Kinetis KE17Z/13Z/12Z with up to 512 KB Flash Reference Manual

Supports: MKE17ZxxxVxx9, MKE13ZxxxVxx9 and MKE12ZxxxVxx9.

Document Number: KE1XZP100M96SF0RM
Rev. 2, 01/2024

Contents

Section number Title Page

Chapter 1
About This Manual

Chapter 2
Introduction

Chapter 3
Core Overview

Chapter 4
Interrupts

Section number Title Page

Chapter 5
System Integration Module (SIM)

Chapter 6
Miscellaneous Control Module (MCM)

Chapter 7
Crossbar Switch Lite (AXBS-Lite)

Chapter 8
Peripheral Bridge (AIPS-Lite)

Chapter 9
Trigger MUX Control (TRGMUX)

Section number Title Page

Section number Title Page

Chapter 10
Direct Memory Access Multiplexer (DMAMUX)

Chapter 11
Enhanced Direct Memory Access (eDMA)

Section number Title Page

Chapter 12

Memory and memory map

Chapter 13
Flash Memory Controller (FMC) / Flash Acceleration Unit (FAU)

Chapter 14
Flash Memory Module (FTFE)

Section number | Title | Page

Chapter 15
Clock Distribution

Chapter 16
System Clock Generator (SCG)

Chapter 17
Peripheral Clock Controller (PCC)

Chapter 18
Reset and Boot

Chapter 19
Kinetis Flashloader

Section number | Title | Page

Chapter 20
Reset Control Module (RCM)

Chapter 21
Power Management

Chapter 22
System Mode Controller (SMC)

Chapter 23

Power Management Controller (PMC)

Chapter 24
Integrity Functions Overview

Chapter 25
External Watchdog Monitor (EWM)

Chapter 26
Watchdog timer (WDOG)

Chapter 27
Cyclic Redundancy Check (CRC)

Chapter 28
Debug

Chapter 29
Micro Trace Buffer (MTB)

Chapter 30
Port Control and Interrupts (PORT)

Chapter 31
General-Purpose Input/Output (GPIO)

Chapter 32
Analog-to-Digital Converter (ADC)

Section number Title Page

Chapter 33
Comparator (CMP)

Chapter 34
FlexTimer Module (FTM)

Section number Title Page

Chapter 35
Low-power Periodic Interrupt Timer (LPIT)

Chapter 36 Pulse Width Timer (PWT)

Chapter 37
Low Power Timer (LPTMR)

Chapter 38
Real Time Clock (RTC)

Section number Title Page

Chapter 39
Low Power Serial Peripheral Interface (LPSPI)

Section number Title Page

Chapter 40
Low Power Inter-Integrated Circuit (LPI2C)

Chapter 41
Low Power Universal Asynchronous Receiver/Transmitter (LPUART)

Chapter 42
Serial Communications Interface (SCI) / Universal Asynchronous Receiver/Transmitter (UART)

Chapter 43
Flexible I/O (FlexIO)

Section number Title Page

Chapter 44
Touch Sensing Input (TSI)

Chapter 1
About This Manual

1.1 Audience

This reference manual is intended for system software and hardware developers and applications programmers who want to develop products with this device. It assumes that the reader understands operating systems, microprocessor system design, and basic principles of software and hardware.

1.2 Organization

This manual has two main sets of chapters.

1. Chapters in the first set contain information that applies to all components on the chip.
2. Chapters in the second set are organized into functional groupings that detail particular areas of functionality.
    - Examples of these groupings are clocking, timers, and communication interfaces.
    - Each grouping includes chapters that provide a technical description of individual modules.

1.3 Module descriptions

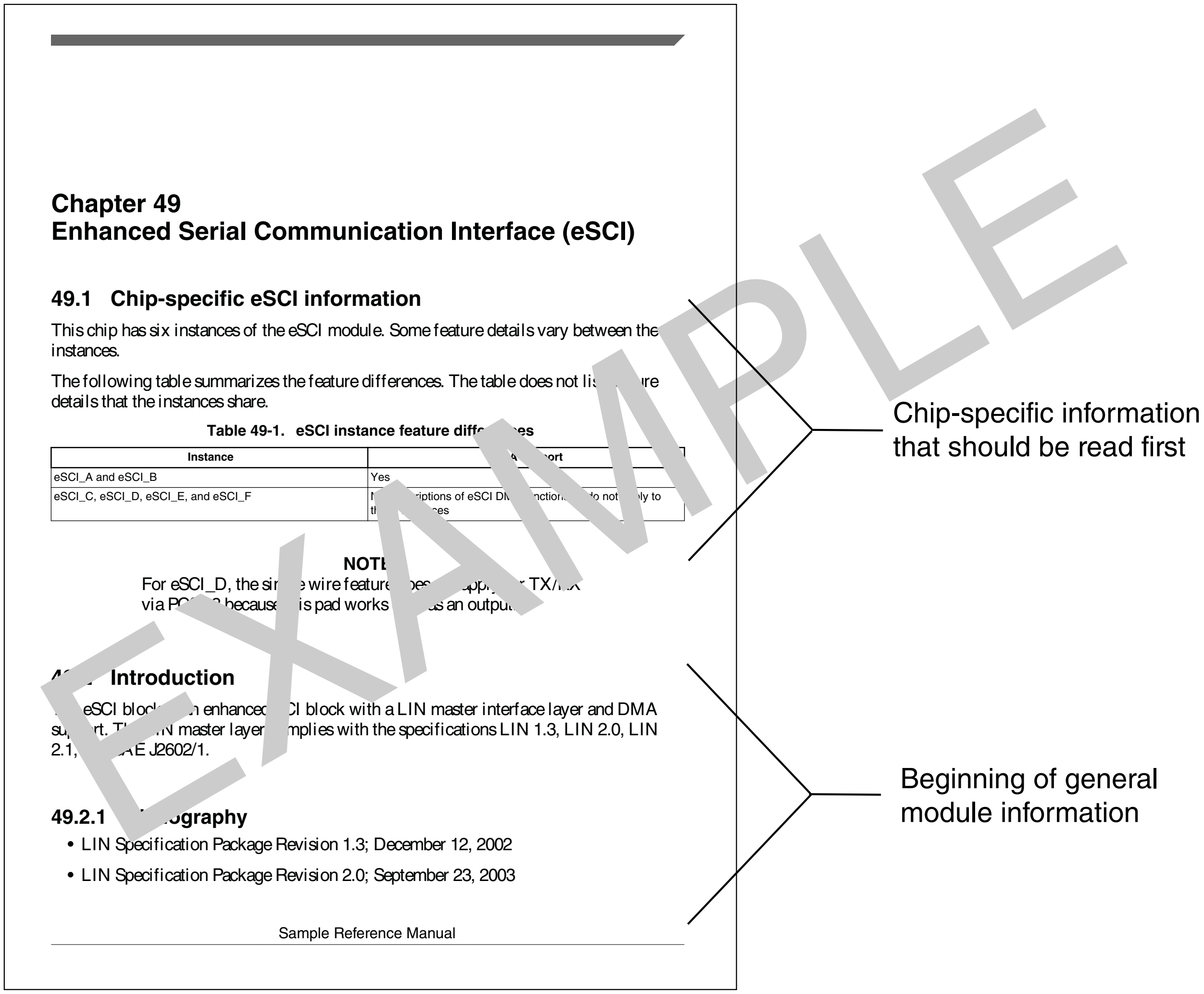
Each module chapter has two main parts:

- **Chip-specific:** The first section, *Chip-specific [module name] information*, includes the number of module instances on the chip and possible implementation differences between the module instances, such as differences in FIFO depths or the number of channels supported. It may also include functional connections between the module instances and other modules. Read this section *first* because its content is crucial to understanding the information in other sections of the chapter.

- **General:** The subsequent sections provide general information about the module, including its signals, registers, and functional description.

NOTE

If there is a conflict between the chip-specific module information (first section) and the general module information (subsequent sections), the chip-specific information supersedes the general information.



Chapter 49
Enhanced Serial Communication Interface (eSCI)

49.1 Chip-specific eSCI information

This chip has six instances of the eSCI module. Some feature details vary between the instances.

The following table summarizes the feature differences. The table does not lis   ure details that the instances share.

Table 49-1. eSCI instance feature diff   es

Instance	A ort
eSCI_A and eSCI_B	Yes
eSCI_C, eSCI_D, eSCI_E, and eSCI_F	N riptions of eSCI DM nction do not ly to th ces

NOTE

For eSCI_D, the sir   e wire feature   oes   apply   r TX/   X via PC   because   is pad works   as an output.

4   Introduction

    eSCI bloc   h enhanced   CI block with a LIN master interface layer and DMA su   rt. T   N master layer   mplies with the specifications LIN 1.3, LIN 2.0, LIN 2.1,   AE J2602/1.

49.2.1   ography

- LIN Specification Package Revision 1.3; December 12, 2002
- LIN Specification Package Revision 2.0; September 23, 2003

Sample Reference Manual

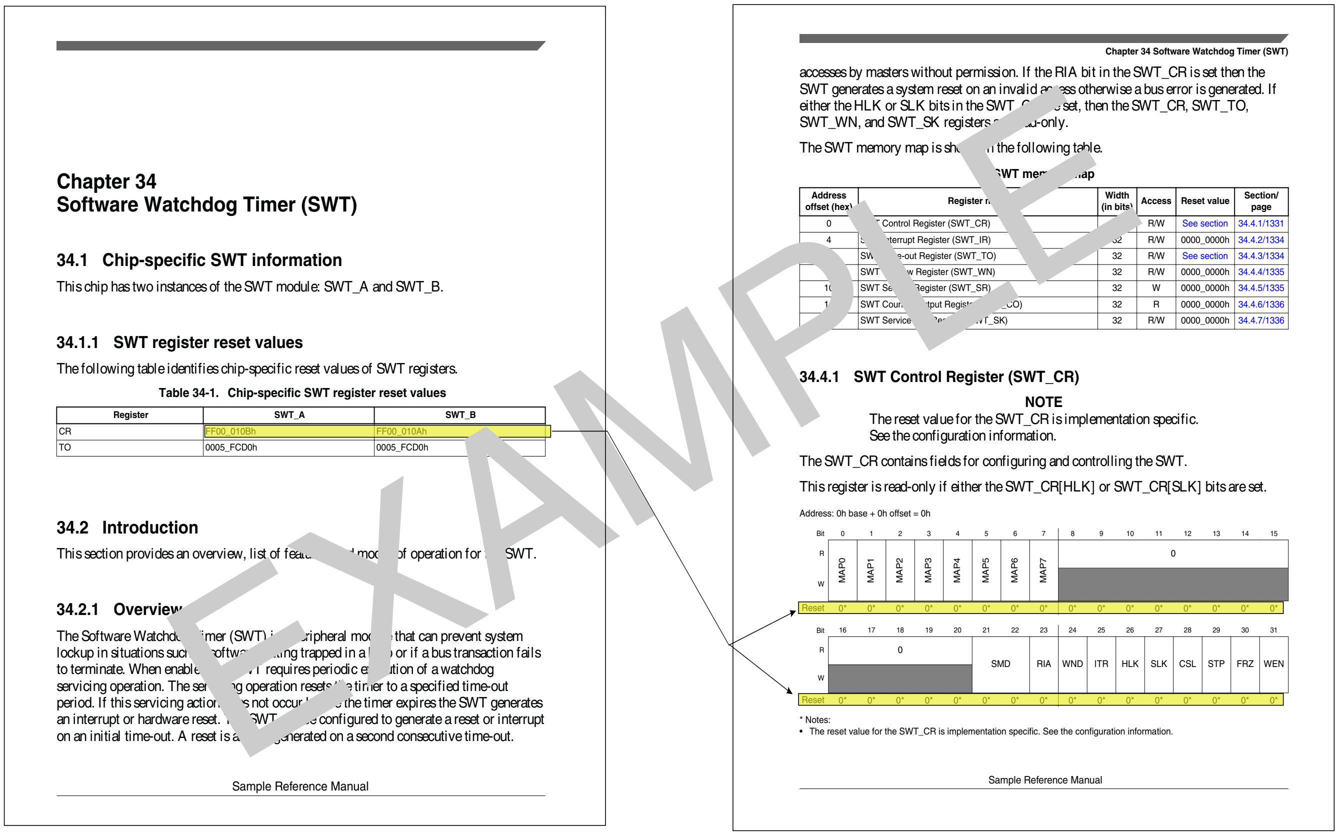
Chip-specific information that should be read first

Beginning of general module information

Figure 1-1. Example: chapter chip-specific information and general module information

1.3.1 Example: chip-specific information that supersedes content in the same chapter

The example below shows chip-specific information that supersedes general module information presented later in the chapter. In this case, the chip-specific register reset values supersede the reset values that appear in the register diagram.



Chapter 34
Software Watchdog Timer (SWT)

34.1 Chip-specific SWT information

This chip has two instances of the SWT module: SWT_A and SWT_B.

34.1.1 SWT register reset values

The following table identifies chip-specific reset values of SWT registers.

Table 34-1. Chip-specific SWT register reset values

Register	SWT_A	SWT_B
CR	FF00_010Bh	FF00_010Ah
TO	0005_FCD0h	0005_FCD0h

34.2 Introduction

This section provides an overview, list of featu d mod of operation for SWT.

34.2.1 Overview

The Software Watchdo imer (SWT) i ripheral mod that can prevent system lockup in situations such oftwa tting trapped in a l or if a bus transaction fails to terminate. When enable VT requires periodic e ution of a watchdog servicing operation. The se ng operation resets e timer to a specified time-out period. If this servicing action s not occur b e the timer expires the SWT generates an interrupt or hardware reset. SWT e configured to generate a reset or interrupt on an initial time-out. A reset is a generated on a second consecutive time-out.

Sample Reference Manual

Chapter 34 Software Watchdog Timer (SWT)

accesses by masters without permission. If the RIA bit in the SWT_CR is set then the SWT generates a system reset on an invalid a ess otherwise a bus error is generated. If either the HLK or SLK bits in the SWT C e set, then the SWT_CR, SWT_TO, SWT_WN, and SWT_SK registers a ad-only.

The SWT memory map is sh n the following table.

SWT mem ap

Address offset (hex)	Register n	Width (in bits)	Access	Reset value	Section/ page
0	T Control Register (SWT_CR)		R/W	See section	34.4.1/1331
4	S terrupt Register (SWT_IR)	32	R/W	0000_0000h	34.4.2/1334
	SW e-out Register (SWT_TO)	32	R/W	See section	34.4.3/1334
	SWT w Register (SWT_WN)	32	R/W	0000_0000h	34.4.4/1335
10	SWT Se Register (SWT_SR)	32	W	0000_0000h	34.4.5/1335
1	SWT Cour tput Regist CO)	32	R	0000_0000h	34.4.6/1336
	SWT Service Re VT_SK)	32	R/W	0000_0000h	34.4.7/1336

34.4.1 SWT Control Register (SWT_CR)

NOTE
The reset value for the SWT_CR is implementation specific.
See the configuration information.

The SWT_CR contains fields for configuring and controlling the SWT.

This register is read-only if either the SWT_CR[HLK] or SWT_CR[SLK] bits are set.

Address: 0h base + 0h offset = 0h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MAP0	MAP1	MAP2	MAP3	MAP4	MAP5	MAP6	MAP7	0							
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0					SMD		RIA	WND	ITR	HLK	SLK	CSL	STP	FRZ	WEN
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

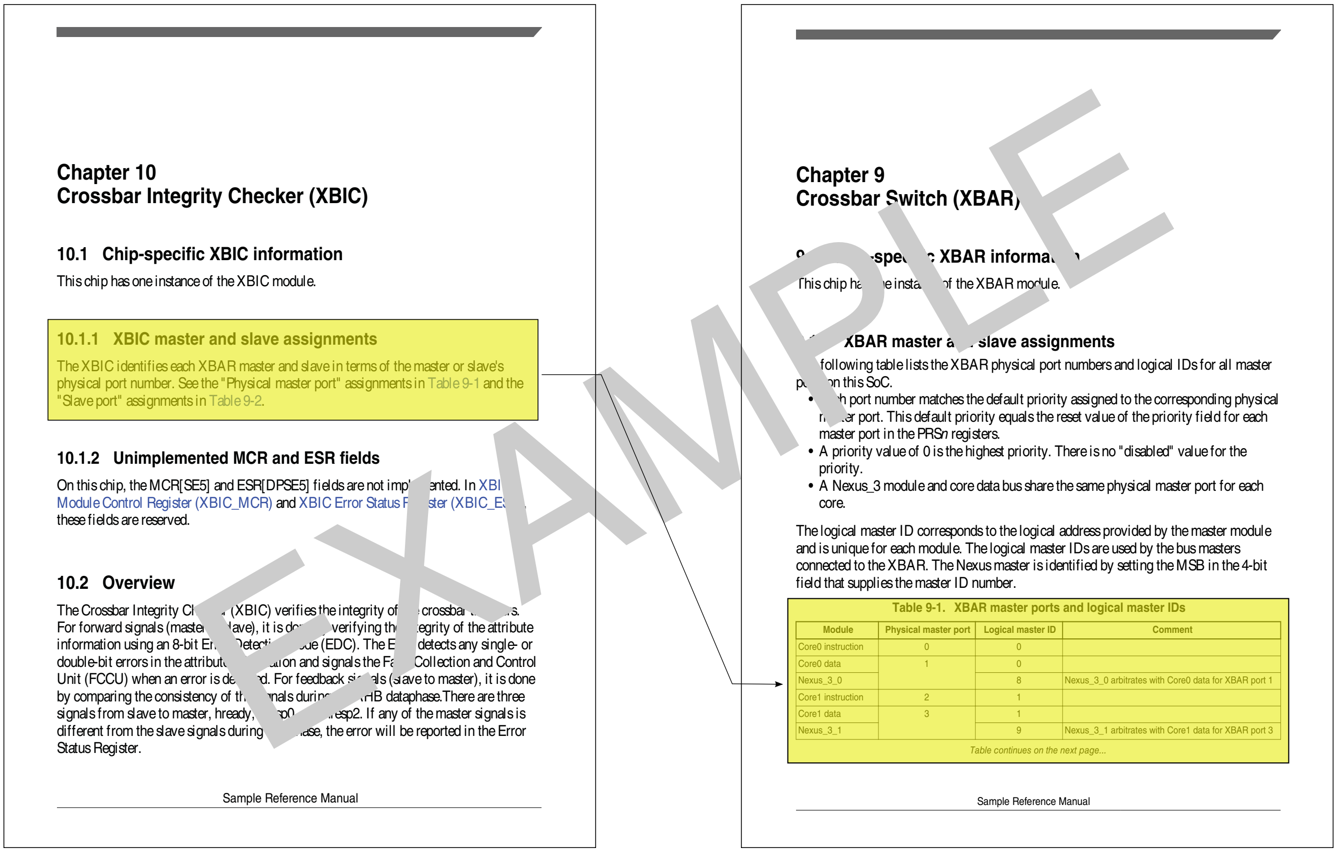
* Notes:
- The reset value for the SWT_CR is implementation specific. See the configuration information.

Sample Reference Manual

Figure 1-2. Example: chip-specific information that supersedes content in the same chapter

1.3.2 Example: chip-specific information that refers to a different chapter

The chip-specific information below refers to another chapter's chip-specific information. In this case, read both sets of chip-specific information before reading further in the chapter.



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC master and slave assignments

The XBIC identifies each XBAR master and slave in terms of the master or slave's physical port number. See the "Physical master port" assignments in Table 9-1 and the "Slave port" assignments in Table 9-2.

10.1.2 Unimplemented MCR and ESR fields

On this chip, the MCR[SE5] and ESR[DPSE5] fields are not imp' ented. In XBI Module Control Register (XBIC_MCR) and XBIC Error Status R ster (XBIC_ES , these fields are reserved.

10.2 Overview

The Crossbar Integrity Ch (XBIC) verifies the integrity of e crossbar rs. For forward signals (master lave), it is do verifying th tegrity of the attribute information using an 8-bit Er Detecti de (EDC). The E detects any single- or double-bit errors in the attribute ation and signals the Fa Collection and Control Unit (FCCU) when an error is de d. For feedback si als (slave to master), it is done by comparing the consistency of th nals durin HB dataphase.There are three signals from slave to master, hready, sp0 resp2. If any of the master signals is different from the slave signals during ase, the error will be reported in the Error Status Register.

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9 -spe c XBAR informa n

This chip ha e insta f the XBAR module.

XBAR master a slave assignments

following table lists the XBAR physical port numbers and logical IDs for all master p on this SoC.

- h port number matches the default priority assigned to the corresponding physical r er port. This default priority equals the reset value of the priority field for each master port in the PRS*n* registers.
- A priority value of 0 is the highest priority. There is no "disabled" value for the priority.
- A Nexus_3 module and core data bus share the same physical master port for each core.

The logical master ID corresponds to the logical address provided by the master module and is unique for each module. The logical master IDs are used by the bus masters connected to the XBAR. The Nexus master is identified by setting the MSB in the 4-bit field that supplies the master ID number.

Table 9-1. XBAR master ports and logical master IDs

Module	Physical master port	Logical master ID	Comment
Core0 instruction	0	0	
Core0 data	1	0	
Nexus_3_0		8	Nexus_3_0 arbitrates with Core0 data for XBAR port 1
Core1 instruction	2	1	
Core1 data	3	1	
Nexus_3_1		9	Nexus_3_1 arbitrates with Core1 data for XBAR port 3

Table continues on the next page...

Sample Reference Manual

Figure 1-3. Example: chip-specific information that refers to a different chapter

1.4 Register descriptions

Module chapters present register information in:

- Memory maps including:
  - Addresses
  - The name and acronym/abbreviation of each register
  - The width of each register (in bits)
  - Each register's reset value
  - The page number on which each register is described
- Register figures
- Field-description tables
- Associated text

The register figures show the field structure using the conventions in the following figure.

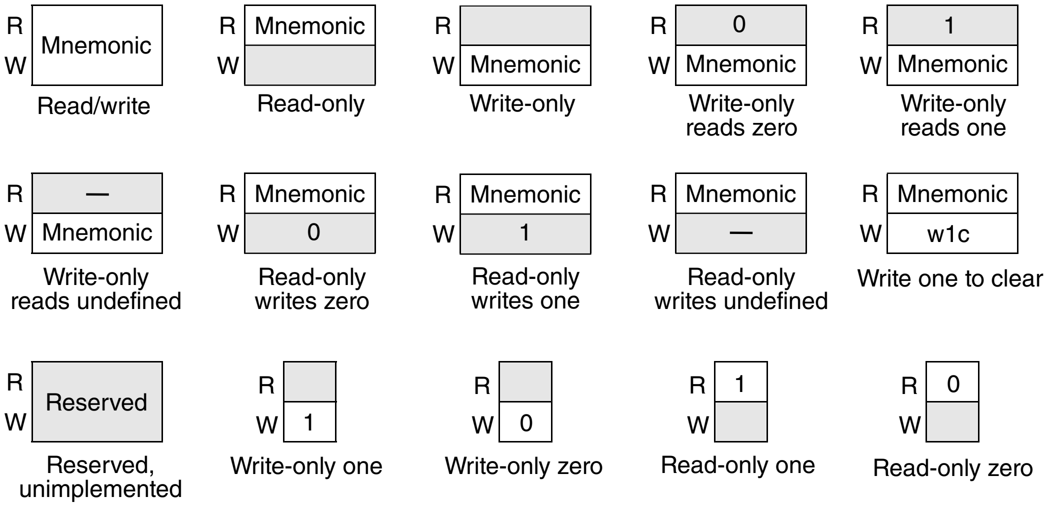


Figure 1-4. Register figure conventions

1.5 Conventions

1.5.1 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*.

1.5.2 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

Table continues on the next page...

Example	Description
	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.

1.5.3 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, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. • Do not modify the default value of a reserved programming setting, such as the reset value of a reserved register field. • Consider undefined locations in memory to be reserved.
w1c	Write 1 to clear: Refers to a register bitfield that must be written as 1 to be "cleared."

Chapter 2
Introduction

2.1 Overview

Information found here provides an overview of this MCU, which is a part of Kinetis E-series of ARM® Cortex®-M0+ MCUs and product family. It also presents high-level descriptions of the modules available on the device covered by this document.

2.2 Block Diagram

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

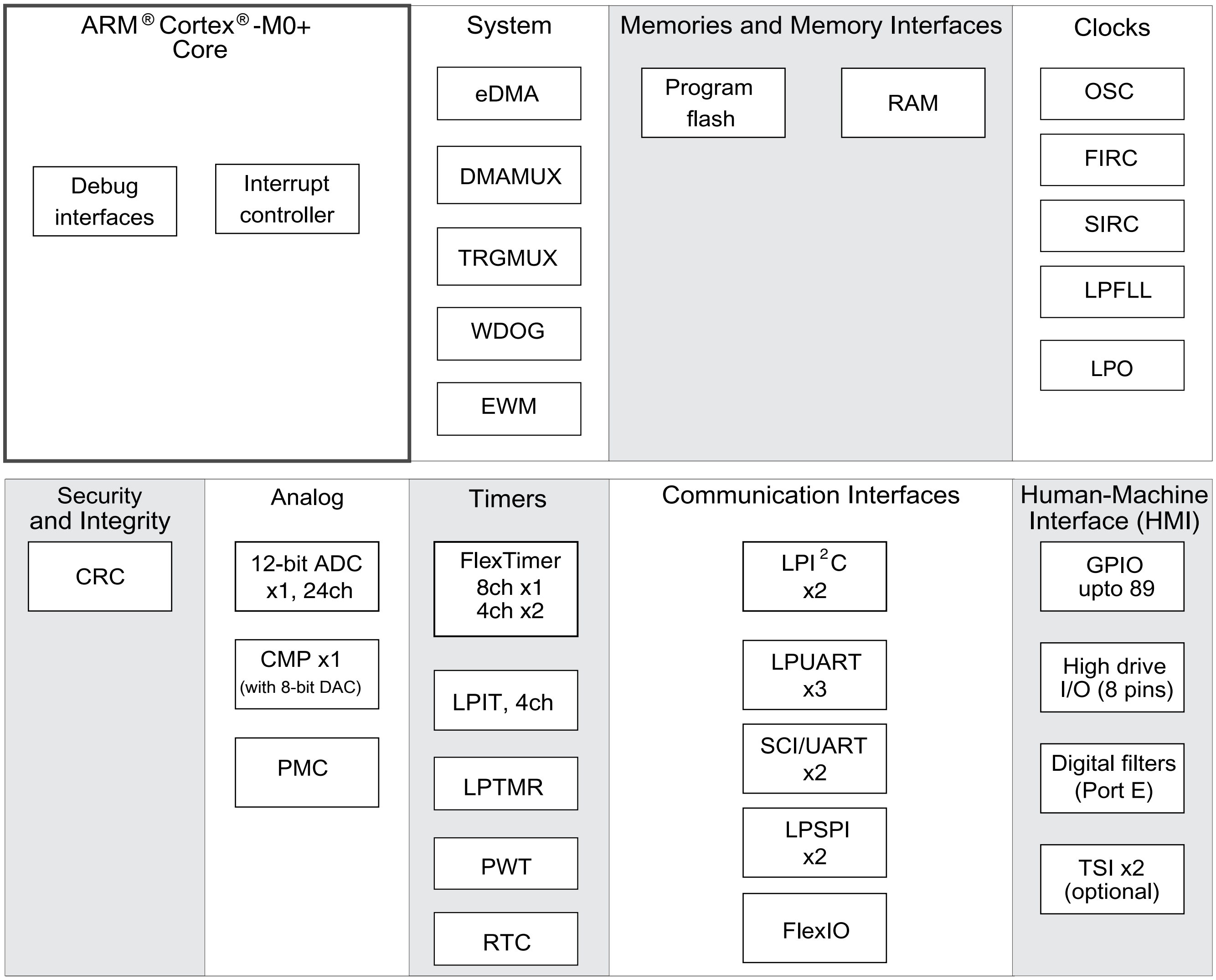


Figure 2-1. MCU block diagram

2.3 Module Functional Categories

The modules on this device are grouped into functional categories. The following sections describe the modules assigned to each category in more detail.

Table 2-1. Module functional categories

Module category	Description
ARM® Cortex®-M0+ core and related modules	• 32-bit MCU core from ARM's Cortex-M class, 96 MHz CPU frequency • Debug interfaces • Serial Wire Debug (SWD) • Micro Trace Buffer (MTB)
System modules	• System integration module (SIM) • System mode controller (SMC) • Miscellaneous control module (MCM) • Crossbar switch (AXBS-Lite) • Peripheral bridge (AIPS-Lite) • Direct memory access (DMA) controller with multiplexer (DMAMUX) to increase available DMA requests. DMA can now handle transfers in VLPS mode • Watchdog (WDOG) • External watchdog monitor (EWM)
Memories and memory interfaces	• Internal memories include: • Program flash memory • On devices with program flash only • SRAM
Clocks	• System clock generator (SCG) • Low-Power-Frequency-locked loop (LPFLL) • Fast internal reference clock (FIRC) • Slow internal reference clock (SIRC) • System oscillator (OSC) • Low Power Oscillator (LPO) • Peripheral Clock Control (PCC)
Integrity functions	• Cyclic Redundancy Check (CRC) module for error detection • 128-bit unique identification (ID) number • ADC self-test and calibration feature
Analog modules	• High speed analog-to-digital converter (ADC) • Comparator (CMP) • Bandgap voltage reference (1V reference voltage) • Power management controllers (PMC) • Multiple power modes available based on high speed run, run, wait, stop, and power-down modes
Timer modules	• FlexTimers (FTM) • Low-power periodic interrupt timer (LPIT) • Low power timer (LPTMR) • Independent real time clock (RTC)
Communication interfaces	• Low-power Serial peripheral interface (LPSPI) • Low-power Inter-integrated circuit (LPI^{2}C) • Low-power UART (LPUART) • SCI/UART • FlexIO

Table continues on the next page...

Table 2-1. Module functional categories (continued)

Module category	Description
Human-machine interfaces (HMI)	• General purpose input/output controller (GPIO) • Capacitive touch sense input (TSI) interface enabled in hardware • High drive I/O pins, see the "Pin properties" section in DataSheet. • Digital filters, see "Ports summary" table in Port control and interrupt module features.

Chapter 3
Core Overview

3.1 ARM Cortex-M0+

The ARM Cortex-M0+ is the member of the Cortex-M Series of processors targeting the micro-controller market. It is an entry-level 32-bit processor designed for very cost sensitive, low power applications. The Cortex-M0+ has a 2-stage pipeline von Neumann architecture. The processor delivers exceptional energy efficiency through extensively optimized design and provides high-end processing hardware including a single-cycle multiplier. It also has an I/O port which supports single cycle loads and stores to tightly-coupled peripherals (e.g. GPIO).

The Cortex-M0+ processor implements the ARMv6-M architecture, which is upward compatible with other Cortex-M profile processors. It is based on the 16-bit Thumb® instruction set and includes Thumb-2 technology (including all but three 16-bit Thumb opcodes plus seven 32-bit instructions). The Cortex-M0+ instruction set provides the exceptional performance expected of a modern 32-bit architecture, with a higher code density than 8-bit and 16-bit microcontrollers.

Cortex-M0+ Processor Features

- Thumb instruction set with Thumb-2 technology
- Nested Vectored Interrupt Controller (NVIC)
- Single-cycle 32-bit hardware multiplier
- Single-cycle I/O port
- Serial-Wire Debug port (SWD)
- Breakpoint & Watchpoint Units
- Micro Trace Buffer (MTB)
- 24-bit system tick timer (SysTick)

The detailed architecture and programming model of Cortex-M0+ processor are discussed in the following documents from ARM.

- Cortex-M0+ Devices Generic User Guide

- Cortex-M0+ Technical Reference Manual
- ARMv6-M Architecture Reference Manual

3.2 Core Buses and Interfaces

The Cortex-M0+ processor provides a single system-level interface using AMBA® technology to provide memory and peripheral accesses, a single-cycle I/O port for high speed access to tightly-coupled peripherals (such as GPIO), a NVIC interface for interrupt handling, a Debug Access Port (DAP) for SWD debug and a Micro Trace Buffer (MTB) interface for trace.

The following interfaces are implemented on the Cortex-M0+ processor of this device.

- A single AHB-Lite bus
- A single-cycle IO port
- PPB bus
- NVIC interface
- MTB interface
- Debug port interface

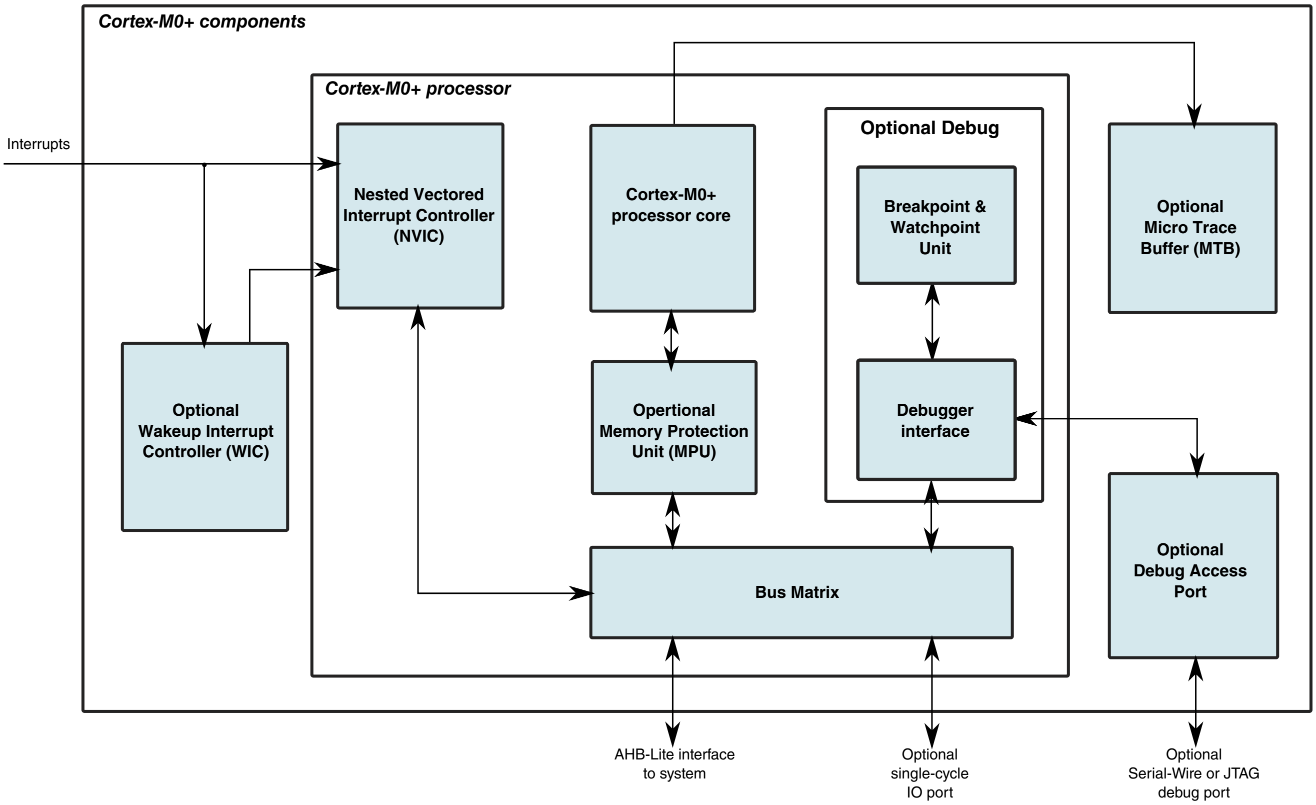

Cortex-M0+ components
Cortex-M0+ processor
Interrupts
Nested Vectored Interrupt Controller (NVIC)
Cortex-M0+ processor core
Optional Debug
Breakpoint & Watchpoint Unit
Optional Micro Trace Buffer (MTB)
Optional Wakeup Interrupt Controller (WIC)
Opertional Memory Protection Unit (MPU)
Debugger interface
Optional Debug Access Port
Bus Matrix
AHB-Lite interface to system
Optional single-cycle IO port
Optional Serial-Wire or JTAG debug port

Figure 3-1. Cortex-M0+ core interfaces

3.3 Core Component Configuration

The processor supports optional tightly-coupled system components. The following table lists the specific configuration of the Cortex-M0+ core on this device.

Component name	Present on this device	Note
Single-cycle Multiplier	YES	
Single-cycle IO Port	YES	
SysTick	YES	
Halting debug	YES	
Watchpoint	YES	Include 2 comparators
Breakpoint	YES	Include 2 comparators
MTB	YES	
WIC	YES	
Vector Table Offset Support	YES	
Unprivileged/Privileged Support	YES	
SWD	YES	
MPU	Not present	

3.4 SysTick Clock Configuration

The System Tick Timer's clock source is always the core clock (CORE_CLK) on this device. This results in the following:

- The CLKSOURCE bit in SysTick Control and Status Register (SYST_CSR) is always set to select the core clock.
- Because the timing reference (CORE_CLK) is a variable frequency, the TENMS bit in the SysTick Calibration Value Register (SYST_CALIB) is always zero.
- The NOREF bit in SysTick Calibration Value Register (SYST_CALIB) is always set, implying that CORE_CLK is the only available source of reference timing.

Chapter 4
Interrupts

4.1 Introduction

The ARM Cortex-M0+ processor includes an interrupt controller called the Nested Vectored Interrupt Controller (NVIC). It is closely coupled to the processor core to provide outstanding interrupt handling abilities and low latency interrupt processing. The NVIC supports nested interrupt, dynamic priority changes, interrupt masking and interrupt tail-chaining. In addition, the NVIC also supports re-locatable vector table and an external Nonmaskable Interrupt (NMI).

The NVIC registers are located within the processor's internal System Control Space (SCS) with base address of 0xE000E000. Most of the NVIC registers are accessible only in privileged mode. The detailed NVIC functionalities and registers descriptions are discussed in the following documents from ARM web.

- Cortex-M0+ Devices Generic User Guide
- Cortex-M0+ Technical Reference Manual

4.2 NVIC configuration

The NVIC supports configurable interrupt number and level of priority. The following sections speficy the exact priority level and interrupt vectors implemented on this device.

4.2.1 Interrupt priority levels

The NVIC on this device supports 4 interrupt priority levels. Therefore, the NVIC_IPR registers contains 2 bits for each interrupt request (IRQ). For example, NVIC_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																																

4.2.2 Non-maskable interrupt

This device supports non-maskable interrupt (NMI) to the NVIC. It is controlled by the external NMI signal from the pin. The pin which the NMI signal is multiplexed on, must be configured for the NMI function to generate the non-maskable interrupt request.

4.3 Interrupt channel assignments

The interrupt source assignments are defined in the following table.

- Vector number — the value stored on the stack when an interrupt is serviced.
- IRQ number — non-core interrupt source count, which is the vector number minus 16.

The IRQ number is used within ARM's NVIC documentation.

Table 4-2. Interrupt vector assignments

Address	Vector	IRQ[1]	NVIC IPR register number[2]	Source module	Source description
ARM Core System Handler Vectors					
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
0x0000_0010	4	–	–	—	—
0x0000_0014	5	–	–	—	—
0x0000_0018	6	–	–	—	—
0x0000_001C	7	–	–	—	—

Table continues on the next page...

