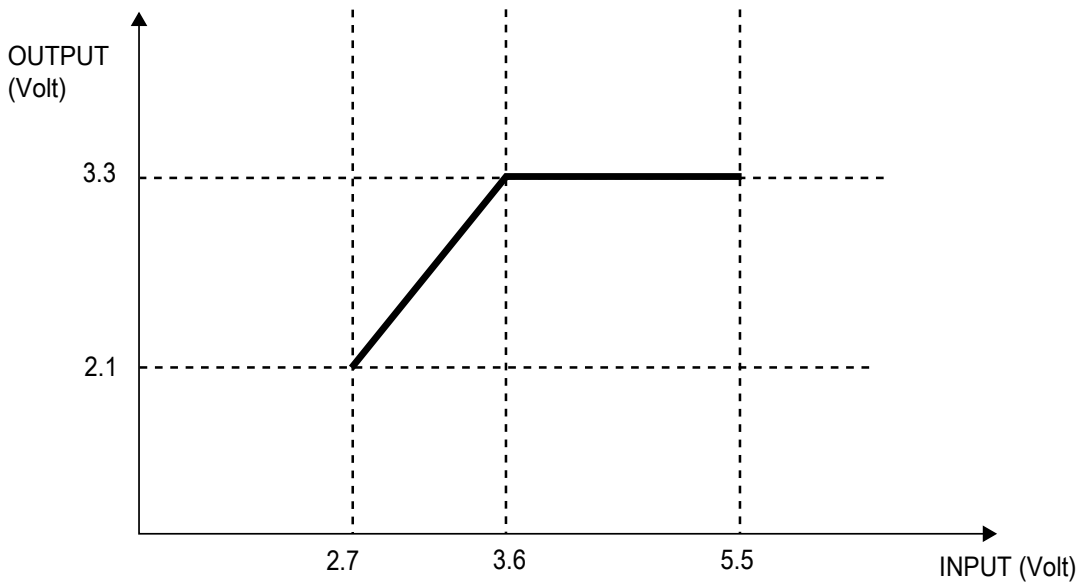


Figure 53-1. Ideal Relation Between the Regulator Output and Input Power Supply

53.1.1 Overview

A simplified block diagram for the USB Voltage Regulator module is shown below.

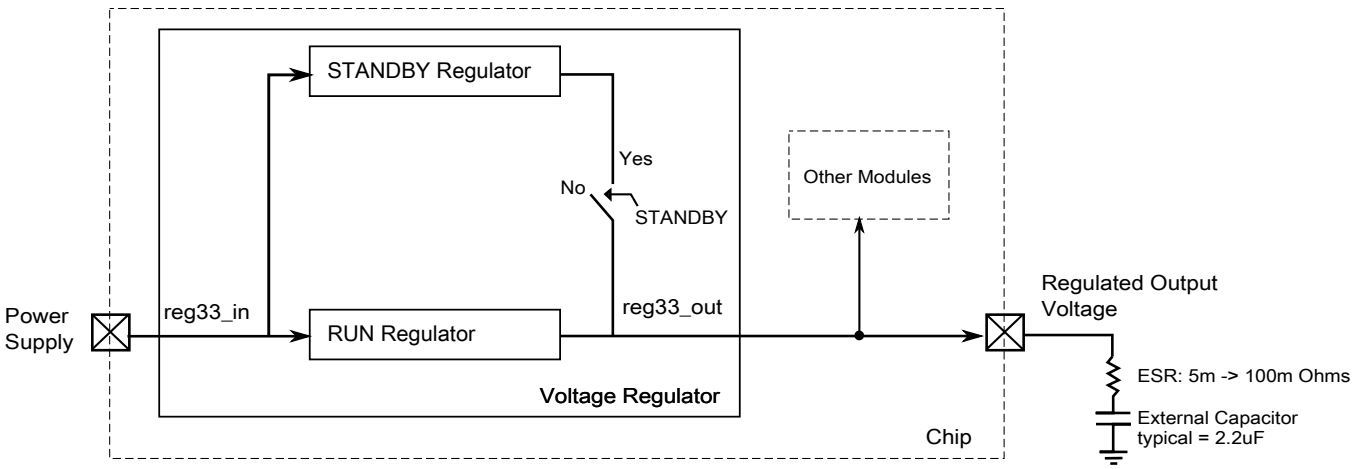


Figure 53-2. USB Voltage Regulator Block Diagram

This module uses 2 regulators in parallel. In run mode, the RUN regulator with the bandgap voltage reference is enabled and can provide up to 120 mA load current. In run mode, the STANDBY regulator and the low power reference are also enabled, but a switch disconnects its output from the external pin. In STANDBY mode, the RUN regulator is disabled and the STANDBY regulator output is connected to the external pin.

Internal power mode signals control whether the module is in RUN or STANDBY mode.

53.1.2 Features

- Low drop-out linear voltage regulator with one power channel (3.3 V).
- Low drop-out voltage: 300 mV.
- Output current: 120 mA.
- Three different power modes: RUN, STANDBY and SHUTDOWN.
- Low quiescent current in RUN mode.
  - Typical value is around 120 µA (one thousand times smaller than the maximum load current).
- Very low quiescent current in STANDBY mode.
  - Typical value is around 1 µA.
- Automatic current limiting if the load current is greater than 290 mA.

- Automatic power-up once some voltage is applied to the regulator input.
- Pass-through mode for regulator input voltages less than 3.6 V
- Small output capacitor: 2.2 μF
- Stable with aluminum, tantalum or ceramic capacitors.

53.1.3 Modes of Operation

The regulator has these power modes:

- RUN—The regulating loop of the RUN regulator and the STANDBY regulator are active, but the switch connecting the STANDBY regulator output to the external pin is open.
- STANDBY—The regulating loop of the RUN regulator is disabled and the standby regulator is active. The switch connecting the STANDBY regulator output to the external pin is closed.
- SHUTDOWN—The module is disabled.

The regulator is enabled by default. This means that once the power supply is provided, the module power-up sequence to RUN mode starts.

53.2 USB Voltage Regulator Module Signal Descriptions

The following table shows the external signals for the regulator.

Table 53-1. USB Voltage Regulator Module Signal Descriptions

Signal	Description	I/O
reg33_in	Unregulated power supply	I
reg33_out	Regulator output voltage	O

Chapter 54
Voltage Reference (VREF)

54.1 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The VREF can be used in applications to provide a reference voltage to external devices, or used internally in the device as a reference to analog peripherals (such as the ADC, DAC, or CMP). The Voltage Reference (VREF) can supply an accurate voltage output that can be trimmed in 0.5 mV steps (for 1.2 V output) or 1.5 mV steps (for 2.1 V output). The voltage reference has 3 operating modes that provide different levels of supply rejection and power consumption.