Table 4-2. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC IPR register number[2]	Source module	Source description
0x0000_0020	8	–	–	—	—
0x0000_0024	9	–	–	—	—
0x0000_0028	10	–	–	—	—
0x0000_002C	11	–	–	ARM core	Supervisor call (SVCall)
0x0000_0030	12	–	–	—	—
0x0000_0034	13	–	–	—	—
0x0000_0038	14	–	–	ARM core	Pendable request for system service (PendableSrvReq)
0x0000_003C	15	–	–	ARM core	System tick timer (SysTick)
Non-Core Vectors					
0x0000_0040	16	0	0	DMA	DMA channel 0 or 4 transfer complete [3]
0x0000_0044	17	1	0	DMA	DMA channel 1 or 5 transfer complete [3]
0x0000_0048	18	2	0	DMA	DMA channel 2 or 6 transfer complete [3]
0x0000_004C	19	3	0	DMA	DMA channel 3 or 7 transfer complete [3]
0x0000_0050	20	4	1	DMA	DMA error interrupt channels 0-7
0x0000_0054	21	5	1	Flash memory	Single interrupt vector for all sources
0x0000_0058	22	6	1	PMC	Low-voltage detect, low-voltage warning
0x0000_005C	23	7	1	Port control module	Pin detect (Port A, E)
0x0000_0060	24	8	2	LPI^{2}C0	Single interrupt vector for all sources
0x0000_0064	25	9	2	LPI^{2}C1	Single interrupt vector for all sources
0x0000_0068	26	10	2	LPSPI0	Single interrupt vector for all sources
0x0000_006C	27	11	2	LPSPI1	Single interrupt vector for all sources
0x0000_0070	28	12	3	LPUART0	Single interrupt vector for all sources
0x0000_0074	29	13	3	LPUART1	Single interrupt vector for all sources
0x0000_0078	30	14	3	LPUART2	Single interrupt vector for all sources
0x0000_007C	31	15	3	ADC0	—
0x0000_0080	32	16	4	CMP0	—
0x0000_0084	33	17	4	FTM0	Single interrupt vector for all sources
0x0000_0088	34	18	4	FTM1	Single interrupt vector for all sources
0x0000_008C	35	19	4	FTM2	Single interrupt vector for all sources
0x0000_0090	36	20	5	RTC	Single interrupt vector for all sources
0x0000_0094	37	21	5	SCI0	Single interrupt vector for all sources
0x0000_0098	38	22	5	LPIT	LPIT channel 0-3
0x0000_009C	39	23	5	FlexIO	—
0x0000_00A0	40	24	6	TSI0	—
0x0000_00A4	41	25	6	TSI1	—
0x0000_00A8	42	26	6	Port control module	Pin detect (Port B, C, D)

Table continues on the next page...

Table 4-2. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC IPR register number[2]	Source module	Source description
0x0000_00AC	43	27	6	SCG	—
0x0000_00B0	44	28	7	WDOG or EWM	Both watchdog modules share this interrupt.
0x0000_00B4	45	29	7	PWT or LPTMR	Single interrupt vector for all sources
0x0000_00B8	46	30	7	SCI1	Single interrupt vector for all sources
0x0000_00BC	47	31	7	RCM	Single interrupt vector for all sources

1. Indicates the NVIC's interrupt source number.
2. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4
3. SIM_MISCTRL[DMA_INT_SEL] configures the DMA channel interrupt OR selection.

4.3.1 Determining the bitfield and register location for configuring a particular interrupt

Suppose you need to configure the low-power timer (LPTMR) interrupt. The following table is an excerpt of the LPTMR row from Interrupt channel assignments (value number as example only).

Table 4-3. LPTMR interrupt vector assignment (example only)

Address	Vector	IRQ[1]	NVIC non-IPR register number[2]	NVIC IPR register number[3]	Source module	Source description
0x0000_0128	74	58	1	14	Low Power Timer	—

1. Indicates the NVIC's interrupt source number.
2. Indicates the NVIC's ISER, ICER, ISPR, ICPR, and IABR register number used for this IRQ. The equation to calculate this value is: IRQ div 32
3. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4

- The NVIC registers you would use to configure the interrupt are:
  - NVIC_ISER1
  - NVIC_ICER1
  - NVIC_ISPR1
  - NVIC_ICPR1
  - NVIC_IABR1
  - NVIC_IPR14
- To determine the particular IRQ's bitfield location within these particular registers:
  - NVIC_ISER1, NVIC_ICER1, NVIC_ISPR1, NVIC_ICPR1, NVIC_IABR1 bit location = IRQ mod 32 = 26
  - NVIC_IPR14 bitfield starting location = 8 × (IRQ mod 4) + 4 = 20

Since the NVIC_IPR bitfields are 2-bit wide (4 priority levels), the NVIC_IPR14 bitfield range is 20-21

Therefore, the following bitfield locations are used to configure the LPTMR interrupts:

- NVIC_ISER1[26]
- NVIC_ICER1[26]
- NVIC_ISPR1[26]
- NVIC_ICPR1[26]
- NVIC_IABR1[26]
- NVIC_IPR14[21:20]

Chapter 5
System Integration Module (SIM)

5.1 Introduction

The System Integration Module (SIM) provides system control and chip configuration registers.

5.1.1 Features

Features of the SIM include:

- System clocking configuration
- Flash and system RAM size configuration
- FlexTimer clock and channel selection and configuration
- ADC trigger selection
- Flash configuration
- System device unique identification (UID)
- LPUART pseudo open drain control

5.2 Memory map and register definition

NOTE

The SIM registers can only be written in the supervisor mode. In the user mode, write accesses are blocked and will result in a bus error.

SIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_8004	Chip Control register (SIM_CHIPCTL)	32	R/W	0000_0300h	5.2.1/58
4004_800C	FTM Option Register 0 (SIM_FTMOPT0)	32	R/W	0000_0000h	5.2.2/60
4004_8018	ADC Options Register (SIM_ADCOPT)	32	R/W	0000_0000h	5.2.3/61
4004_801C	FTM Option Register 1 (SIM_FTMOPT1)	32	R/W	0000_0000h	5.2.4/62
4004_8024	System Device Identification Register (SIM_SDID)	32	R	See section	5.2.5/64
4004_804C	Flash Configuration Register 1 (SIM_FCFG1)	32	R/W	See section	5.2.6/65
4004_8050	Flash Configuration Register 2 (SIM_FCFG2)	32	R/W	See section	5.2.7/67
4004_8054	Unique Identification Register High (SIM_UIDH)	32	R	See section	5.2.8/68
4004_8058	Unique Identification Register Mid-High (SIM_UIDMH)	32	R	See section	5.2.9/69
4004_805C	Unique Identification Register Mid Low (SIM_UIDML)	32	R	See section	5.2.10/69
4004_8060	Unique Identification Register Low (SIM_UIDL)	32	R	See section	5.2.11/70
4004_806C	Miscellaneous Control register (SIM_MISCTRL)	32	R/W	0000_0000h	5.2.12/70

5.2.1 Chip Control register (SIM_CHIPCTL)

SIM_CHIPCTL contains the controls for selecting PWT alternative clock source, ADC COCO trigger, trace clock, and clock out source.

Address: 4004_8000h base + 4h offset = 4004_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												RTC32KCLKSEL		PWTCLKSEL	
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	FLEXIOTRIGSEL		Reserved		CLKOUTSEL		CLKOUTDIV		0		0	
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

SIM_CHIPCTL field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SIM_CHIPCTL field descriptions (continued)

Field	Description
19–18 RTC32KCLKSEL	RTC 32 kHz clock input select 00 Reserved 01 RTC_CLKIN 10 SOSC_CLK 11 Reserved
17–16 PWTCLKSEL	PWT clock source select 00 PWT alternative clock is from the TCLK0 pin. 01 PWT alternative clock is from the TCLK1 pin. 10 PWT alternative clock is from the TCLK2 pin. 11 Reserved
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 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–10 FLEXIOTRIGSEL	FLEXIO Trigger0/1 source Select Bit10 – FLEXIOTRIGSEL[0]. • value 0: from TRGMUX • value 1: from async clock of PCC slot 98 Bit11 – FLEXIOTRIGSEL[1]. • value 0: from TRGMUX • value 1: from async clock of PCC slot 99
9–8 Reserved	This field is reserved.
7–6 CLKOUTSEL	CLKOUT Select Selects the clock to output on the CLKOUT pin. 00 Reserved 01 SCGCLKOUT(SIRC/FIRC/SOSC/LPFLL), see SCG_CLKOUTCNFG register. 10 Reserved 11 LPO clock (128 kHz)
5–4 CLKOUTDIV	CLKOUT divider ratio 00 Divided by 1 01 Divided by 2 10 Divided by 4 11 Divided by 8
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

5.2.2 FTM Option Register 0 (SIM_FTMOPT0)

Address: 4004_8000h base + Ch offset = 4004_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		FTM2CLKSEL		FTM1CLKSEL		FTM0CLKSEL		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													FTM0FLTxSEL		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_FTMOPT0 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29–28 FTM2CLKSEL	FTM2 External Clock Pin Select Selects the external pin used to drive the clock to the FTM2 module. **NOTE:** The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM2 external clock driven by TCLK0 pin. 01 FTM2 external clock driven by TCLK1 pin. 10 FTM2 external clock driven by TCLK2 pin. 11 No clock input
27–26 FTM1CLKSEL	FTM1 External Clock Pin Select Selects the external pin used to drive the clock to the FTM1 module. **NOTE:** The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM1 external clock driven by TCLK0 pin. 01 FTM1 external clock driven by TCLK1 pin. 10 FTM1 external clock driven by TCLK2 pin. 11 No clock input
25–24 FTM0CLKSEL	FTM0 External Clock Pin Select Selects the external pin used to drive the clock to the FTM0 module. **NOTE:** The selected pin must also be configured for the FTM external clock function through the appropriate Pin Control Register in the Port Control module. 00 FTM0 external clock driven by TCLK0 pin. 01 FTM0 external clock driven by TCLK1 pin. 10 FTM0 external clock driven by TCLK2 pin. 11 No clock input

Table continues on the next page...

SIM_FTMOPT0 field descriptions (continued)

Field	Description
23–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FTM0FLTxSEL	FTM0 Fault x Select Selects the source of FTM0 fault. Every bit means one fault input respectively. **NOTE:** The pin source for fault must be configured for the FTM module fault function through the appropriate pin control register in the port control module when it comes from external fault pin. TRGMUX_FTM0 SELx is corresponding to FTM0 Fault x input. Bit value = 0: FTM0_FLTx pin Bit value = 1: TRGMUX_FTM0 out

5.2.3 ADC Options Register (SIM_ADCOPT)

Address: 4004_8000h base + 18h offset = 4004_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										ADC0PRETRGSEL		ADC0SWPRETRG			ADC0TRGSEL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_ADCOPT field descriptions

Field	Description
31–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 ADC0PRETRGSEL	ADC0 pre-trigger source select Selects pre-trigger source for ADC0. 00 Reserved 01 TRGMUX output 10 ADC0 software pre-trigger 11 Reserved

Table continues on the next page...

SIM_ADCOPT field descriptions (continued)

Field	Description
3–1 ADC0SWPRETRG	ADC0 software pre-trigger sources 000 software pre-trigger disabled 001 - 011 Reserved (do not use) 100 software pre-trigger 0 101 software pre-trigger 1 110 software pre-trigger 2 111 software pre-trigger 3
0 ADC0TRGSEL	ADC0 trigger source select Selects trigger source for ADC0. 0 Reserved 1 TRGMUX output

5.2.4 FTM Option Register 1 (SIM_FTMOPT1)

Address: 4004_8000h base + 1Ch offset = 4004_801Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								FTM0_OUTSEL							
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							FTM2CH1SEL	FTM2CH0SEL		FTM1CH0SEL		0	FTM2SYNCBIT	FTM1SYNCBIT	FTM0SYNCBIT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_FTMOPT1 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 FTM0_OUTSEL	FTM0 channel modulation select with FTM1_CH1 Bit 7 to 0 are for channel 7 to 0 respectively. 0 No modulation with FTM1_CH1 1 Modulation with FTM1_CH1

Table continues on the next page...

SIM_FTMOPT1 field descriptions (continued)

Field	Description
15–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 FTM2CH1SEL	FTM2 CH1 Select Selects FTM2 CH1 input 0 FTM2_CH1 input 1 exclusive OR of FTM2_CH0, FTM2_CH1, and FTM1_CH1
7–6 FTM2CH0SEL	FTM2 CH0 Select Selects FTM2 CH0 input 00 FTM2_CH0 input 01 CMP0 output 10 Reserved 11 Reserved
5–4 FTM1CH0SEL	FTM1 CH0 Select Selects FTM1 CH0 input 00 FTM1_CH0 input 01 CMP0 output 10 Reserved 11 Reserved
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 FTM2SYNCBIT	FTM2 Sync Bit Software control for FTM2 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM2. Software must clear this bit to allow other trigger sources to assert.
1 FTM1SYNCBIT	FTM1 Sync Bit Software control for FTM1 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM1. Software must clear this bit to allow other trigger sources to assert.
0 FTM0SYNCBIT	FTM0 Sync Bit Software control for FTM0 hardware trigger synchronization 0 No effect. 1 Write 1 to assert the TRIG1 input to FTM0. Software must clear this bit to allow other trigger sources to assert.

5.2.5 System Device Identification Register (SIM_SDID)

NOTE

Reset value loaded during System Reset from Flash IFR.

Address: 4004_8000h base + 24h offset = 4004_8024h

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	FAMILYID				SUBFAMID				SERIESID				RAMSIZE				REVID				PROJECTID					PINID						
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	0	0	1	0	0	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

SIM_SDID field descriptions

Field	Description
31–28 FAMILYID	Kinetis E-series Family ID Specifies the Kinetis E-series family of the device. 0001 KE1x Family (Enhanced features)
27–24 SUBFAMID	Kinetis E-series Sub-Family ID Specifies the Kinetis E-series sub-family of the device.
23–20 SERIESID	Kinetis Series ID Specifies the Kinetis series of the device. 0010 Kinetis E+ series
19–16 RAMSIZE	RAM size This field specifies the amount of system RAM available on the device. 0111 48 KB 1000 96 KB Others Reserved
15–12 REVID	Device revision number Specifies the silicon implementation number for the device.
11–7 PROJECTID	Project ID Specifies the silicon feature set identication number for the device. 00100 for this device.
PINID	Pin identification Specifies the pin count of the device.

Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
	0000110 48-pin 0000111 64-pin 0001010 100-pin

5.2.6 Flash Configuration Register 1 (SIM_FCFG1)

NOTE

Reset value of PFSIZE is loaded during System Reset from Flash IFR.

Address: 4004_8000h base + 4Ch offset = 4004_804Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				PFSIZE				0							
W																
Reset	0	0	0	0	x*	x*	x*	x*	0	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										FLASHDOZE	FLASHDIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

* Notes:

- x = Undefined at reset.

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 This field specifies the amount of program flash memory available on the device . Undefined values are reserved. 1001 256 KB of program flash memory, 8 KB protection region 1011 512 KB of program flash memory, 16 KB protection region
23–16 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.
11–2 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. An attempt by the bus master to access the Flash when the Flash is disabled will result in a bus error. This bit should 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 wakeup time from Doze mode is extended when this bit is set. 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 bit 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

5.2.7 Flash Configuration Register 2 (SIM_FCFG2)

NOTE

Reset values of MAXADDR0 are loaded during System Reset from Flash IFR.

Address: 4004_8000h base + 50h offset = 4004_8050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PFLASHSWAP	MAXADDR0							1	MAXADDR1						
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	0															
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- Reset value loaded during System Reset from Flash IFR.x = Undefined at reset.

SIM_FCFG2 field descriptions

Field	Description
31 PFLASHSWAP	Program Flash Swap bit Indicates that swap is active . 0 Swap is not active. 1 Swap is active.
30–24 MAXADDR0	Max address block 0 This field concatenated with 13 trailing zeros indicates the first invalid address of program flash (block 0). For example, if MAXADDR0 = 0x20, the first invalid address of program flash (block 0) is 0x0004_0000. This would be the MAXADDR0 value for a device with 256 KB program flash in flash block 0.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
22–16 MAXADDR1	Max address block 1 This field concatenated with 13 trailing zeros indicates the first invalid address of data flash (block 1).
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

5.2.8 Unique Identification Register High (SIM_UIDH)

Address: 4004_8000h base + 54h offset = 4004_8054h

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	UID127_96																															
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*

* Notes:

- Reset value loaded during System Reset from Flash IFR.

SIM_UIDH field descriptions

Field	Description
UID127_96	Unique Identification Unique identification for the device.

5.2.9 Unique Identification Register Mid-High (SIM_UIDMH)

Address: 4004_8000h base + 58h offset = 4004_8058h

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	UID95_64																															
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*

* Notes:
- Reset value loaded during System Reset from Flash IFR.

SIM_UIDMH field descriptions

Field	Description
UID95_64	Unique Identification Unique identification for the device.

5.2.10 Unique Identification Register Mid Low (SIM_UIDML)

Address: 4004_8000h base + 5Ch offset = 4004_805Ch

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	UID63_32																															
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*

* Notes:
- Reset value loaded during System Reset from Flash IFR.

SIM_UIDML field descriptions

Field	Description
UID63_32	Unique Identification Unique identification for the device.

5.2.11 Unique Identification Register Low (SIM_UIDL)

Address: 4004_8000h base + 60h offset = 4004_8060h

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	UID31_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*

* Notes:

- Reset value loaded during System Reset from Flash IFR.

SIM_UIDL field descriptions

Field	Description
UID31_0	Unique Identification Unique identification for the device.

5.2.12 Miscellaneous Control register (SIM_MISCTRL)

Address: 4004_8000h base + 6Ch offset = 4004_806Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													UART2ODE	UART1ODE	UART0ODE
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								DMA_INT_SEL				0			SW_ TRG
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_MISCTRL field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 UART2ODE	UART2 Open Drain Enable

Table continues on the next page...

SIM_MISCTRL field descriptions (continued)

Field	Description
	0 Open drain is disabled on UART2 1 Open drain is enabled on UART2
17 UART1ODE	UART1 Open Drain Enable 0 Open drain is disabled on UART1 1 Open drain is enabled on UART1
16 UART0ODE	UART0 Open Drain Enable 0 Open drain is disabled on UART0 1 Open drain is enabled on UART0
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–4 DMA_INT_SEL	DMA channel interrupt OR select Bit 7 of SIM_MISCTRL DMA channel 7 and channel 3 interrupt select bit (logic 1 is ch7 OR ch3, while logic 0 is ch3) Bit 6 of SIM_MISCTRL DMA channel 6 and channel 2 interrupt select bit (logic 1 is ch6 OR ch2, while logic 0 is ch2) Bit 5 of SIM_MISCTRL DMA channel 5 and channel 1 interrupt select bit (logic 1 is ch5 OR ch1, while logic 0 is ch1) Bit 4 of SIM_MISCTRL DMA channel 4 and channel 0 interrupt select bit (logic 1 is ch4 OR ch0, while logic 0 is ch0)
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 SW_TRG	Software Trigger bit to TRGMUX If you expect to implement a software trigger, then configure this field to get the corresponding trigger event. See the "Module Interconnectivity" section in TRGMUX chapter for more details.

Chapter 6
Miscellaneous Control Module (MCM)

6.1 Introduction

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

6.1.1 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

6.2 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 (MCM_PLASC)	16	R	0007h	6.2.1/74
F000_300A	Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)	16	R	0005h	6.2.2/74
F000_300C	Platform Control Register (MCM_PLACR)	32	R/W	0000_0250h	6.2.3/75
F000_3040	Compute Operation Control Register (MCM_CPO)	32	R/W	0000_0000h	6.2.4/78

6.2.1 Crossbar Switch (AXBS) Slave Configuration (MCM_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	0	1	1	1

MCM_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.

6.2.2 Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)

PLAMC is a 16-bit read-only register identifying the presence/absence of bus master connections to the device's crossbar switch.

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	0	1	0	1

MCM_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.

Table continues on the next page...

MCM_PLAMC field descriptions (continued)

Field	Description
	0 A bus master connection to AXBS input port *n* is absent 1 A bus master connection to AXBS input port *n* is present

6.2.3 Platform Control Register (MCM_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

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															ESFC
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	DFCS	EFDS	DFCC	DFCIC	DFCDA	0	ARB	0								
W						CFCC										
Reset	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0

MCM_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.

Table continues on the next page...

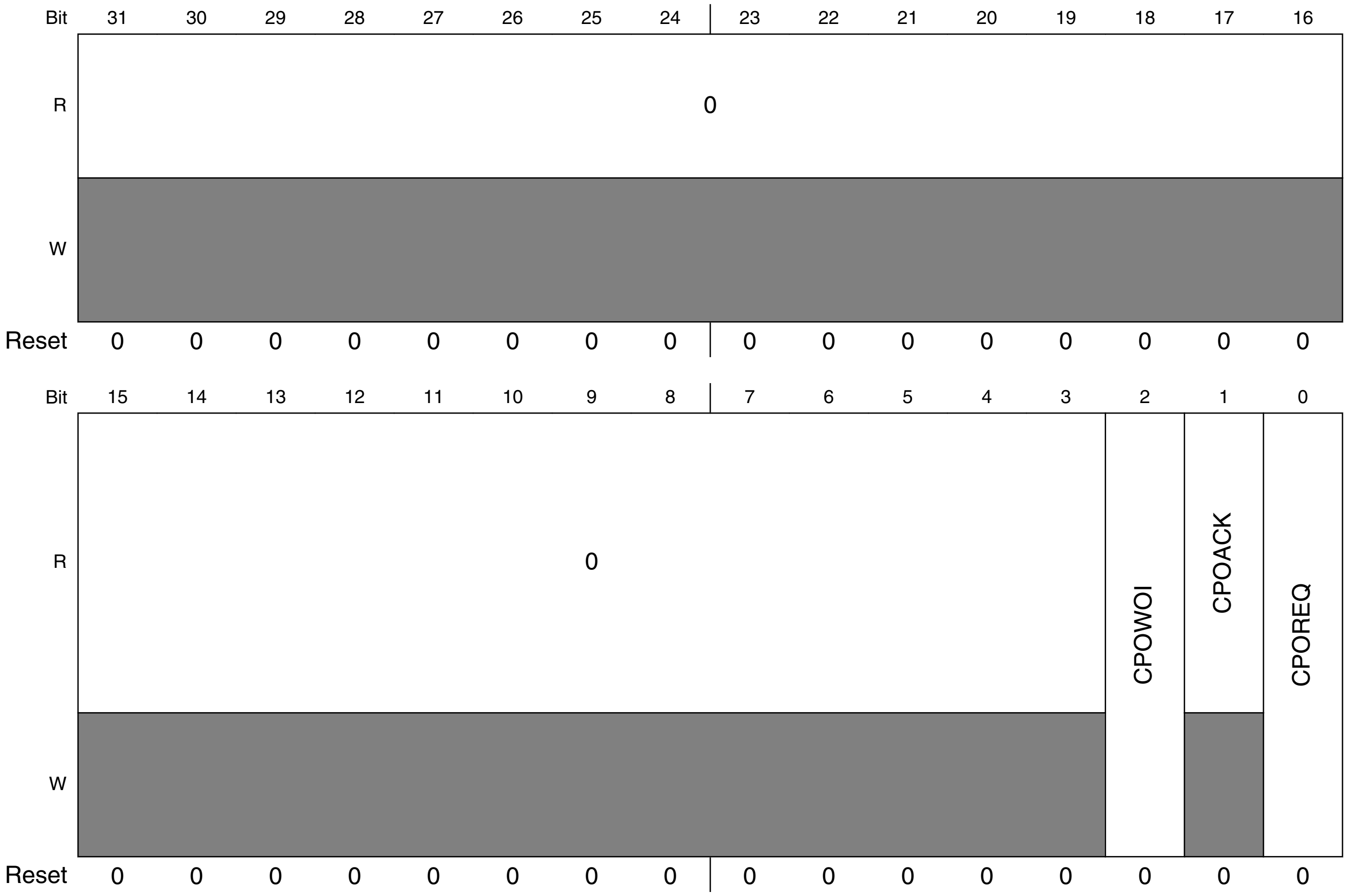
MCM_PLACR field descriptions (continued)

Field	Description
	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 0 Fixed-priority arbitration for the crossbar masters 1 Round-robin arbitration for the crossbar masters
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

6.2.4 Compute Operation Control Register (MCM_CPO)

This register controls the Compute Operation.

Address: F000_3000h base + 40h offset = F000_3040h



MCM_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.

Table continues on the next page...

MCM_CPO field descriptions (continued)

Field	Description
	0 Request is cleared. 1 Request Compute Operation.

Chapter 7
Crossbar Switch Lite (AXBS-Lite)

7.1 Chip-specific Information for this Module

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

Master module	Master port number
ARM core I/D bus	0
ARM core system bus	1
DMA	2

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

Slave module	Slave port number
Flash memory controller	0
SRAM controllers	1
Peripheral bridge 0 / GPIO[1]	2

[1] See ""System memory map"" for access restrictions.

NOTE

This crossbar switch has no memory mapped configuration registers. The arbitration method in the crossbar switch is programmable by MCM registers.

NOTE

The AXBS master and slave configuration information can be read from MCM registers.

7.2 Overview

This section 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.

7.2.1 Features

- Symmetric crossbar bus switch implementation
    - Allows concurrent access from different masters to different slaves
- Single-clock 32-bit transfer
- Programmable configuration for fixed-priority or round-robin slave port arbitration (see the chip-specific information).
- 32-bit AHB crossbar bus switch compatible with ARM's Advanced Microcontroller Bus Architecture (AMBA) Specification v2.0

7.3 Functional description

Information about general operation and arbitration are provided in this section.

7.3.1 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 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 does not know whether it owns the slave port it is targeting. The master waits while it does not have control of the slave port it is targeting.

After the master acquires control of the slave port, it controls the port until it relinquishes the 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.

7.3.2 Arbitration

The crossbar switch supports the following arbitration algorithms:

- Fixed priority
- Round-robin

The selection of the global slave port arbitration algorithm is described in the crossbar switch chip-specific information.

7.3.2.1 Arbitration during undefined length bursts

Undefined length bursts can be interrupted.

7.3.2.2 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level with the highest numbered master having the highest priority (for example, in a system with 5 masters, master 1 has lower priority than master 3). If two masters request access to the same slave port, the master with the highest priority gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing access from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port.

Table 7-1. Methods of how the crossbar switch grants control of a slave port to a master

When	Then the crossbar switch grants control to the requesting master
Both of the following are true: • The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	At the next clock edge
Both of the following are true: • The current master is running an undefined length burst transfer. • The requesting master's priority level is higher than that of the current master.	At the next arbitration point for the undefined length burst transfer
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

7.3.2.3 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requests the master owns the slave bus at the next transfer boundary. Priority is based on how far ahead the ID of the requesting master is of the ID of the last master.

After a master is granted access to a slave port, a master may perform as many transfers as desired to that port until another master requests the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle, if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume that the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and masters 0, 4, and 5 make simultaneous requests, they are serviced in this order: 4,5, and then 0.

The round-robin arbitration mode generally provides a more fair allocation of the available slave-port bandwidth (compared to fixed priority) because the fixed master priority does not affect the master selection.

7.3.2.4 Clocking

This module has no clocking considerations.

7.3.2.5 Interrupts

This module has no interrupts.

7.4 External signals

This module has no external signals.

7.5 Initialization/application information

No initialization is required for the crossbar switch.

Chapter 8
Peripheral Bridge (AIPS-Lite)

8.1 Chip-specific information for this module

8.1.1 Peripheral slot assignment

The peripheral bridge is used to access the registers of most of the modules on this device. See Peripheral Bridge (AIPS-Lite) Memory Map for the memory slot assignment.

8.2 Overview

AIPS_Lite converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

This peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 4 KB each. All the peripherals may not be used. See the memory map chapter for details on slot assignments. The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

8.2.1 Features

Following are the key features of the peripheral bridge:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width

8.2.2 General operation

The slave devices connected to the peripheral bridge are modules that contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge.

The register maps of the peripherals are located on 4-KB boundaries. Each peripheral is allocated one or more 4-KB block(s) of the memory map.

8.3 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus.

The peripheral bridge manages all transactions for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

8.3.1 Access support

Aligned and misaligned 32-bit, 16-bit, and byte accesses are supported for 32-bit peripherals. Misaligned accesses are supported to allow memory to be placed on the slave peripheral bus. Peripheral registers must not be misaligned, although no explicit checking is performed by the peripheral bridge. All accesses are performed with a single transfer.

All accesses to the peripheral slots must be sized less than or equal to the designated peripheral slot size. If an access is attempted that is larger than the targeted port, an error response is generated.

8.3.2 Clocking

This module has no clocking considerations.

8.3.3 Interrupts

This module has no interrupts.

8.4 External signals

This module has no external signals.

8.5 Memory map and register definition

The AIPS module(s) on this chip do(es) not contain any user-programmable registers.

Chapter 9
Trigger MUX Control (TRGMUX)

9.1 Chip-specific information for this module

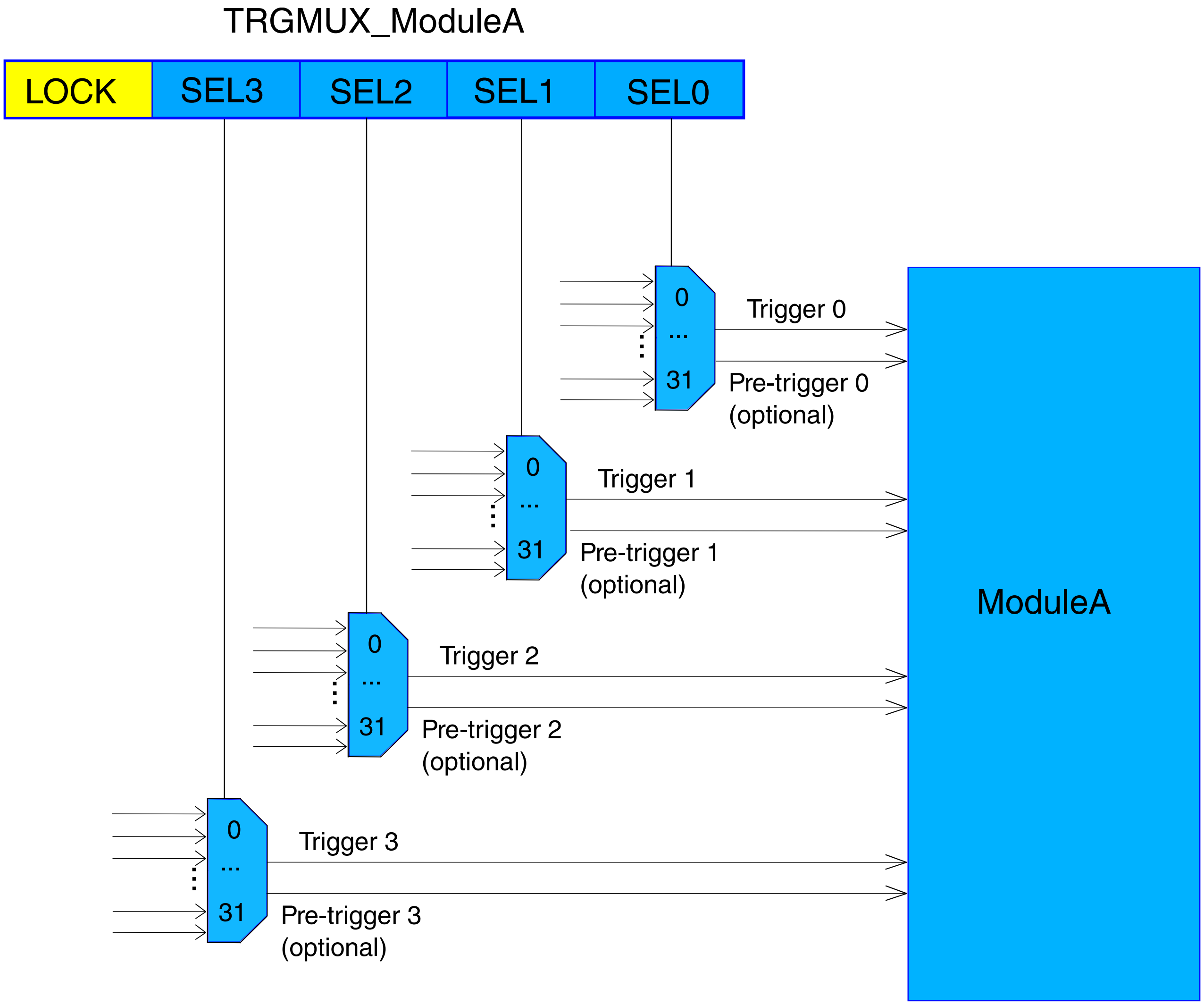
9.1.1 Module Interconnectivity

The module interconnectivity scheme is based on the TRGMUX. The TRGMUX introduces an extremely flexible methodology for connecting various trigger sources to multiple pins/peripherals. This TRGMUX design has removed some trigger inputs, and added one pre-stage trigger source TRGMUX1 for the TRGMUX0. TRGMUX1 supports up to 32 trigger sources and has 8 outputs. These 8 outputs will be the trigger inputs of TRGMUX0. TRGMUX0 supports up to 32 input sources, and its output will be the target modules.

With the TRGMUX, each peripheral which accepts external triggers will usually have one specific 32-bit trigger control register. Each control register supports up to 4 triggers, and each trigger can be selected from up to 32 inputs.

For some trigger sources, there is optional pre-trigger. The trigger and the pre-trigger are 1-1 paired up, and are both selected by the same trigger control register. Not every module has pre-trigger input, please refer to the respective module chapter for details.

Following is the main structure of TRGMUX, and take ModuleA as an example.



NOTE

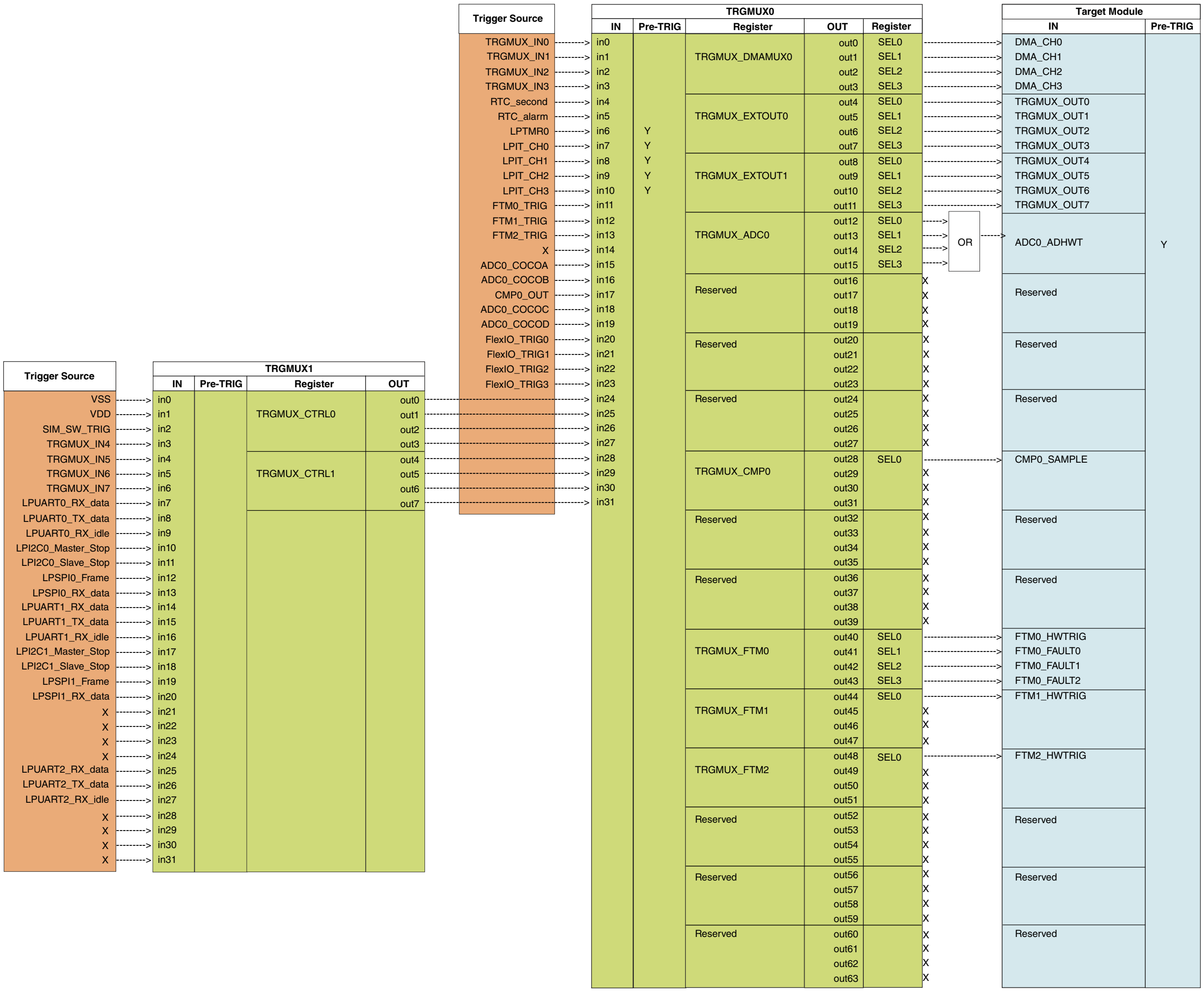
Each TRGMUX control register supports up to 4 trigger channels, but it's not necessary for each module to implement all of the 4 triggers. For those modules (e.g. external output, etc.) which needs more than 4 trigger inputs, multiple control registers are created to support that.

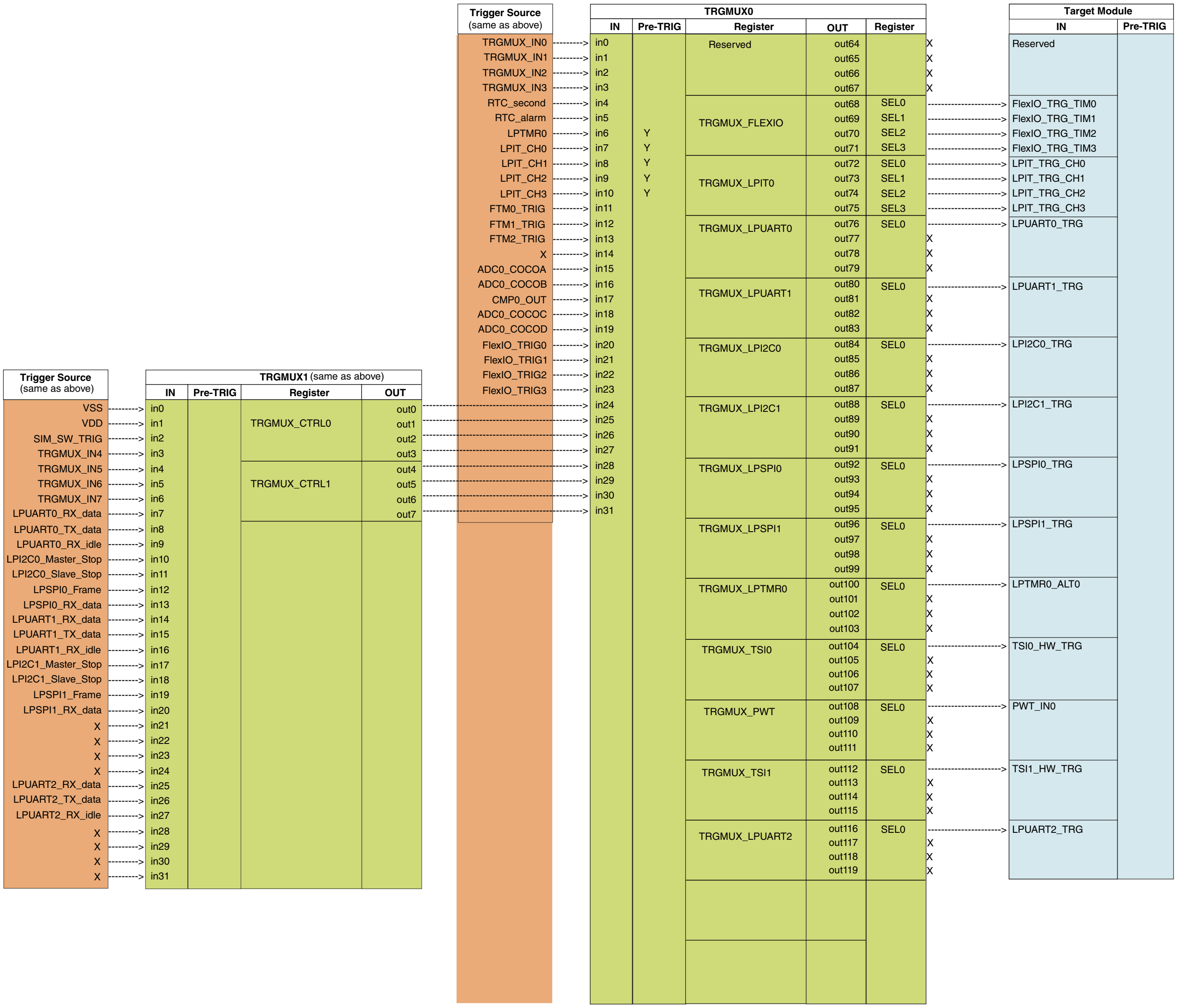
The trigger input and peripheral trigger control are assigned as the following figure indication.

Trigger source	Explanation
VSS	VSS trigger
VDD	VDD trigger
SIM_SW_TRG	Software trigger controlled by SIM module
TRGMUX_INx	TRGMUX external trigger input x
LPUARTx_RX_data	LPUARTx receive end of word trigger

Table continues on the next page...

LPUARTx_TX_data	LPUARTx transmit end of word trigger
LPUARTx_RX_idle	LPUARTx receive idle detected trigger
LPI2Cx_Master_Stop	LPI2Cx master stop or repeated start trigger
LPI2Cx_Slave_Stop	LPI2Cx slave stop or repeated start trigger
LPSPIx_Frame	LPSPIx end of frame trigger
LPSPIx_RX_data	LPSPIx receive data trigger
ADCx_COCOA	ADCx conversion complete trigger for data result A
ADCx_COCOB	ADCx conversion complete trigger for data result B
ADCx_COCOC	ADCx conversion complete trigger for data result C
ADCx_COCOD	ADCx conversion complete trigger for data result D
RTC_second	RTC second trigger
RTC_alarm	RTC alarm trigger
LPTMRx	LPTMRx timer counter match trigger
LPIT_CHx	LPIT channel x timer counter match trigger
FTMx_TRIG	FTMx counter initialization trigger (init_trig) and channel match trigger (ext_trig)
CMPx_OUT	CMPx output trigger
FlexIO_TRIGx	FlexIO timer x counter match trigger


Trigger Source
TRGMUX0
IN
Pre-TRIG
Register
OUT
Register
Target Module
IN
Pre-TRIG
TRGMUX_IN0 in0 out0 SEL0 DMA_CH0
TRGMUX_IN1 in1 out1 SEL1 DMA_CH1
TRGMUX_IN2 in2 out2 SEL2 DMA_CH2
TRGMUX_IN3 in3 out3 SEL3 DMA_CH3
TRGMUX_DMAMUX0
RTC_second in4 out4 SEL0 TRGMUX_OUT0
RTC_alarm in5 out5 SEL1 TRGMUX_OUT1
LPTMR0 in6 Y out6 SEL2 TRGMUX_OUT2
LPIT_CH0 in7 Y out7 SEL3 TRGMUX_OUT3
TRGMUX_EXTOUT0
LPIT_CH1 in8 Y out8 SEL0 TRGMUX_OUT4
LPIT_CH2 in9 Y out9 SEL1 TRGMUX_OUT5
LPIT_CH3 in10 Y out10 SEL2 TRGMUX_OUT6
FTM0_TRIG in11 out11 SEL3 TRGMUX_OUT7
TRGMUX_EXTOUT1
FTM1_TRIG in12 out12 SEL0
FTM2_TRIG in13 out13 SEL1
X in14 out14 SEL2
ADC0_COCOA in15 out15 SEL3
TRGMUX_ADC0
OR
ADC0_ADHWT
Y
ADC0_COCOB in16 out16 X
CMP0_OUT in17 out17 X
ADC0_COCOC in18 out18 X
ADC0_COCOD in19 out19 X
Reserved
Reserved
FlexIO_TRIG0 in20 out20 X
FlexIO_TRIG1 in21 out21 X
FlexIO_TRIG2 in22 out22 X
FlexIO_TRIG3 in23 out23 X
Reserved
Reserved
in24 out24 X
in25 out25 X
in26 out26 X
in27 out27 X
Reserved
Reserved
in28 out28 SEL0 CMP0_SAMPLE
in29 out29 X
in30 out30 X
in31 out31 X
TRGMUX_CMP0
out32 X
out33 X
out34 X
out35 X
Reserved
Reserved
out36 X
out37 X
out38 X
out39 X
Reserved
Reserved
out40 SEL0 FTM0_HWTRIG
out41 SEL1 FTM0_FAULT0
out42 SEL2 FTM0_FAULT1
out43 SEL3 FTM0_FAULT2
TRGMUX_FTM0
out44 SEL0 FTM1_HWTRIG
out45 X
out46 X
out47 X
TRGMUX_FTM1
out48 SEL0 FTM2_HWTRIG
out49 X
out50 X
out51 X
TRGMUX_FTM2
out52 X
out53 X
out54 X
out55 X
Reserved
Reserved
out56 X
out57 X
out58 X
out59 X
Reserved
Reserved
out60 X
out61 X
out62 X
out63 X
Reserved
Reserved
Trigger Source
TRGMUX1
IN
Pre-TRIG
Register
OUT
VSS in0 out0
VDD in1 out1
SIM_SW_TRIG in2 out2
TRGMUX_IN4 in3 out3
TRGMUX_CTRL0
TRGMUX_IN5 in4 out4
TRGMUX_IN6 in5 out5
TRGMUX_IN7 in6 out6
LPUART0_RX_data in7 out7
TRGMUX_CTRL1
LPUART0_TX_data in8
LPUART0_RX_idle in9
LPI2C0_Master_Stop in10
LPI2C0_Slave_Stop in11
LPSPI0_Frame in12
LPSPI0_RX_data in13
LPUART1_RX_data in14
LPUART1_TX_data in15
LPUART1_RX_idle in16
LPI2C1_Master_Stop in17
LPI2C1_Slave_Stop in18
LPSPI1_Frame in19
LPSPI1_RX_data in20
X in21
X in22
X in23
X in24
LPUART2_RX_data in25
LPUART2_TX_data in26
LPUART2_RX_idle in27
X in28
X in29
X in30
X in31



NOTE

When using the TRGMUX to trigger DMA, DMAMUX must be configured (in the DMAMUX_CHCFG register) with ENBL, TRIG bit set, meanwhile SOURCE bits must be !=0 .

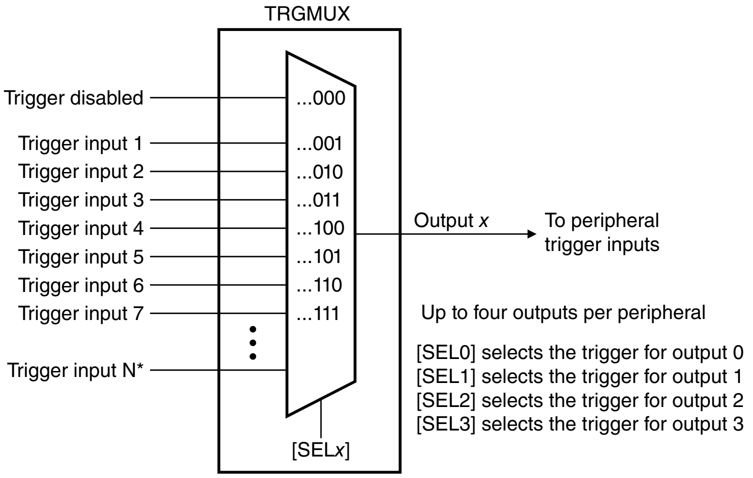
NOTE

For each ADC, the four triggers are OR'ed together to provide a flexible trigger scheme for the hardware trigger of each ADC, while the pre-triggers are not OR'ed. The LPIT pre-triggers can be pre-triggers for each ADC. Please refer to the ADC chapter for details on ADC trigger implementation on this device.

9.2 Overview

TRGMUX allows you to configure the trigger inputs for various peripherals.

9.2.1 Block diagram


TRGMUX
Trigger disabled
Trigger input 1
Trigger input 2
Trigger input 3
Trigger input 4
Trigger input 5
Trigger input 6
Trigger input 7
Trigger input N*
...000
...001
...010
...011
...100
...101
...110
...111
[SELx]
Output x
To peripheral trigger inputs
Up to four outputs per peripheral
[SEL0] selects the trigger for output 0
[SEL1] selects the trigger for output 1
[SEL2] selects the trigger for output 2
[SEL3] selects the trigger for output 3

* Up to 255 trigger inputs may be available for SEL0, SEL1, and SEL2. For SEL3, up to 127 trigger inputs may be available. When the number of trigger inputs is 255, SEL3 is not available and becomes reserved. See the chip-specific TRGMUX information for the maximum number of trigger inputs supported on this chip.

Figure 9-1. Block diagram

9.2.2 Features

- Configurable trigger sources for peripherals
- Dedicated TRGMUX register for each peripheral

9.3 Functional description

9.3.1 Clocking

This module has no clocking considerations.

9.3.2 Interrupts

This module has no interrupts.

9.4 External signals

This module has no external signals.

9.5 Initialization

This module does not require initialization.

9.6 TRGMUX register descriptions

9.6.1 TRGMUX memory map

You can only write to TRGMUX registers in Supervisor mode.

Table 9-1. Select bit fields

Field	Description
SEL*x*	Specifies the MUX select for the peripheral trigger inputs. Use this field to select the trigger sources for peripheral modules. 0h - TRGMUX_IN0 is selected. 1h - TRGMUX_IN1 is selected. 2h - TRGMUX_IN2 is selected. 3h - TRGMUX_IN3 is selected. 4h - RTC_second is selected. 5h - RTC_alarm is selected. 6h - LPTMR0 is selected. 7h - LPIT_CH0 is selected. 8h - LPIT_CH1 is selected. 9h - LPIT_CH2 is selected. Ah - LPIT_CH3 is selected. Bh - FTM0_TRG is selected.

Table 9-1. Select bit fields

Field	Description
	Ch - FTM1_TRG is selected. Dh - FTM2_TRG is selected. Eh - Fh - ADC0_COCOA is selected. 10h - ADC0_COCOB is selected. 11h - CMP0_OUT is selected. 12h - ADC0_COCOC is selected. 13h - ADC0_COCOD is selected. 14h - FLEXIO_TRIG0 is selected. 15h - FLEXIO_TRIG1 is selected. 16h - FLEXIO_TRIG2 is selected. 17h - FLEXIO_TRIG3 is selected. 18h - TRGMUX1 Output 0 is selected. 19h - TRGMUX1 Output 1 is selected. 1Ah - TRGMUX1 Output 2 is selected. 1Bh - TRGMUX1 Output 3 is selected. 1Ch - TRGMUX1 Output 4 is selected. 1Dh - TRGMUX1 Output 5 is selected. 1Eh - TRGMUX1 Output 6 is selected. 1Fh - TRGMUX1 Output 7 is selected.

TRGMUX0 base address: 4006_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX DMAMUX0 (DMAMUX0)	32	RW	0000_0000h
4h	TRGMUX EXTOUT0 (EXTOUT0)	32	RW	0000_0000h
8h	TRGMUX EXTOUT1 (EXTOUT1)	32	RW	0000_0000h
Ch	TRGMUX ADC0 (ADC0)	32	RW	0000_0000h
1Ch	TRGMUX CMP0 (CMP0)	32	RW	0000_0000h
28h	TRGMUX FTM0 (FTM0)	32	RW	0000_0000h
2Ch	TRGMUX FTM1 (FTM1)	32	RW	0000_0000h
30h	TRGMUX FTM2 (FTM2)	32	RW	0000_0000h
44h	TRGMUX FLEXIO (FLEXIO)	32	RW	0000_0000h
48h	TRGMUX LPIT0 (LPIT0)	32	RW	0000_0000h
4Ch	TRGMUX LPUART0 (LPUART0)	32	RW	0000_0000h
50h	TRGMUX LPUART1 (LPUART1)	32	RW	0000_0000h
54h	TRGMUX LPI2C0 (LPI2C0)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
58h	TRGMUX LPI2C1 (LPI2C1)	32	RW	0000_0000h
5Ch	TRGMUX LPSPI0 (LPSPI0)	32	RW	0000_0000h
60h	TRGMUX LPSPI1 (LPSPI1)	32	RW	0000_0000h
64h	TRGMUX LPTMR0 (LPTMR0)	32	RW	0000_0000h
68h	TRGMUX TSI0 (TSI0)	32	RW	0000_0000h
6Ch	TRGMUX PWT (PWT)	32	RW	0000_0000h
70h	TRGMUX TSI1 (TSI1)	32	RW	0000_0000h
74h	TRGMUX LPUART2 (LPUART2)	32	RW	0000_0000h

9.6.2 TRGMUX DMAMUX0 (DMAMUX0)

9.6.2.1 Offset

Register	Offset
DMAMUX0	0h

9.6.2.2 Function

Configures the DMAMUX0 module.

9.6.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.2.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.3 TRGMUX EXTOUT0 (EXTOUT0)

9.6.3.1 Offset

Register	Offset
EXTOUT0	4h

9.6.3.2 Function

Configures the EXTOUT0 module.

9.6.3.3 Diagram

<table>
<tr><td>Bits</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td rowspan="2">LK</td><td colspan="2">0</td><td colspan="5" rowspan="2">SEL3</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL2</td></tr>
<tr><td>W</td><td colspan="2"></td><td colspan="3"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bits</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL1</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL0</td></tr>
<tr><td>W</td><td colspan="3"></td><td colspan="3"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

9.6.3.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.4 TRGMUX EXTOUT1 (EXTOUT1)

9.6.4.1 Offset

Register	Offset
EXTOUT1	8h

9.6.4.2 Function

Configures the EXTOUT1 module.

9.6.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.4.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21	Reserved

Table continues on the next page...

Field	Function
—	
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.5 TRGMUX ADC0 (ADC0)

9.6.5.1 Offset

Register	Offset
ADC0	Ch

9.6.5.2 Function

Configures the ADC0 module.

9.6.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.5.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.6 TRGMUX CMP0 (CMP0)

9.6.6.1 Offset

Register	Offset
CMP0	1Ch

9.6.6.2 Function

Configures the CMP0 module.

9.6.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.6.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.7 TRGMUX FTM0 (FTM0)

9.6.7.1 Offset

Register	Offset
FTM0	28h

9.6.7.2 Function

Configures the FTM0 module.

9.6.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.7.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.

Table continues on the next page...

Field	Function
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.8 TRGMUX FTM1 (FTM1)

9.6.8.1 Offset

Register	Offset
FTM1	2Ch

9.6.8.2 Function

Configures the FTM1 module.

9.6.8.3 Diagram

<table>
<tr><td>Bits</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td rowspan="2">LK</td><td colspan="7">0</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL2</td></tr>
<tr><td>W</td><td colspan="7"></td><td colspan="3"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bits</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL1</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL0</td></tr>
<tr><td>W</td><td colspan="3"></td><td colspan="3"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

9.6.8.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.9 TRGMUX FTM2 (FTM2)

9.6.9.1 Offset

Register	Offset
FTM2	30h

9.6.9.2 Function

Configures the FTM2 module.

9.6.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.9.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.10 TRGMUX FLEXIO (FLEXIO)

9.6.10.1 Offset

Register	Offset
FLEXIO	44h

9.6.10.2 Function

Configures the FLEXIO module.

9.6.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W	▨	▨	▨						▨	▨	▨					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.10.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.

Table continues on the next page...

Field	Function
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.11 TRGMUX LPIT0 (LPIT0)

9.6.11.1 Offset

Register	Offset
LPIT0	48h

9.6.11.2 Function

Configures the LPIT0 module.

9.6.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.11.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-1 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-1 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-1 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.12 TRGMUX LPUART0 (LPUART0)

9.6.12.1 Offset

Register	Offset
LPUART0	4Ch

9.6.12.2 Function

Configures the LPUART0 module.

9.6.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.12.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.13 TRGMUX LPUART1 (LPUART1)

9.6.13.1 Offset

Register	Offset
LPUART1	50h

9.6.13.2 Function

Configures the LPUART1 module.

9.6.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.13.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved

Table continues on the next page...

Field	Function
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.14 TRGMUX LPI2C0 (LPI2C0)

9.6.14.1 Offset

Register	Offset
LPI2C0	54h

9.6.14.2 Function

Configures the LPI2C0 module.

9.6.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.14.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.15 TRGMUX LPI2C1 (LPI2C1)

9.6.15.1 Offset

Register	Offset
LPI2C1	58h

9.6.15.2 Function

Configures the LPI2C1 module.

9.6.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.15.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.16 TRGMUX LPSPI0 (LPSPI0)

9.6.16.1 Offset

Register	Offset
LPSPI0	5Ch

9.6.16.2 Function

Configures the LPSPI0 module.

9.6.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.16.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.17 TRGMUX LPSPI1 (LPSPI1)

9.6.17.1 Offset

Register	Offset
LPSPI1	60h

9.6.17.2 Function

Configures the LPSPI1 module.

9.6.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.17.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved

Table continues on the next page...

Field	Function
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.18 TRGMUX LPTMR0 (LPTMR0)

9.6.18.1 Offset

Register	Offset
LPTMR0	64h

9.6.18.2 Function

Configures the LPTMR0 module.

9.6.18.3 Diagram

<table>
<tr><td>Bits</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td rowspan="2">LK</td><td colspan="7">0</td><td colspan="8">0</td></tr>
<tr><td>W</td><td colspan="7"></td><td colspan="8"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bits</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="8">0</td><td colspan="3">0</td><td colspan="5" rowspan="2">SEL0</td></tr>
<tr><td>W</td><td colspan="8"></td><td colspan="3"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

9.6.18.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.19 TRGMUX TSI0 (TSI0)

9.6.19.1 Offset

Register	Offset
TSI0	68h

9.6.19.2 Function

Configures the TSI0 module.

9.6.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.19.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.20 TRGMUX PWT (PWT)

9.6.20.1 Offset

Register	Offset
PWT	6Ch

9.6.20.2 Function

Configures the PWT module.

9.6.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.20.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.21 TRGMUX TSI1 (TSI1)

9.6.21.1 Offset

Register	Offset
TSI1	70h

9.6.21.2 Function

Configures the TSI1 module.

9.6.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.21.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved

Table continues on the next page...

Field	Function
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.6.22 TRGMUX LPUART2 (LPUART2)

9.6.22.1 Offset

Register	Offset
LPUART2	74h

9.6.22.2 Function

Configures the LPUART2 module.

9.6.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							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								0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.6.22.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable

Table continues on the next page...

Field	Function
	1b - Register is not writable until the next system reset
30-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-1 for field values.

9.7 TRGMUX register descriptions

9.7.1 TRGMUX memory map

You can only write to TRGMUX registers in Supervisor mode.

Table 9-2. Select bit fields

Field	Description
SEL*x*	Specifies the MUX select for the peripheral trigger inputs. Use this field to select the trigger sources for peripheral modules. 0h - VSS is selected. 1h - VDD is selected. 2h - SIM_SW_TRIG is selected. 3h - TRGMUX_IN4 is selected. 4h - TRGMUX_IN5 is selected. 5h - TRGMUX_IN6 is selected. 6h - TRGMUX_IN7 is selected. 7h - LPUART0_RX_data is selected. 8h - LPUART0_TX_data is selected. 9h - LPUART0_RX_idle is selected. Ah - LPI2C0_Master_Stop is selected. Bh - LPI2C0_Slave_Stop is selected. Ch - LPSPI0_Frame is selected. Dh - LPSPI0_RX_data is selected.

Table 9-2. Select bit fields

Field	Description
	Eh - LPUART1_RX_data is selected. Fh - LPUART1_TX_data is selected. 10h - LPUART1_RX_idle is selected. 11h - LPI2C1_Master_Stop is selected. 12h - LPI2C1_Slave_Stop is selected. 13h - LPSPI1_Frame is selected. 14h - LPSPI1_RX_data is selected. 15h - 16h - 17h - 18h - 19h - LPUART2_RX_data is selected. 1Ah - LPUART2_TX_data is selected. 1Bh - LPUART2_RX_idle is selected. 1Ch - 1Dh - 1Eh - 1Fh -

TRGMUX1 base address: 4006_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX CTRL0 (CTRL0)	32	RW	0000_0000h
4h	TRGMUX CTRL1 (CTRL1)	32	RW	0000_0000h

9.7.2 TRGMUX CTRL0 (CTRL0)

9.7.2.1 Offset

Register	Offset
CTRL0	0h

9.7.2.2 Function

Configures the CTRL0 module.

9.7.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.7.2.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-2 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-2 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-2 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-2 for field values.

9.7.3 TRGMUX CTRL1 (CTRL1)

9.7.3.1 Offset

Register	Offset
CTRL1	4h

9.7.3.2 Function

Configures the CTRL1 module.

9.7.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0		SEL3					0			SEL2				
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			SEL1					0			SEL0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.7.3.4 Fields

Field	Function
31 LK	TRGMUX Register Lock Disables writing to the register. You can write to this field once after system reset. When this field is 1, you cannot write to SEL*x* until the next system reset. This field becomes 0 after system reset. 0b - Register is writable 1b - Register is not writable until the next system reset
30-29 —	Reserved

Table continues on the next page...

Field	Function
28-24 SEL3	TRGMUX Source Select 3 Specifies the source select for output 3. See Table 9-2 for field values.
23-21 —	Reserved
20-16 SEL2	TRGMUX Source Select 2 Specifies the source select for output 2. See Table 9-2 for field values.
15-13 —	Reserved
12-8 SEL1	TRGMUX Source Select 1 Specifies the source select for output 1. See Table 9-2 for field values.
7-5 —	Reserved
4-0 SEL0	TRGMUX Source Select 0 Specifies the source select for output 0. See Table 9-2 for field values.

9.8 Usage Guide

The TRGMUX is an extremely flexible module interconnectivity scheme. The trigger source could be from various peripherals and external input pins, to multiple pins/ peripherals. The module level interconnections and trigger scheme offload the intervention of CPU, which is also useful when CPU is in WAIT/STOP mode. The following are some typical use-cases for TRGMUX.

9.8.1 ADC Trigger Source

The following triggers are via the TRGMUX:

- CMP out to trigger each ADC
- LPIT capable to trigger each ADC
- RTC capable to trigger each ADC
- LPTMR capable to trigger each ADC

For details, please refer to “ADC Trigger Sources” section.

For details, please refer to “ADC Trigger Concept – Use Case ” section.

9.8.2 CMP Window/Sample Input

LPIT could be used to generate pulse output which can be used as sampling windows of CMP block via TRGMUX.

For details, please refer to “Window Mode” section in the CMP chapter.

9.8.3 FTM Fault Detection Input / Hardware Triggers and Synchronization

Please refer to the FTM chapter for more details.

Chapter 10
Direct Memory Access Multiplexer (DMAMUX)

10.1 Chip-specific information for this module

10.1.1 DMAMUX request sources

This device includes a DMA request mux that allows up to 63 DMA request signals to be mapped to any of the 8 DMA channels. The DMA request sources could be peripheral DMA requests or always-on slots. Because of the mux, there is not a hard correlation between any of the DMA request sources and a specific DMA channel.

Some of the modules support Asynchronous DMA operation as indicated by the last column in the following DMA source assignment table. Asynchronous DMA requests can be used to activate a DMA channel in WAIT or STOP mode.

Table 10-1. DMA request sources - MUX 0

Source number	Source module	Source description	Async DMA capable
0	—	Channel disabled[1]	
1	TSI0	TSI0 DMA Transfer	Yes
2	LPUART0	Receive	Yes
3	LPUART0	Transmit	Yes
4	LPUART1	Receive	Yes
5	LPUART1	Transmit	Yes
6	LPUART2	Receive	Yes
7	LPUART2	Transmit	Yes
8	Reserved	—	
9	Reserved	—	
10	FlexIO	Shifter0	Yes
11	FlexIO	Shifter1	Yes
12	FlexIO	Shifter2	Yes

Table continues on the next page...

Table 10-1. DMA request sources - MUX 0 (continued)

Source number	Source module	Source description	Async DMA capable
13	FlexIO	Shifter3	Yes
14	LPSPI0	Receive	Yes
15	LPSPI0	Transmit	Yes
16	LPSPI1	Receive	Yes
17	LPSPI1	Transmit	Yes
18	LPI^{2}C0	Receive	Yes
19	LPI^{2}C0	Transmit	Yes
20	FTM0	Channel 0	
21	FTM0	Channel 1	
22	FTM0	Channel 2	
23	FTM0	Channel 3	
24	FTM0	Channel 4	
25	FTM0	Channel 5	
26	FTM0	Channel 6	
27	FTM0	Channel 7	
28	FTM1	Channel 0	
29	FTM1	Channel 1	
30	FTM2	Channel 0	
31	FTM2	Channel 1	
32	LPI^{2}C1	LPI^{2}C1 Receive	Yes for LPI^{2}C1
33	LPI^{2}C1	LPI^{2}C1 Transmit	Yes for LPI^{2}C1
34	Reserved	—	
35	Reserved	—	
36	SCI0	Receive	
37	SCI0	Transmit	
38	SCI1	Receive	
39	SCI1	Transmit	
40	ADC0	ADC0 COCO	Yes
41	Reserved	—	
42	Reserved	—	
43	CMP0	—	Yes
44	Reserved	—	
45	Reserved	—	
46	Reserved	—	
47	Reserved	—	
48	TSI1	TSI1 DMA Transfer	Yes
49	Port control module	Port A	Yes

Table continues on the next page...

Table 10-1. DMA request sources - MUX 0 (continued)

Source number	Source module	Source description	Async DMA capable
50	Port control module	Port B	Yes
51	Port control module	Port C	Yes
52	Port control module	Port D	Yes
53	Port control module	Port E	Yes
54			
55			
56			
57	FTM1	OR of ch2-ch3	
58	FTM2	OR of ch2-ch3	
59	LPTMR0	—	Yes
60	DMAMUX	Always enabled	
61	DMAMUX	Always enabled	
62	DMAMUX	Always enabled	
63	DMAMUX	Always enabled	

1. Configuring a DMA channel to select source 0 or any of the reserved sources disables that DMA channel.

10.1.2 DMA trigger sources

The DMAMUX on this device also supports a periodic trigger mode. The trigger sources are from TRGMUX output showed in following table. The triggers from TRGMUX module can trigger a DMA transfer on the first four DMA channels (channel 0 -3), for example, the LPIT can trigger DMA via TRGMUX.

Table 10-2. DMAMUX trigger sources

Trigger number	Trigger module	Trigger description
0	TRGMUX	TRGMUX trigger out0
1	TRGMUX	TRGMUX trigger out1
2	TRGMUX	TRGMUX trigger out2
3	TRGMUX	TRGMUX trigger out3

10.2 Introduction

10.2.1 Overview

DMAMUX routes DMA sources, called slots, to any of the 8 DMA channels. See the chip-specific information for details.

10.2.2 Block diagram

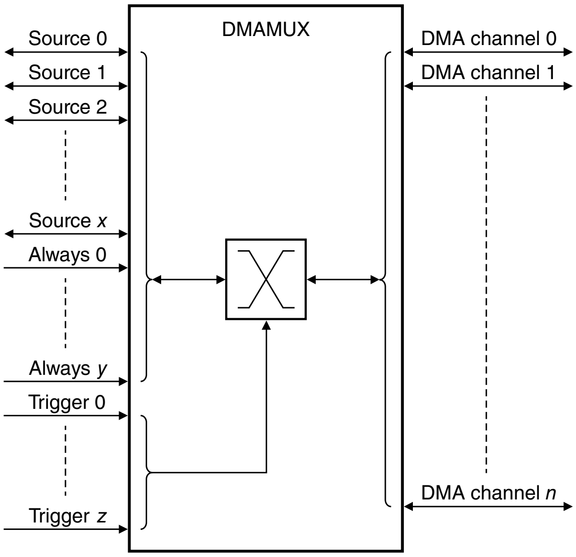


Figure 10-1. Block diagram

10.2.3 Features

- Ability to route up to 59 peripheral slots and up to 4 always-on slots to 8 channels
- Supports 8 independently selectable DMA channel routers (the first 4 channels provide an additional trigger functionality)
- Allows assignment of each channel router to one of the possible peripheral DMA slots or always-on slots

10.2.4 Modes of operation

The following table describes the functional operating modes of DMAMUX.

Table 10-3. Modes of operation

Mode	Description
Disabled	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. You can also use this mode to temporarily suspend a DMA channel while system reconfiguration takes place, for example, by changing the period of a DMA trigger.
Normal	In this mode, a DMA source is routed directly to the specified DMA channel. The operation of DMAMUX in this mode is completely transparent to the system.
Periodic trigger	In this mode, a DMA source can only request a DMA transfer, such as, when a transmit buffer becomes empty or a receive buffer becomes full, periodically. You configure the period using an external periodic interrupt timer (LPIT). This mode is available only for channels 0–3. See the chip-specific section for more details.

10.3 Functional description

The primary purpose of DMAMUX is to provide flexibility in the system's use of the available DMA channels. Configuration of DMAMUX is intended to be a static procedure performed during the execution of the system boot code. However, if you follow the procedure outlined in Enabling and configuring sources, you can change the configuration of DMAMUX during the normal operation of the system. All DMA channels must be inactive before and during a configuration change.

Functionally, DMAMUX channels can be divided into two classes:

- Channels that implement the normal routing functionality plus the periodic triggering capability
- Channels that implement only the normal routing functionality

10.3.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first four channels of DMAMUX provide a special periodic triggering capability that you can use to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention (see Figure 10-2).

An external periodic interrupt timer (LPIT) generates the trigger; for example, by configuring the external periodic timer, you configure the periodic triggering interval.

Note

Because of the dynamic nature of the system (owing to DMA channel priorities, bus arbitration, interrupt service routine lengths, and so on), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

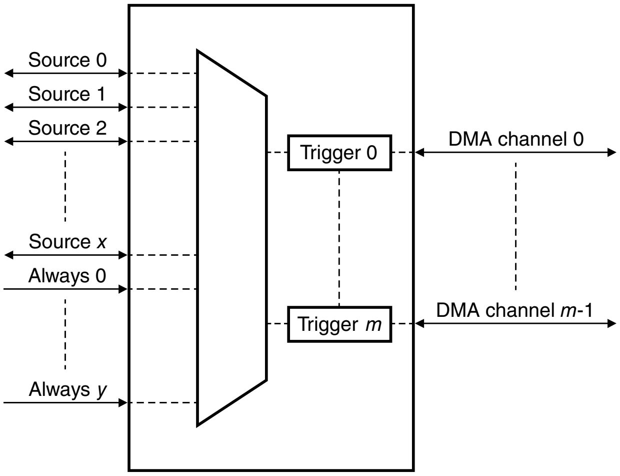


Figure 10-2. DMAMUX triggered channels

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 DMA until a trigger event is observed (see Figure 10-3).

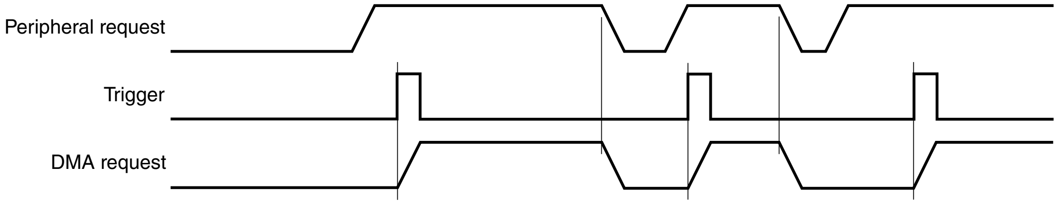


Figure 10-3. DMAMUX channel triggering: normal operation

After the DMA request is serviced, the peripheral negates 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 is ignored (see Figure 10-4).

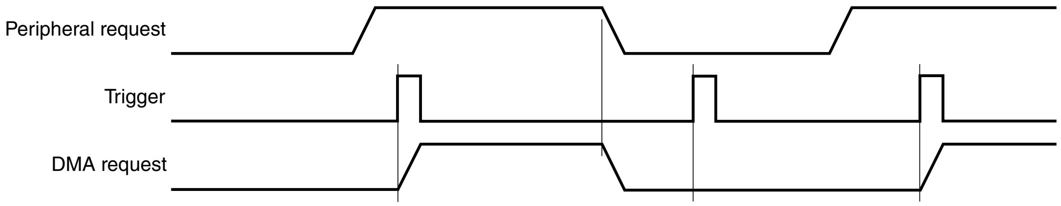


Figure 10-4. DMAMUX channel triggering: ignored trigger

You can use this triggering capability with any peripheral that supports DMA transfers, and is most useful in the following situations:

- Periodic polling of 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 in the aforementioned text. After periodic polling is set up, SPI requests 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 SPI, you can configure SPI and DMA to transfer the received data into memory, effectively implementing a method to periodically read data from external devices and transferring the results into memory without processor intervention.
- Using the GPIO ports to drive or sample waveforms: by configuring 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 DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in a tabular form in on-chip memory.

10.3.2 DMA channels with no triggering capability

The other channels of DMAMUX provide the normal routing functionality as described in Modes of operation.

10.3.3 Always-enabled DMA sources

In addition to the peripherals that you can use as DMA sources, 4 additional DMA sources 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 data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to and from GPIO: moving data to and from 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 software must explicitly start or activate

In cases where software must 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 as follows.

- Transfer all data in a single minor loop: by configuring 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 of this option is reduced granularity in determining the load that the DMA transfer imposes on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use explicit software reactivation: using this option, 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: using this option, 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, you must enable the DMA channel and point it to an "always-enabled" source. Channel reactivation 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 data packets from one source to another, without processor intervention.

10.4 Initialization and application information

This section provides instructions for initializing the DMA channel MUX.

10.4.1 Reset

The reset state of each field is shown in Memory map and register definition. In summary, after reset, all channels are disabled and you must enable them explicitly before use.

10.4.2 Enabling and configuring sources

10.4.2.1 Enable a source with periodic triggering

Perform the following procedure to enable a source with periodic triggering:

1. Determine the DMA channel with which the source is associated. Only the first 4 DMA channels have periodic triggering capability.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Ensure that the DMA channel is properly configured in DMA. You can enable the DMA channel 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 CHCFG*n*[ENBL] and CHCFG*n*[TRIG] are 1.

The following is an example. See the chip-specific information for the number of DMA channels on this chip that have triggering capability.

Example: To configure source 5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0h to CHCFG1.
2. Configure channel 1 in DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write C5h to CHCFG1.

The following code example illustrates steps 1 and 4 above:

Listing 10-1. Configuring source 5 with DMA channel 1 (with periodic triggering)

```
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+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                                 /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0xC5;                                 /* ENBL = 1  DMA Channel is enabled */
                                                /* TRIG = 1  Triggering is enabled */
                                                /* SOURCE = 5  DMA Source 5 is selected */
```

10.4.2.2 Enable a source without periodic triggering

Perform the following procedure to enable a source without periodic triggering:

1. Determine the DMA channel with which the source is associated. Only the first 4 DMA channels have periodic triggering capability.
2. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Ensure that the DMA channel is properly configured in DMA. You can enable the DMA channel at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] is 1 while CHCFG*n*[TRIG] is 0.

The following is an example. See the chip configuration details for the number of DMA channels on this chip that have triggering capability.

Example: To configure source 5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0h to CHCFG1.
2. Configure channel 1 in DMA, including enabling the channel.
3. Write 85h to CHCFG1.

The following code example illustrates steps 1 and 3 above:

Listing 10-2. Configuring source 5 with DMA channel 1 (without periodic triggering)

```
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+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                                 /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0x85;                                 /* ENBL = 1  DMA Channel is enabled */
                                                /* TRIG = 0  Triggering is disabled */
                                                /* SOURCE = 5  DMA Source 5 is selected */
```

10.4.2.3 Disable a source

You can disable a particular DMA source by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configurations may be necessary. See the "Enhanced Direct Memory Access (eDMA)" chapter for 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. Write 0 to CHCFG*n*[ENBL] and CHCFG*n*[TRIG] of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG*n*[ENBL] and CHCFG*n*[TRIG] are 1.

Example: 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 does not have the triggering capability.
2. Write 0h to CHCFG8.
3. Write 87h to CHCFG8. (In this example, writing 1 to CHCFG[TRIG] has no effect because of the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

Listing 10-3. Switching DMA channel 8 from source 5 transmit to source 7 transmit

```
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+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;                               /* Clear all the fields of CHCFG1 register */
*CHCFG8 = 0x87;                               /* ENBL = 1  DMA Channel is enabled */
                                              /* TRIG = 0  Triggering is disabled */
                                              /* SOURCE = 7  DMA Source 7 is selected */
```

10.5 Memory map and register definition

This section provides a detailed description of all memory-mapped registers in DMAMUX.

10.5.1 DMAMUX register descriptions

10.5.1.1 DMAMUX memory map

DMAMUX base address: 4002_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7h	Channel Configuration (CHCFG0 - CHCFG7)	8	RW	00h

10.5.1.2 Channel Configuration (CHCFG0 - CHCFG7)

10.5.1.2.1 Offset

For a = 0 to 7:

Register	Offset
CHCFGa	0h + (a × 1h)

10.5.1.2.2 Function

Provides you with the configuration to enable or disable, and to associate each of the DMA channels with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

> Writing to multiple CHCFG registers with the same source value results in unpredictable behavior. This is true, even if a channel is disabled (CHCFG*n*[ENBL] == 0).
>
> Before changing the trigger or source settings, you must disable a DMA channel using CHCFG*n*[ENBL].

10.5.1.2.3 Diagram

Bits	7	6	5	4	3	2	1	0
R W	ENBL	TRIG	SOURCE					
Reset	0	0	0	0	0	0	0	0

10.5.1.2.4 Fields

Field	Function
7 ENBL	DMA Channel Enable Enables the DMA channel. The condition when the DMA channel is disabled (ENBL = 0) is called Disabled mode. This mode is primarily used during the configuration of DMAMUX. DMA has separate channel enables or disables, which you must use to disable or reconfigure a DMA channel. 0b - Disable 1b - Enable
6 TRIG	DMA Channel Trigger Enable Enables the periodic triggering capability for the triggered DMA channel. If triggering is disabled and ENBL = 1, the DMA channel routes the specified source to the DMA channel (Normal mode).