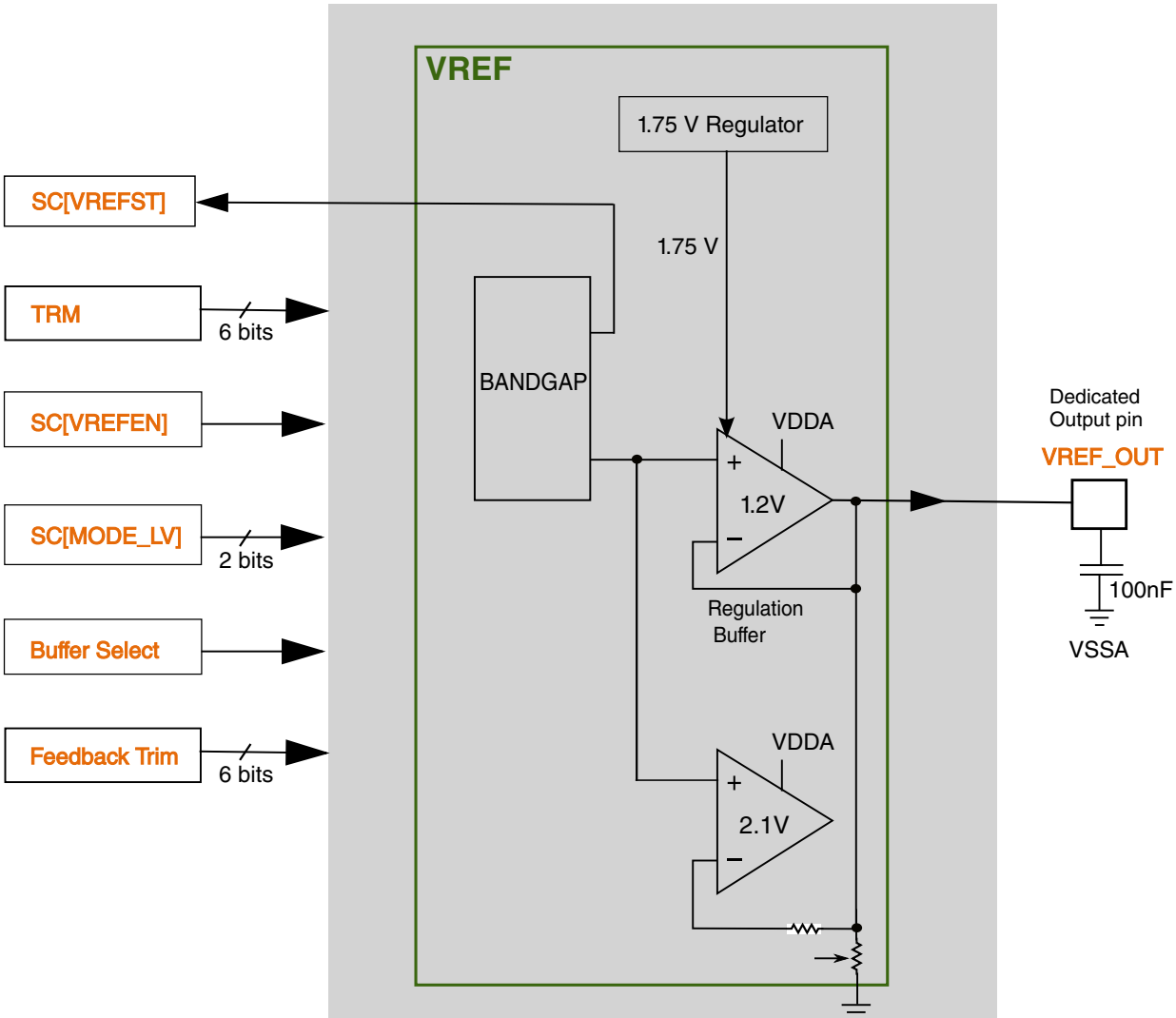


Figure 54-1. Voltage reference block diagram

54.1.1 Overview

The Voltage Reference provides a buffered reference voltage for use as an external reference, which can be set to either 1.2V or 2.1V. In addition, the buffered reference is available internally for use with on-chip peripherals (such as ADCs and DACs). Refer to the chip configuration details. When the VREF is enabled, the reference voltage is placed on a dedicated output pin.

- In 1.2V mode, the Voltage Reference output can be trimmed with 0.5mV resolution, using the TRM register TRIM[5:0] bitfield.
- In 2.1V mode, the Voltage Reference output can be trimmed with 1.5mV resolution, using the TRM register VREF_TRM4 TRIM2V1[5:0] bitfield.

54.1.2 Features

- Programmable trim register with 0.5 mV steps in 1.2V mode and 1.5mV steps in 2.1V mode; upon reset, the trim register is automatically loaded with a factory-trimmed value
- Programmable buffer mode selection:
  - Off
  - Bandgap enabled/standby (output buffer disabled)
  - Low power buffer mode (output buffer enabled)
  - High power buffer mode (output buffer enabled)
- Selectable 1.2V or 2.1V output at room temperature
- Dedicated output pin, VREF_OUT

54.1.3 Modes of Operation

The Voltage Reference continues normal operation in Run, Wait, and Stop modes. The Voltage Reference can also run in Very Low Power Run (VLPR), Very Low Power Wait (VLPW) and Very Low Power Stop (VLPS). If it is desired to use the VREF regulator and/or the chop oscillator in the very low power modes, the system reference voltage (also referred to as the bandgap voltage reference) must be enabled in these modes. Refer to the chip configuration details for information on enabling this mode of operation. Having the VREF regulator enabled does increase current consumption. In very low power modes it may be desirable to disable the VREF regulator to minimize current consumption. Note however that the accuracy of the output voltage will be reduced (by as much as several mVs) when the VREF regulator is not used.

NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

54.1.4 VREF Signal Descriptions

The following table shows the Voltage Reference signals properties.

Table 54-1. VREF Signal Descriptions

Signal	Description	I/O
VREF_OUT	Internally-generated Voltage Reference output	O

NOTE

When the VREF output buffer is disabled, the status of the VREF_OUT signal is high-impedence.

54.2 Memory Map and Register Definition

VREF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_2000	VREF Trim Register (VREF_TRM)	8	R/W	See section	54.2.1/1368
4007_2001	VREF Status and Control Register (VREF_SC)	8	R/W	00h	54.2.2/1370
4007_2005	VREF Trim Register 4 (VREF_TRM4)	8	R/W	00h	54.2.3/1371

54.2.1 VREF Trim Register (VREF_TRM)

This register contains the trim data for the Voltage Reference.

Address: 4007_2000h base + 0h offset = 4007_2000h

Bit	7	6	5	4	3	2	1	0
Read Write	FLIP	CHOPEN	TRIM					
Reset	0	0	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

VREF_TRM field descriptions

Field	Description
7 FLIP	Reverses the amplifier polarity. FLIP can be used to test bandgap amplifier offset. The trim value is stored in IFR and loaded out of reset.

Table continues on the next page...

VREF_TRM field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>6
CHOPEN</td><td>Chop oscillator enable. When set, the internal chopping operation is enabled and the internal analog offset will be minimized.

CHOPEN bit is set during factory trimming of the VREF voltage. CHOPEN bit should be written to 1 to achieve the performance stated in the data sheet.

If the internal voltage regulator is being used (REGEN bit is set to 1), then the chop oscillator must also be enabled.

If the chop oscillatorwill be used in very low power modes, then the system (bandgap) voltage reference must also be enabled. See the chip-specific VREF information (also known as "chip configuration" details) to see how this can be achieved.