Table continues on the next page...

Field	Function
	If triggering is enabled and ENBL = 1, then DMAMUX is in Periodic Trigger mode. 0b - Disable 1b - Enable
5-0 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 peripherals and their slot numbers.

Chapter 11
Enhanced Direct Memory Access (eDMA)

11.1 Overview

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

11.1.1 Block diagram

The following figure illustrates the components of the eDMA system, including the eDMA module ("engine").

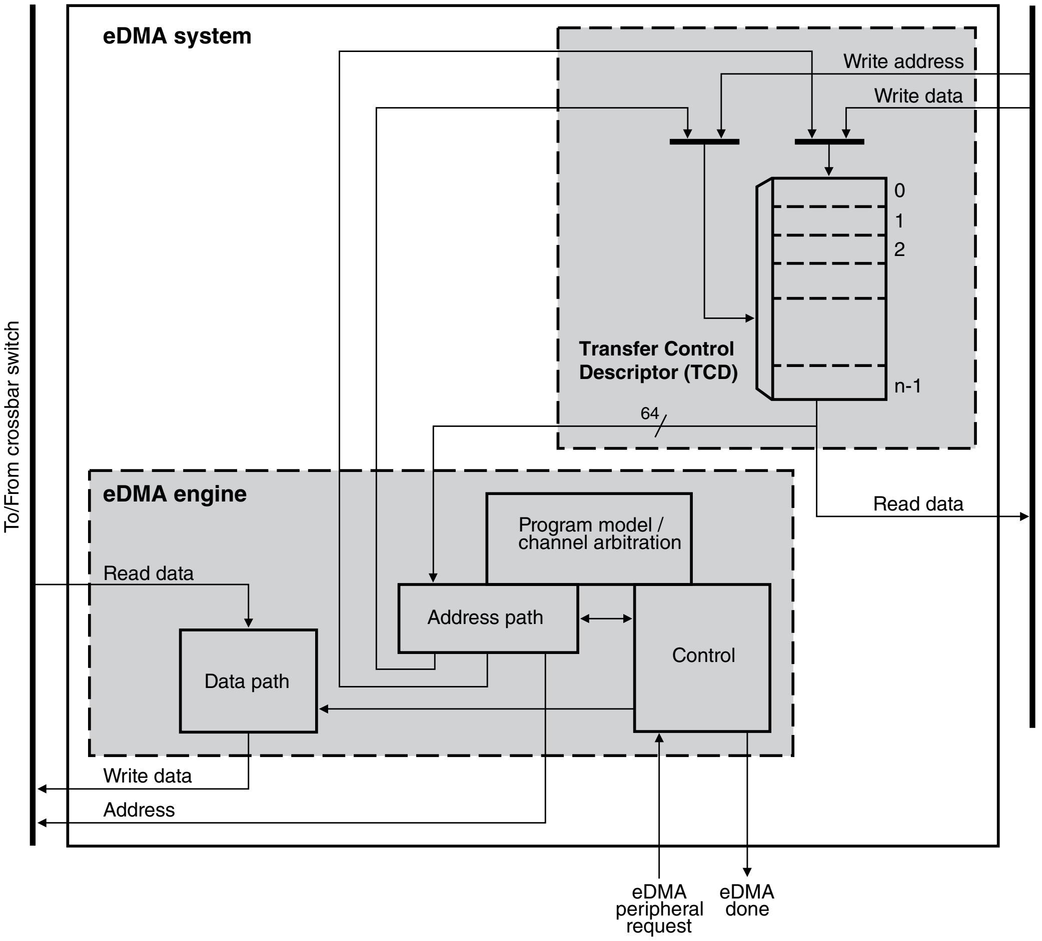


Figure 11-1. Block diagram

11.1.2 Block parts

The eDMA module comprises two major modules: the eDMA engine and the transfer-control descriptor local memory.

Table 11-1 describes the eDMA engine submodules.

Table 11-1. eDMA engine submodules

Submodule	Function
Address path	The address path block: • Provides registered versions of two channel Transfer Control Descriptors (TCDs)—channel x (normal start) and channel y (preemption start) • Manages all master bus-address calculations All channels provide the same functionality. This structure enables preemption of data transfers associated with an active channel (after completion of a read/write sequence) if the eDMA engine asserts a higher priority channel activation.

Table continues on the next page...

Table 11-1. eDMA engine submodules (continued)

Submodule	Function
	After eDMA activates a channel, it runs until the minor loop completes, unless preempted by a higher priority channel. This provides a mechanism (enabled by DCHPRI*n*[ECP]) in which the eDMA engine can preempt a large data move operation to minimize the time another channel stalls. When the eDMA engine selects a channel to execute, it reads the contents of the channel TCD from local memory and loads it into one of the following: • The address path channel x registers (normal start) • The address path channel y registers (preemption start) After the minor loop execution completes, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count completes, the eDMA engine performs additional processing, including: • Final address pointer updates • Reloading the TCD*n*_CITER field • A possible fetch of the next TCD*n* from memory as part of a scatter/gather operation.
Data path	The data path block implements the bus master read/write data path. 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 two-stage pipelined internal bus. The address path module represents the first stage of the bus pipeline (address phase). The data path module implements the second stage of the pipeline (data phase).
Programming model/ channel arbitration	This block implements: • The first section of the eDMA programming model • Channel arbitration logic The programming model registers connect to the chip's internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs also connect to this block (via control logic).
Control	The control block provides all control functions for the eDMA engine. For data transfers in which the source size (SSIZE) and destination size (DSIZE) are equal, the eDMA engine performs a series of source read/destination write operations until it has transferred the number of bytes specified in the minor loop byte count (NBYTES). For TCDs in which the source and destination sizes are not equal, the eDMA engine executes multiple accesses of the smaller size data for each reference of the larger size. For example, if the source size (SSIZE) references 16-bit data and the destination size (DSIZE) is 32-bit data, eDMA performs two reads, then one 32-bit write.

Table 11-2 explains the partitioning of the TCD local memory.

Table 11-2. Transfer control descriptor memory

Submodule	Description
Memory controller	The Memory controller logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. If simultaneous accesses occur, the eDMA engine receives priority and the peripheral transaction stalls.
Memory array	The Memory array provides TCD storage for the transfer profile for each channel.

11.1.3 Features

The eDMA module is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. Use it for applications where you statically know the size of the data to be transferred and do not define the size 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 performs complex data transfers with minimal intervention from a host processor
    - Connections to the crossbar switch (AXBS) for bus mastering the data movement
- TCD supports 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 notification via programmable interrupt requests
    - One interrupt per channel. eDMA engine can generate an interrupt when major iteration count completes
    - 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

NOTE

In the discussion of this module, *n* is the channel number.

11.2 Functional description

The operation of the eDMA is described in the following subsections.

11.2.1 Modes of operation

eDMA operates in the following modes:

Table 11-3. Modes of operation

Mode	Description
Normal	In Normal mode, eDMA transfers data from a source to a destination. The source and destination can be a memory block or an I/O block capable of operation with eDMA. A *service request* initiates a transfer of a specific number of bytes (NBYTES) as specified in the TCD. • The *minor loop* is the sequence of read and write operations that transfers the NBYTES of data for a service request. • Each service request executes one iteration of the major loop, transferring NBYTES of data.
Debug	DMA operation is configurable in Debug mode via Control (CR) • If CR[EDBG] = 0, eDMA continues to operate normally when the chip is in debug mode. • If CR[EDBG] = 1, eDMA stops transferring data when the chip enters debug mode. If a channel is active when eDMA enters Debug mode, eDMA continues operation until the channel retires.
Wait	Before entering Wait mode, eDMA attempts to complete any transfer that is in progress. After the transfer completes, the chip enters Wait mode.

11.2.2 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:

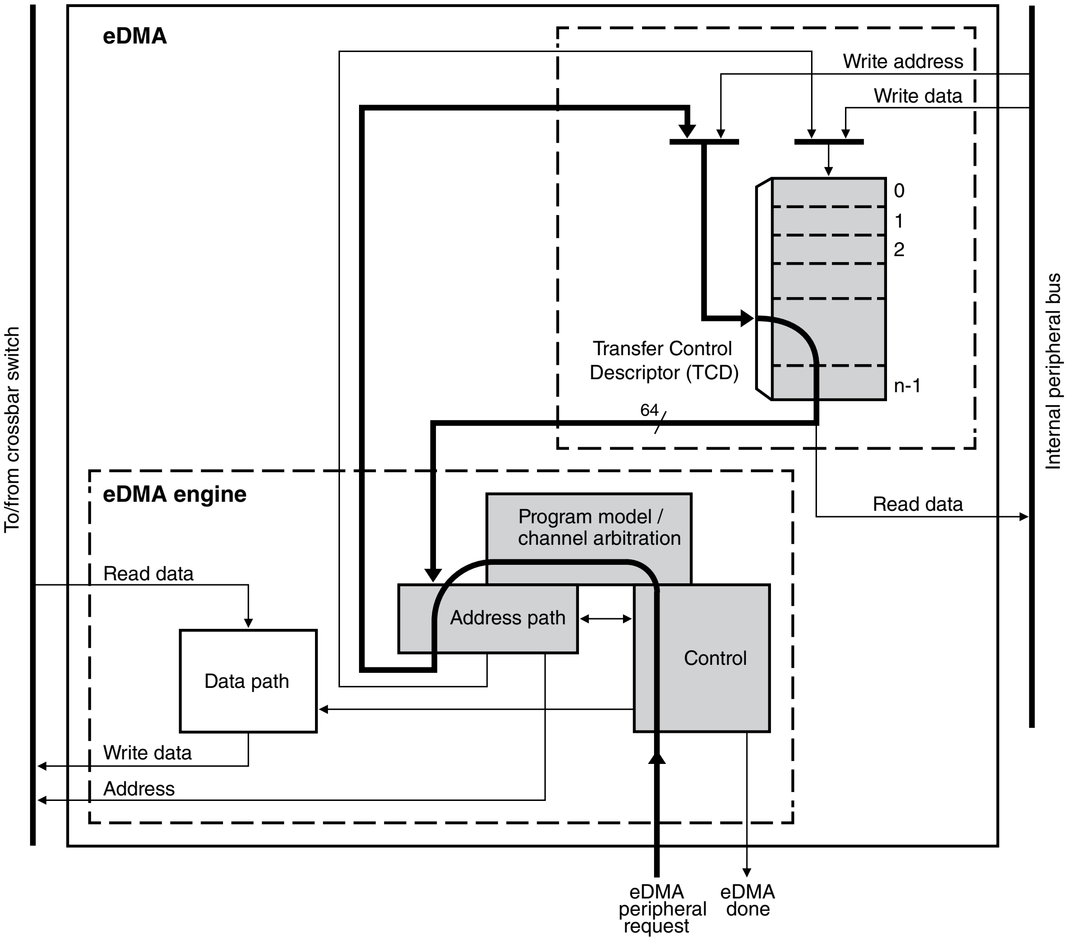


Figure 11-2. eDMA operation, part 1

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 executes, using either 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's internal register file. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the internal register file.

The following diagram illustrates the second part of the basic data flow:

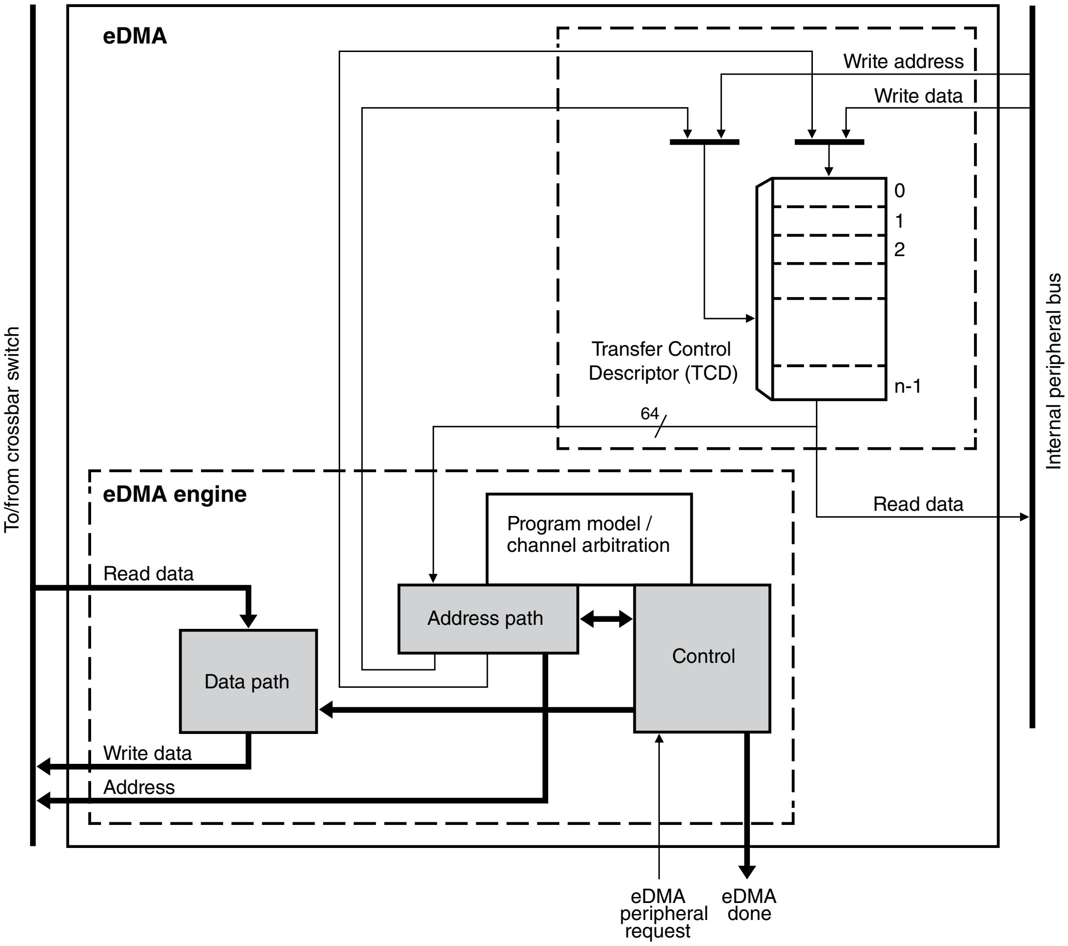


Figure 11-3. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) execute sequentially through the required source reads and destination writes to perform the 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.

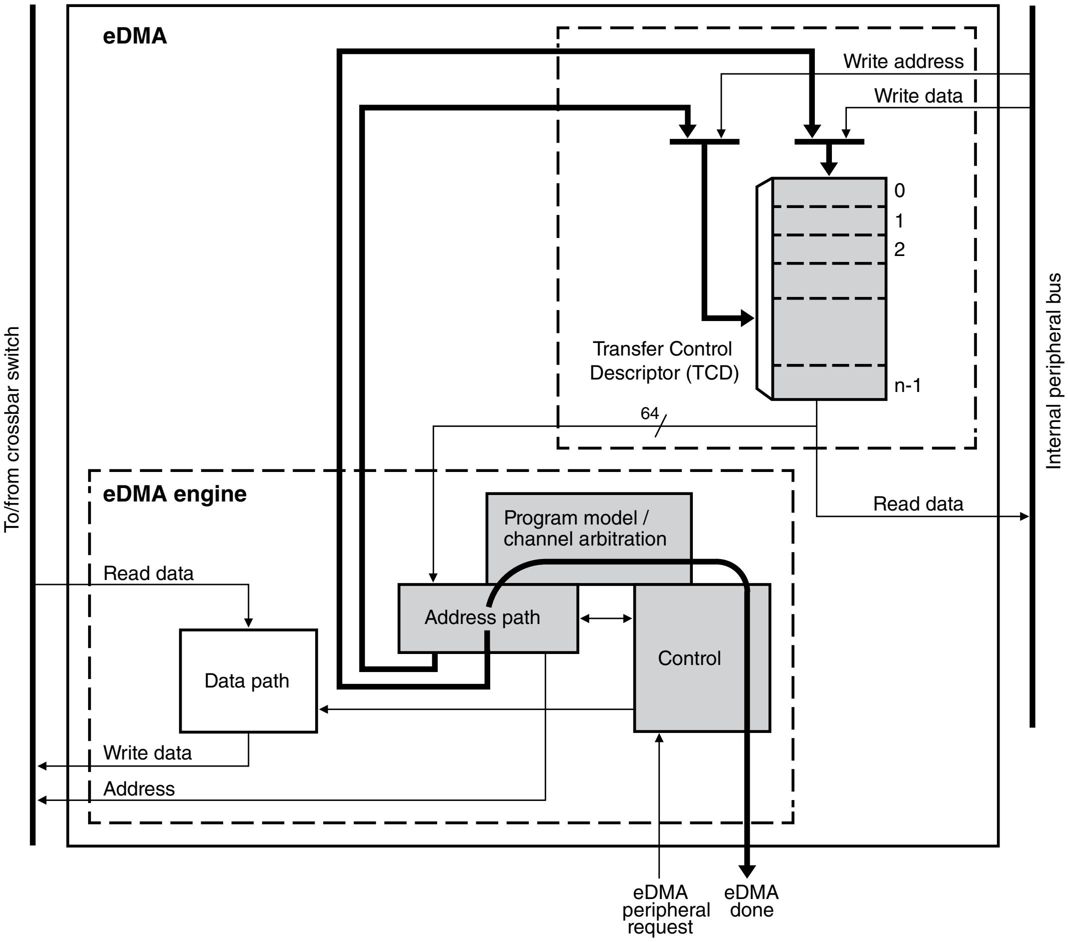


Figure 11-4. eDMA operation, part 3

11.2.3 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 is 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 is reported in the Error Status register. However, the channel number is not 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 channel priorities within a group are not unique, the DMA is halted after the CPE error is recorded. The DMA remains halted and does not process any channel service requests. After 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[ELINK] bit does not equal the TCDn_BITER[ELINK] 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 canceled 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 canceled channel number and ECX is set. The TCD of a canceled 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 canceled 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 enables you to stop a large data transfer when 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 manage 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 because 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.

11.2.4 Channel preemption

Channel preemption is enabled on a per-channel basis by setting DCHPRIn[ECP]. Channel preemption enables the executing channel's data transfers to temporarily be suspended in favor of starting a higher priority channel. After the preempting channel has completed 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 preemption ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This enables 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.

11.2.5 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.

11.2.5.1 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 11-4. 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
48.0 MHz, 32 bit	96.0	48.0	38.4
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.

11.2.5.2 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 11-5. 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 11-6. eDMA peak request rate (MReq/sec)

System frequency (MHz)	Request rate with zero wait states	Request rate with wait states
48.0	5.3	4.2
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 11-7. 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)

11.2.5.3 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.

11.2.6 Clocking

This module has no clocking considerations.

11.2.7 Interrupts

Software can enable the interrupt for each channel for the following events:

1. The major loop is half complete (INTHALF)
2. The major loop is complete (INTMAJOR)
3. A configuration error occurs (EEI)

11.3 External signals

This module has no external signals.

11.4 Initialization

The following sections discuss initialization of the eDMA and programming considerations.

11.4.1 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 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 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 in the programming model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in Table 11-8, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the system 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 have been 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 11-8. 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)
DREQ	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
ESG	Control bit to enable scatter/gather feature
INTHALF	Control bit to enable interrupt when major loop is half complete
INTMAJOR	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).

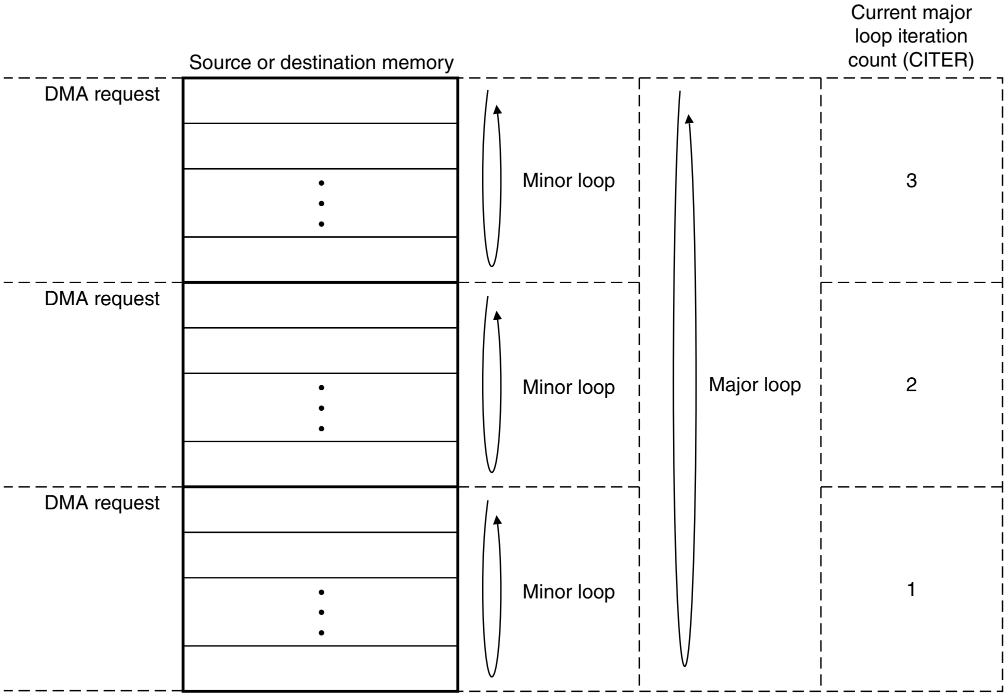


Figure 11-5. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

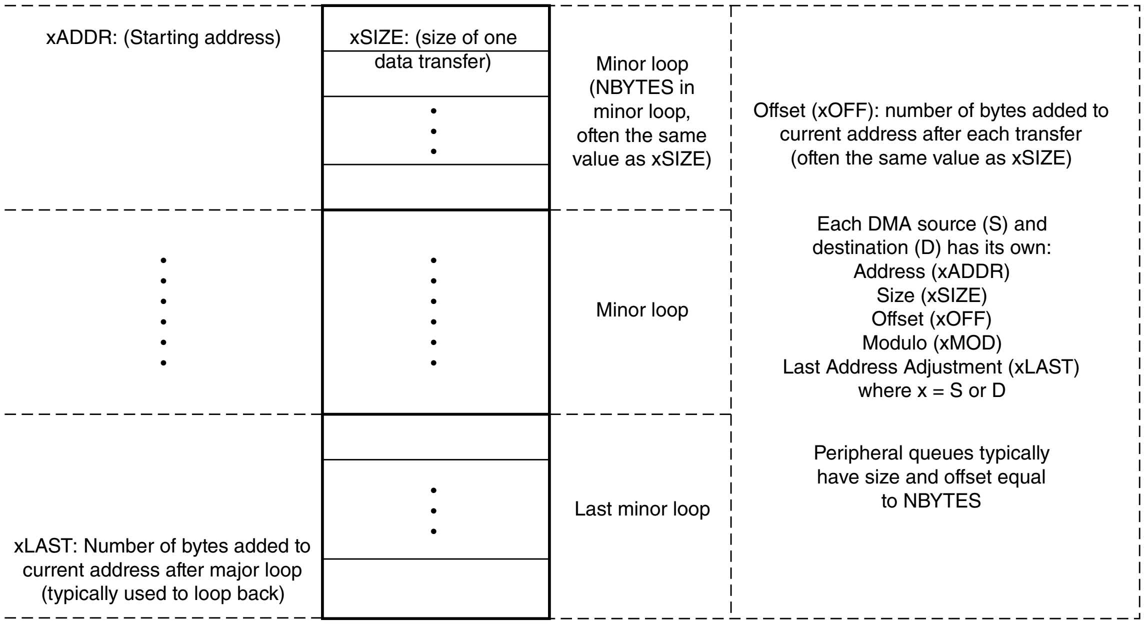

xADDR: (Starting address)
xSIZE: (size of one data transfer)
Minor loop (NBYTES in minor loop, often the same value as xSIZE)
Offset (xOFF): number of bytes added to current address after each transfer (often the same value as xSIZE)
Minor loop
Each DMA source (S) and destination (D) has its own:
Address (xADDR)
Size (xSIZE)
Offset (xOFF)
Modulo (xMOD)
Last Address Adjustment (xLAST)
where x = S or D
Last minor loop
Peripheral queues typically have size and offset equal to NBYTES
xLAST: Number of bytes added to current address after major loop (typically used to loop back)

Figure 11-6. Memory array terms

11.4.2 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 are associated with the selected channel.

11.4.3 Arbitration mode considerations

This section discusses arbitration considerations for the eDMA.

11.4.3.1 Fixed channel arbitration

In this mode, the channel service request from the highest priority channel is selected to execute.

11.4.3.2 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.

11.4.4 DMA transfer examples

This section presents examples of how to perform DMA transfers with the eDMA.

11.4.4.1 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, TCD*n*_CSR[DONE] 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 an 8-bit 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[INTMAJOR] = 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.

11.4.4.2 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, the eDMA peripheral, 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*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.

11.4.4.3 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 zero 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 11-9. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

11.4.5 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

11.4.5.1 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 TCD*n*_CSR[START] and TCD*n*_CSR[ACTIVE]. The minor-loop-complete condition is indicated by both bits reading zero after 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 service requests initiated by hardware, that is, peripherals, 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.

11.4.5.2 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 when a channel executes. The true values of 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.

11.4.5.3 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, 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.

11.4.6 Channel linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD*n*_CSR[START] bit of another channel (or itself), thus 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[ELINK] 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[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJOR_ELINK] = 1
TCDn_CSR[MAJOR_LINKCH] = 0x3
```

executes as:

1. Minor loop done → set TCD2_CSR[START] bit.
2. Minor loop done → set TCD2_CSR[START] bit.

3. Minor loop done → set TCD2_CSR[START] bit.
4. Minor loop done, major loop done → set TCD3_CSR[START] bit.

When minor loop linking is enabled (TCD*n*_CITER[ELINK] = 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[ELINK] = 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[ELINK] bit and the TCD*n*_BITER[ELINK] bit must be 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, that is, use another channel's TCD, at the end of a loop.

Table 11-10. Channel linking parameters

Desired link behavior	TCD control field name	Description
Link at end of minor loop	CITER[ELINK]	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_ELINK]	Enable channel-to-channel linking on major loop completion
	CSR[MAJOR_LINKCH]	Link channel number when linking at end of major loop

11.4.7 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

11.4.7.1 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.

11.4.7.2 Dynamic channel linking

Dynamic channel linking is the process of setting TCDn_CSR[MAJORELINK] during channel execution (see the diagram in TCD structure). This field is read from the TCD local memory at the end of channel execution, thus enabling you to enable the feature during channel execution.

Because you can change the configuration during execution, a coherency model is needed. Consider the scenario where you attempt to execute a dynamic channel link by enabling TCDn_CSR[MAJORELINK] at the same time the eDMA engine is retiring the channel. TCDn_CSR[MAJORELINK] 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 one to TCDn_CSR[MAJORELINK].
2. Read back TCDn_CSR[MAJORELINK].
3. Test the TCDn_CSR[MAJORELINK] request status:
    - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
    - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces TCDn_CSR[MAJORELINK] 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

You must clear TCDn_CSR[DONE] before writing TCDn_CSR[MAJORELINK]. The eDMA engine automatically clears TCDn_CSR[DONE] after a channel begins execution.

11.4.7.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It enables 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 you are able to change the configuration during execution, a coherency model is needed. Consider the scenario where you attempt to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] bit at the same time the eDMA engine is retiring the channel. The ESG bit 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 MAJORLINKCH field and the ESG bit with a single read. For both dynamic channel linking and scatter/gather requests, the TCD local memory controller forces the TCD MAJOR[ELINK] and ESG bits to zero on any writes to a channel's TCD word 7 if that channel's TCD[DONE] bit is set, indicating the major loop is complete.

NOTE

> The user must clear the TCDn_CSR[DONE] bit before writing the MAJORELINK or ESG bits. The TCDn_CSR[DONE] bit is cleared automatically by the eDMA engine after a channel begins execution.

11.4.7.3.1 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 TCDn_CSR[MAJORELINK] is zero, TCDn_CSR[MAJORLINKCH] is not used by the eDMA. In this case, MAJORLINKCH may be used for other purposes. This method uses the MAJORLINKCH field as a TCD identification (ID).

1. When the descriptors are built, write a unique TCD ID in TCDn_CSR[MAJORLINKCH] for each TCD associated with a channel using dynamic scatter/gather.
2. Write one to TCDn_CSR[DREQ].

   Should a dynamic scatter/gather attempt fail, setting the DREQ bit prevents a future hardware activation of the 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_SGA final offset value.

3. Write the TCDn_DLASTSGA register with the scatter/gather address.
4. Write one to TCDn_CSR[ESG].

5. Read back the 16-bit TCD control/status field.
6. Test the ESG request status and MAJORLINKCH value in the TCDn_CSR register:

   If ESG = 1, the dynamic link attempt was successful.

   If ESG = 0 and MAJORLINKCH (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If ESG = 0 and MAJORLINKCH (ID) changed, the dynamic link attempt was successful (the new TCD's ESG value cleared the ESG bit).

11.4.7.3.2 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 identification (ID).

1. Write one to TCDn_CSR[DREQ].

   Should a dynamic scatter/gather attempt fail, setting TCDn_CSR[DREQ] prevents a future hardware activation of the 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_SGA final offset value.

2. Write the TCDn_DLAST_SGA register with the scatter/gather address.
3. Write one to TCDn_CSR[ESG].
4. Read back TCDn_CSR[ESG].
5. Test the ESG request status:

   If ESG = 1, the dynamic link attempt was successful.

   If ESG = 0, read the 32-bit TCDn_DLAST_SGA field.

   If ESG = 0 and TCDn_DLAST_SGA did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If ESG = 0 and TCDn_DLAST_SGA changed, the dynamic link attempt was successful (the new TCD's ESG value cleared the ESG bit).

11.4.8 Suspend/resume a DMA channel with active hardware service requests

The DMA enables you to move data from memory or peripheral registers to another location in memory or peripheral registers without CPU interaction. After the DMA and peripherals have been configured and are active, it is rare to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, a specific procedure must be followed. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (SPI), ADC, or other module.

11.4.8.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status register (DMA_HRSn) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ bit on appropriate DMA channel.

11.4.8.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting the relevant ERQ bit.
2. Enable the DMA service request at the peripheral.

For example, assume the SPI is set as a master for transmitting data via a DMA service request when the SPI_TXFIFO has an empty slot. The DMA transfers the next command and data to the TXFIFO upon the request. You must suspend the DMA/SPI transfer loop and perform the following steps:

1. Disable the DMA service request at the source by writing zero to SPI_RSER[TFFF_RE]. Confirm that SPI_RSER[TFFF_RE] is zero.
2. Ensure there is no DMA service request from the SPI by verifying that DMA_HRS[HRS*n*] is zero for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ bit. If a service request is present, wait until the request has been processed and the HRS bit reads zero.

11.5 Memory map/register definition

The eDMA programming model consists of registers that provide:

- Control and status functions
- Channel configuration functions
- TCD definition functions

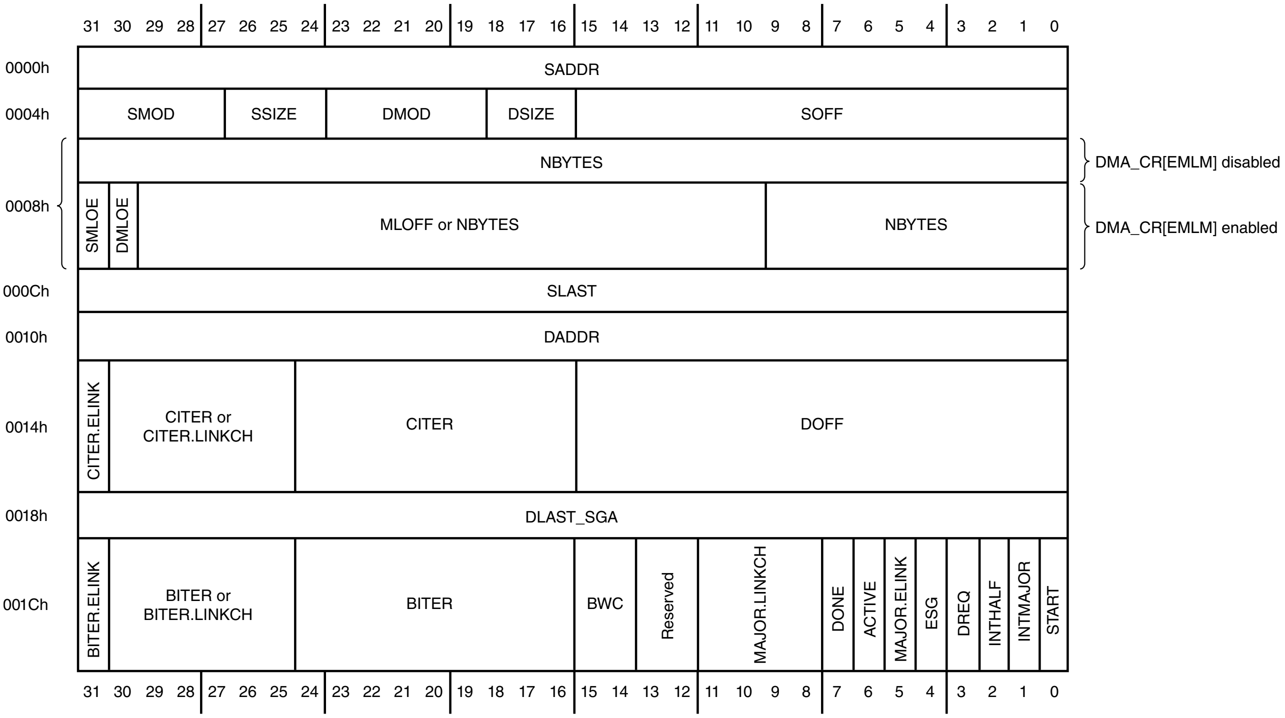
11.5.1 TCD memory

Each channel requires a 32-byte TCD to define the desired data movement operation. The channel descriptors are in local memory in sequential order: channel 0, channel 1, ... channel 7. Each TCD*n* definition comprises 11 registers of 16 or 32 bits.

11.5.2 TCD initialization

Before activating a channel, you must initialize its TCD with the appropriate transfer profile.

11.5.3 TCD structure



11.5.4 Reserved memory and fields

- Reading reserved fields in a register returns the value of zero.
- The eDMA ignores writes to reserved bits in a register.
- Reading or writing a reserved memory location generates a bus error.

11.5.5 DMA register descriptions

11.5.5.1 DMA memory map

DMA base address: 4000_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	Table 11-10
4h	Error Status (ES)	32	R	0000_0000h
Ch	Enable Request (ERQ)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
14h	Enable Error Interrupt (EEI)	32	RW	0000_0000h
18h	Clear Enable Error Interrupt (CEEI)	8	W	00h
19h	Set Enable Error Interrupt (SEEI)	8	W	00h
1Ah	Clear Enable Request (CERQ)	8	W	00h
1Bh	Set Enable Request (SERQ)	8	W	00h
1Ch	Clear DONE Status Bit (CDNE)	8	W	00h
1Dh	Set START Bit (SSRT)	8	W	00h
1Eh	Clear Error (CERR)	8	W	00h
1Fh	Clear Interrupt Request (CINT)	8	W	00h
24h	Interrupt Request (INT)	32	RW	0000_0000h
2Ch	Error (ERR)	32	RW	0000_0000h
34h	Hardware Request Status (HRS)	32	R	0000_0000h
44h	Enable Asynchronous Request in Stop (EARS)	32	RW	0000_0000h
100h	Channel Priority (DCHPRI3)	8	RW	03h
101h	Channel Priority (DCHPRI2)	8	RW	02h
102h	Channel Priority (DCHPRI1)	8	RW	01h
103h	Channel Priority (DCHPRI0)	8	RW	00h
104h	Channel Priority (DCHPRI7)	8	RW	07h
105h	Channel Priority (DCHPRI6)	8	RW	06h
106h	Channel Priority (DCHPRI5)	8	RW	05h
107h	Channel Priority (DCHPRI4)	8	RW	04h
1000h - 10E0h	TCD Source Address (TCD0_SADDR - TCD7_SADDR)	32	RW	Table 11-10
1004h - 10E4h	TCD Signed Source Address Offset (TCD0_SOFF - TCD7_SOFF)	16	RW	Table 11-10
1006h - 10E6h	TCD Transfer Attributes (TCD0_ATTR - TCD7_ATTR)	16	RW	Table 11-10
1008h - 10E8h	TCD Minor Byte Count (Minor Loop Mapping Disabled) (TCD0_NBYTES_MLNO - TCD7_NBYTES_MLNO)	32	RW	Table 11-10
1008h - 10E8h	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (TCD0_NBYTES_MLOFFNO - TCD7_NBYTES_MLOFFNO)	32	RW	Table 11-10
1008h - 10E8h	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (TCD0_NBYTES_MLOFFYES - TCD7_NBYTES_MLOFFYES)	32	RW	Table 11-10
100Ch - 10ECh	TCD Last Source Address Adjustment (TCD0_SLAST - TCD7_SLAST)	32	RW	Table 11-10
1010h - 10F0h	TCD Destination Address (TCD0_DADDR - TCD7_DADDR)	32	RW	Table 11-10
1014h - 10F4h	TCD Signed Destination Address Offset (TCD0_DOFF - TCD7_DOFF)	16	RW	Table 11-10
1016h - 10F6h	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD7_CITER_ELINKNO)	16	RW	Table 11-10

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1016h - 10F6h	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD7_CITER_ELINKYES)	16	RW	Table 11-10
1018h - 10F8h	TCD Last Destination Address Adjustment/Scatter Gather Address (TCD0_DLASTSGA - TCD7_DLASTSGA)	32	RW	Table 11-10
101Ch - 10FCh	TCD Control and Status (TCD0_CSR - TCD7_CSR)	16	RW	Table 11-10
101Eh - 10FEh	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD7_BITER_ELINKNO)	16	RW	Table 11-10
101Eh - 10FEh	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD7_BITER_ELINKYES)	16	RW	Table 11-10

11.5.5.2 Control (CR)

11.5.5.2.1 Offset

Register	Offset
CR	0h

11.5.5.2.2 Function

This register defines the basic operating configuration of the eDMA module.

You can configure arbitration to use either a fixed-priority or a round-robin scheme. For fixed-priority arbitration, eDMA selects and executes the highest-priority channel that requests service. The channel priority registers assign the priorities (see the Channel Priority (DCHPRI0 - DCHPRI7) registers). For round-robin arbitration, the eDMA engine ignores channel priorities and cycles through channels from high to low channel number without regard to priority.

NOTE

For correct operation, you must write to this register only when the eDMA channels are inactive—that is, when TCD*n*_CSR[ACTIVE] = 0.

Minor loop offsets are address-offset values to be added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) when the minor loop completes. When you enable minor loop offsets, eDMA adds the minor loop offset (MLOFF) value to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both, before it writes the addresses back into the TCD. If the major loop is complete, eDMA ignores the minor loop offset, and uses the major loop address offsets (TCDn_SLAST and TCDn_DLAST_SGA) to compute the next TCDn_SADDR and TCDn_DADDR values.

Enabling minor loop mapping (EMLM = 1) redefines TCDn word2. eDMA uses a portion of TCDn word2 for multiple fields:

- A source enable field (SMLOE) to specify the minor loop offset is to be applied to the source address (TCDn_SADDR) when the minor loop completes
- A destination enable field (DMLOE) to specify the minor loop offset to be applied to the destination address (TCDn_DADDR) when the minor loop completes
- The sign extended minor loop offset value (MLOFF).

eDMA uses the same offset value (MLOFF) for both source and destination minor loop offsets. When you enable either minor loop offset (SMLOE = 1 or DMLOE = 1), the NBYTES field reduces in size to 10 bits. When you disable both minor loop offsets (SMLOE = 0 and and DMLOE = 0), the NBYTES field is a 30-bit vector.

When you disable minor loop mapping (EMLM = 0), the NBYTES field contains all 32 bits of TCDn word2.

11.5.5.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ACTIVE	Reserved							0						CX	ECX
W																
Reset	0	u	u	u	u	u	u	u	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								EMLM	CLM	HALT	HOE	Reserved	ERCA	EDBG	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.5.5.2.4 Fields

Field	Function
31 ACTIVE	eDMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel

Table continues on the next page...

Field	Function
30-24 —	Reserved
23-18 —	Reserved
17 CX	Cancel Transfer When you write 1 to this field, the following actions take place: • Stop the executing channel • Force the minor loop to finish. The cancellation takes effect after the last write of the current read/write sequence. This field is automatically written with 0 after the cancellation completes. The cancellation retires the channel normally as if the minor loop completed. 0b - Normal operation 1b - Cancel the remaining data transfer
16 ECX	Error Cancel Transfer When you write a 1 to this field, the following actions take place: • Stop the executing channel • Force the minor loop to finish. The cancellation takes effect after the last write of the current read/write sequence. This field is automatically reset to 0 after the cancellation completes. In addition to cancelling the transfer, eDMA: • Treats the cancel as an error condition • Updates the Error Status register (DMAx_ES) • Optionally generates an error interrupt. 0b - Normal operation 1b - Cancel the remaining data transfer
15-8 —	Reserved
7 EMLM	Enable Minor Loop Mapping When the value of this field is 0, TCDn.word2 is a 32-bit NBYTES field. When the value of this field is 1, TCDn.word2 includes: • Individual enable fields • An offset field • 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 reduces in size when either offset is enabled. 0b - Disabled 1b - Enabled
6 CLM	Continuous Link Mode When the value of this field is 0, a minor loop channel link made to itself goes through channel arbitration before being activated again. When the value of this field is 1, a minor loop channel link made to itself does not go through channel arbitration before being activated again. When the minor loop completes, 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. **NOTE:** Do not use continuous link mode with a channel linking to itself if there is only one minor loop iteration per service request, for example, 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. 0b - Continuous link mode is off

Table continues on the next page...

Field	Function
	1b - Continuous link mode is on
5 HALT	Halt eDMA Operations When this field is 1 the following actions take place: • eDMA stalls the start of any new channels • Executing channels are allowed to complete. When you write 0 to this field, channel execution resumes. 0b - Normal operation 1b - eDMA operations halted
4 HOE	Halt On Error When this field is 1, any error causes the eDMA engine to write 1 to the HALT field. Subsequently, the eDMA engine ignores all service requests until you write 0 to the HALT field. 0b - Normal operation 1b - Error causes HALT field to be automatically set to 1
3 —	Reserved
2 ERCA	Enable Round Robin Channel Arbitration When you write 1 to this field, eDMA uses round robin arbitration for channel selection. Otherwise, eDMA uses fixed priority arbitration for channel selection. 0b - Fixed priority arbitration 1b - Round robin arbitration
1 EDBG	Enable Debug When this field is 0 and the chip enters Debug mode, eDMA continues operation. When this field is 1, entry of the chip into Debug mode causes the eDMA to stall the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the chip exits Debug mode or you write 0 to this field. 0b - When the chip is in Debug mode, the eDMA continues to operate. 1b - When the chip is in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete.
0 —	Reserved

11.5.5.3 Error Status (ES)

11.5.5.3.1 Offset

Register	Offset
ES	4h

11.5.5.3.2 Function

The ES register provides information concerning the most-recently recorded channel error. Channel errors can be caused by:

- A configuration error, that is:
  - An illegal setting in the transfer-control descriptor
  - An illegal priority register setting in fixed arbitration
- An error termination to a bus master read or write cycle
- A cancel transfer with error field that is 1 when a transfer is canceled via the corresponding cancel transfer control field

See Fault reporting and handling for more details.

11.5.5.3.3 Diagram

Bits	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

Bits	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

11.5.5.3.4 Fields

Field	Function
31 VLD	Logical OR of all ERR status fields 0b - No ERR fields are 1 1b - At least one ERR field has a value of 1, indicating a valid error exists that has not been cleared
30-17 —	Reserved
16 ECX	Transfer Canceled 0b - No canceled transfers 1b - The most-recently recorded entry was a canceled transfer initiated by the error cancel transfer field
15 —	Reserved
14 CPE	Channel Priority Error 0b - No channel priority error. 1b - The most-recently recorded error was a configuration error in the channel priorities. Channel priorities are not unique.
13-11 —	Reserved
10-8 ERRCHN	Error Channel Number or Canceled Channel Number

Table continues on the next page...

Field	Function
	The channel number of the most-recently recorded error, excluding CPE errors or most-recently recorded error canceled transfer.
7 SAE	Source Address Error 0b - No source address configuration error. 1b - The most-recently 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 0b - No source offset configuration error. 1b - The most-recently 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 0b - No destination address configuration error. 1b - The most-recently 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 0b - No destination offset configuration error. 1b - The most-recently 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 0b - No NBYTES/CITER configuration error. 1b - The most-recently 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] = 0, or TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK].
2 SGE	Scatter/Gather Configuration Error When 1, this field indicates the most-recently recorded error was a configuration error detected in the TCDn_DLASTSGA field. eDMA checks This field 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. 0b - No scatter/gather configuration error. 1b - The most-recently recorded error was a configuration error detected in the TCDn_DLASTSGA field.
1 SBE	Source Bus Error 0b - No source bus error. 1b - The most-recently recorded error was a bus error on a source read.
0 DBE	Destination Bus Error 0b - No destination bus error. 1b - The most-recently recorded error was a bus error on a destination write.

11.5.5.4 Enable Request (ERQ)

11.5.5.4.1 Offset

Register	Offset
ERQ	Ch

11.5.5.4.2 Function

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 this register.

DMA request input signals and this enable request field must be set to 1 before a channel's hardware service request is accepted. The state of the DMA enable request field does not affect a channel service request made explicitly through software or a linked channel request.

NOTE

> Disable a channel's hardware service request at the source before writing 0 to the channel's ERQ field.

11.5.5.4.3 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								ERQ7	ERQ6	ERQ5	ERQ4	ERQ3	ERQ2	ERQ1	ERQ0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.5.5.4.4 Fields

Field	Function
31-8 —	Reserved
7 ERQ7	Enable DMA Request 7 0b - The DMA request signal for channel 7 is disabled 1b - The DMA request signal for channel 7 is enabled
6 ERQ6	Enable DMA Request 6 0b - The DMA request signal for channel 6 is disabled 1b - The DMA request signal for channel 6 is enabled
5 ERQ5	Enable DMA Request 5 0b - The DMA request signal for channel 5 is disabled 1b - The DMA request signal for channel 5 is enabled

Table continues on the next page...

Field	Function
4 ERQ4	Enable DMA Request 4 0b - The DMA request signal for channel 4 is disabled 1b - The DMA request signal for channel 4 is enabled
3 ERQ3	Enable DMA Request 3 0b - The DMA request signal for channel 3 is disabled 1b - The DMA request signal for channel 3 is enabled
2 ERQ2	Enable DMA Request 2 0b - The DMA request signal for channel 2 is disabled 1b - The DMA request signal for channel 2 is enabled
1 ERQ1	Enable DMA Request 1 0b - The DMA request signal for channel 1 is disabled 1b - The DMA request signal for channel 1 is enabled
0 ERQ0	Enable DMA Request 0 0b - The DMA request signal for channel 0 is disabled 1b - The DMA request signal for channel 0 is enabled

11.5.5.5 Enable Error Interrupt (EEI)

11.5.5.5.1 Offset

Register	Offset
EEI	14h

11.5.5.5.2 Function

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 registers. 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 field must be set to 1 before an error interrupt request for a given channel is sent to the interrupt controller.

11.5.5.5.3 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								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

11.5.5.5.4 Fields

Field	Function
31-8 —	Reserved
7 EEI7	Enable Error Interrupt 7 0b - An error on channel 7 does not generate an error interrupt 1b - An error on channel 7 generates an error interrupt request
6 EEI6	Enable Error Interrupt 6 0b - An error on channel 6 does not generate an error interrupt 1b - An error on channel 6 generates an error interrupt request
5 EEI5	Enable Error Interrupt 5 0b - An error on channel 5 does not generate an error interrupt 1b - An error on channel 5 generates an error interrupt request
4 EEI4	Enable Error Interrupt 4 0b - An error on channel 4 does not generate an error interrupt 1b - An error on channel 4 generates an error interrupt request
3 EEI3	Enable Error Interrupt 3 0b - An error on channel 3 does not generate an error interrupt 1b - An error on channel 3 generates an error interrupt request
2 EEI2	Enable Error Interrupt 2 0b - An error on channel 2 does not generate an error interrupt 1b - An error on channel 2 generates an error interrupt request
1 EEI1	Enable Error Interrupt 1 0b - An error on channel 1 does not generate an error interrupt 1b - An error on channel 1 generates an error interrupt request
0 EEI0	Enable Error Interrupt 0 0b - An error on channel 0 does not generate an error interrupt 1b - An error on channel 0 generates an error interrupt request

11.5.5.6 Clear Enable Error Interrupt (CEEI)

11.5.5.6.1 Offset

Register	Offset
CEEI	18h

11.5.5.6.2 Function

The CEEI provides a simple memory-mapped mechanism to write 0 to a given field in the EEI register to disable the error interrupt for a given channel. The data value on a register write causes the corresponding field in the EEI register to be written to 0. Writing 1 to the CAEE field provides a global clear to 0 function, forcing the EEI contents to be written to 0, disabling all DMA request inputs.

If the NOP field is written with 1, the command is ignored. This enables you to write 1 to a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word must all have their NOP field set to 1 so that these registers are not affected by the write.

Reads of this register return all zeroes.

11.5.5.6.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	CAEE	0			CEEI		
Reset	0	0	0	0	0	0	0	0

11.5.5.6.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation, ignore the other fields in this register
6 CAEE	Clear All Enable Error Interrupts 0b - Write 0 only to the EEI field specified in the CEEI field 1b - Write 0 to all fields in EEI
5-3 —	Reserved
2-0 CEEI	Clear Enable Error Interrupt Writes 0 to the corresponding field in EEI

11.5.5.7 Set Enable Error Interrupt (SEEI)

11.5.5.7.1 Offset

Register	Offset
SEEI	19h

11.5.5.7.2 Function

The SEEI register provides a simple memory-mapped mechanism to write 1 to a given field in the EEI register to enable the error interrupt for a given channel. The data value on a register write causes the corresponding field in the EEI to be written to 1. Writing 1 to the SAEE field provides a global set to 1 function, forcing the entire EEI register contents to be written with 1.

If the NOP field is 1, the command is ignored. This enables you to write 1 to a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word must all have their NOP field set to 1 so that these registers are not affected by the write.

Reads of this register return all zeroes.

11.5.5.7.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	SAEE	0			SEEI		
Reset	0	0	0	0	0	0	0	0

11.5.5.7.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation, ignore the other fields in this register
6	Set All Enable Error Interrupts 0b - Write 1 only to the EEI field specified in the SEEI field

Table continues on the next page...

Field	Function
SAEE	1b - Writes 1 to all fields in EEI
5-3 —	Reserved
2-0 SEEI	Set Enable Error Interrupt Writes 1 to the corresponding field in EEI

11.5.5.8 Clear Enable Request (CERQ)

11.5.5.8.1 Offset

Register	Offset
CERQ	1Ah

11.5.5.8.2 Function

The CERQ provides a simple memory-mapped mechanism to write 0 to a given field in the ERQ register to disable the DMA request for a given channel. The data value on a register write causes the corresponding field in the ERQ register to be written with 0. Setting the CAER field provides a global clear to 0 function, forcing the entire contents of the ERQ register to be written with 0, disabling all DMA request inputs.

If the NOP field is 1, the command is ignored. This enables you to write 1 to a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word must all have their NOP field written with 1 so that these registers are not affected by the write.

Reads of this register return all zeroes.

NOTE

Disable a channel's hardware service request at the source before writing 0 to the channel's ERQ field.

11.5.5.8.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	CAER	0			CERQ		
Reset	0	0	0	0	0	0	0	0

11.5.5.8.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation, ignore the other fields in this register
6 CAER	Clear All Enable Requests 0b - Write 0 to only the ERQ field specified in the CERQ field 1b - Write 0 to all fields in ERQ
5-3 —	Reserved
2-0 CERQ	Clear Enable Request Writes 0 to the corresponding field in ERQ.

11.5.5.9 Set Enable Request (SERQ)

11.5.5.9.1 Offset

Register	Offset
SERQ	1Bh

11.5.5.9.2 Function

The SERQ provides a simple memory-mapped mechanism to write 1 to a given field in the ERQ register to enable the DMA request for a given channel. The data value on a register write causes the corresponding field in the ERQ register to be set. Writing 1 to the SAER field provides a global set to 1 function, forcing the entire contents of ERQ register to be 1.

If the NOP field is 1, the command is ignored. This enables you to write 1 to a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word must all have their NOP field written with 1 so that these registers are not affected by the write.

Reads of this register returns all zeroes.

11.5.5.9.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	SAER	0			SERQ		
Reset	0	0	0	0	0	0	0	0

11.5.5.9.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation, ignore the other fields in this register
6 SAER	Set All Enable Requests 0b - Write 1 to only the ERQ field specified in the SERQ field 1b - Write 1 to all fields in ERQ
5-3 —	Reserved
2-0 SERQ	Set Enable Request Writes 1 to the corresponding field in ERQ.

11.5.5.10 Clear DONE Status Bit (CDNE)

11.5.5.10.1 Offset

Register	Offset
CDNE	1Ch

11.5.5.10.2 Function

The CDNE provides a simple memory-mapped mechanism to write 0 to the DONE field in the TCD of the given channel. The data value on a register write causes the DONE field in the corresponding TCD to be written with 0. Writing 1 to the CADN field provides a global clear function, forcing all DONE fields to be written with 0.

If the NOP field is 1, the command is ignored. This enables you to write 1 to a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word must all have their NOP field written with 1 so that these registers are not affected by the write.

Reads of this register return all zeroes.

11.5.5.10.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	CADN	0			CDNE		
Reset	0	0	0	0	0	0	0	0

11.5.5.10.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation; all other fields in this register are ignored.
6 CADN	Clears All DONE fields 0b - Writes 0 to only the TCDn_CSR[DONE] field specified in the CDNE field 1b - Writes 0 to all bits in TCDn_CSR[DONE]
5-3 —	Reserved
2-0 CDNE	Clear DONE field Writes 0 to the corresponding field in TCDn_CSR[DONE]

11.5.5.11 Set START Bit (SSRT)

11.5.5.11.1 Offset

Register	Offset
SSRT	1Dh

11.5.5.11.2 Function

The SSRT register provides a simple memory-mapped mechanism to write 1 to the START field in the TCD of the given channel. The data value on a register write causes the START field in the corresponding TCD to be written with 1. Writing 1 to the SAST field provides a global set to 1 function, forcing all START fields to be written with 1.

If the NOP field is 1, the command is ignored. This enables you to write 1 to a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word must all have their NOP field written with 1 so that these registers are not affected by the write.

Reads of this register return all zeroes.

11.5.5.11.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	SAST	0			SSRT		
Reset	0	0	0	0	0	0	0	0

11.5.5.11.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation; all other fields in this register are ignored.
6 SAST	Set All START fields (activates all channels) 0b - Write 1 to only the TCDn_CSR[START] field specified in the SSRT field 1b - Write 1 to all bits in TCDn_CSR[START]
5-3 —	Reserved
2-0 SSRT	Set START field Sets the corresponding field in TCDn_CSR[START]

11.5.5.12 Clear Error (CERR)

11.5.5.12.1 Offset

Register	Offset
CERR	1Eh

11.5.5.12.2 Function

The CERR provides a simple memory-mapped mechanism to write 0 to a given field in the ERR register to disable the error condition field for a given channel. The given value on a register write causes the corresponding field in the ERR register to be written with 0. Writing 1 to the CAEI field provides a global clear to 0 function, forcing the ERR register contents to be written with 0, clearing all channel error indicators. If the NOP field is 1, the command is ignored. This enables you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

11.5.5.12.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	CAEI	0			CERR		
Reset	0	0	0	0	0	0	0	0

11.5.5.12.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation; all other fields in this register are ignored.
6 CAEI	Clear All Error Indicators 0b - Write 0 to only the ERR field specified in the CERR field 1b - Write 0 to all fields in ERR
5-3 —	Reserved
2-0 CERR	Clear Error Indicator Writes 0 to the corresponding field in ERR

11.5.5.13 Clear Interrupt Request (CINT)

11.5.5.13.1 Offset

Register	Offset
CINT	1Fh

11.5.5.13.2 Function

The CINT register provides a simple, memory-mapped mechanism to clear a given field in the INT register to disable the interrupt request for a given channel. The given value on a register write causes the corresponding field in the INT register to be cleared. Setting the CAIR field provides a global clear function, forcing the entire contents of the INT to be cleared, disabling all DMA interrupt requests.

If the NOP field is 1, the command is ignored. This enables you to set a single, byte-wide register with a 32-bit write that does not affect the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP field set to 1 so that these registers are not affected by the write.

Reads of this register return all zeroes.

11.5.5.13.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0				0		
W	NOP	CAIR	0			CINT		
Reset	0	0	0	0	0	0	0	0

11.5.5.13.4 Fields

Field	Function
7 NOP	No Op Enable 0b - Normal operation 1b - No operation; all other fields in this register are ignored.
6	Clear All Interrupt Requests 0b - Clear only the INT field specified in the CINT field

Table continues on the next page...

Field	Function
CAIR	1b - Clear all bits in INT
5-3 —	Reserved
2-0 CINT	Clear Interrupt Request Clears the corresponding field in INT

11.5.5.14 Interrupt Request (INT)

11.5.5.14.1 Offset

Register	Offset
INT	24h

11.5.5.14.2 Function

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 write 0 to 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 0 in any bit position has no effect 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.

11.5.5.14.3 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								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

11.5.5.14.4 Fields

Field	Function
31-8 —	Reserved
7 INT7	Interrupt Request 7 0b - The interrupt request for channel 7 is cleared 1b - The interrupt request for channel 7 is active
6 INT6	Interrupt Request 6 0b - The interrupt request for channel 6 is cleared 1b - The interrupt request for channel 6 is active
5 INT5	Interrupt Request 5 0b - The interrupt request for channel 5 is cleared 1b - The interrupt request for channel 5 is active
4 INT4	Interrupt Request 4 0b - The interrupt request for channel 4 is cleared 1b - The interrupt request for channel 4 is active
3 INT3	Interrupt Request 3 0b - The interrupt request for channel 3 is cleared 1b - The interrupt request for channel 3 is active
2 INT2	Interrupt Request 2 0b - The interrupt request for channel 2 is cleared 1b - The interrupt request for channel 2 is active
1 INT1	Interrupt Request 1 0b - The interrupt request for channel 1 is cleared 1b - The interrupt request for channel 1 is active
0 INT0	Interrupt Request 0 0b - The interrupt request for channel 0 is cleared 1b - The interrupt request for channel 0 is active

11.5.5.15 Error (ERR)

11.5.5.15.1 Offset

Register	Offset
ERR	2Ch

11.5.5.15.2 Function

The ERR register provides a bit map for the 8 channels, signaling the presence of an error for each channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate field in this register. The outputs of this register are enabled by the contents of the EEI register, 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 reset the appropriate bit to 0, 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 TCD DONE field to 1 and the possible generation 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 fields. 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 1 in any bit position clears the corresponding channel's error status. A 0 in any bit position has no effect on the corresponding channel's current error status. The CERR is provided so the error indicator for a single channel can easily be reset to 0.

11.5.5.15.3 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								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

11.5.5.15.4 Fields

Field	Function
31-8 —	Reserved
7 ERR7	Error In Channel 7 0b - No error in this channel has occurred 1b - An error in this channel has occurred
6 ERR6	Error In Channel 6 0b - No error in this channel has occurred 1b - An error in this channel has occurred
5 ERR5	Error In Channel 5 0b - No error in this channel has occurred 1b - An error in this channel has occurred
4 ERR4	Error In Channel 4 0b - No error in this channel has occurred 1b - An error in this channel has occurred
3 ERR3	Error In Channel 3 0b - No error in this channel has occurred 1b - An error in this channel has occurred
2 ERR2	Error In Channel 2 0b - No error in this channel has occurred 1b - An error in this channel has occurred
1 ERR1	Error In Channel 1 0b - No error in this channel has occurred 1b - An error in this channel has occurred
0 ERR0	Error In Channel 0 0b - No error in this channel has occurred 1b - An error in this channel has occurred