0 Chop oscillator is disabled.
1 Chop oscillator is enabled.</td></tr>
<tr><td>TRIM</td><td>Trim bits

These bits change the resulting VREF by approximately ± 0.5 mV for each step. The trim value is stored in IFR and loaded out of reset.
When test_mux_en=1:

<table>
<tr><th>TRIM[2:0]</th><th>test_mux_out_pos</th><th>test_mux_out_neg</th></tr>
<tr><td>000</td><td>vbg</td><td>test_res</td></tr>
<tr><td>001</td><td>vbg</td><td>vssa</td></tr>
<tr><td>010</td><td>vbe</td><td>vssa</td></tr>
<tr><td>011</td><td>p7_trim</td><td>vssa</td></tr>
<tr><td>1xx</td><td>v_1p75</td><td>vssa</td></tr>
</table>

NOTE: Min = minimum and max = maximum voltage reference output. For minimum and maximum voltage reference output values, refer to the Data Sheet for this chip.

000000 Min
000001 Max–(31 mV)
....
111111 Max</td></tr>
</table>

54.2.2 VREF Status and Control Register (VREF_SC)

This register contains the control bits that enable the internal voltage reference and select the buffer mode to be used.

Address: 4007_2000h base + 1h offset = 4007_2001h

Bit	7	6	5	4	3	2	1	0
Read	VREFEN	REGEN	ICOMPEN	TRESEN	TMUXEN	VREFST	MODE_LV	
Write								
Reset	0	0	0	0	0	0	0	0

VREF_SC field descriptions

Field	Description
7 VREFEN	Internal Voltage Reference enable Enables the bandgap reference within the Voltage Reference module. **NOTE:** After the VREF is enabled, turning off the clock to the VREF module via the corresponding clock gate register will not disable the VREF. VREF must be disabled using this VREFEN bit. 0 The module is disabled. 1 The module is enabled.
6 REGEN	Regulator enable Enables the internal 1.75 V regulator to produce a constant internal voltage supply in order to reduce the sensitivity to external supply noise and variation. If it is desired to keep the regulator enabled in very low power modes, refer to the Chip Configuration details to see how this can be achieved. REGEN bit should be written to 1 to achieve the performance stated in the data sheet. **NOTE:** See section "Internal voltage regulator" for details on the required sequence to enable the internal regulator. 0 Internal 1.75 V regulator is disabled. 1 Internal 1.75 V regulator is enabled.
5 ICOMPEN	Second order curvature compensation enable ICOMPEN bit should be written to 1 to achieve the performance stated in the data sheet. 0 Disabled 1 Enabled
4 TRESEN	Test second order curvature compensation enable TRESEN bit is accessible only in functional test mode. 0 Disabled 1 Enabled
3 TMUXEN	Test MUX enable

Table continues on the next page...

VREF_SC field descriptions (continued)

Field	Description
	TMUXEN bit is accessible only in functional test mode. 0 Disabled 1 Enabled
2 VREFST	Internal Voltage Reference stable Indicates that the bandgap reference within the Voltage Reference module has finished its startup and stabilization. **NOTE:** VREFST bit is valid only when the chop oscillator is not being used. 0 The module is disabled or not stable. 1 The module is stable.
MODE_LV	Buffer Mode selection Selects the buffer modes for the Voltage Reference module. 00 Bandgap on only, for stabilization and startup 01 High power buffer mode enabled 10 Low-power buffer mode enabled 11 Reserved

54.2.3 VREF Trim Register 4 (VREF_TRM4)

This register contains the enable and trim for VREF 2.1V generation.

Address: 4007_2000h base + 5h offset = 4007_2005h

Bit	7	6	5	4	3	2	1	0
Read	VREF2V1_EN	0	TRIM2V1					
Write								
Reset	0	0	0	0	0	0	0	0

VREF_TRM4 field descriptions

Field	Description
7 VREF2V1_EN	Internal Voltage Reference (2.1V) Enable Enables the bandgap reference within the Voltage Reference module for generating the 2.1V output 0 VREF 2.1V is enabled 1 VREF 2.1V is disabled
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRIM2V1	VREF 2.1V Trim Bits These bits change the resulting VREF by approximately ± 1.5 mV for each step. The trim value is stored in IFR and loaded out of reset.

54.3 Functional Description

The Voltage Reference is a bandgap buffer system. Unity gain amplifiers are used.

The VREF_OUT signal can be used by both internal and external peripherals in low and high power buffer mode. A 100 nF capacitor must always be connected between VREF_OUT and VSSA if the VREF is being used.

The following table shows all possible function configurations of the Voltage Reference.

Table 54-2. Voltage Reference function configurations

SC[VREFEN]	SC[MODE_LV]	Configuration	Functionality
0	X	Voltage Reference disabled	Off
1	00	Voltage Reference enabled, bandgap on only	Startup and standby
1	01	Voltage Reference enabled, high-power buffer on	VREF_OUT available for internal and external use. 100 nF capacitor is required.
1	10	Voltage Reference enabled, low power buffer on	VREF_OUT available for internal and external use. 100 nF capacitor is required.
1	11	Reserved	Reserved

54.3.1 Voltage Reference Disabled, SC[VREFEN] = 0

When SC[VREFEN] = 0, the Voltage Reference is disabled, the VREF bandgap and the output buffers are disabled. The Voltage Reference is in off mode.

54.3.2 Voltage Reference Enabled, SC[VREFEN] = 1

When SC[VREFEN] = 1, the Voltage Reference is enabled, and different modes should be set by the SC[MODE_LV] bits.

54.3.2.1 SC[MODE_LV]=00

The internal VREF bandgap is enabled to generate an accurate 1.2 V output that can be trimmed with the TRM register's TRIM[5:0] bitfield. The bandgap requires some time for startup and stabilization. SC[VREFST] can be monitored to determine if the stabilization and startup is complete when the chop oscillator is not enabled.

If the chop oscillator is being used, the internal bandgap reference voltage settles within the chop oscillator start up time, Tchop_osc_stup.

The output buffer is disabled in this mode, and there is no buffered voltage output. The Voltage Reference is in standby mode. If this mode is first selected and the low power or high power buffer mode is subsequently enabled, there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet.

54.3.2.2 SC[MODE_LV] = 01

The internal VREF bandgap is on. The high power buffer is enabled to generate a buffered 1.2 V voltage to VREF_OUT. It can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from the standby mode (SC[MODE_LV] = 00, SC[VREFEN] = 1) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of Tstup or until SC[VREFST] = 1 when the chop oscillator is not enabled. If the chop oscillator is being used, you must wait the time specified by Tchop_osc_stup (chop oscillator start up time) to ensure the VREF output has stabilized.

In this mode, a 100 nF capacitor is required to connect between the VREF_OUT pin and VSSA.

54.3.2.3 SC[MODE_LV] = 10

The internal VREF bandgap is on. The low power buffer is enabled to generate a buffered 1.2 V voltage to VREF_OUT. It can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from the standby mode (SC[MODE_LV] = 00, SC[VREFEN] = 1) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of Tstup or until SC[VREFST] = 1 when the chop oscillator is not enabled. If the chop oscillator is being used, you must wait the time specified by Tchop_osc_stup (chop oscillator start up time) to ensure the VREF output has stabilized.

In this mode, a 100 nF capacitor is required to connect between the VREF_OUT pin and VSSA.

54.3.2.4 SC[MODE_LV] = 11

Reserved

54.3.3 Internal voltage regulator

The VREF module contains an internal voltage regulator that can be enabled to provide additional supply noise rejection. It is recommended that when possible, this regulator be enabled to provide the optimum VREF performance.

If the internal voltage regulator is being used, the chop oscillator must also be enabled. A specific sequence must be followed when enabling the internal regulator as follows:

1. Enable the chop oscillator (VREF_TRM[CHOPEN] = 1)
2. Configure the VREF_SC register to the desired settings with the internal regulator disabled, VREF_SC[REGEN] = 0
3. Wait > 300ns
4. Enable the internal regulator by setting VREF_SC[REGEN] to 1

54.4 Initialization/Application Information

The Voltage Reference requires some time for startup and stabilization. After SC[VREFEN] = 1, SC[VREFST] can be monitored to determine if the stabilization and startup is completed when the chop oscillator is not enabled. When the chop oscillator is enabled, the settling time of the internal bandgap reference is defined by Tchop_osc_stup (chop oscillator start up time). You must wait this time (Tchop_osc_stup) after the internal bandgap has been enabled to ensure the VREF internal reference voltage has stabilized.

When the Voltage Reference is already enabled and stabilized, changing SC[MODE_LV] will not clear SC[VREFST] but there will be some startup time before the output voltage at the VREF_OUT pin has settled. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. Also, there will be some settling time when a step change of the load current is applied to the VREF_OUT pin. When the 1.75V VREF regulator is disabled, the VREF_OUT voltage will be more sensitive to supply voltage variation. It is recommended to use this regulator to achieve optimum VREF_OUT performance.

The TRM[CHOPEN], SC[REGEN] and SC[ICOMPEN] bits must be written to 1 to achieve the performance stated in the device data sheet.

NOTE

See section "Internal voltage regulator" for details on the required sequence to enable the internal regulator.

Chapter 55
Watchdog timer (WDOG32)

55.1 Introduction

The Watchdog Timer (WDOG) module is an independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDOG module is not serviced (refreshed) within a certain period, it resets the MCU.

55.1.1 Features

Features of the WDOG32 module include:

- Configurable clock source inputs independent from the bus clock
  - Bus clock (slow clock)
  - LPO (1 kHz clock from PMC)
  - SIRC (8 MHz IRC from SCG)
  - ERCLK (external reference clock from SCG)
- Programmable timeout period
  - Programmable 16-bit timeout value
  - Optional fixed 256 clock prescaler when longer timeout periods are needed
- Robust write sequence for counter refresh
  - Refresh sequence of writing 0x02A6 and then 0x80B4 within 16 bus clocks
- Window mode option for the refresh mechanism
  - Programmable 16-bit window value

- Provides robust check that program flow is faster than expected
- Early refresh attempts trigger a reset.

- Optional timeout interrupt to allow post-processing diagnostics
  - Interrupt request to CPU with interrupt vector for an interrupt service routine (ISR)
  - Forced reset occurs 128 bus clocks after the interrupt vector fetch.
- Configuration bits are write-once-after-reset to ensure watchdog configuration cannot be mistakenly altered.
- Robust write sequence for unlocking write-once configuration bits
  - Unlock sequence of writing 0x20C5 and then 0x28D9 within 16 bus clocks for allowing updates to write-once configuration bits
  - Software must make updates within 128 bus clocks after unlocking and before WDOG32 closes the unlock window.

55.1.2 Block diagram

The following figure shows a block diagram of the WDOG module.

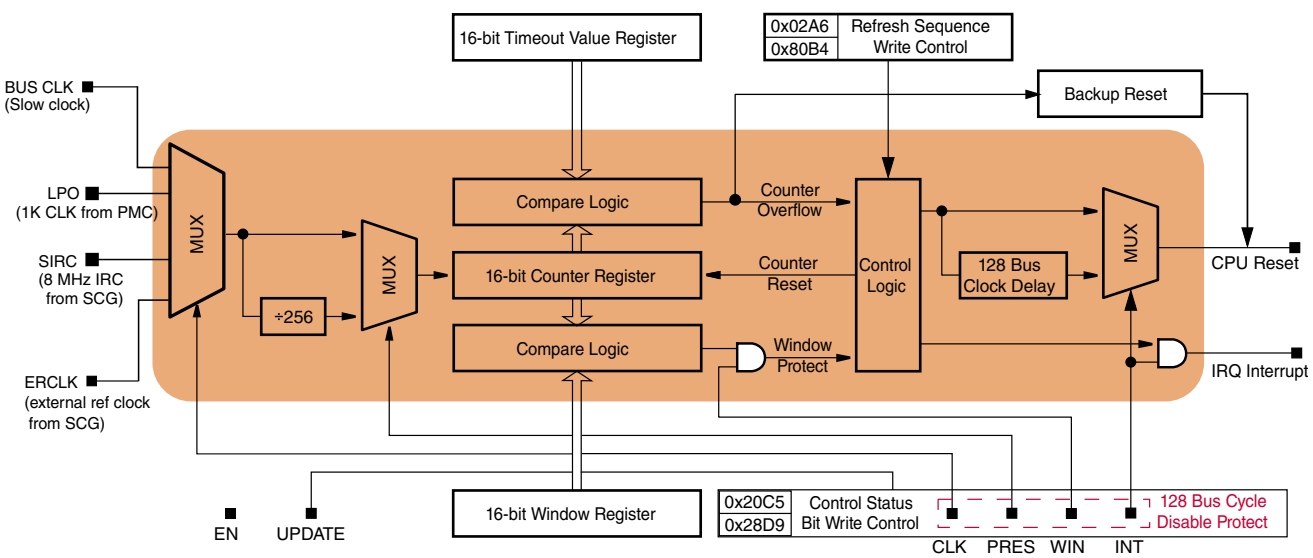


Figure 55-1. WDOG block diagram

55.2 Memory map and register definition

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_6000	Watchdog Control and Status Register (WDOG0_CS)	32	R/W	See section	55.2.1/1379
4007_6004	Watchdog Counter Register (WDOG0_CNT)	32	R/W	0000_0002h	55.2.2/1381
4007_6008	Watchdog Timeout Value Register (WDOG0_TOVAL)	32	R/W	0000_0400h	55.2.3/1382
4007_600C	Watchdog Window Register (WDOG0_WIN)	32	R/W	0000_0000h	55.2.4/1383