11.5.5.16 Hardware Request Status (HRS)

11.5.5.16.1 Offset

Register	Offset
HRS	34h

11.5.5.16.2 Function

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.

Each HRS field for its respective channel is 1 when a hardware request is present on the channel. After the request is completed and channel is free, the HRS field is automatically changed to 0 by hardware.

11.5.5.16.3 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								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

11.5.5.16.4 Fields

Field	Function
31-8 —	Reserved
7 HRS7	Hardware Request Status Channel 7 0b - A hardware service request for channel 7 is not present 1b - A hardware service request for channel 7 is present
6 HRS6	Hardware Request Status Channel 6 0b - A hardware service request for channel 6 is not present 1b - A hardware service request for channel 6 is present
5	Hardware Request Status Channel 5 0b - A hardware service request for channel 5 is not present

Table continues on the next page...

Field	Function
HRS5	1b - A hardware service request for channel 5 is present
4 HRS4	Hardware Request Status Channel 4 0b - A hardware service request for channel 4 is not present 1b - A hardware service request for channel 4 is present
3 HRS3	Hardware Request Status Channel 3 0b - A hardware service request for channel 3 is not present 1b - A hardware service request for channel 3 is present
2 HRS2	Hardware Request Status Channel 2 0b - A hardware service request for channel 2 is not present 1b - A hardware service request for channel 2 is present
1 HRS1	Hardware Request Status Channel 1 0b - A hardware service request for channel 1 is not present 1b - A hardware service request for channel 1 is present
0 HRS0	Hardware Request Status Channel 0 0b - A hardware service request for channel 0 is not present 1b - A hardware service request for channel 0 is present

11.5.5.17 Enable Asynchronous Request in Stop (EARS)

11.5.5.17.1 Offset

Register	Offset
EARS	44h

11.5.5.17.2 Function

The EARS register is used to enable or disable the DMA requests in Enable Request (ERQ) by AND'ing the bits of these two registers.

11.5.5.17.3 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								EDREQ_7	EDREQ_6	EDREQ_5	EDREQ_4	EDREQ_3	EDREQ_2	EDREQ_1	EDREQ_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.5.5.17.4 Fields

Field	Function
31-8 —	Reserved
7 EDREQ_7	Enable asynchronous DMA request in stop mode for channel 7. 0b - Disable asynchronous DMA request for channel 7 1b - Enable asynchronous DMA request for channel 7
6 EDREQ_6	Enable asynchronous DMA request in stop mode for channel 6. 0b - Disable asynchronous DMA request for channel 6 1b - Enable asynchronous DMA request for channel 6
5 EDREQ_5	Enable asynchronous DMA request in stop mode for channel 5. 0b - Disable asynchronous DMA request for channel 5 1b - Enable asynchronous DMA request for channel 5
4 EDREQ_4	Enable asynchronous DMA request in stop mode for channel 4. 0b - Disable asynchronous DMA request for channel 4 1b - Enable asynchronous DMA request for channel 4
3 EDREQ_3	Enable asynchronous DMA request in stop mode for channel 3. 0b - Disable asynchronous DMA request for channel 3 1b - Enable asynchronous DMA request for channel 3
2 EDREQ_2	Enable asynchronous DMA request in stop mode for channel 2. 0b - Disable asynchronous DMA request for channel 2 1b - Enable asynchronous DMA request for channel 2
1 EDREQ_1	Enable asynchronous DMA request in stop mode for channel 1. 0b - Disable asynchronous DMA request for channel 1 1b - Enable asynchronous DMA request for channel 1
0 EDREQ_0	Enable asynchronous DMA request in stop mode for channel 0. 0b - Disable asynchronous DMA request for channel 0 1b - Enable asynchronous DMA request for channel 0

11.5.5.18 Channel Priority (DCHPRI0 - DCHPRI7)

11.5.5.18.1 Offset

For n = 0 to 7:

Register	Offset
DCHPRIn	100h + (n + 3 - 2 × (n mod 4))

11.5.5.18.2 Function

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, and so on. 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.

11.5.5.18.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	ECP	DPA	0			CHPRI		
W								
Reset	See #d997e5551a1310.							

11.5.5.18.4 Register reset values

Register	Reset value
DCHPRI0	00h
DCHPRI1	01h
DCHPRI2	02h
DCHPRI3	03h
DCHPRI4	04h
DCHPRI5	05h
DCHPRI6	06h
DCHPRI7	07h

11.5.5.18.5 Fields

Field	Function
7 ECP	Enable Channel Preemption. This field resets to 0. 0b - Channel n cannot be suspended by a higher priority channel's service request 1b - Channel n can be temporarily suspended by the service request of a higher priority channel
6 DPA	Disable Preempt Ability. This field resets to 0. 0b - Channel n can suspend a lower priority channel 1b - Channel n cannot suspend any channel, regardless of channel priority
5-3 —	Reserved

Table continues on the next page...

Field	Function
2-0 CHPRI	Channel n Arbitration Priority Channel priority when fixed-priority arbitration is enabled.

11.5.5.19 TCD Source Address (TCD0_SADDR - TCD7_SADDR)

11.5.5.19.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_SADDR	1000h + (n × 20h)

11.5.5.19.2 Function

This register contains the source address of the transfer.

11.5.5.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.19.4 Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

11.5.5.20 TCD Signed Source Address Offset (TCD0_SOFF - TCD7_SOFF)

11.5.5.20.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_SOFF	1004h + (n × 20h)

11.5.5.20.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SOFF															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.20.3 Fields

Field	Function
15-0 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.

11.5.5.21 TCD Transfer Attributes (TCD0_ATTR - TCD7_ATTR)

11.5.5.21.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_ATTR	1006h + (n × 20h)

11.5.5.21.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SMOD					SSIZE			DMOD					DSIZE		
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.21.3 Fields

Field	Function
15-11 SMOD	Source Address Modulo Any non-zero value in this field 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. 0_0000b - Source address modulo feature is disabled 0_0001b-1_1111b - Value defines address range used to set up circular data queue
10-8 SSIZE	Source data transfer size **NOTE:** 1. Using a reserved value causes a configuration error. 2. The eDMA defaults to privileged data access for all transactions. 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - Reserved 100b - 16-byte 101b - 32-byte 110b - Reserved 111b - Reserved
7-3 DMOD	Destination Address Modulo See the SMOD definition.
2-0 DSIZE	Destination data transfer size See the SSIZE definition.

11.5.5.22 TCD Minor Byte Count (Minor Loop Mapping Disabled) (TCD0_NBYTES_MLNO - TCD7_NBYTES_MLNO)

11.5.5.22.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_NBYTES_MLNO	1008h + (n × 20h)

11.5.5.22.2 Function

This register, or one of the next two registers (TCD_NBYTES_MLOFFNO, TCD_NBYTES_MLOFFYES), that defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, is enabled but not used for this channel, or is 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.

11.5.5.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NBYTES															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NBYTES															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.22.4 Fields

Field	Function
31-0 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 are performed 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, and 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.

11.5.5.23 TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (TCD0_NBYTES_MLOFFNO - TCD7_NBYTES_MLOFFNO)

11.5.5.23.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_NBYTES_MLOFFNO	1008h + (n × 20h)

11.5.5.23.2 Function

One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFYES), that defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, is enabled but not used for this channel, or is 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 = 1 or DMLOE = 1, refer to the TCD_NBYTES_MLOFFYES register description. If minor loop mapping is disabled, refer to the TCD_NBYTES_MLNO register description.

11.5.5.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SMLOE	DMLOE	NBYTES													
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NBYTES															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.23.4 Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Specifies whether the minor loop offset is applied to the source address when the minor loop completes. 0b - The minor loop offset is not applied to the SADDR 1b - The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset Enable Specifies whether the minor loop offset is applied to the destination address when the minor loop completes. 0b - The minor loop offset is not applied to the DADDR 1b - The minor loop offset is applied to the DADDR
29-0 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 are performed 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, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

11.5.5.24 TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (TCD0_NBYTES_MLOFFYES - TCD7_NBYTES_MLOFFYES)

11.5.5.24.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_NBYTES_MLOFFYES	1008h + (n × 20h)

11.5.5.24.2 Function

One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFNO), that defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, is enabled but not used for this channel, or is 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 = 0 and DMLOE = 0, refer to the TCD_NBYTES_MLOFFNO register description. If minor loop mapping is disabled, refer to the TCD_NBYTES_MLNO register description.

11.5.5.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SMLOE	DMLOE	MLOFF													
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MLOFF						NBYTES									
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.24.4 Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Specifies whether the minor loop offset is applied to the source address when the minor loop completes. 0b - The minor loop offset is not applied to the SADDR 1b - The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset Enable Specifies whether the minor loop offset is applied to the destination address when the minor loop completes. 0b - The minor loop offset is not applied to the DADDR 1b - The minor loop offset is applied to the DADDR
29-10 MLOFF	If SMLOE = 1 or DMLOE = 1, 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.
9-0 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 are performed 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, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

11.5.5.25 TCD Last Source Address Adjustment (TCD0_SLAST - TCD7_SLAST)

11.5.5.25.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_SLAST	100Ch + (n × 20h)

11.5.5.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SLAST															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLAST															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.25.3 Fields

Field	Function
31-0 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.

11.5.5.26 TCD Destination Address (TCD0_DADDR - TCD7_DADDR)

11.5.5.26.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_DADDR	1010h + (n × 20h)

11.5.5.26.2 Function

This register contains the destination address of the transfer.

11.5.5.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.26.4 Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

11.5.5.27 TCD Signed Destination Address Offset (TCD0_DOFF - TCD7_DOFF)

11.5.5.27.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_DOFF	1014h + (n × 20h)

11.5.5.27.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DOFF															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.27.3 Fields

Field	Function
15-0 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.

11.5.5.28 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD7_CITER_ELINKNO)

11.5.5.28.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_CITER_ELINKNO	1016h + (n × 20h)

11.5.5.28.2 Function

This register contains the minor-loop channel-linking configuration and the channel's current iteration count. It is the same register as TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD7_CITER_ELINKYES), but its fields are defined differently based on the state of the ELINK field. If the ELINK field is 0, this register is defined as follows.

11.5.5.28.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ELINK	CITER														
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.28.4 Fields

Field	Function
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this field 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 TCDn_CSR[START] 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 field must be equal to BITER[ELINK]; otherwise, a configuration error is reported. 0b - Channel-to-channel linking is disabled 1b - Channel-to-channel linking is enabled
14-0 CITER	Current Major Iteration Count This field is 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. It optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** 1. When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. 2. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

11.5.5.29 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD7_CITER_ELINKYES)

11.5.5.29.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_CITER_ELINKYES	1016h + (n × 20h)

11.5.5.29.2 Function

This register contains the minor-loop channel-linking configuration and the channel's current iteration count. It is the same register as TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD7_CITER_ELINKNO), but its fields are defined differently based on the state of the ELINK field. If the ELINK field is 1, this register is defined as follows.

11.5.5.29.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ELINK				LINKCH			CITER								
W		0														
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.29.4 Fields

Field	Function
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this field 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 TCDn_CSR[START] 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 field must be equal to BITER[ELINK]; otherwise, a configuration error is reported. 0b - Channel-to-channel linking is disabled 1b - Channel-to-channel linking is enabled
14-12 —	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].
8-0 CITER	Current Major Iteration Count This field is 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. It optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** 1. When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field.

Field	Function
	2. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

11.5.5.30 TCD Last Destination Address Adjustment/Scatter Gather Address (TCD0_DLASTSGA - TCD7_DLASTSGA)

11.5.5.30.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_DLASTSGA	1018h + (n × 20h)

11.5.5.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DLASTSGA															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DLASTSGA															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.30.3 Fields

Field	Function
31-0 DLASTSGA	Destination last address adjustment, or next memory address TCD for channel (scatter/gather) If (TCDn_CSR[ESG] = 0) then: • This is the 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 TCD 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.

11.5.5.31 TCD Control and Status (TCD0_CSR - TCD7_CSR)

11.5.5.31.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_CSR	101Ch + (n × 20h)

11.5.5.31.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BWC					MAJORLINKCH			DONE	ACTIVE	MAJORELINK	ESG	DREQ	INTHALF	INTMAJOR	START
W			0													
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.31.3 Fields

Field	Function
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. **NOTE:** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. **NOTE:** When executing a large, zero wait-stated memory-to-memory transfer, insert bandwidth control using the TCD_CSR[BWC] bits to avoid: • Starvation of another master accessing the memory. • Any delay in writing a TCD during the transfer. 00b - No eDMA engine stalls 01b - Reserved 10b - eDMA engine stalls for 4 cycles after each R/W 11b - eDMA engine stalls for 8 cycles after each R/W
13-11 —	Reserved

Table continues on the next page...

Field	Function
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 START bit.
7 DONE	Channel Done This field indicates whether the eDMA has completed the major loop. The eDMA engine sets the value of this field to 1 when the CITER count reaches zero. The value of this field is reset to 0 by the hardware (when the channel is activated) or by software. **NOTE:** This field must be 0 to write the MAJORELINK or ESG fields.
6 ACTIVE	Channel Active This field indicates whether the channel is currently in execution. The eDMA sets the value of this field to 1 when channel service begins, and resets it to 0 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 field controls linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets TCDn_CSR[START] of the specified channel. **NOTE:** To support the dynamic linking coherency model, this field is forced to zero when written to when TCDn_CSR[DONE] is set. 0b - Channel-to-channel linking is disabled 1b - Channel-to-channel linking is enabled
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this field controls scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo 32-bit address containing a 32-byte data structure loaded as the TCD into local memory. **NOTE:** To support the dynamic scatter/gather coherency model, this field is forced to zero when written to when TCDn_CSR[DONE] is set. 0b - The current channel's TCD is normal format 1b - The current channel's TCD specifies a scatter gather format
3 DREQ	Disable Request If the value of this field is 1, eDMA hardware automatically writes 0 to the corresponding ERQ field when the current major iteration count reaches zero. 0b - The channel's ERQ field is not affected 1b - The channel's ERQ field value changes to 0 when the major loop is complete
2 INTHALF	Enable an interrupt when major counter is half complete. If the value of this field is 1, the channel generates an interrupt request by setting the appropriate field 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. 0b - Half-point interrupt is disabled 1b - Half-point interrupt is enabled
1	Enable an interrupt when major iteration count completes.

Table continues on the next page...

Field	Function
INTMAJOR	If the value of this field is 1, the channel generates an interrupt request by setting the appropriate field in the INT when the current major iteration count reaches zero. 0b - End of major loop interrupt is disabled 1b - End of major loop interrupt is enabled
0 START	Channel Start If the value of this field is 1, the channel is requesting service. eDMA hardware automatically writes 0 to this field after the channel begins execution. 0b - Channel is not explicitly started 1b - Channel is explicitly started via a software initiated service request

11.5.5.32 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD7_BITER_ELINKNO)

11.5.5.32.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_BITER_ELINKNO	101Eh + (n × 20h)

11.5.5.32.2 Function

If TCDn_BITER[ELINK] is 0, the TCDn_BITER register is defined as follows.

11.5.5.32.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ELINK	BITER														
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.32.4 Fields

Field	Function
15 ELINK	Enables channel-to-channel linking on minor loop complete

Table continues on the next page...

Field	Function
	As the channel completes the minor loop, this field enables linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets TCDn_CSR[START] 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. 0b - Channel-to-channel linking is disabled 1b - Channel-to-channel linking is enabled
14-0 BITER	Starting Major Iteration Count As the TCD 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. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field. 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.

11.5.5.33 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD7_BITER_ELINKYES)

11.5.5.33.1 Offset

For n = 0 to 7:

Register	Offset
TCDn_BITER_ELINKYES	101Eh + (n × 20h)

11.5.5.33.2 Function

If TCDn_BITER[ELINK] is 1, the TCDn_BITER register is defined as follows.

11.5.5.33.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ELINK				LINKCH			BITER								
W		0														
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

11.5.5.33.4 Fields

Field	Function
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this field enables linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets TCDn_CSR[START] 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. 0b - Channel-to-channel linking is disabled 1b - Channel-to-channel linking is enabled
14-12 —	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]. **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.
8-0 BITER	Starting major iteration count As the TCD 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. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field. 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.

11.6 Usage Guide

NOTE

User should configure DMA_TCD*n*_CSR[BWC] (bit 15-14) as 10 when another DMA channel is active.

Related application notes on this DMA module are as follows.

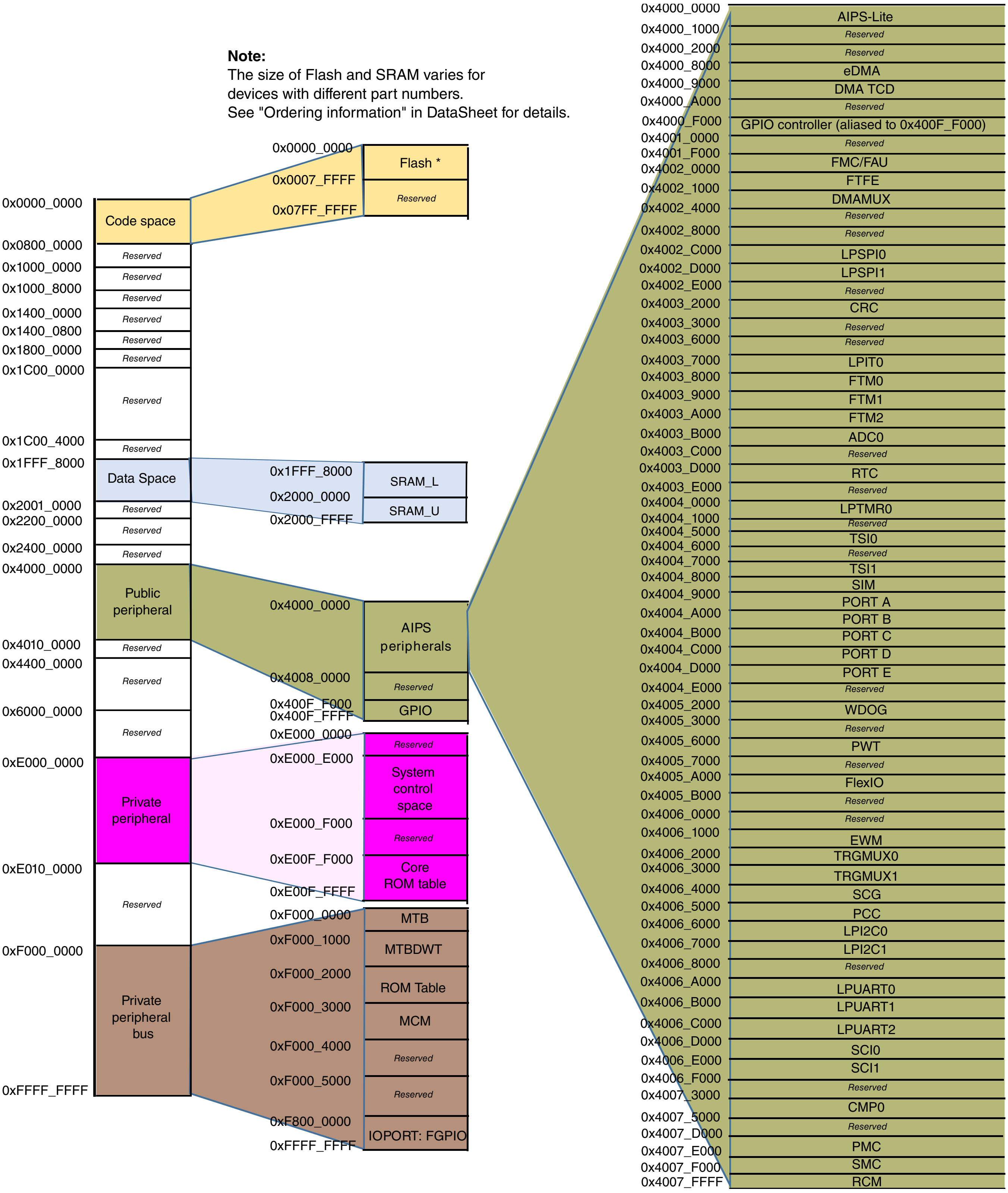
- Using DMA for pulse counting on Kinetis
- Using DMA and GPIO to emulate timer functionality on Kinetis Family devices
- Using DMA to Emulate ADC Flexible Scan Mode on Kinetis K Series

Chapter 12
Memory and memory map

12.1 Introduction

This device contains various memories and memory-mapped peripherals which are located in one 4 GB (32-bit address) contiguous memory space. This chapter describes the memory and peripheral locations within that memory space.

The following figure shows the system memory and peripheral locations.



Note:
The size of Flash and SRAM varies for devices with different part numbers.
See "Ordering information" in DataSheet for details.

Figure 12-1. Memory map

12.2 Flash memory

12.2.1 Flash memory types

This device contains the following types of flash memory:

- Program flash memory — non-volatile flash memory that can execute program code

12.2.2 Flash memory sizes

The devices covered in this document contain:

- 2 blocks (256 KB each) of program flash consisting of 2 KB sectors

The amounts of flash memory and the address range for the devices is shown in following table.

Device	Program flash (KB)	Address range
KE17Z512 KE13Z512 KE12Z512	512	0x0000_0000–0x0007_FFFF (P-Flash)

12.3 SRAM memory

12.3.1 SRAM sizes

This device contains SRAM accessed by bus masters through the cross-bar switch. The on-chip SRAM is split into SRAM_L and SRAM_U regions where the SRAM_L and SRAM_U ranges form a contiguous block in the memory map anchored at address 0x2000_0000. As such:

- SRAM_L is anchored to 0x1FFF_FFFF and occupies the space before this ending address.
- SRAM_U is anchored to 0x2000_0000 and occupies the space after this beginning address.

NOTE

Burst-access cannot occur across the 0x2000_0000 boundary that separates the two SRAM arrays. The two arrays should be treated as separate memory ranges for burst accesses.

The amount of SRAM for the devices covered in this document is shown in the following table.

Device	SRAM_L size (KB)	SRAM_U size (KB)	Total SRAM (KB)	Address Range
KE1xZ512 (x=7/3/2)	32	64	96	0x1FFF_8000-0x2000_FFFF

12.3.2 SRAM retention in low power modes

The SRAM is retained power on to all power modes on this device.

12.4 System memory map

The following table shows the high-level device memory map. This map provides the complete architectural address space definition for the various sections. Based on the physical sizes of the memories and peripherals, the actual address regions used may be smaller.

Table 12-1. System memory map

System 32-bit Address Range	Destination Slave	Access
0x0000_0000–0x07FF_FFFF [1]	Program flash and read-only data (Includes exception vectors in first 1024 bytes)	All masters
0x0800_0000–0x0FFF_FFFF	Reserved	–
0x1000_0000–0x13FF_FFFF	Reserved	Reserved
0x1400_0000–0x17FF_FFFF	Reserved	Reserved
0x1800_0000–0x1BFF_FFFF	Reserved	–
0x1C00_0000–0x1C00_3FFF	Reserved	Reserved
0x1C00_4000–0x1FEF_FFFF	Reserved	–
0x1FF0_0000–0x1FFF_FFFF [2]	SRAM_L: Lower SRAM	All masters
0x2000_0000–0x200F_FFFF [2]	SRAM_U: Upper SRAM	All masters
0x2010_0000–0x201F_FFFF	Reserved	–
0x2020_0000–0x21FF_FFFF	Reserved	–
0x2200_0000–0x23FF_FFFF	Reserved	–

Table continues on the next page...

Table 12-1. System memory map (continued)

System 32-bit Address Range	Destination Slave	Access
0x2400_0000-0x2FFF_FFFF	Reserved	–
0x3000_0000-0x33FF_FFFF	Reserved	–
0x3400_0000-0x3FFF_FFFF	Reserved	–
0x4000_0000–0x4007_FFFF	AIPS Peripherals	Cortex-M0+ core & DMA
0x4008_0000–0x400F_EFFF	Reserved	–
0x400F_F000–0x400F_FFFF	General purpose input/output (GPIO)	Cortex-M0+ core & DMA
0x4010_0000–0x41FF_FFFF	Reserved	–
0x4200_0000–0x43FF_FFFF	Reserved	–
0x4400_0000–0x5FFF_FFFF	Reserved	Reserved
0x6000_0000–0xDFFF_FFFF	Reserved	–
0xE000_0000–0xE00F_FFFF	Private peripherals	Cortex-M0+ core only
0xE010_0000–0xEFFF_FFFF	Reserved	–
0xF000_0000–0xF000_0FFF	Micro Trace Buffer (MTB) registers	Cortex-M0+ core only
0xF000_1000–0xF000_1FFF	MTB Data Watchpoint and Trace (MTBDWT) registers	Cortex-M0+ core only
0xF000_2000–0xF000_2FFF	ROM table	Cortex-M0+ core only
0xF000_3000–0xF000_3FFF	Miscellaneous Control Module (MCM)	Cortex-M0+ core only
0xF000_4000–0xF000_4FFF	Reserved	Reserved
0xF000_5000–0xF7FF_FFFF	Reserved	–
0xF800_0000–0xFFFF_FFFF	IOPORT: FGPIO (single cycle)	Cortex-M0+ core only

1. This map provides the complete architectural address space definition for the flash. Based on the physical sizes of the memories implemented for a particular device, the actual address regions used may be smaller. See "Flash memory sizes" for details.
2. This range varies depending on amount of SRAM implemented for a particular device. See SRAM sizes for details.

NOTE

1. Access rights to AIPS-Lite peripheral bridge and general purpose input/output (GPIO) module address space is limited to the core, DMA.

12.5 Peripheral memory map

The peripheral memory map is accessible via a crossbar slave port and the AIPS peripheral bridge. The peripheral bridge converts register access from AHB bus domain to peripheral bus domain.

For peripherals that have clock gating control bits (CGC bit) in PCC module, the associated peripherals could be enabled/disabled by these control bits. Access to a disabled peripheral or unimplemented AIPS slot results in 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 results in a transfer error termination.

12.5.1 Peripheral Bridge (AIPS-Lite) Memory Map

Table 12-2. Peripheral bridge slot assignments

System 32-bit base address	Slot number	Module
0x4000_0000	0	—
0x4000_1000	1	
0x4000_2000	2	—
0x4000_3000	3	—
0x4000_4000	4	—
0x4000_5000	5	—
0x4000_6000	6	—
0x4000_7000	7	—
0x4000_8000	8	DMA controller
0x4000_9000	9	DMA controller transfer control descriptors
0x4000_A000	10	—
0x4000_B000	11	—
0x4000_C000	12	—
0x4000_D000	13	—
0x4000_E000	14	—
0x4000_F000	15	RGPIO controller (aliased to 0x400F_F000)
0x4001_0000	16	—
0x4001_1000	17	—
0x4001_2000	18	—
0x4001_3000	19	—
0x4001_4000	20	—

Table continues on the next page...

Table 12-2. Peripheral bridge slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4001_5000	21	—
0x4001_6000	22	—
0x4001_7000	23	—
0x4001_8000	24	—
0x4001_9000	25	—
0x4001_A000	26	—
0x4001_B000	27	—
0x4001_C000	28	—
0x4001_D000	29	—
0x4001_E000	30	—
0x4001_F000	31	—
0x4002_0000	32	Flash memory
0x4002_1000	33	DMA channel mutiplexer 0
0x4002_2000	34	—
0x4002_3000	35	—
0x4002_4000	36	—
0x4002_5000	37	—
0x4002_6000	38	—
0x4002_7000	39	—
0x4002_8000	40	—
0x4002_9000	41	—
0x4002_A000	42	—
0x4002_B000	43	—
0x4002_C000	44	Low Power SPI (LPSPI) 0
0x4002_D000	45	Low Power SPI (LPSPI) 1
0x4002_E000	46	—
0x4002_F000	47	—
0x4003_0000	48	—
0x4003_1000	49	
0x4003_2000	50	CRC
0x4003_3000	51	—
0x4003_4000	52	—
0x4003_5000	53	—
0x4003_6000	54	—
0x4003_7000	55	Low-power Periodic interrupt timer (LPIT0)
0x4003_8000	56	FlexTimer (FTM) 0
0x4003_9000	57	FlexTimer (FTM) 1
0x4003_A000	58	FlexTimer (FTM) 2
0x4003_B000	59	Analog-to-digital converter (ADC) 0

Table continues on the next page...

Table 12-2. Peripheral bridge slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4003_C000	60	—
0x4003_D000	61	Real-time clock (RTC)
0x4003_E000	62	—
0x4003_F000	63	—
0x4004_0000	64	Low-power timer (LPTMR0)
0x4004_1000	65	—
0x4004_2000	66	—
0x4004_3000	67	—
0x4004_4000	68	—
0x4004_5000	69	Touch sense interface (TSI0)
0x4004_6000	70	—
0x4004_7000	71	Touch sense interface (TSI1)
0x4004_8000	72	System integration module (SIM)
0x4004_9000	73	Port A multiplexing control
0x4004_A000	74	Port B multiplexing control
0x4004_B000	75	Port C multiplexing control
0x4004_C000	76	Port D multiplexing control
0x4004_D000	77	Port E multiplexing control
0x4004_E000	78	—
0x4004_F000	79	—
0x4005_0000	80	—
0x4005_1000	81	—
0x4005_2000	82	Software watchdog (WDOG)
0x4005_3000	83	—
0x4005_4000	84	—
0x4005_5000	85	—
0x4005_6000	86	Pulse Width Timer (PWT)
0x4005_7000	87	—
0x4005_8000	88	—
0x4005_9000	89	—
0x4005_A000	90	Flexible IO (FlexIO)
0x4005_B000	91	—
0x4005_C000	92	—
0x4005_D000	93	—
0x4005_E000	94	—
0x4005_F000	95	—
0x4006_0000	96	—
0x4006_1000	97	External watchdog (EWM)
0x4006_2000	98	Trigger Multiplexing Control (TRGMUX 0)

Table continues on the next page...

Table 12-2. Peripheral bridge slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4006_3000	99	Trigger Multiplexing Control (TRGMUX 1)
0x4006_4000	100	System Clock Generator (SCG)
0x4006_5000	101	Peripheral Clock Control (PCC)
0x4006_6000	102	Low Power I^{2}C (LPI^{2}C 0)
0x4006_7000	103	Low Power I^{2}C (LPI^{2}C 1)
0x4006_8000	104	—
0x4006_9000	105	—
0x4006_A000	106	Low Power UART (LPUART 0)
0x4006_B000	107	Low Power UART (LPUART 1)
0x4006_C000	108	Low Power UART (LPUART 2)
0x4006_D000	109	SCI 0
0x4006_E000	110	SCI 1
0x4006_F000	111	—
0x4007_0000	112	—
0x4007_1000	113	—
0x4007_2000	114	—
0x4007_3000	115	Analog comparator (CMP 0)
0x4007_4000	116	—
0x4007_5000	117	—
0x4007_6000	118	—
0x4007_7000	119	—
0x4007_8000	120	—
0x4007_9000	121	—
0x4007_A000	122	—
0x4007_B000	123	—
0x4007_C000	124	—
0x4007_D000	125	Power management controller (PMC)
0x4007_E000	126	System Mode controller (SMC)
0x4007_F000	127	Reset Control Module (RCM)
0x400F_F000		GPIO controller

12.6 Private Peripheral Bus (PPB) memory map

The PPB is part of the defined ARM bus architecture and provides access to select processor-local modules. These resources are only accessible from the core; other system masters do not have access to them.

Table 12-3. PPB memory map

System 32-bit Address Range	Resource	Additional Range Detail	Resource
0xE000_0000–0xE000_DFFF	Reserved		
0xE000_E000–0xE000_EFFF	System Control Space (SCS)	0xE000_E000–0xE000_E00F	Reserved
		0xE000_E010–0xE000_E0FF	SysTick
		0xE000_E100–0xE000_ECFF	NVIC
		0xE000_ED00–0xE000_ED8F	System Control Block
		0xE000_ED90–0xE000_EDEF	Reserved
		0xE000_EDF0–0xE000_EEFF	Debug
		0xE000_EF00–0xE000_EFFF	Reserved
0xE000_F000–0xE00F_EFFF	Reserved		
0xE00F_F000–0xE00F_FFFF	Core ROM Space (CRS)		

Chapter 13
Flash Memory Controller (FMC) / Flash Acceleration Unit (FAU)

13.1 Introduction

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.

13.1.1 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.

13.1.2 Features

The FMC's features include:

- Interface between the device and the flash memory:
  - 8-bit, 16-bit, and 32-bit read operations to program flash memory.
  - For bank 0 and bank 1: Read accesses to consecutive 32-bit spaces in memory return the second read data with no wait states. The memory returns 64 bits via the 32-bit bus access.
- For bank 0 and bank 1: Acceleration of data transfer from program flash memory to the device:

- 64-bit prefetch speculation buffer with controls for instruction/data access
- 4-way, 8-set, 64-bit line size cache for a total of thirty-two 64-bit entries
- Single-entry buffer per bank

13.2 Modes of operation

The FMC only operates when a bus master accesses the flash memory.

For any device power mode where the flash memory cannot be accessed, the FMC is disabled.

13.3 External signal description

The FMC has no external signals.

13.4 Functional description

The FMC is a flash acceleration unit with flexible buffers for user configuration. Besides managing the interface between the device and the flash memory, the FMC can be used to restrict access from crossbar switch masters and —for program flash only—to customize the cache and buffers to provide single-cycle system-clock data-access times. Whenever a hit occurs for the prefetch speculation buffer, the cache, or the single-entry buffer, the requested data is transferred within a single system clock.

13.4.1 Default configuration

Upon system reset, the FMC is configured to provide a significant level of buffering for transfers from the flash memory:

- Crossbar masters 0, 1, 2, 3 have read access to bank 0 and bank 1.
- For bank 0 and bank 1:
  - Prefetch support for data and instructions is enabled for crossbar masters 0, 1, 2, 3.
  - The cache is configured for least recently used (LRU) replacement for all four ways.
  - The cache is configured for data or instruction replacement.
  - The single-entry buffer is enabled.

13.4.2 Speculative reads

The FMC has a single buffer that reads ahead to the next word in the flash memory if there is an idle cycle. Speculative prefetching is programmable for each bank for instruction and/or data accesses using MCM_CPCR[14] and MCM_CPCR[15]. Because many code accesses are sequential, using the speculative prefetch buffer improves performance in most cases.

When speculative reads are enabled, the FMC immediately requests the next sequential address after a read completes. By requesting the next word immediately, speculative reads can help to reduce or even eliminate wait states when accessing sequential code and/or data.

For example, consider the following scenario:

- Assume a system with a 4:1 core-to-flash clock ratio and with speculative reads enabled.
- The core requests four sequential longwords in back-to-back requests, meaning there are no core cycle delays except for stalls waiting for flash memory data to be returned.
- None of the data is already stored in the cache or speculation buffer.

In this scenario, the sequence of events for accessing the four longwords is as follows:

1. The first longword read requires 4 to 7 core clocks.
2. Due to the 64-bit data bus of the flash memory, the second longword read takes only 1 core clock because the data is already available inside the FMC. While the data for the second longword is being returned to the core, the FMC also starts reading the third and fourth longwords from the flash memory.
3. Accessing the third longword requires 3 core clock cycles. The flash memory read itself takes 4 clocks, but the first clock overlaps with the second longword read.
4. Reading the fourth longword, like the second longword, takes only 1 clock due to the 64-bit flash memory data bus.

13.5 Initialization and application information

The FMC does not require user initialization. Flash acceleration features are enabled by default.

13.6 Usage Guide

For many systems the on-chip flash is the main memory. The Flash Acceleration Unit (FAU) is the interface between the flash memory blocks and the system. In a typical configuration, the core and system bus clock speeds are clock significantly faster than the flash memory clock. The FAU includes features designed to accelerate flash accesses.

For more detailed information, refer to the FMC (same module as FAU) section in AN4745: Optimizing Performance on Kinetis K-series MCUs.

Chapter 14
Flash Memory Module (FTFE)

14.1 Introduction

The FTFE module includes the following accessible memory regions:

- Program flash memory for vector space and code store
- Programming acceleration RAM to speed flash programming

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 FTFE 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.

14.1.1 Features

The FTFE module includes the following features.

NOTE

See Memories and Memory Interfaces chapter for the exact amount of flash memory available on your device.

14.1.1.1 Program Flash Memory Features

- Sector size of 2 Kbytes
- Program flash protection scheme prevents accidental program or erase of stored data
- Automated, built-in, program and erase algorithms with verify
- Section programming for faster bulk programming times
- Read access to one program flash block is possible while programming or erasing data in another program flash block

14.1.1.2 Programming Acceleration RAM features

- RAM to support section programming

14.1.1.3 Other FTFE 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

14.1.2 Block diagram

The block diagram of the FTFE module is shown in the following figure.

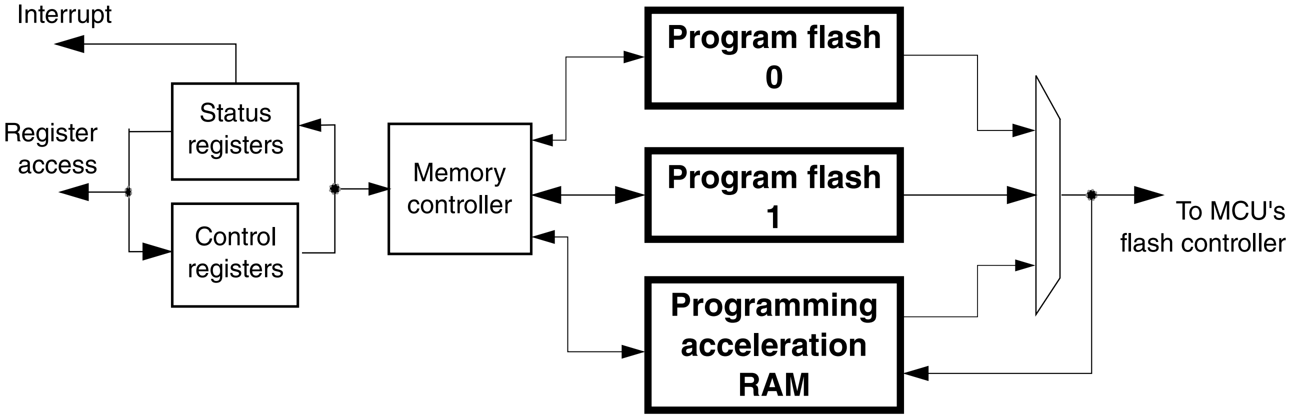


Figure 14-1. FTFE block diagram

14.1.3 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 FTFE 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 FTFE module.

Flash block — A macro within the FTFE module which provides the nonvolatile memory storage.

FTFE 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 FTFE 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.

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 FTFE 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 FTFE module.

Phrase — 64 bits of data with an aligned phrase having byte-address[2:0] = 000.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

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.

Section program buffer — Lower quarter of the programming acceleration RAM allocated for storing large amounts of data for programming via the Program Section command.

Secure — An MCU state conveyed to the FTFE module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

14.2 External signal description

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

14.3 Memory map and registers

This section describes the memory map and registers for the FTFE module. Data read from unimplemented memory space in the FTFE module is undefined. Writes to unimplemented or reserved memory space (registers) in the FTFE module are ignored.

14.3.1 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 FTFE module.

NOTE

The flash configuration field offset addresses are relative byte addresses. Check your device specific memory map for the location of the program flash memory.

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 MCU 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).

14.3.2 Program flash 0 IFR map

The program flash 0 IFR is a 1 Kbyte nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see Read Resource command, Read Once command, Program Once command). The program flash 0 IFR is located within the program flash 0 memory block. The contents of the program flash 0 IFR are summarized in the following table.

Offset Address Range	Size (Bytes)	Field Description
0x000 – 0x3BF	960	Reserved
0x3C0 – 0x3FF	64	Program Once ID Field (index = 0x00 - 0x07)

14.3.2.1 Program Once field

The Program Once field in the program flash 0 IFR provides 64 bytes of user data storage separate from the program flash 0 main array. The user can program the Program Once field one time only as there is no erase mechanism available for the program flash 0 IFR. The Program Once field can be read any number of times. This section of the program flash 0 IFR is accessed in 8 byte records using the Read Once command and Program Once command.

14.3.3 Register descriptions

The FTFE 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).

14.3.3.1 FTFE register descriptions

14.3.3.1.1 FTFE memory map

FTFE base address: 4002_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Flash Status Register (FSTAT)	8	RW	00h
1h	Flash Configuration Register (FCNFG)	8	RW	02h
2h	Flash Security Register (FSEC)	8	R	Table 14-
3h	Flash Option Register (FOPT)	8	R	Table 14-
4h	Flash Common Command Object Registers (FCCOB3)	8	RW	00h
5h	Flash Common Command Object Registers (FCCOB2)	8	RW	00h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
6h	Flash Common Command Object Registers (FCCOB1)	8	RW	00h
7h	Flash Common Command Object Registers (FCCOB0)	8	RW	00h
8h	Flash Common Command Object Registers (FCCOB7)	8	RW	00h
9h	Flash Common Command Object Registers (FCCOB6)	8	RW	00h
Ah	Flash Common Command Object Registers (FCCOB5)	8	RW	00h
Bh	Flash Common Command Object Registers (FCCOB4)	8	RW	00h
Ch	Flash Common Command Object Registers (FCCOBB)	8	RW	00h
Dh	Flash Common Command Object Registers (FCCOBA)	8	RW	00h
Eh	Flash Common Command Object Registers (FCCOB9)	8	RW	00h
Fh	Flash Common Command Object Registers (FCCOB8)	8	RW	00h
10h	Program Flash Protection Registers (FPROT3)	8	RW	Table 14-
11h	Program Flash Protection Registers (FPROT2)	8	RW	Table 14-
12h	Program Flash Protection Registers (FPROT1)	8	RW	Table 14-
13h	Program Flash Protection Registers (FPROT0)	8	RW	Table 14-
2Eh	Flash Error Status Register (FERSTAT)	8	RW	00h
2Fh	Flash Error Configuration Register (FERCNFG)	8	RW	00h

14.3.3.1.2 Flash Status Register (FSTAT)

14.3.3.1.2.1 Offset

Register	Offset
FSTAT	0h

14.3.3.1.2.2 Function

The FSTAT register reports the operational status of the FTFE 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).

14.3.3.1.2.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	CCIF	RDCOLERR	ACCERR	FPVIOL	0			MGSTAT0
W	W1C	W1C	W1C	W1C				
Reset	0	0	0	0	0	0	0	0

14.3.3.1.2.4 Fields

Field	Function
7 CCIF	Command Complete Interrupt Flag The CCIF flag indicates that a FTFE 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. The CCIF bit 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. 0b - FTFE command in progress 1b - FTFE command has completed
6 RDCOLERR	FTFE Read Collision Error Flag The RDCOLERR error bit indicates that the MCU attempted a read from an FTFE resource that was being manipulated by an FTFE 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. 0b - No collision error detected 1b - Collision error detected
5 ACCERR	Flash Access Error Flag The ACCERR error bit indicates an illegal access has occurred to an FTFE resource caused by a violation of the command write sequence or issuing an illegal FTFE 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. 0b - No access error detected 1b - Access error detected
4 FPVIOL	Flash Protection Violation Flag The FPVIOL error bit 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. 0b - No protection violation detected 1b - Protection violation detected
3-1 —	Reserved

Table continues on the next page...

Field	Function
0 MGSTAT0	Memory Controller Command Completion Status Flag The MGSTAT0 status flag is set if an error is detected during execution of an FTFE command or during the flash reset sequence. As a status flag, this bit cannot (and need not) be cleared by the user like the other error flags in this register. 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.

14.3.3.1.3 Flash Configuration Register (FCNFG)

14.3.3.1.3.1 Offset

Register	Offset
FCNFG	1h

14.3.3.1.3.2 Function

This register provides information on the current functional state of the FTFE module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. RAMRDY is a read-only status bit while PFLSH and EEERDY are reserved bits.

14.3.3.1.3.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	CCIE	RDCOLLIE	ERSAREQ	ERSSUSP	SWAP	PFLSH	RAMRDY	EEERDY
W								
Reset	0	0	0	0	0	0	1	0

14.3.3.1.3.4 Fields

Field	Function
7 CCIE	Command Complete Interrupt Enable The CCIE bit controls interrupt generation when an FTFE command completes. 0b - Command complete interrupt disabled

Table continues on the next page...

Field	Function
	1b - Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6 RDCOLLIE	Read Collision Error Interrupt Enable The RDCOLLIE bit controls interrupt generation when an FTFE read collision error occurs. 0b - Read collision error interrupt disabled 1b - Read collision error interrupt enabled. An interrupt request is generated whenever an FTFE read collision error is detected (see the description of FSTAT[RDCOLERR]).
5 ERSAREQ	Erase All Request This bit 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. The ERSAREQ bit sets when an erase all request is triggered external to the FTFE and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the FTFE when the operation completes. 0b - No request or request complete 1b - Request to: 1) run the Erase All Blocks command, 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 The ERSSUSP bit allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing. 0b - No suspend requested 1b - Suspend the current Erase Flash Sector command execution
3 SWAP	Swap The SWAP flag indicates which half of the program flash space is located at relative address 0x0000. The state of the SWAP flag is set by the FTFE during the reset sequence. See the description of the Swap Control command for information on swap management. 0b - Program flash 0 block is located at relative address 0x0000 1b - Program flash 1 block is located at relative address 0x0000
2 PFLSH	This bit is reserved and always has the value 0.
1 RAMRDY	RAM Ready This flag indicates the current status of the programming acceleration RAM. This bit should always be set. 0b - Programming acceleration RAM is not available 1b - Programming acceleration RAM is available
0 EEERDY	EEERDY This bit is reserved and always has the value 0. 0b - See RAMRDY for availability of programming acceleration RAM 1b - Reserved

14.3.3.1.4 Flash Security Register (FSEC)

14.3.3.1.4.1 Offset

Register	Offset
FSEC	2h

14.3.3.1.4.2 Function

This read-only register holds all bits associated with the security of the MCU and FTFE 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.

14.3.3.1.4.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	KEYEN		MEEN		FSLACC		SEC	
W								
Reset	u	u	u	u	u	u	u	u

14.3.3.1.4.4 Fields

Field	Function
7-6 KEYEN	Backdoor Key Security Enable These bits enable and disable backdoor key access to the FTFE module. 00b - Backdoor key access disabled 01b - Backdoor key access disabled (preferred KEYEN state to disable backdoor key access) 10b - Backdoor key access enabled 11b - Backdoor key access disabled
5-4 MEEN	Mass Erase Enable Bits Enables and disables mass erase capability of the FTFE module. When the SEC field is set to unsecure, the MEEN setting does not matter. 00b - Mass erase is enabled 01b - Mass erase is enabled 10b - Mass erase is disabled 11b - Mass erase is enabled
3-2 FSLACC	Factory Security Level Access Code These bits enable or disable 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.

Table continues on the next page...

Field	Function
	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 the SEC bits are set to secure. When the SEC field is set to unsecure, the FSLACC setting does not matter. 00b - Factory access granted 01b - Factory access denied 10b - Factory access denied 11b - Factory access granted
1-0 SEC	Flash Security These bits define the security state of the MCU. In the secure state, the MCU limits access to FTFE module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the FTFE module is unsecured using backdoor key access, the SEC bits are forced to 10b. 00b - MCU security status is secure 01b - MCU security status is secure 10b - MCU security status is unsecure (The standard shipping condition of the FTFE is unsecure.) 11b - MCU security status is secure

14.3.3.1.5 Flash Option Register (FOPT)

14.3.3.1.5.1 Offset

Register	Offset
FOPT	3h

14.3.3.1.5.2 Function

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.

14.3.3.1.5.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	OPT							
W								
Reset	u	u	u	u	u	u	u	u

14.3.3.1.5.4 Fields

Field	Function
7-0 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.

14.3.3.1.6 Flash Common Command Object Registers (FCCOB0 - FCCOBB)

14.3.3.1.6.1 Offset

For a = 0 to B (0 to 11):

Register	Offset
FCCOBa	4h + (a + 3 - 2 × (a mod 4))

14.3.3.1.6.2 Function

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.

14.3.3.1.6.3 Diagram

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

14.3.3.1.6.4 Fields

Field	Function
7-0 CCOBn	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 commandâs execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit,

Field	Function
	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 FTFE 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 FTFE command, typically an address and/or data values. **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 FTFE 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

1. Refers to FCCOB register name, not register address

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).

1. Refers to FCCOB register name, not register address

14.3.3.1.7 Program Flash Protection Registers (FPROT0 - FPROT3)

14.3.3.1.7.1 Offset

Register	Offset
FPROT3	10h
FPROT2	11h

Table continues on the next page...

Register	Offset
FPROT1	12h
FPROT0	13h

14.3.3.1.7.2 Function

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 FTFE command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow up to 32 protectable regions of equal memory size.

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.

14.3.3.1.7.3 Diagram

Bits	7	6	5	4	3	2	1	0
R W	PROT							
Reset	u	u	u	u	u	u	u	u

14.3.3.1.7.4 Fields

Field	Function
7-0 PROT	Program Flash Region Protect Each program flash region can be protected from program and erase operations by setting the associated PROT bit to the protected state. 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. **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. 0000_0000b - Program flash region is protected. 0000_0001b - Program flash region is not protected

14.3.3.1.8 Flash Error Status Register (FERSTAT)

14.3.3.1.8.1 Offset

Register	Offset
FERSTAT	2Eh

14.3.3.1.8.2 Function

This register reports the detection of uncorrected ECC errors during read access to the FTFE module.

The DFDIF flag is readable and writable. The unassigned bits read 0 and are not writable.

14.3.3.1.8.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0						DFDIF	0
W							W1C	
Reset	0	0	0	0	0	0	0	0

14.3.3.1.8.4 Fields

Field	Function
7-2 —	Reserved
1 DFDIF	Double Bit Fault Detect Interrupt Flag The DFDIF flag indicates an uncorrectable ECC fault was detected during a valid flash read access from the platform flash controller. The DFDIF flag is cleared by writing a 1 to it. Writing a 0 to DFDIF has no effect. 0b - Double bit fault not detected during a valid flash read access from the platform flash controller 1b - Double bit fault detected (or FERCNFG[FDFD] is set) during a valid flash read access from the platform flash controller
0 —	Reserved

14.3.3.1.9 Flash Error Configuration Register (FERCNFG)

14.3.3.1.9.1 Offset

Register	Offset
FERCNFG	2Fh

14.3.3.1.9.2 Function

This register enables the force and interrupt of uncorrected ECC errors detected during read access to the FTFE module.

The FDFD and DFDIE bits are readable and writable. The unassigned bits read 0 and are not writable.

14.3.3.1.9.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0		FDFD	0			DFDIE	0
W								
Reset	0	0	0	0	0	0	0	0

14.3.3.1.9.4 Fields

Field	Function
7-6 —	Reserved
5 FDFD	Force Double Bit Fault Detect The FDFD bit enables the user to emulate the setting of the FERSTAT[DFDIF] flag to check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. 0b - FERSTAT[DFDIF] sets only if a double bit fault is detected during read access from the platform flash controller 1b - FERSTAT[DFDIF] sets during any valid flash read access from the platform flash controller. An interrupt request is generated if the DFDIE bit is set.
4-2 —	Reserved
1 DFDIE	Double Bit Fault Detect Interrupt Enable The DFDIE bit controls interrupt generation when an uncorrectable ECC fault is detected during a valid flash read access from the platform flash controller. 0b - Double bit fault detect interrupt disabled 1b - Double bit fault detect interrupt enabled. An interrupt request is generated whenever the FERSTAT[DFDIF] flag is set.
0 —	Reserved

14.4 Functional Description

The following sections describe functional details of the FTFE module.

14.4.1 Program flash memory swap

The user can configure the memory map of the program flash space such that either half of the program flash memory can exist at relative address 0x0000. This swap feature enables the lower half of the program flash space to be operational while the upper half is being updated for future use.

The Swap Control command handles swapping the two halves of program flash memory within the memory map. See Swap Control command for details.

14.4.2 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* — Four registers protect 32 regions of the program flash memory as shown in the following figure

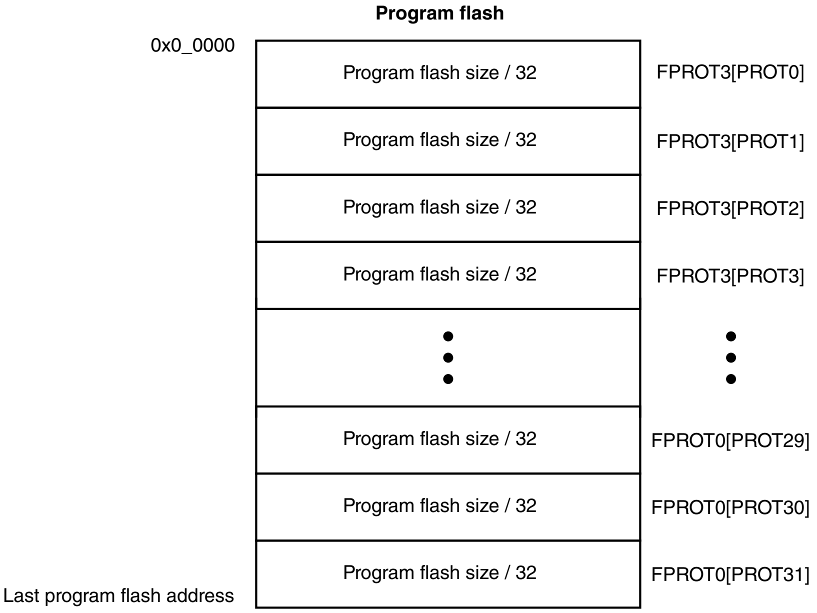


Figure 14-2. Program flash protection

NOTE

Flash protection features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Some features described in the application note may not be available on this device.

14.4.3 Interrupts

The FTFE module can generate interrupt requests to the MCU upon the occurrence of various FTFE events. These interrupt events and their associated status and control bits are shown in the following table.

Table 14-1. FTFE Interrupt Sources

FTFE Event	Readable Status Bit	Interrupt Enable Bit
FTFE Command Complete	FSTAT[CCIF]	FCNFG[CCIE]

Table continues on the next page...

Table 14-1. FTFE Interrupt Sources (continued)

FTFE Event	Readable Status Bit	Interrupt Enable Bit
FTFE Read Collision Error	FSTAT[RDCOLERR]	FCNFG[RDCOLLIE]
FTFE ECC Error Detection	FERSTAT[DFDIF]	FERCNFG[DFDIE]

Note

Vector addresses and their relative interrupt priority are determined at the MCU level.

14.4.4 Flash Operation in Low-Power Modes

14.4.4.1 Wait Mode

When the MCU enters wait mode, the FTFE module is not affected. The FTFE module can recover the MCU from wait via the command complete interrupt (see Interrupts).

14.4.4.2 Stop Mode

When the MCU requests stop mode, if an FTFE 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 FTFE command is running (CCIF = 0).

NOTE

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the FTFE module does not accept flash commands.

14.4.5 Flash memory reads and ignored writes

The FTFE module requires only the flash address to execute a flash memory read. MCU read access is available to all flash memory.

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).

14.4.6 Read while write (RWW)

The following simultaneous accesses are allowed:

- The user may read from one program flash memory block while commands are active in the other program flash memory block.

Simultaneous operations are further discussed in Allowed simultaneous flash operations.

14.4.7 Flash Program and Erase

All flash functions except read require the user to setup and launch an FTFE command through a series of peripheral bus writes. The user cannot initiate any further FTFE commands until notified that the current command has completed. The FTFE command structure and operation are detailed in FTFE Command Operations.

14.4.8 FTFE Command Operations

FTFE command operations are typically used to modify flash memory contents. The next sections describe:

- The command write sequence used to set FTFE command parameters and launch execution
- A description of all FTFE commands available

14.4.8.1 Command Write Sequence

FTFE commands are specified using a command write sequence illustrated in Command Execution and Error Reporting. The FTFE 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 an FTFE command in VLP mode will be ignored. Attempts to launch an FTFE command in HSRUN mode will be trapped with the ACCERR flag being set.

14.4.8.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired FTFE command. The individual registers that make up the FCCOB data set can be written in any order.

14.4.8.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing the FSTAT[CCIF] bit by writing a '1' to it. The CCIF flag remains zero until the FTFE command completes.

The FSTAT register contains a blocking mechanism, which prevents a new command from launching (can't clear 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.

14.4.8.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The FTFE 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. 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, the 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 the FSTAT[CCIF] bit.

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 the FSTAT[MGSTAT0] bit. 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 FTFE sets the FSTAT[CCIF] bit signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

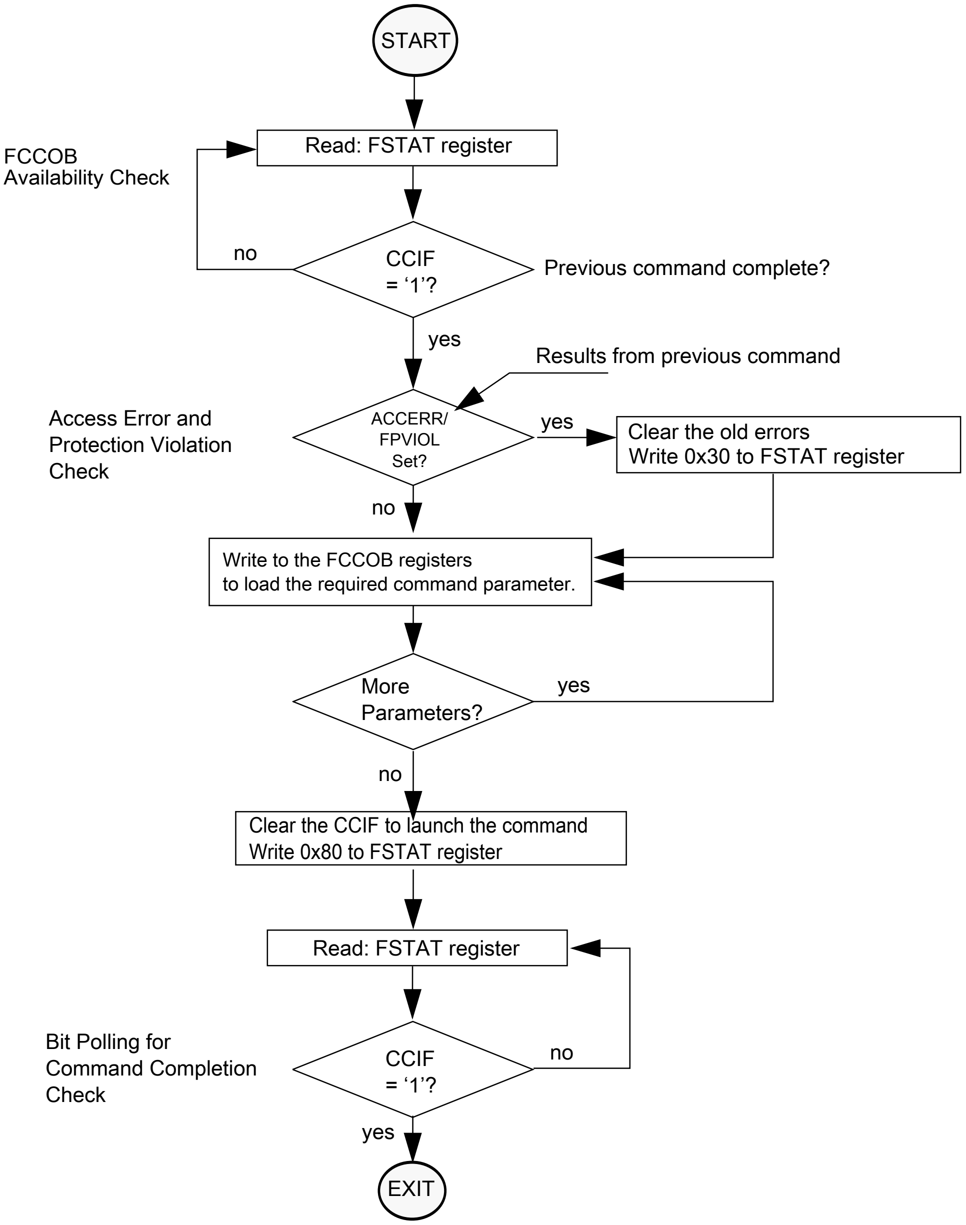


Figure 14-3. Generic Flash Command Write Sequence Flowchart

14.4.8.2 Flash commands

The following table summarizes the function of all flash commands. If any column is marked with an 'X', the flash command is relevant to that particular memory resource.

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 phrases at margin read levels.
0x03	Read Resource	IFR,ID	IFR	Read 8 bytes from program flash IFR or version ID.
0x07	Program Phrase	×	×	Program 8 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.
0x0B	Program Section	×	×	Program data from the Section Program Buffer to a program flash block.
0x40	Read 1s All Blocks	×	×	Verify that all program flash blocks are erased then release MCU security.
0x41	Read Once	IFR		Read 8 bytes of an indexed field in the program flash 0 IFR.
0x43	Program Once	IFR		One-time program of 8 bytes of an indexed field in the program flash 0 IFR.
0x44	Erase All Blocks	×	×	Erase all program flash blocks, program flash swap IFR. Then, verify-erase and release MCU security. **NOTE**: An erase is only possible when all memory locations are unprotected.
0x45	Verify Backdoor Access Key	×	x	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
0x46	Swap Control	x	x	Handles swap-related activities.
0x49	Erase All Blocks Unsecure	×	×	Erase all program flash blocks, program flash swap IFR. Then, verify-erase, program the security byte to the unsecure state, and release MCU security.

14.4.8.3 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 14-2. Allowed Simultaneous Memory Operations

		Program flash X[1]			
		Read	**Program Phrase**	**Erase Flash Sector**	**Erase Flash Block**
Program flash Y[1]	**Read**		OK	OK	OK
	Program Phrase	OK			
	Erase Flash Sector	OK			
	Erase Flash Block	OK			

[1] P-Flash X refers to any of the P-Flash blocks (0, 1) and P-Flash Y refers to any of the P-Flash blocks (0, 1), but not the same block. Thus, it is possible to read from any of the blocks while programming or erasing another.

14.4.9 Margin Read Commands

The Read-1s commands (Read 1s All Blocks, Read 1s Block, Read 1s Section) 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. All simple (uncommanded) flash array reads to the MCU always 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.

14.4.10 Flash command descriptions

This section describes all flash commands that can be launched by a command write sequence. The FTFE 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 the ACCERR and FPVIOL bits in the FSTAT register 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 FTFE is running a command (CCIF = 0) on that same block. The FTFE 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.

14.4.10.1 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 block is erase-verified.

Table 14-3. 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 64-bit aligned (Flash address [2:0] = 000).

After clearing CCIF to launch the Read 1s Block command, the FTFE sets the read margin for 1s according to Table 14-4 and then reads all locations within the selected program flash block.

Table 14-4. 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 14-5. 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 64-bit aligned	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

14.4.10.2 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 14-6. 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 64-bit aligned (Flash address [2:0] = 000).

Upon clearing CCIF to launch the Read 1s Section command, the FTFE sets the read margin for 1s according to Table 14-7 and then reads all locations within the specified section of flash memory.

If the FTFE fails to read all 1s (i.e. the flash section is not erased), the FSTAT(MGSTAT0) bit is set. The CCIF flag sets after the Read 1s Section operation completes.

Table 14-7. 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 14-8. 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 64-bit aligned	FSTAT[ACCERR]
The requested section crosses a flash block boundary	FSTAT[ACCERR]
The requested number of phrases is zero	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

14.4.10.3 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 14-9. 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 FTFE sets the read margin for 1s based on the provided margin choice according to Table 14-10. The Program Check operation then 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, the MGSTAT0 bit is set.

The FTFE will then set the read margin for 0s based on the provided margin choice.The Program Check operation will then read the specified longword and compare the actual read data to the expected data provided by the FCCOB. If the comparison at margin-0 fails, the MGSTAT0 bit will be set. The CCIF flag will set after the Program Check operation has completed.

The starting address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 0 data is expected at the supplied 32-bit aligned address,
- Byte 1 data is expected at byte address specified + 0b01,
- Byte 2 data is expected at byte address specified + 0b10, and
- Byte 3 data is expected at byte address specified + 0b11.

NOTE

See the description of margin reads, Margin Read Commands

Table 14-10. 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 14-11. 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]
Flash address is not longword aligned	FSTAT[ACCERR]
An invalid margin choice is supplied	FSTAT[ACCERR]
Either of the margin reads does not match the expected data	FSTAT[MGSTAT0]

14.4.10.4 Read Resource command

The Read Resource command is provided for the user to read data from special-purpose memory resources located within the Flash module. The special-purpose memory resources available include program flash IFR space, and the Version ID field. The Version ID field contains an 8 byte code that indicates a specific FTFE implementation.

Table 14-12. 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]
4	Resource select code (see Table 14-13)
Returned values	
4	Read Data [64:56]

Table continues on the next page...

Table 14-12. Read Resource Command FCCOB Requirements (continued)

FCCOB Number	FCCOB contents [7:0]
5	Read Data [55:48]
6	Read Data [47:40]
7	Read Data [39:32]
8	Read Data [31:24]
9	Read Data [23:16]
A	Read Data [15:8]
B	Read Data [7:0]

1. Must be 64-bit aligned (Flash address [2:0] = 000).

Table 14-13. Read Resource Select Codes

Resource Select Code	Description	Resource Size	Local Address Range
0x00	Program Flash 0 IFR	1024 Bytes	0x00_0000 - 0x00_03FF
0x00	Program Flash Swap IFR	1024 Bytes	0x04_0000 - 0x04_03FF
0x01	Version ID	8 Bytes	0x00_0008 - 0x00_000F

After clearing CCIF to launch the Read Resource command, eight consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag will set after the Read Resource operation has completed. 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 14-14. 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 64-bit aligned	FSTAT[ACCERR]

14.4.10.5 Program Phrase command

The Program Phrase command programs eight 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 14-15. Program Phrase Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x07 (PGM8)
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
8	Byte 4 program value
9	Byte 5 program value
A	Byte 6 program value
B	Byte 7 program value

[1] Must be 64-bit aligned (Flash address [2:0] = 000)

Upon clearing CCIF to launch the Program Phrase command, the FTFE programs the data bytes into the flash using the supplied address. The protection status is always checked. If the swap system is enabled, the double-phrase containing the swap indicator address in each half of the program flash space is implicitly protected from programming. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Phrase 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 MGSTAT0. The CCIF flag is set after the Program Phrase operation completes.

The starting address must be 64-bit aligned (flash address [2:0] = 000):

- Byte 0 data is written to the starting address ('start'),
- Byte 1 data is programmed to byte address start+0b01,
- Byte 2 data is programmed to byte address start+0b10, and
- Byte 3 data is programmed to byte address start+0b11, etc.

Table 14-16. Program Phrase 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 64-bit aligned	FSTAT[ACCERR]
Flash address points to a protected area	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation.	FSTAT[MGSTAT0]

14.4.10.6 Erase Flash Block command

The Erase Flash Block operation erases all addresses in a single program flash.

Table 14-17. 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 64-bit aligned (Flash address [2:0] = 000).

Upon clearing CCIF to launch the Erase Flash Block command, the FTFE 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 program flash protection (FPROT) registers). If the swap system is enabled, the swap indicator address is implicitly protected from block erase unless the swap system is in the UPDATE or UPDATE-ERASED state and the program flash block being erased is the non-active block that contains the swap indicator address. If the erase verify fails, the MGSTAT0 bit in FSTAT is set. The CCIF flag will set after the Erase Flash Block operation has completed.

Table 14-18. 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 64-bit aligned	FSTAT[ACCERR]
Any area of the selected flash block 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 Block command to verify all bits are erased.

14.4.10.7 Erase Flash Sector command

The Erase Flash Sector operation erases all addresses in a flash sector.

Table 14-19. 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 64-bit aligned (Flash address [2:0] = 000).

After clearing CCIF to launch the Erase Flash Sector command, the FTFE 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 swap system is enabled, the swap indicator address in each program flash block is implicitly protected from sector erase unless the swap system is in the UPDATE or UPDATE-ERASED state and the program flash sector containing the swap indicator address being erased is in the non-active block. 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 Aborting a Suspended Erase Flash Sector Operation).

Table 14-20. 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 64-bit aligned	FSTAT[ACCERR]
The selected program flash sector 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 Section command to verify all bits are erased.

14.4.10.7.1 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 samples the state of the ERSSUSP bit at convenient points. If the FTFE detects that the ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the FTFE 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 FTFE detects that a suspend request has been made, the FTFE clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the FTFE 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 FTFE has acknowledged it.

14.4.10.7.2 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 FTFE 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.

14.4.10.7.3 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 FTFE 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.

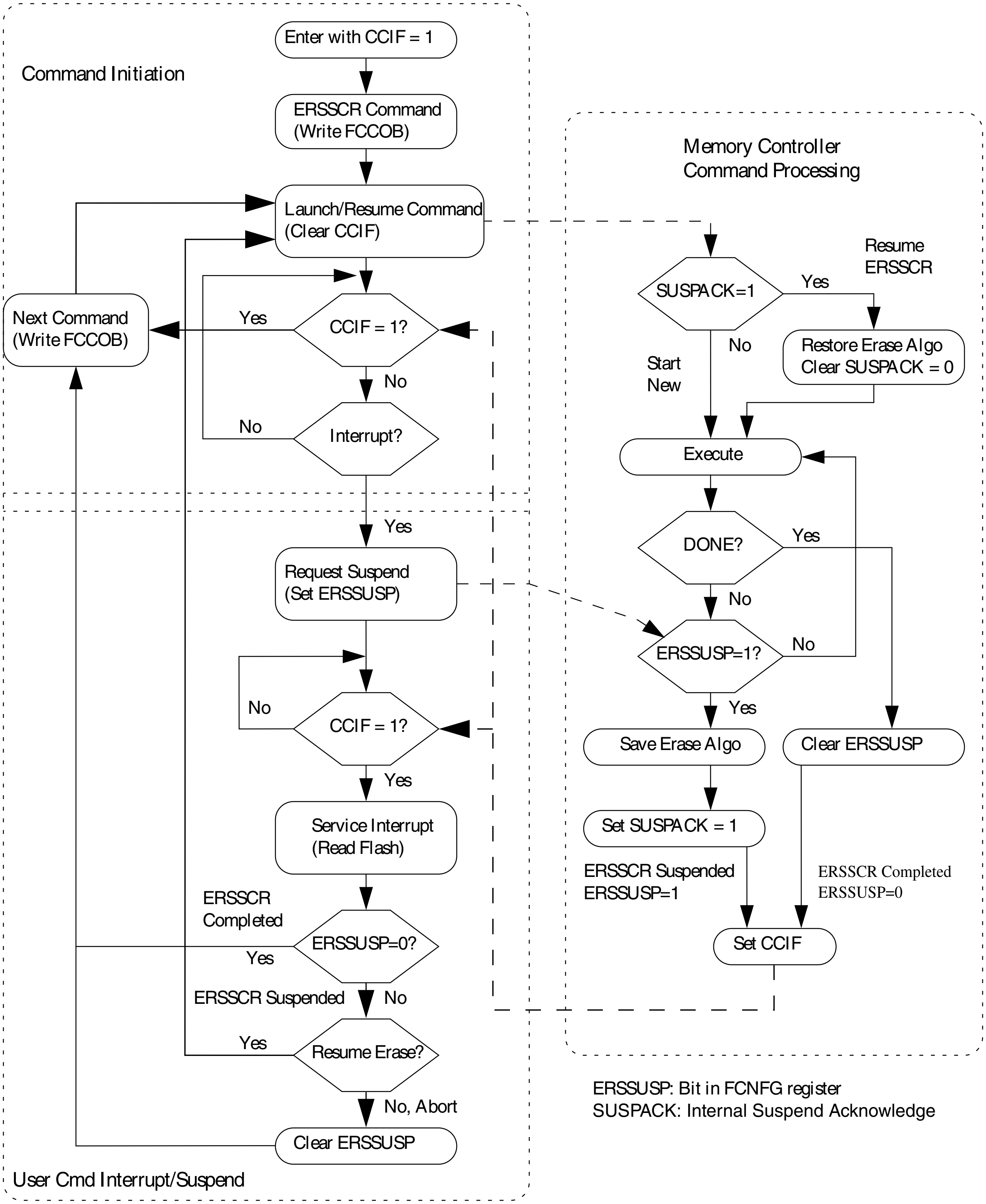

Command Initiation
Enter with CCIF = 1
ERSSCR Command (Write FCCOB)
Launch/Resume Command (Clear CCIF)
Next Command (Write FCCOB)
Yes
CCIF = 1?
No
Interrupt?
No
Yes
Request Suspend (Set ERSSUSP)
No
CCIF = 1?
Yes
Service Interrupt (Read Flash)
ERSSCR Completed
ERSSUSP=0?
Yes
ERSSCR Suspended
No
Yes
Resume Erase?
No, Abort
Clear ERSSUSP
User Cmd Interrupt/Suspend
Memory Controller Command Processing
SUSPACK=1
Yes
Resume ERSSCR
Restore Erase Algo Clear SUSPACK = 0
No
Start New
Execute
DONE?
Yes
No
ERSSUSP=1?
No
Yes
Save Erase Algo
Clear ERSSUSP
Set SUSPACK = 1
ERSSCR Suspended ERSSUSP=1
ERSSCR Completed ERSSUSP=0
Set CCIF
ERSSUSP: Bit in FCNFG register
SUSPACK: Internal Suspend Acknowledge

Figure 14-4. Suspend and Resume of Erase Flash Sector Operation

14.4.10.8 Program Section command

The Program Section operation programs the data found in the section program buffer to previously erased locations in the flash memory using an embedded algorithm. Data is preloaded into the section program buffer (see Flash sector programming).

The section program buffer is limited to the lower quarter of the programming acceleration RAM (relative byte addresses 0x0000-0x03FF - be sure to check your device specific memory map for the location of the programming acceleration RAM). Data written to the remainder of the programming acceleration RAM is ignored and may be overwritten during Program Section command execution.

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 14-21. Program Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x0B (PGMSEC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Number of phrases to program [15:8]
5	Number of phrases to program [7:0]

[1] Must be 64-bit aligned (Flash address [2:0] = 000).

After clearing CCIF to launch the Program Section command, the FTFE will block access to the programming acceleration RAM and program the data residing in the Section Program Buffer into the flash memory starting at the flash address provided.

The starting address must be unprotected (see the description of the FPROT registers) to permit execution of the Program Section operation. If the swap system is enabled, the phrase containing the swap indicator in each half of the program flash space is implicitly protected from programming. If the phrase containing the swap indicator address is encountered during the Program Section operation, it will be bypassed without setting FPVIOL and the contents will not be programmed. Programming, which is not allowed to cross a flash sector boundary, continues until all requested phrases have been programmed.

After the Program Section operation has completed, the CCIF flag will set. The contents of the Section Program Buffer are not changed by the Program Section operation unless the swap system is enabled and the Program Section operation is targeting the phrase containing the swap indicator in which case that phrase is changed to all ones.

Table 14-22. Program Section 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 64-bit aligned	FSTAT[ACCERR]
The requested section crosses a program flash sector boundary	FSTAT[ACCERR]
The requested number of phrases is zero	FSTAT[ACCERR]
The space required to store data for the requested number of phrases is more than one quarter the size of the programming acceleration RAM	FSTAT[ACCERR]
The flash address falls in a protected area	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

14.4.10.8.1 Flash sector programming

The process of programming an entire flash sector using the Program Section command is as follows:

1. Launch the Erase Flash Sector command to erase the flash sector to be programmed.
2. Beginning with the starting address of the programming acceleration RAM, sequentially write enough data to the RAM to fill an entire flash sector, or as much data is allowed due to RAM size versus flash sector size. This area of the RAM serves as the section program buffer. The section program buffer can be written to while the operation launched in step 1 is executing, i.e. while CCIF = 0.
3. Execute the Program Section command to program the contents of the section program buffer into the selected flash sector.
4. Repeat steps 2 through 3 to complete the entire flash sector, if necessary.
5. To program additional flash sectors, repeat steps 1 through 4.

14.4.10.9 Read 1s All Blocks command

The Read 1s All Blocks command checks if the program flash blocks 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 14-23. 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 FTFE :