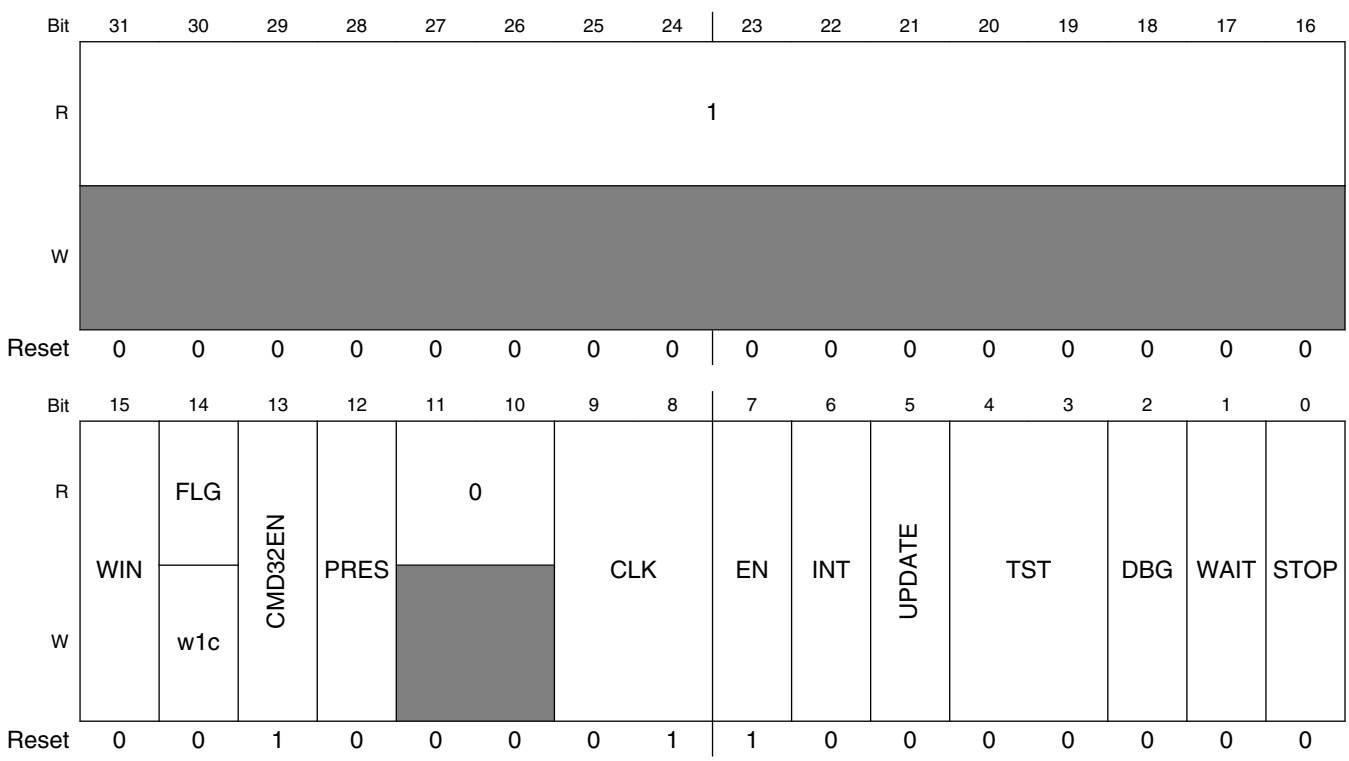
55.2.1 Watchdog Control and Status Register (WDOGx_CS)

This section describes the function of Watchdog Control and Status Register.

NOTE

TST is cleared (0:0) on POR only. Any other reset does not affect the value of this field.

Address: 4007_6000h base + 0h offset = 4007_6000h



WDOGx_CS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
15 WIN	Watchdog Window This write-once bit enables window mode. See the Window mode section. 0 Window mode disabled. 1 Window mode enabled.
14 FLG	Watchdog Interrupt Flag This bit is an interrupt indicator when INT is set in control and status register 1. Write 1 to clear it. 0 No interrupt occurred. 1 An interrupt occurred.
13 CMD32EN	Enables or disables WDOG support for 32-bit (or 16-bit or 8-bit) refresh/unlock command write words 0 Disables support for 32-bit (or 16-bit or 8-bit) refresh/unlock command write words 1 Enables support for 32-bit (or 16-bit or 8-bit) refresh/unlock command write words
12 PRES	Watchdog Prescalar This write-once bit enables a fixed 256 pre-scaling of watchdog counter reference clock. (The block diagram shows this clock divider option.) 0 256 prescalar disabled. 1 256 prescalar enabled.
11–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9–8 CLK	Watchdog Clock This write-once field indicates the clock source that feeds the watchdog counter. See the Clock source section. 00 Bus clock. 01 Internal low-power oscillator (LPOCLK). 10 8 MHz internal reference clock. 11 External clock source.
7 EN	Watchdog Enable This write-once bit enables the watchdog counter to start counting. 0 Watchdog disabled. 1 Watchdog enabled.
6 INT	Watchdog Interrupt This write-once bit configures the watchdog to immediately generate an interrupt request upon a reset-triggering event (timeout or illegal write to the watchdog), before forcing a reset. After the interrupt vector fetch (which comes after the reset-triggering event), the reset occurs after a delay of 128 bus clocks. 0 Watchdog interrupts are disabled. Watchdog resets are not delayed. 1 Watchdog interrupts are enabled. Watchdog resets are delayed by 128 bus clocks from the interrupt vector fetch.

Table continues on the next page...

WDOGx_CS field descriptions (continued)

Field	Description
5 UPDATE	Allow updates This write-once bit allows software to reconfigure the watchdog without a reset. 0 Updates not allowed. After the initial configuration, the watchdog cannot be later modified without forcing a reset. 1 Updates allowed. Software can modify the watchdog configuration registers within 128 bus clocks after performing the unlock write sequence.
4–3 TST	Watchdog Test Enables the fast test mode. The test mode allows software to exercise all bits of the counter to demonstrate that the watchdog is functioning properly. See the Fast testing of the watchdog section. This write-once field is cleared (0:0) on POR only. Any other reset does not affect the value of this field. 00 Watchdog test mode disabled. 01 Watchdog user mode enabled. (Watchdog test mode disabled.) After testing the watchdog, software should use this setting to indicate that the watchdog is functioning normally in user mode. 10 Watchdog test mode enabled, only the low byte is used. CNT[CNTLOW] is compared with TOVAL[TOVALLOW]. 11 Watchdog test mode enabled, only the high byte is used. CNT[CNTHIGH] is compared with TOVAL[TOVALHIGH].
2 DBG	Debug Enable This write-once bit enables the watchdog to operate when the chip is in debug mode. 0 Watchdog disabled in chip debug mode. 1 Watchdog enabled in chip debug mode.
1 WAIT	Wait Enable This write-once bit enables the watchdog to operate when the chip is in wait mode. 0 Watchdog disabled in chip wait mode. 1 Watchdog enabled in chip wait mode.
0 STOP	Stop Enable This write-once bit enables the watchdog to operate when the chip is in stop mode. 0 Watchdog disabled in chip stop mode. 1 Watchdog enabled in chip stop mode.

55.2.2 Watchdog Counter Register (WDOGx_CNT)

This section describes the watchdog counter register.

The watchdog counter register provides access to the value of the free-running watchdog counter. Software can read the counter register at any time.

Software cannot write directly to the watchdog counter; however, two write sequences to these registers have special functions:

1. The *refresh sequence* resets the watchdog counter to 0x0000. See the Refreshing the Watchdog section.
2. The *unlock sequence* allows the watchdog to be reconfigured without forcing a reset (when CS[UPDATE] = 1). See the Example code: Reconfiguring the Watchdog section.

NOTE

All other writes to this register are illegal and force a reset.

Address: 4007_6000h base + 4h offset = 4007_6004h

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

WDOGx_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 CNTHIGH	High byte of the Watchdog Counter
CNTLOW	Low byte of the Watchdog Counter

55.2.3 Watchdog Timeout Value Register (WDOGx_TOVAL)

This section describes the watchdog timeout value register. TOVAL contains the 16-bit value used to set the timeout period of the watchdog.

The watchdog counter (CNT) is continuously compared with the timeout value (TOVAL). If the counter reaches the timeout value, the watchdog forces a reset.

NOTE

Do not write 0 to the Watchdog Timeout Value Register; otherwise, the watchdog always generates a reset.

Address: 4007_6000h base + 8h offset = 4007_6008h

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

WDOGx_TOVAL field descriptions

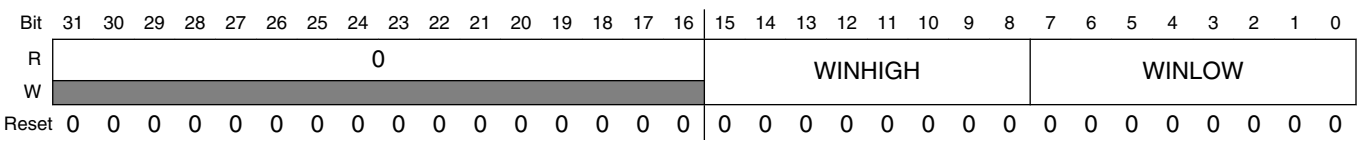
Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 TOVALHIGH	High byte of the timeout value;
TOVALLOW	Low byte of the timeout value

55.2.4 Watchdog Window Register (WDOGx_WIN)

This section describes the watchdog window register. When window mode is enabled (CS[WIN] is set), The WIN register determines the earliest time that a refresh sequence is considered valid. See the Watchdog refresh mechanism section.

The WIN register value must be less than the TOVAL register value.

Address: 4007_6000h base + Ch offset = 4007_600Ch



WDOGx_WIN field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 WINHIGH	High byte of Watchdog Window
WINLOW	Low byte of Watchdog Window

55.3 Functional description

The WDOG module provides a fail safe mechanism to ensure the system can be reset to a known state of operation in case of system failure, such as the CPU clock stopping or there being a run away condition in the software code. The watchdog counter runs continuously off a selectable clock source and expects to be serviced (refreshed) periodically. If it is not, it generates a reset triggering event.

The timeout period, window mode, and clock source are all programmable but must be configured within 128 bus clocks after a reset.

55.3.1 Watchdog refresh mechanism

The watchdog resets the MCU if the watchdog counter is not refreshed. A robust refresh mechanism makes it very unlikely that the watchdog can be refreshed by runaway code.

To refresh the watchdog counter, software must execute a refresh write sequence before the timeout period expires. In addition, if window mode is used, software must not start the refresh sequence until after the time value set in the WIN register. See the following figure.

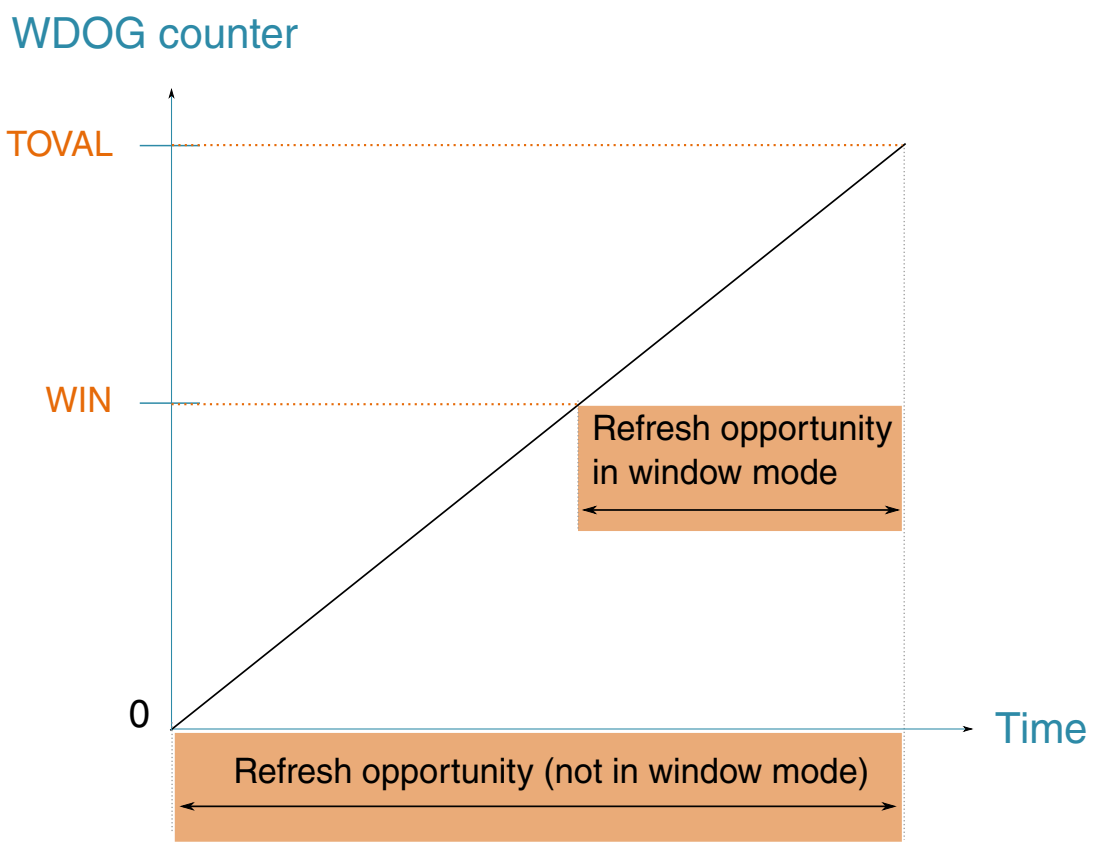


Figure 55-2. Refresh opportunity for the Watchdog counter

55.3.1.1 Window mode

Software finishing its main control loop faster than expected could be an indication of a problem. Depending on the requirements of the application, the WDOG can be programmed to force a reset when refresh attempts are early.

When Window mode is enabled, the watchdog must be refreshed after the counter has reached a minimum expected time value; otherwise, the watchdog resets the MCU. The minimum expected time value is specified in the WIN register. Setting CS[WIN] enables Window mode.

55.3.1.2 Refreshing the Watchdog

The refresh write sequence can be the following:

- One 32-bit write (0xB480_A602) to the CNT register.

This must occur before the WDOG timeout; otherwise, the watchdog resets the MCU.

Note

Before starting the refresh sequence, disable the global interrupts. Otherwise, an interrupt could effectively invalidate the refresh sequence, if the interrupt occurs before the refresh writes finish. After the sequence finishes, re-enable the global interrupts.

55.3.1.3 Example code: Refreshing the Watchdog

The following code segment shows the refresh write sequence of the WDOG module. The refresh command access size is 32-bit.

```
/* Refresh watchdog */

for (;;) // main loop
{
   ...

   DisableInterrupts; // disable global interrupt

   WDOG_CNT = 0xB480A602; // write the refresh word

   EnableInterrupts; // enable global interrupt

   ...
}
```

55.3.2 Configuring the Watchdog Once

All watchdog control bits, timeout value, and window value are write-once after reset *within 128 bus clocks*. This means that after a write has occurred they cannot be changed unless a reset occurs. This is guaranteed by the user configuring the window and timeout value first, followed by the other control bits, and ensuring that CS[UPDATE] is also set to 0.

This provides a robust mechanism to configure the watchdog and ensure that a runaway condition cannot mistakenly disable or modify the watchdog configuration after configured.

The new configuration takes effect only after all registers except CNT are written after reset. Otherwise, the WDOG uses the reset values by default. If window mode is not used (CS[WIN] is 0), writing to WIN is not required to make the new configuration take effect.

55.3.2.1 Reconfiguring the Watchdog

In some cases (like when supporting a bootloader function), you may want to reconfigure or disable the watchdog, *without forcing a reset first*.

- By setting CS[UPDATE] to 1 on the initial configuration of the watchdog after a reset, you can reconfigure the watchdog at any time by executing an unlock sequence.
- Conversely, if CS[UPDATE] remains 0, the only way to reconfigure the watchdog is by initiating a reset.

The unlock sequence is similar to the refresh sequence but uses different values.

55.3.2.2 Unlocking the Watchdog

The unlock sequence can be the following:

- One 32-bit write of 0xD928_C520 to the CNT register

This should occur within 16 bus clocks at any time after the watchdog has been configured. On completing the unlock sequence, the user must reconfigure the watchdog within 128 bus clocks; otherwise, the watchdog closes the unlock window.

NOTE

Due to the 128 bus clocks requirement for reconfiguring the watchdog, some delays must be inserted before executing STOP or WAIT instructions after reconfiguring the watchdog. This ensures that the watchdog's new configuration takes effect

before the MCU enters low power mode. Otherwise, the MCU may not be waken up from low power mode.

55.3.2.3 Example code: Reconfiguring the Watchdog

The following code segment shows an example reconfiguration of the WDOG module. The unlock command access size is 32-bit.

```
/* Initialize watchdog with ~1-kHz clock source, ~1s time-out */
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; // unlock word
WDOG_TOVAL = 1000; // setting timeout value
WDOG_CS = WDOG_CS_CLK_MASK; // setting 1-kHz clock source
WDOG_CS = WDOG_CS_EN_MASK; // enable counter running
EnableInterrupts; // enable global interrupt
```

55.3.3 Clock source

The watchdog counter has the following clock source options selected by programming CS[CLK]:

- bus clock
- internal Low-Power Oscillator clock (LPO_CLK) (This is the default source.)
- internal 8 MHz clock (SIRC)
- external clock (SOSC)

The options allow software to select a clock source independent of the bus clock for applications that need to meet more robust safety requirements. Using a clock source other than the bus clock ensures that the watchdog counter continues to run if the bus clock is somehow halted; see Backup reset.

An optional fixed prescaler for all clock sources allows for longer timeout periods. When CS[PRES] is set, the clock source is prescaled by 256 before clocking the watchdog counter.

The following table summarizes the different watchdog timeout periods available.

Table 55-1. Watchdog timeout availability

Reference clock	Prescaler	Watchdog time-out availability
Internal LPO_CLK	Pass through	~1 ms–65.5 s (if LPO_CLK = 1 kHz); (~1 ms–65.5 s)/128 (if LPO_CLK = 128 kHz). [1]
	÷256	~256 ms–16,777.2 s (if LPO_CLK = 1 kHz); ~2 ms–131.1 s (if LPO_CLK = 128 kHz).
Internal 8 MHz (SIRC)	Pass through	125 ns–8.1925 ms
	÷256	32 µs–2.09728 s
1 MHz (from bus or external)	Pass through	1 µs–65.54 ms
	÷256	256 µs–16.777 s
20 MHz (from bus or external)	Pass through	50 ns–3.277 ms
	÷256	12.8 µs–838.8 ms

1. The default timeout value after reset is approximately 4 ms (if LPO_CLK = 1 kHz), or 4/128 ms (if LPO_CLK = 128 kHz).

NOTE

When the programmer switches clock sources during reconfiguration, the watchdog hardware holds the counter at zero for 2.5 periods of the previous clock source and 2.5 periods of the new clock source after the configuration time period (128 bus clocks) ends. This delay ensures a smooth transition before restarting the counter with the new configuration.

55.3.4 Using interrupts to delay resets

- **When interrupts are enabled (CS[INT] = 1):** After a reset-triggering event (like a counter timeout or invalid refresh attempt), the watchdog first generates an interrupt request. Next, the watchdog delays 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event) before forcing a reset, to allow the interrupt service routine (ISR) to perform tasks (like analyzing the stack to debug code).
- **When interrupts are disabled (CS[INT] = 0):** the watchdog does not delay the forcing a reset.

55.3.5 Backup reset

NOTE

A clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the backup reset function is not available.

The backup reset function is a safeguard feature that independently generates a reset in case the main WDOG logic loses its clock (the bus clock) and can no longer monitor the counter. If the watchdog counter overflows twice in succession (without an intervening reset), the backup reset function takes effect and generates a reset.

55.3.6 Functionality in debug and low-power modes

By default, the watchdog is not functional in Debug mode, Wait mode, or Stop mode. However, the watchdog can remain functional in these modes as follows:

- **For Debug mode,** set CS[DBG]. (This way the watchdog is functional in Debug mode even when the CPU is held by the Debug module.)
- **For Wait mode,** set CS[WAIT].
- **For Stop mode,** set CS[STOP], CS[WAIT], and ensure the clock source is active in STOP mode.

NOTE

The watchdog can not generate interrupt in Stop mode even if CS[STOP] is set and will not wake the MCU from Stop mode. It can generate reset during Stop mode.

For Debug mode and Stop mode, in addition to the above configurations, a clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the watchdog cannot function.

55.3.7 Fast testing of the watchdog

Before executing application code in safety critical applications, users are required to test that the watchdog works as expected and resets the MCU. Testing every bit of a 16-bit counter by letting it run to the overflow value takes a relatively long time (64 kHz clocks).

To help minimize the startup delay for application code after reset, the watchdog has a feature to test the watchdog more quickly by splitting the counter into its constituent byte-wide stages. The low and high bytes are run independently and tested for timeout against the corresponding byte of the timeout value register. (For complete coverage when testing the high byte of the counter, the test feature feeds the input clock via the 8th bit of the low byte, thus ensuring that the overflow connection from the low byte to the high byte is tested.)

Using this test feature reduces the test time to 512 clocks (not including overhead, such as user configuration and reset vector fetches). To further speed testing, use a faster clock (such as the bus clock) for the counter reference.

On a power-on reset, the POR bit in the system reset register is set, indicating the user should perform the WDOG fast test.

55.3.7.1 Testing each byte of the counter

The test procedure follows these steps:

1. Program the preferred watchdog timeout value in the TOVAL register during the watchdog configuration time period.
2. Select a byte of the counter to test using the CS[TST] = 10b for the low byte; CS[TST] = 11b for the high byte.
3. Wait for the watchdog to timeout. Optionally, in the idle loop, increment RAM locations as a parallel software counter for later comparison. Because the RAM is not affected by a watchdog reset, the timeout period of the watchdog counter can be compared with the software counter to verify the timeout period has occurred as expected.
4. The watchdog counter times out and forces a reset.
5. Confirm the WDOG flag in the system reset register is set, indicating that the watchdog caused the reset. (The POR flag remains clear.)
6. Confirm that CS[TST] shows a test (10b or 11b) was performed.

If confirmed, the count and compare functions work for the selected byte. Repeat the procedure, selecting the other byte in step 2.

NOTE

CS[TST] is cleared by a POR only and not affected by other resets.

55.3.7.2 Entering user mode

After successfully testing the low and high bytes of the watchdog counter, the user can configure CS[TST] to 01b to indicate the watchdog is ready for use in application user mode. Thus if a reset occurs again, software can recognize the reset trigger as a real watchdog reset caused by runaway or faulty application code.

As an ongoing test when using the default LPO clock source, software can periodically read the CNT register to ensure the counter is being incremented.

Appendix A
Release Notes for Rev. 3, 3.1, and 4

A.1 General changes throughout document

• Added LK field to PORTx_PCRn register, bit 15. • Removed information on 64-pin package and indicated 121-pin package as Package Your Way. • Removed references to USB_CLK_RECOVER_IRC_EN register.
• Removed information about Keep Alive mode.
• Updated register addresses in Cryptographic Acceleration Unit (CAU) • Updated section Noise detection mode in Touch Sensing Input (TSI)

A.2 About This Document chapter changes

• No substantial content changes

A.3 Introduction chapter changes

• In the block diagram, changed "USB RAM" to "USB SRAM"

A.4 Chip Configuration chapter changes

• In section Universal Serial Bus (USB) FS Subsystem, added a note stating "For the USB FS OTG controller to operate, the minimum system clock frequency is 20 MHz."
• In AWIC stop wake-up sources, added USB. • Renamed section "USB SRAM in VLLS Power Modes" to "USB SRAM" and updated the section. • Added the following note in Universal Serial Bus (USB) FS Subsystem - "USB OTG is not functional in VLPx, VLLSx, and any Stop modes."

A.5 Memory Map changes

- No substantial content changes

A.6 Clock Distribution changes

- In table Table 5-1, added the following footnote to USB - "For the USB FS OTG controller to operate, the minimum required DIVCORE_CLK is 24 MHz (half of USB functional clock).

A.7 Reset and Boot chapter changes

- No substantial content changes

A.8 Power Management chapter changes

- Added a row for TSTMR in table Module operation in low power modes.

A.9 Security chapter changes

- No substantial content changes

A.10 Debug chapter changes

- Updated section Debug and security

A.11 Signal Multiplexing and Signal Descriptions chapter changes

- No substantial content changes

A.12 ADC changes

- No substantial content changes

A.13 Crossbar switch module changes

- No substantial content changes
- Fixed-priority operation : Removed the note referring to MGPCR from the "How the Crossbar Switch grants control of a slave port to a master" table.
- Initialization/application information :
  - Changed wording of sentence about arbitration scheme.

A.14 BME configuration changes

- In topic Introduction, removed reference to M2B.

A.15 Kinetis ROM Bootloader changes

- No substantial content changes

A.16 MMCAU changes

- No substantial content changes

A.17 CMP changes

- No substantial content changes

A.18 CRC changes

- No substantial content changes

A.19 DAC changes

- No substantial content changes

A.20 DMAMUX module changes

- No substantial content changes

A.21 eDMA module changes

- Error Status Register (DMA_ES) : Removed bullet about uncorrectable TCD SRAM errors.
- eDMA initialization : In bullet beginning with "Enable any hardware service requests..." replaced "ERQH and ERQL registers" with "ERQ register."

A.22 EMVSIM changes

- No substantial content changes

A.23 FMC changes

- No substantial content changes

A.24 FMC changes

- No substantial content changes

A.25 FTFA changes

• Add ACCERR check for sector size larger than segment size in Error Handling table for RD1XA and ERSXA commands • Modify FSEC[MEEN] register field description • Modify Flash Commands by Mode table entries for Read 1s All Blocks and Erase All Blocks commands • Add Read 1s All Execute-only Segments and Erase All Execute-only Segments commands; modify list of Margin Read Commands • Add reference to AN5112 in Flash Access Protection • Add ACCERR check for mode/security in Error Handling table for Verify Backdoor Access Key and Read 1s All Blocks commands • Change column heading from Byte to Offset Address in configuration field description table • Add suggestion to bit poll FSTAT[CCIF] for command completion in Generic flash command write sequence flowchart • Clarify that ACCERR and FPVIOL flags must be clear before ERSSUSP can be set in Suspending an Erase Flash Sector Operation • Remove erroneous reference to the flash configuration field in Suspending an Erase Flash Sector Operation • Specify minimum time of 4.3 msec between request to resume and suspend erase in Resuming an Erase Flash Sector Operation • Correct FPROT register description when supporting 2KB sectors • Add list of specific commands impacted by Flash Access Protection • Clarify writability of ACCERR and FPVIOL while CCIF is set in FSTAT register description
• In FTFA_FACSN[NUMSG], changed bit field value from 0x40 to 0x4x

A.26 INTMUX changes

• No substantial content changes

A.27 LLWU changes

• No substantial content changes

A.28 LPI2C changes

• No substantial content changes

A.29 LPIT changes

• In "Modes of operation" section, removed Low Leakage Stop row from "Chip modes supported by the LPIT module" table, because it is not supported by the device.

- In Module Control Register (MCR), in M_CEN field (Module Clock Enable), in the note, added MSR register to the list of registers that generate transfer errors when M_CEN is not enabled.
- In "Functional Description / Initialization" section, in the note, added MSR register to the list of registers that generate transfer errors when M_CEN is not enabled.

A.30 LPSPI changes

- No substantial content changes

A.31 LPTMR changes

- No substantial content changes

A.32 LPUART changes

- No substantial content changes

A.33 MMDVSQ changes

- No substantial content changes

A.34 MCM changes

- No substantial content changes

A.35 MSCM changes

- No substantial content changes

A.36 MTB configuration changes

• No substantial content changes

A.37 PCC changes

• No substantial content changes

A.38 PMC changes

• No substantial content changes

A.39 PORT changes

For the "Digital filter" section, clarified the maximum latency through a digital filter.
• In Pin Control Register (PCRn), added additional details to Interrupt Configuration field (IRQC) description.

A.40 RCM changes

• No substantial content changes

A.41 GPIO changes

• No substantial content changes

A.42 RTC changes

• No substantial content changes
• Updated field descriptions of the IER[TSIE]. • Updated CR[CPE] field.

A.43 I2S/SAI changes

• No substantial content changes

A.44 SCG changes

• No substantial content changes

A.45 SIM changes

• No substantial content changes

A.46 SMC changes

• Removed RAM2 content.
• For the PARAM register, clarified the bitfield descriptions.
• For PMCTRL[RUNM], clarified that any reset while in HSRUN mode clears RUNM. • For PMSTAT[PMSTAT], clarified that any reset while in HSRUN mode clears PMSTAT. • In section "Stop mode entry sequence," added clarification to step 5. • In section "High Speed Run (HSRUN) mode," added clarifications to the last paragraph.

A.47 TPM changes

• No substantial content changes

A.48 TRGMUX changes

• No substantial content changes

A.49 Touch sense input chapter changes

- No substantial content changes

A.50 TSTMR changes

- No substantial content changes

A.51 USB full speed OTG controller changes

Changed CLK_RECOVER_IRC_EN to reserved.

A.52 USB VREG changes

- No substantial content changes

A.53 VREF_2V1 changes

- Updated the bit field values of VREF_TRM4[VREF2V1_EN] to the following:

  0 VREF 2.1V is enabled

  1 VREF 2.1V is disabled

A.54 WDOG changes

- No substantial content changes

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number MKL28ZRM
Revision 4, 06/2016