- sets the read margin for 1s according to Table 14-24,
- checks the contents of the program flash are in the erased state.

If the FTFE confirms that these memory resources are erased, 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 flash memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The EEERDY and RAMRDY bits are clear during the Read 1s All Blocks operation and are restored at the end of the Read 1s All Blocks operation.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

Table 14-24. 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 14-25. 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]

14.4.10.10 Read Once command

The Read Once command provides read access to indexed 8-byte records located in the program flash 0 IFR (see Program flash 0 IFR map and Program Once field) . Each Program Once record index is programmed using the Program Once command.

Table 14-26. Read Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x41 (RDONCE)
1	Program Once record index (0x00 - 0x07)
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
9	Program Once byte 5 value
A	Program Once byte 6 value
B	Program Once byte 7 value

After clearing CCIF to launch the Read Once command, an 8-byte Program Once record is read from the program flash 0 IFR (index 0x00 to 0x07) and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. During execution of the Read Once command, any attempt to read addresses within the flash block containing the 8-byte record returns invalid data. The Read Once command can be executed any number of times.

Table 14-27. 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]

14.4.10.11 Program Once command

The Program Once command enables programming of indexed 8-byte records located in the program flash 0 IFR (see Program flash 0 IFR map and Program Once field) . The Program Once field can be read using the Read Once command or using the Read Resource command. Each Program Once record in program flash 0 IFR can be programmed only once since the program flash 0 IFR cannot be erased.

Table 14-28. Program Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x43 (PGMONCE)
1	Program Once record index (0x00 - 0x07)

Table continues on the next page...

Table 14-28. Program Once Command FCCOB Requirements (continued)

FCCOB Number	FCCOB Contents [7:0]
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
9	Program Once Byte 5 value
A	Program Once Byte 6 value
B	Program Once Byte 7 value

After clearing CCIF to launch the Program Once command and verifying that the selected record is erased, the selected record is programmed using the values provided into the program flash 0 IFR (index 0x00 to 0x07). The Program Once command also verifies that the programmed values read back correctly. Any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. The CCIF flag is set after the Program Once operation has completed. During execution of the Program Once command, any attempt to read addresses within the flash block containing the 8-byte record returns invalid data.

Table 14-29. 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-erased 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_FFFF_FFFF, the Program Once command is allowed to execute again on that same record.

14.4.10.12 Erase All Blocks command

The Erase All Blocks operation erases all flash memory, verifies all memory contents, and releases MCU security.

Table 14-30. 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 FTFE erases all program flash memory, program flash swap IFR space, then verifies that all are erased.

If the FTFE verifies that all flash memories were properly erased, security is released by setting the FSEC[SEC] field to the unsecure state and the FCNFG[RAMRDY] bit is set. The Erase All Blocks command aborts if any flash region is protected. The swap indicator address in the program flash blocks are not implicitly protected from the erase operation. 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.

Table 14-31. 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.

14.4.10.12.1 Triggering an erase all external to the flash 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 FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory, program flash swap IFR space regardless of the state of the FSTAT[ACCERR and FPVIOL] flags or the protection settings or the state of the flash swap system. If the post-erase verify passes, the routine releases security by setting the FSEC[SEC] field register to the unsecure state and the FCNFG[RAMRDY] bit sets. 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, except FPVIOL, is available as described in Erase All Blocks command/Erase All Blocks Unsecure command.

CAUTION

Since the program flash swap IFR containing the swap indicator address is erased during the Erase All Blocks command operation, the swap system becomes uninitialized. The Swap Control command must be run with the initialization code to set the swap indicator address and initialize the swap system.

14.4.10.13 Verify Backdoor Access Key command

Execution of the Verify Backdoor Access Key command is 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. The column labeled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

Table 14-32. 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 FTFE checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the FTFE sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the FTFE 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 FTFE 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 14-33. 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]

14.4.10.14 Swap Control command

The Swap Control command handles specific activities associated with swapping the two halves of program flash memory within the memory map.

Table 14-34. Swap Control Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x46 (SWAP)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Swap Control Code: 0x01 - Initialize Swap System 0x02 - Set Swap in Update State 0x04 - Set Swap in Complete State 0x08 - Report Swap Status
Returned values	
5	Current Swap Mode: 0x00 - Uninitialized 0x01 - Ready 0x02 - Update 0x03 - Update-Erased 0x04 - Complete
6	Current Swap Block Status: For devices with FlexNVM: 0x00 - Program flash block 0 at 0x0_0000 0x01 - Program flash block 1 at 0x0_0000 For devices with program flash only: 0x00 - Program flash block 0 at 0x0_0000 0x01 - Program flash block 1 at 0x0_0000

Table continues on the next page...

Table 14-34. Swap Control Command FCCOB Requirements (continued)

FCCOB Number	FCCOB Contents [7:0]
7	Next Swap Block Status (after any reset): For devices with FlexNVM: 0x00 - Program flash block 0 at 0x0_0000 0x01 - Program flash block 1 at 0x0_0000 For devices with program flash only: 0x00 - Program flash block 0 at 0x0_0000 0x01 - Program flash block 1 at 0x0_0000

1. Must be 64-bit aligned (Flash address [2:0] = 000).

Upon clearing CCIF to launch the Swap Control command, the FTFE will handle swap-related activities based on the Swap Control code provided in FCCOB4 as follows:

- 0x01 (Initialize Swap System to UPDATE-ERASED State) - After verifying that the current swap state is UNINITIALIZED, and that both phrases which will contain the swap indicators (located in each half of the Program flash memory within the relative phrase flash address provided) are erased, and that the flash address provided is in the lower half of Program flash memory but not in the Flash Configuration Field, the flash address provided (shifted with bits[2:0] removed) will be programmed into the swap indicator address field found in the program flash swap IFR. After the swap indicator address has been programmed into the program flash swap IFR, the swap enable word will be programmed to 0x0000. After the swap enable word has been programmed, the swap indicator located in the lower half of the Program flash memory will be programmed to 0xFF00.
- 0x02 (Progress Swap to UPDATE State) - After verifying that the current swap state is READY and that the aligned flash address provided matches the one stored in the program flash swap IFR, the swap indicator located in the currently active program flash block will be programmed to 0xFF00.
- 0x04 (Progress Swap to COMPLETE State) - After verifying that the current swap state is UPDATE-ERASED and that the aligned flash address provided matches the one stored in the program flash swap IFR, the swap indicator located in the currently active program flash block will be programmed to 0x0000. Before executing with this Swap Control code, the user must erase the non-active swap indicator using the Erase Flash Block or Erase Flash Sector commands and update the application code or data as needed.
- 0x08 (Report Swap Status) - After verifying that the aligned flash address provided is in the lower half of Program flash memory but not in the Flash Configuration Field, the status of the swap system will be reported as follows:
  - FCCOB5 (Current Swap State) - indicates the current swap state based on the status of the swap enable word and the swap indicators. If the MGSTAT0 flag is

set after command completion, the swap state returned was not successfully transitioned from and the appropriate swap command code must be attempted again. If the current swap state is UPDATE and the non-active swap indicator is 0xFFFF, the current swap state is changed to UPDATE-ERASED.
- FCCOB6 (Current Swap Block Status) - indicates which program flash block is currently located at relative flash address 0x0_0000.
- FCCOB7 (Next Swap Block Status) - indicates which program flash block will be located at relative flash address 0x0_0000 after the next reset of the FTFE module.

NOTE

It is recommended that the user execute the Swap Control command to report swap status (code 0x08) after any reset to determine if issues with the swap system were detected during the swap state determination procedure.

NOTE

It is recommended that the user write 0xFF to FCCOB5, FCCOB6, and FCCOB7 since the Swap Control command will not always return the swap state and status fields when an ACCERR is detected.

The CCIF flag is set after the Swap Control operation has completed.

The swap indicators are implicitly protected from being programmed during Program Phrase or Program Section command operations and are implicitly unprotected during Swap Control command operations. The swap indicators are implicitly protected from being erased during Erase Flash Block and Erase Flash Sector command operations unless the swap indicator being erased is in the non-active program flash block and the swap system is in the UPDATE or UPDATE-ERASED state. The Erase All Blocks command or erase-all function can be used to place the swap system in the UNINITIALIZED state.

Table 14-35. Swap Control Command Error Handling

Error Condition	Swap Control Code	Error Bit
Command not available in current mode/security[1]	All	FSTAT[ACCERR]
Flash address is not in the lower half of program flash memory	All	FSTAT[ACCERR]
Flash address is in the Flash Configuration Field	1, 8	FSTAT[ACCERR]
Flash address is not 64-bit aligned	All	FSTAT[ACCERR]
Flash address does not match the swap indicator address in the IFR	2, 4	FSTAT[ACCERR]
Swap initialize requested when phrase containing swap indicator (in each half of program flash memory) is not in the erased state	1	FSTAT[ACCERR]

Table continues on the next page...

Table 14-35. Swap Control Command Error Handling (continued)

Error Condition	Swap Control Code	Error Bit
Swap initialize requested when swap system is not in the uninitialized state	1	FSTAT[ACCERR]
Swap update requested when swap system is not in the ready state	2	FSTAT[ACCERR]
Swap complete requested when swap system is not in the update-erased state	4	FSTAT[ACCERR]
An undefined swap control code is provided	-	FSTAT[ACCERR]
Any errors have been encountered during the swap determination and program-verify operations	1, 2, 4	FSTAT[MGSTAT0]
Any brownouts were detected during the swap determination procedure	8	FSTAT[MGSTAT0]

1. Returned fields will not be updated, i.e. no swap state or status reporting

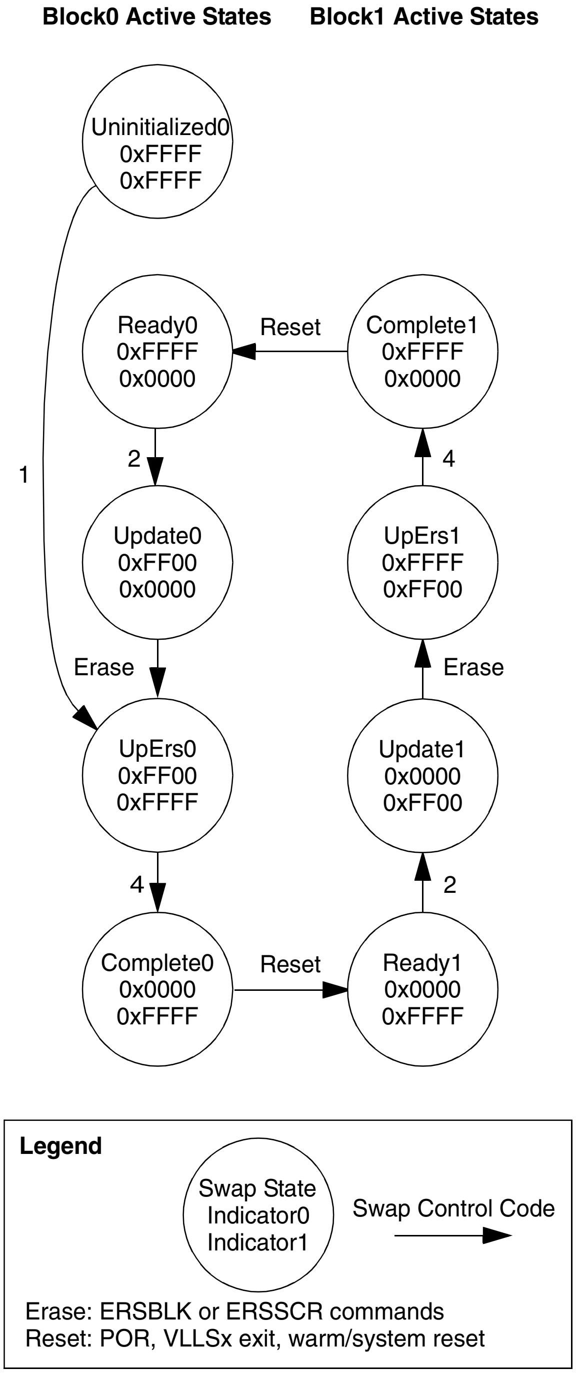


Figure 14-5. Valid Swap State Sequencing

Table 14-36. Swap State Report Mapping

Case	Swap Enable Field[1]	Swap Indicator 0[1]	Swap Indicator 1[1]	Swap State[2]	State Code	MGSTAT0	Active Block
1	0xFFFF	-	-	Uninitialized	0	0	0
2	0x0000	0xFF00	0x0000	Update	2	0	0
3	0x0000	0xFF00	0xFFFF	Update-Erased	3	0	0
4	0x0000	0x0000	0xFFFF[3]	Complete[4]	4	0	0
5	0x0000	0x0000	0xFFFF	Ready[5]	1	0	1
6	0x0000	0x0000	0xFF00	Update	2	0	1
7	0x0000	0xFFFF	0xFF00	Update-Erased	3	0	1
8	0x0000	0xFFFF[3]	0x0000	Complete[4]	4	0	1
9	0x0000	0xFFFF	0x0000	Ready[5]	1	0	0
10	0xXXXX	-	-	Uninitialized	0	1	0
11	0x0000	0xFFFF	0xFFFF	Uninitialized	0	1	0
12	0x0000	0xFFXX	0xFFFF	Ready	1	1	0
13	0x0000	0xFFXX	0x0000	Ready	1	1	0
14	0x0000	0xXXXX	0x0000	Ready	1	1	0
15[6]	0x0000	0xFFFF	0xFFXX	Ready	1	1	1
16	0x0000	0x0000	0xFFXX	Ready	1	1	1
17[6]	0x0000	0x0000	0xXXXX	Ready	1	1	1
18	0x0000	0xFF00	0xFFFF	Update	2	1	0
19	0x0000	0xFF00	0xXXXX	Update	2	1	0
20	0x0000	0xFF(00)	0xFFXX	Update	2	1	0
21[6]	0x0000	0x0000	0x0000	Update	2	1	0
22[6]	0x0000	0xXXXX	0xXXXX	Update	2	1	0
23	0x0000	0xFFFF[7]	0xFF00	Update	2	1	1
24	0x0000	0xXXXX	0xFF00	Update	2	1	1
25	0x0000	0xFFXX	0xFF(00)	Update	2	1	1
26	0x0000	0xXX00	0xFFFF	Update-Erased	3	1	0
27	0x0000	0xXXXX	0xFFFF	Update-Erased	3	1	0
28	0x0000	0xFFFF	0xXX00	Update-Erased	3	1	1
29	0x0000	0xFFFF	0xXXXX	Update-Erased	3	1	1

[1] 0xXXXX, 0xFFXX, 0xXX00 indicates a non-valid value was read; 0xFF(00) indicates more 0's than other indicator (if same number of 0's, then swap system defaults to block 0 active)

[2] Cases 10-29 due to brownout (abort) detected during program or erase steps related to swap

[3] Must read 0xFFFF with erase verify level before transition to Complete allowed

[4] No reset since successful Swap Complete execution

[5] Reset after successful Swap Complete execution

[6] Not a valid case

[7] Fails to read 0xFFFF at erase verify level

14.4.10.14.1 Swap state determination

During the reset sequence, the state of the swap system is determined by evaluating the swap related fields in the program flash swap IFR and the swap indicator phrase found at the relative swap indicator address within each half of the program flash memory.

Table 14-37. Program Flash Swap IFR Fields

Address Range	Size (Bytes)	Field Description
0x000 – 0x001	2	Swap Indicator Address
0x002 – 0x003	2	Swap Enable Word
0x004 – 0x3FF	1020	Reserved

14.4.10.15 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 14-38. 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 FTFE erases all program flash memory, program flash swap IFR space, then verifies that all are erased.

If the FTFE verifies that all flash memories were properly erased, security is released by setting the FSEC[SEC] field to the unsecure state, the security byte (see Flash configuration field description) is programmed to the unsecure state by the Erase All Blocks Unsecure command, and the FCNFG[RAMRDY] bit is set. The swap indicator address in the program flash blocks are not implicitly protected from the erase operation. 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.

Table 14-39. 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]

14.4.11 Security

The FTFE module provides security information to the MCU based on contents of the FSEC security register. The MCU then limits access to FTFE resources as defined in the device's Chip Configuration details. During reset, the FTFE 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. Details of the settings are described in the FSEC register description.

Flash security features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Some features described in the application note may not be available on this device.

Table 14-40. FSEC fields

FSEC field	Description
KEYEN	Backdoor Key Access
MEEN	Mass Erase Capability
FSLACC	Factory Security Level Access
SEC	MCU security

14.4.11.1 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 effect after the next MCU reset.

14.4.11.1.1 Unsecuring the MCU Using Backdoor Key Access

The MCU 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 which 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 MCU. The entire 8-byte key cannot be all 0s or all 1s, i.e. 0x0000_0000_0000_0000 and 0xFFFF_FFFF_FFFF_FFFF 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 MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Verify Backdoor Access Key command
2. If the Verify Backdoor Access Key command is successful, the MCU 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 future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the MCU 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 MCU, the security state of the FTFE 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 MCU has full control of the contents of the Flash Configuration Field. The MCU 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.

14.5 Reset Sequence

On each system reset the FTFE module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FOPT, and FSEC registers and the FCNFG[SWAP, PFLSH, RAMRDY, EEERDY] bits.

CCIF is cleared throughout the reset sequence. The FTFE module holds off all CPU access for a portion of 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 FTFE 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 15
Clock Distribution

15.1 Introduction

This chapter presents the clock architecture overview of this device, the clock distribution and module clocks, and a clock terminology section. The clocking generation and configuration can be divided into 3 parts:

1. Clock sources generation
    - FIRC, SIRC, SOSC, LPFLL, all from the SCG module
    - LPO from PMC
2. Peripheral Clock Controller (PCC)
3. Module level clock control (Inside specific peripherals)

The System Clock Generator (SCG) module is used on this device for main system clock generation. It generates clock sources like Fast IRC (FIRC, 48 MHz, within 1% accuracy), Slow IRC (SIRC, 2/8 MHz, within 3% accuracy), System Oscillator (SOSC) and LPFLL. It controls which clock source is used to derive the system clocks. The SCG also divides the selected clock source into a variety of clock domains, including the clocks for the system bus masters, system bus slaves, and flash memory .

Besides the clocks generated by SCG, there are other clock generator: LPO from PMC.

Clock selection for most modules is controlled by the Peripheral Clock Controller (PCC) module. The PCC also implements module-specific clock gating to allow granular shutoff of modules.

Various modules have module-specific clocks that can be generated from the FIRC_CLK, SIRC_CLK, SOSC_CLK, FLL_CLK clock. In addition, there are various other module-specific clocks that have other alternate sources. While clock selection for most modules is controlled by the PCC module, some peripherals have clock source selection/divider inside the module, for details, please see the "Peripheral Clock Summary" table for more information.

15.2 High-level clocking diagram

The following diagram shows the high-level clocking architecture and various clock sources for this device.

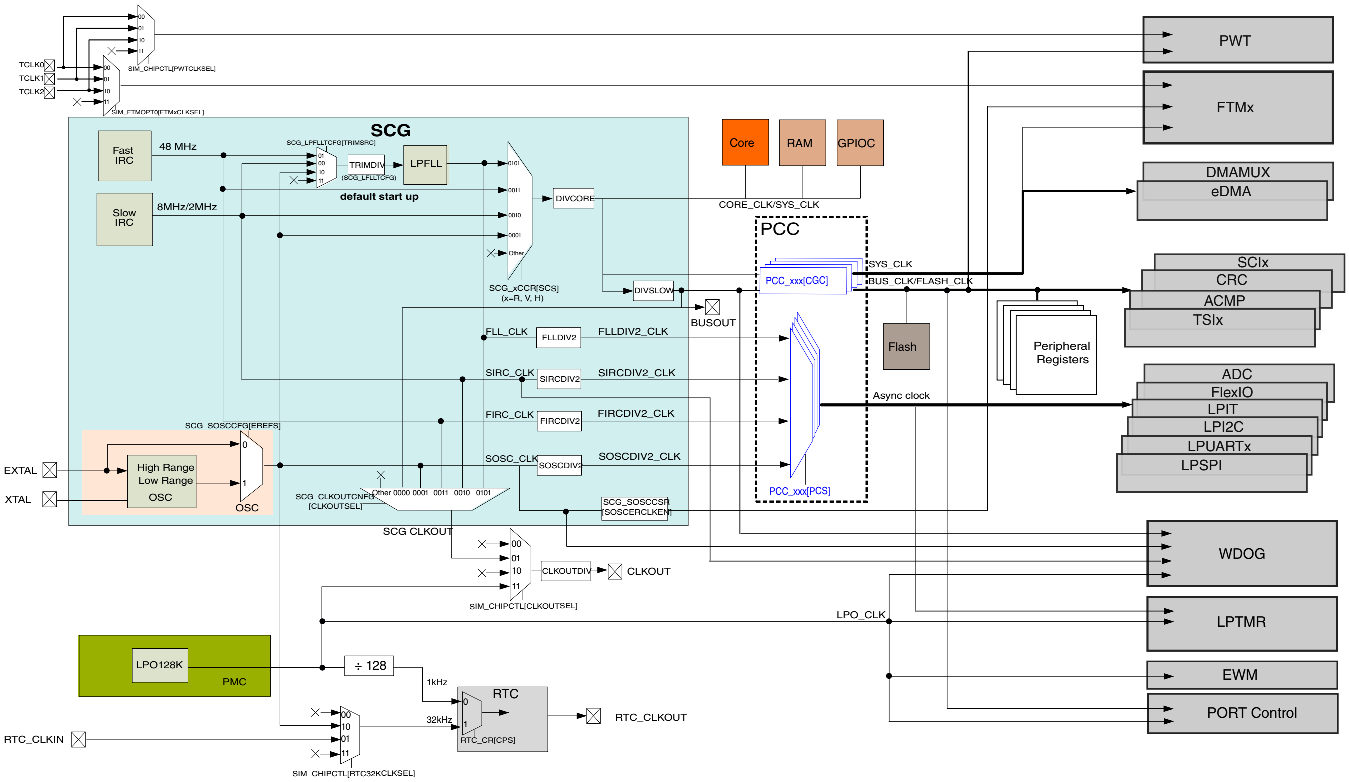


Figure 15-1. Clocking Diagram

15.3 Clock definitions

The following table describes the clocks in the previous block diagram and other sections of this document.

Clock name	Description
CORE_CLK	Clocks the ARM core, divided by DIVCORE bits inside SCG
SYS_CLK	Clocks the Crossbar, NVIC, Flash controller, FTM , etc. SYS_CLK can run up to CORE_CLK and divided by DIVCORE bits inside SCG.
BUS_CLK	Clocks the Peripherals, divided by DIVSLOW bits inside SCG
FLASH_CLK	Clocks the flash module, divided by DIVSLOW bits inside SCG

Table continues on the next page...

Clock name	Description
FLL_CLK	Optional divided FLL source for peripherals
SIRC_CLK	Optional divided SIRC source for peripherals
FIRC_CLK	Optional divided FIRC source for peripherals
SOSC_CLK[1]	Optional divided System Oscillator clock for peripherals. **NOTE:** SOSC_CLK/ERCLK/OSCERCLK stands for the same clock source, in some module chapters.
LPO_CLK	Always on low power oscillator clock inside PMC
RTC_CLKOUT	Clock output from RTC module for both internal and external
CLKOUT	Optional output clock source for external devices
BUSOUT	Optional output bus clock through pin for external devices or diagnostics

[1]
- For WDOG, its SOSC_CLK is with no dividers, and not gated by SCG_SOSCCSR[SOSCERCLKEN].
- For FTM, its SOSC_CLK is with no dividers, but gated by SCG_SOSCCSR[SOSCERCLKEN].
- For other peripherals (LPUART etc.), its SOSC_CLK is divided by DIVx, and not gated by SCG_SOSCCSR[SOSCERCLKEN].

15.4 Typical Clock Configuration

The clock dividers are programmed via the SCG module's clock divider registers. The following requirements must be met when configuring the clocks for this device:

The following are a few of the more common clock configurations for this device:

15.4.1 Default start-up clock

In default out of reset, the CPU is clocked from internal Fast IRC (IRC48M). The clocks, e.g. core clock and bus clock, are programmed via the SCG module. For the default reset value of divider, please refer to SCG chapter for details.

15.4.2 VLPR mode clocking

The clock dividers should not be changed while in VLPR mode. They must be programmed prior to entering VLPR mode to guarantee:

- the core/system clocks are less than or equal to 4 MHz
- the flash memory clock is less than or equal to 1 MHz

15.5 Clock Gating

The clock to most of the modules can be individually gated on and off using the PCC module. After any reset, PCC disables part of the clock to the corresponding module to conserve power. Prior to initializing a module, set the corresponding clock gating control bits in PCC register to enable the clock. Before turning off the clock, make sure to disable the module.

Any bus access to a peripheral that has its bus interface clock disabled (CGC=0 in PCC module) will generate a bus fault. While any bus access to a peripheral that has its functional clock disabled (PCS=0 in PCC module) will not return a fault, but the module cannot work properly.

NOTE

Changes to clock source should be done when clock is gated by PCC to avoid glitches to output clock.

15.6 Module clocks

The following table summarizes the clocks associated with each module.

Table 15-1. Peripheral Clock Summary

Module Name	Bus Interface Clock Gating	Peripheral Functional Clock		Max Frequency of Clock Source
	Gated by [CGC] bit of PCC	Clocks controlled by [PCS] bits of PCC [1]	Clocks controlled by registers inside module	
Communications				
LPUART0 – LPUART2	Yes	FIRC_CLK, SIRC_CLK, FLL_CLK, SOSC_CLK	-	Max: 96 MHz
SCI0 – SCI1	Yes	–	–	Max: BUS_CLK
LPSPI0 – LPSPI1	Yes	FIRC_CLK, SIRC_CLK, FLL_CLK, SOSC_CLK	-	Max: 96 MHz SCK clock Max: 24 MHz (Rx), 48 MHz (Tx)
LPI^{2}C0 – LPI^{2}C1	Yes	FIRC_CLK, SIRC_CLK, FLL_CLK, SOSC_CLK	-	Max: 60 MHz
FlexIO	Yes	FIRC_CLK, SIRC_CLK,	-	Max: 96 MHz

Table continues on the next page...

Table 15-1. Peripheral Clock Summary (continued)

Module Name	Bus Interface Clock Gating	Peripheral Functional Clock		Max Frequency of Clock Source
	Gated by [CGC] bit of PCC	Clocks controlled by [PCS] bits of PCC [1]	Clocks controlled by registers inside module	
		FLL_CLK, SOSC_CLK		
Timers				
LPTMR	Yes	FIRC_CLK, SIRC_CLK, FLL_CLK, SOSC_CLK	LPO_CLK	Max: 48 MHz LPO_CLK: 128kHz
LPIT	Yes	FIRC_CLK, SIRC_CLK, FLL_CLK, SOSC_CLK	-	Max: 96 MHz
RTC	Yes	-	LPO_CLK, RTC_CLKIN	Max: BUS_CLK LPO_CLK: 1 kHz
FlexTimer0 - FlexTimer2	Yes	-	SYS_CLK, SOSC_CLK, TCLKx	Max: SYS_CLK
PWT	Yes	-	BUS_CLK, TCLKx	Max: BUS_CLK
System Modules				
Watchdog	No	-	BUS_CLK, SIRC_CLK, LPO_CLK, SOSC_CLK	Max: BUS_CLK LPO_CLK: 128kHz
EWM	Yes	-	LPO_CLK	LPO_CLK: 128kHz
PMC	No	-	BUS_CLK, LPO_CLK	Max: BUS_CLK
RCM	No	-	BUS_CLK, LPO_CLK	Max: BUS_CLK LPO_CLK: 1kHz
Port Control	Yes	-	BUS_CLK, LPO_CLK	Max: BUS_CLK LPO_CLK: 128kHz
SIM	No	BUS_CLK		Max: BUS_CLK
CRC	Yes	BUS_CLK		Max: BUS_CLK
GPIOC	No	SYS_CLK		Max: SYS_CLK
DMA	Yes	SYS_CLK		Max: SYS_CLK
Memory Modules				
FTFE	Yes	FLASH_CLK		Max: FLASH_CLK
SYS RAM	No	SYS_CLK		Max: SYS_CLK
Analog Modules				
ADC0	Yes	FIRC_CLK, SIRC_CLK, FLL_CLK, SOSC_CLK	-	Max: 50 MHz
ACMP0	Yes	BUS_CLK		Max: BUS_CLK
TSI0–TSI1	Yes	BUS_CLK		Max: BUS_CLK

1. The clock sources undergo clock divider DIV*x* in SCG (output to PCC). For details, see the "High-Level clocking diagram" section in Clocking chapter and the "Chip-specific information" section in each module chapter.

15.6.1 LPO clock distribution

See the section "High-Level clocking diagram" for details.

15.6.2 EWM clocks

This table shows the EWM clocks and the corresponding chip clocks.

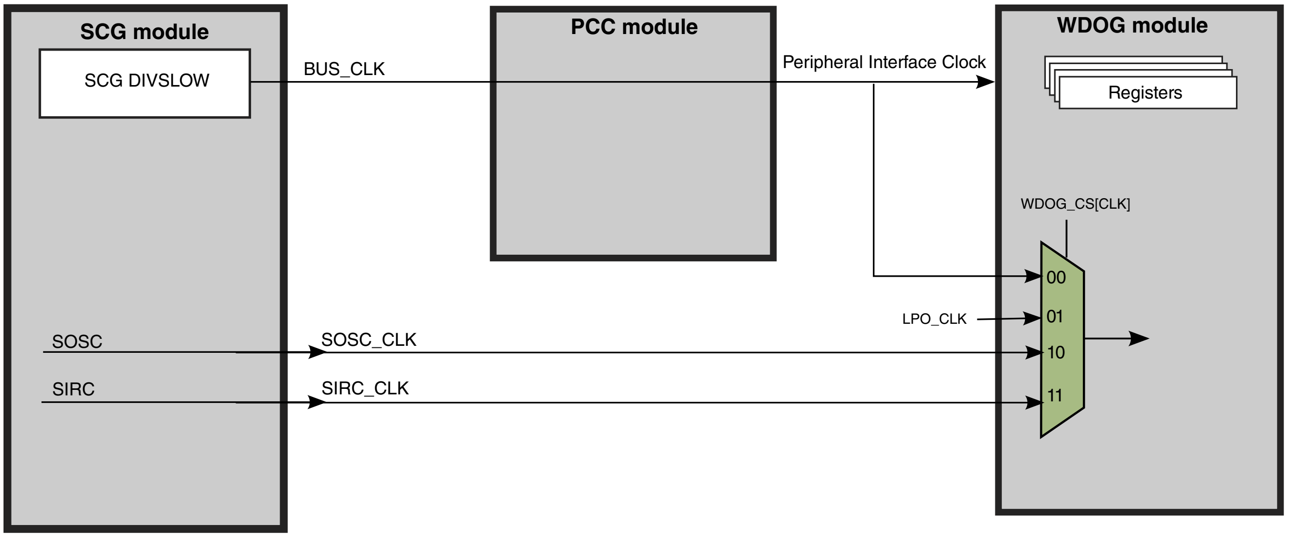
Table 15-2. EWM clock connections

Module clock	Chip clock
Low Power Clock	128 kHz LPO Clock (LPO_CLK)

15.6.3 WDOG Clocking Information

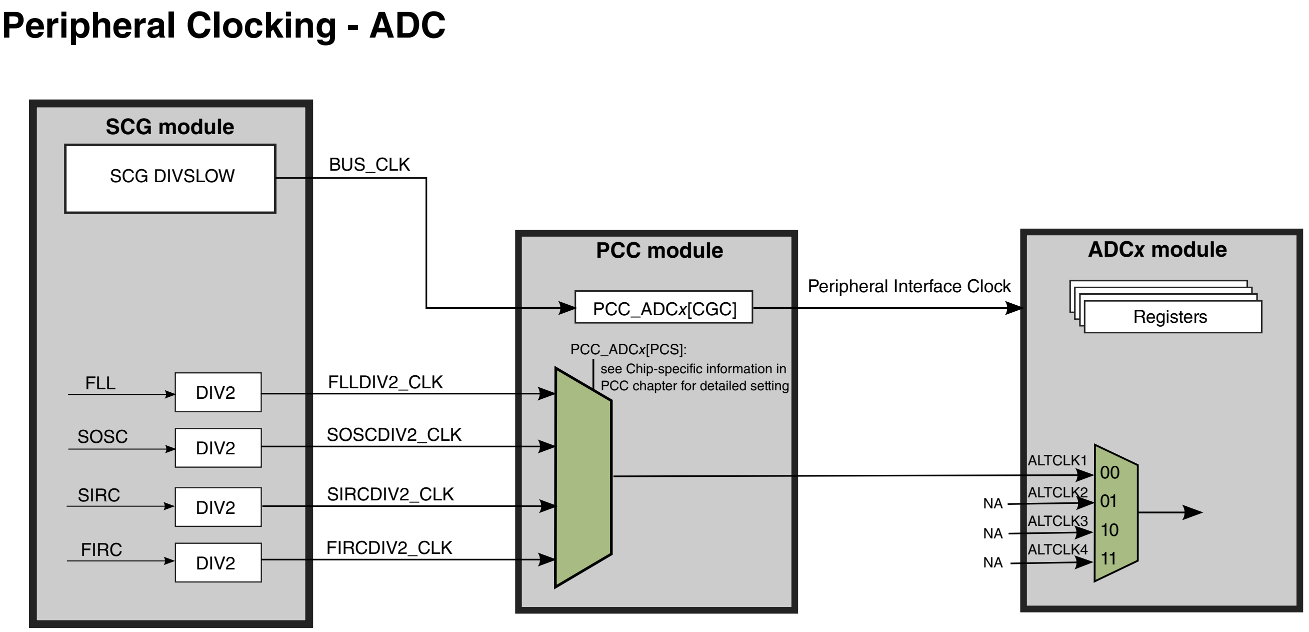
The following figure shows the input clock sources available for this module.

Peripheral Clocking - WDOG



15.6.4 ADC Clocking Information

The following figure shows the input clock sources available for this module.



NOTE

ALTCLK2~4 are not connected on this chip.

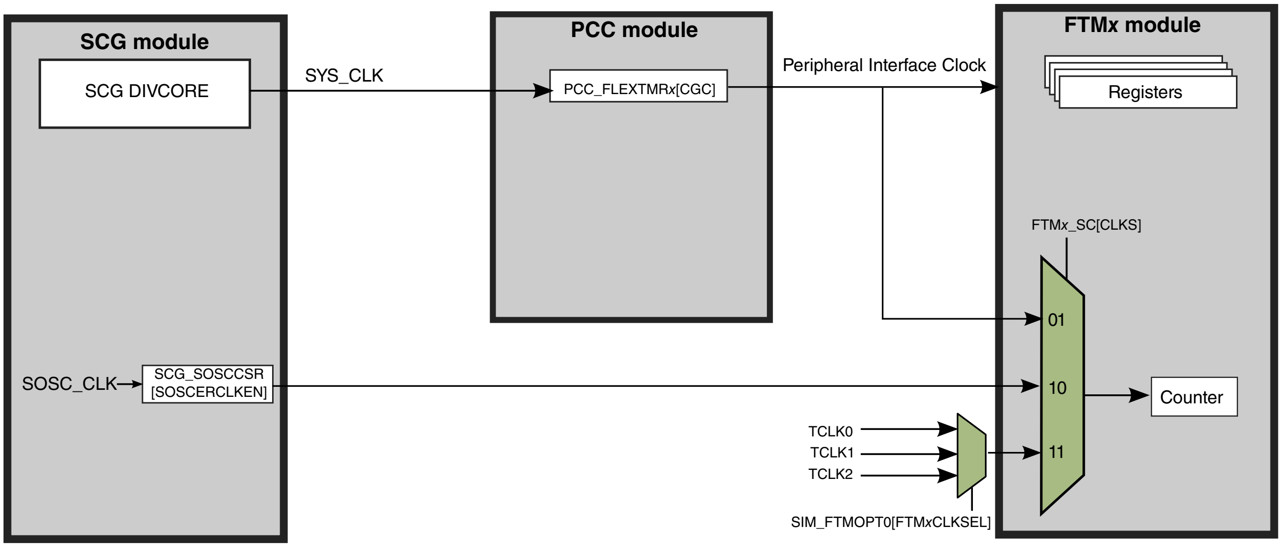
15.6.5 FTM Clocking Information

The following figure shows the input clock sources available for this module.

NOTE

It is recommended to clear the FTM channel (n) flag bits CHF right after writing a non-zero value to CLKS[1:0]. This procedure guarantees that the FTM will not capture spurious inputs edges in its input modes while CLKS[1:0] is 00b.

Peripheral Clocking - FTM



NOTE

Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM system clock frequency (SYS_CLK).

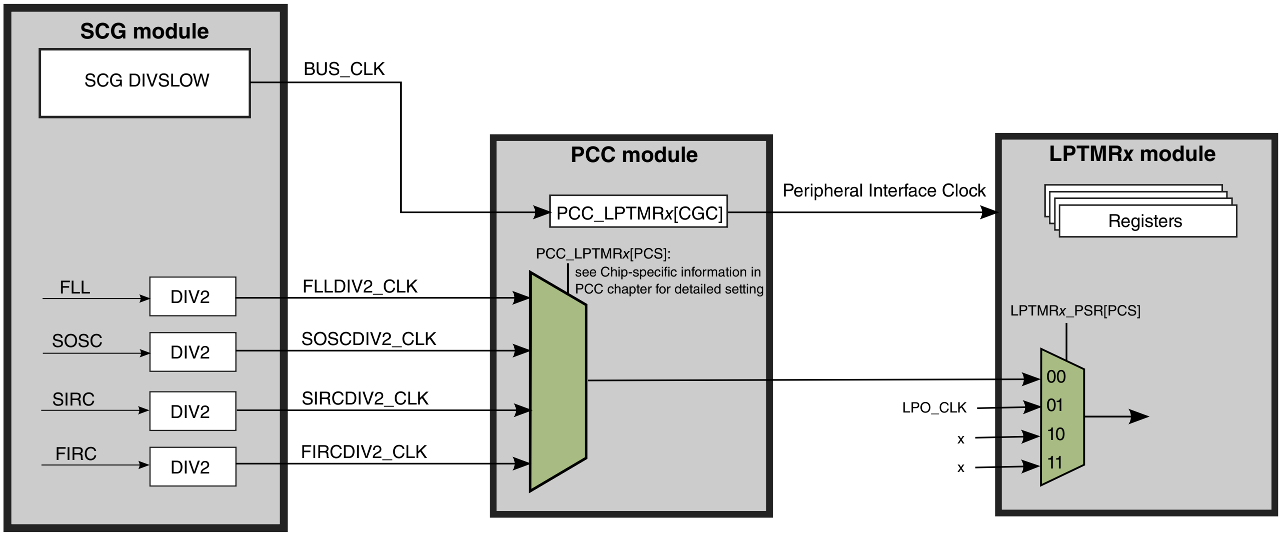
NOTE

The external clock are synchronized by FTM system clock (SYS_CLK). Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

15.6.6 LPTMR prescaler/glitch filter clocking options

The following figure shows the input clock sources available for this module.

Peripheral Clocking - LPTMR



NOTE

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

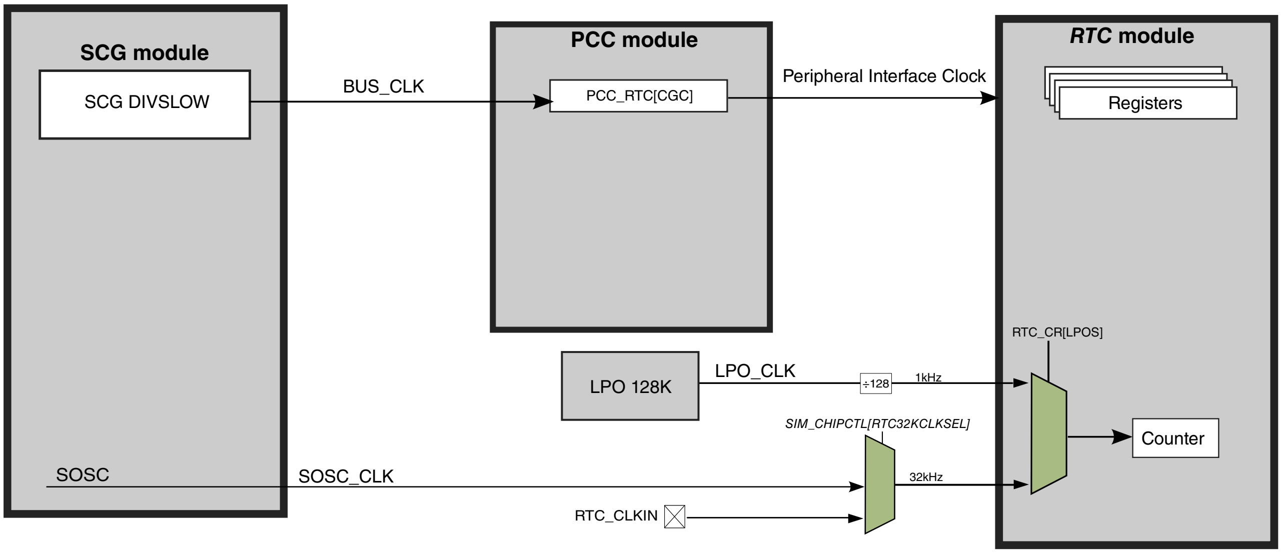
15.6.7 RTC Clocking Information

The following figure shows the input clock sources available for this module.

NOTE

No 32 kHz crystal in this device. See the clocking figure below, for more details about RTC clock source.

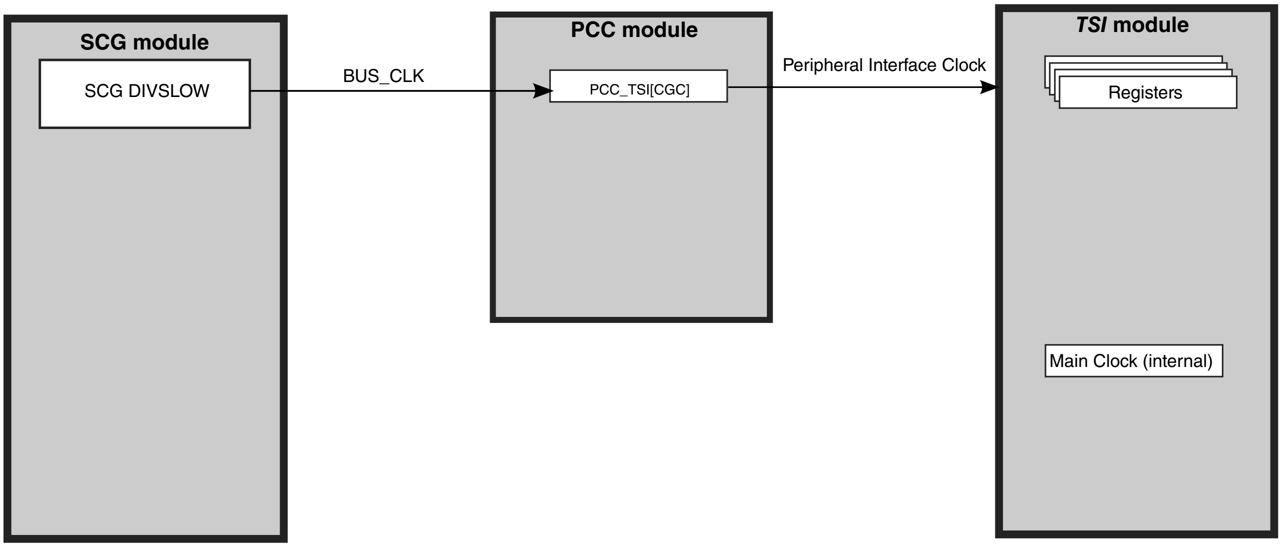
Peripheral Clocking - RTC



15.6.8 TSI Clocking Information

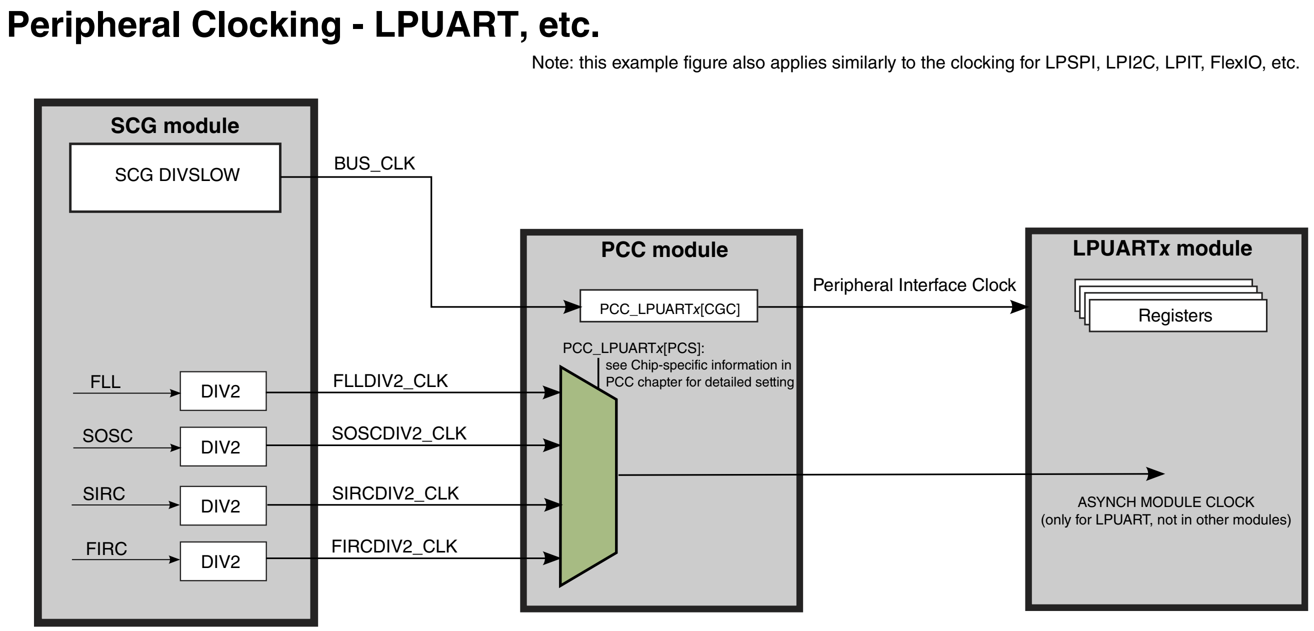
This following figure shows the TSI clocks.

Peripheral Clocking - TSI



15.6.9 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

The following figure shows the input clock sources available for this module.


Peripheral Clocking - LPUART, etc.
Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, LPIT, FlexIO, etc.
SCG module
SCG DIVSLOW
BUS_CLK
PCC module
PCC_LPUARTx[CGC]
Peripheral Interface Clock
LPUARTx module
Registers
PCC_LPUARTx[PCS]:
see Chip-specific information in PCC chapter for detailed setting
FLL
DIV2
FLLDIV2_CLK
SOSC
DIV2
SOSCDIV2_CLK
SIRC
DIV2
SIRCDIV2_CLK
FIRC
DIV2
FIRCDIV2_CLK
ASYNCH MODULE CLOCK
(only for LPUART, not in other modules)

Chapter 16
System Clock Generator (SCG)

16.1 Chip-specific information for this module

16.1.1 Instantiation Information

16.1.1.1 Information of SCG on this device

NOTE

For the clocking dividers, DIV1 is not used on this device, and DIV2 is used in SCG for peripheral clocking. See the "High-level clocking diagram" in the Clock Distribution chapter.

NOTE

FIRC is trimmed to 48 MHz only, in this device. Other values are reserved in SCG_FIRCCFG[RANGE].

Writing to SCG_FIRCSTAT register can cause hard fault when auto trim is disabled.

ERCLK (External Reference Clock) is either from an external pin or from the SCG internal OSC (SOSC), and configured with the SCG_SOSCCFG[EREFS] bit.

For the supported frequency range of OSC, see the "Oscillator electrical specifications" section in the Data Sheet.

16.1.1.1.1 SCG clock mode transitions

The following figure shows the valid clock mode transitions supported by SCG, for this device. For more information, see the Functional description section.

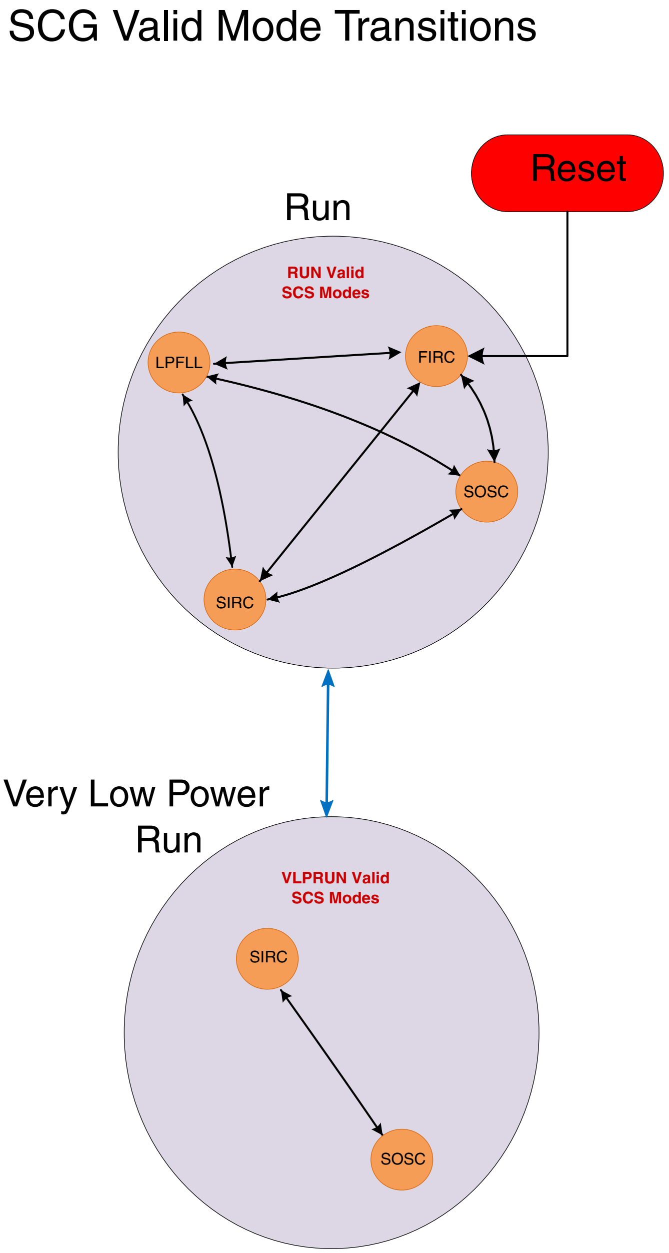


Figure 16-1. SCG Valid Mode Transition Diagram

16.2 Introduction

The system clock generator (SCG) module provides the system clocks for the MCU. The SCG takes clock inputs from a variety of sources and generates the types of clocks that the MCU requires.

- Available clock source inputs for the SCG include:
  - System oscillator clock (SOSC)

  - Slow internal reference clock (SIRC)
  - Fast internal reference clock (FIRC)

Clock dividers are available for:

- core clock (DIVCORE)
- peripheral interface clock (DIVSLOW)

16.2.1 Block diagram

See the Clock Distribution Chapter for more information.

NOTE

> To identify the oscillators used in your specific device, see the chip-specific SCG information or clocking chapters.

16.2.2 Features

Key features of the SCG module are:

- Low Power Frequency Locked-Loop (LPFLL):
  - Programmable multiplier for up to 4 different frequency ranges
  - Internal reference clocks or oscillators reference clocks can be used as the LPFLL source for trimming purposes
  - Can be selected as the clock source for the system clocks
- 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
  - Either the slow or the fast clock can be selected as the clock source for the system clocks
- System Crystal Oscillator:

  - Can be used as a source for the LPFLL
  - Can be selected as the clock source for the system clocks
- Clock monitor with reset and interrupt request capability for SOSC, clocks

16.3 Functional description

16.3.1 SCG Clock Mode Transitions

The following figure shows the valid clock mode transitions supported by SCG.

Slow IRC (SIRC) boot mode is not supported on this device.

Figure 16-2. SCG Valid Mode Transitions

The SCG will restrict programming into invalid clock modes and writes to System Clock Source (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 Run Clock Control Register (RCCR) and VLPR Clock Control Register (VCCR). 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 16-1. 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: One of the following scenarios is true • 0001 is written to VCCR[SCS] and VCCR[SCS] matches RCCR[SCS] • 0001 is written to VCCR[SCS], VCCR[SCS] differs from RCCR[SCS]) and 1 is written to SOSCCSR[SOSCLPEN] 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).

Table continues on the next page...

Table 16-1. SCG modes of operation (continued)

Mode	Description
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: One of the following scenarios is true • 0010 is written to VCCR[SCS] and VCCR[SCS] matches RCCR[SCS] • 0010 is written to VCCR[SCS], VCCR[SCS] differs from RCCR[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 the default clock mode of operation and is entered when all the following conditions occur: • RUN MODE: • 0011 is written to RCCR[SCS] VLRUN MODE: One of the following scenarios is true • 0011 is written to VCCR[SCS] and VCCR[SCS] matches RCCR[SCS] • 0011 is written to VCCR[SCS], VCCR[SCS] differs from RCCR[SCS]) and 1 is written to FIRCCSR[FIRCLPEN] 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. 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.
Low Power FLL (LPFLL)	Low Power FLL (LPFLL) mode is entered when all the following conditions occur: • RUN MODE: • 0101 is written to RCCR[SCS] VLRUN MODE: • Invalid mode. Programming SCG into LPFLL mode will be ingored HSRUN MODE: • 0101 is written to HCCR[SCS] • LPFLLEN = 1 • LPFLLVLD = 1 In LPFLL mode, SCSCLKOUT and system clocks are derived from the Low Power FLL (LPFLL). By default the LPFLL will be running in open-loop mode using default trim values. In closed-loop mode (LPFLLTREN=1 and LPFLLTRUP=1) LPFLL will be auto trimmed using a selectable reference clock as specified by its corresponding SCG_LPFLLTCFG[TRIMSRC]. The LPFLL will lock its frequency to the LPFLL factor, as specified by the SCG_LPFLLCFG[FSEL].

Table continues on the next page...

Table 16-1. 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, including SCG system clocks (DIVCORE,DIVSLOW). There are exceptions; the following clocks can continue to run and stay enabled in the following cases: SIRC is available in Normal Stop and VLPS modes when all the following conditions become true: • SIRCCSR[SIRCEN] = 1 • SIRCCSR[SIRCSTEN] = 1 • SIRCCSR[SIRCLPEN] = 1 in VLPS FIRC is available in Normal Stop and VLPS modes when all the following conditions become true: • FIRCCSR[FIRCEN] = 1 • FIRCCSR[FIRCSTEN] = 1 • FIRCCSR[FIRCLPEN] = 1 in VLPS SOSC is available in following low power stop modes (Normal Stop, VLPS) when all the below conditions are true. • SOSCCSR[SOSCEN] = 1 • SOSCCSR[SOSCSTEN] = 1 • SOSCCSR[SOSCLPEN] = 1 (required only for Low Power Stop modes (VLPS)

16.3.2 Clocks

The following table describes the clocks of SCG.

Table 16-2. Clocks and their description

Clock	Type	Description
Slow Bus Clock	Input	SCG register read/write clock source.
System Oscillating Clock (SOSC)	Input	Reference clock used to generate system clocks when RCCR[SCS]=0001.
Slow Internal Reference Clock (SIRC)	Output	Reference clock used to generate system clocks when RCCR[SCS]=0010.
Fast Internal Reference Clock (FIRC)	Output	Reference clock used to generate system clocks when RCCR[SCS]=0011.
Real Time Clock Oscillator (RTCOSC)	Output	Reference clock used to generate system clocks when RCCR[SCS]=0100.
Low Power FLL (LPFLL)	Output	Reference clock used to generate system clocks when RCCR[SCS]=0101.
Sys PLL (SPLL)	Output	Reference clock used to generate system clocks when RCCR[SCS]=0110).

16.3.3 Interrupts/Resets

SCG includes several sources for interrupt requests. The following table summarizes these sources.

Each of these sources includes a mask control field (such as SOSCCSR[SOSCCMRE], LPFLLCSR[LPFLLCMRE]])

Table 16-3. Interrupts

Flag	Functionality	Can generate interrupt?	Can generate reset?
SOSCCSR[SOSCERR]	Indicates whether SOSC source clock error has been detected either because of a loss of clock or clock has gone invalid.	Yes, if SOSCCSR[SOSCCMRE] = 0	Yes, if SOSCCSR[SOSCCMRE] = 1
LPFLLCSR[LPFLLCMRE]	Indicates whether LPFLL source clock error has been detected either because of a loss of lock or loss of clock has been detected.	Yes, if LPFLLCSR[LPFLLCMRE] = 0	Yes, if LPFLLCSR[LPFLLCMRE] = 1

16.4 External signals

SCG has no external signals.

16.5 Initialization

SCG is enabled by default and its default clock mode will be FIRC clock mode. SCG Modes of operation table describes how to configure SCG into a different clocking mode.

16.6 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.

16.6.1 SCG register descriptions

- For any writeable SCG registers, only 32-bit writes are allowed. 8-bit or 16-bit writes will result in transfer errors.

16.6.1.1 SCG memory map

SCG base address: 4006_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	R	0100_0000h
4h	Parameter Register (PARAM)	32	R	8800_002Eh
10h	Clock Status Register (CSR)	32	R	Table 16-3
14h	Run Clock Control Register (RCCR)	32	RW	Table 16-3
18h	VLPR Clock Control Register (VCCR)	32	RW	Table 16-3
1Ch	HSRUN Clock Control Register (HCCR)	32	RW	0300_0001h
20h	SCG CLKOUT Configuration Register (CLKOUTCNFG)	32	RW	0300_0000h
100h	System OSC Control Status Register (SOSCCSR)	32	RW	Table 16-3
104h	System OSC Divide Register (SOSCDIV)	32	RW	0000_0000h
108h	System Oscillator Configuration Register (SOSCCFG)	32	RW	0000_0010h
200h	Slow IRC Control Status Register (SIRCCSR)	32	RW	0000_0005h
204h	Slow IRC Divide Register (SIRCDIV)	32	RW	0000_0000h
208h	Slow IRC Configuration Register (SIRCCFG)	32	RW	0000_0001h
300h	Fast IRC Control Status Register (FIRCCSR)	32	RW	0300_0001h
304h	Fast IRC Divide Register (FIRCDIV)	32	RW	0000_0000h
308h	Fast IRC Configuration Register (FIRCCFG)	32	RW	0000_0000h
30Ch	Fast IRC Trim Configuration Register (FIRCTCFG)	32	RW	0000_0000h
318h	Fast IRC Status Register (FIRCSTAT)	32	RW	Table 16-3
500h	Low Power FLL Control Status Register (LPFLLCSR)	32	RW	0000_0000h
504h	Low Power FLL Divide Register (LPFLLDIV)	32	RW	0000_0000h
508h	Low Power FLL Configuration Register (LPFLLCFG)	32	RW	0000_0000h
50Ch	Low Power FLL Trim Configuration Register (LPFLLTCFG)	32	RW	0000_0000h
514h	Low Power FLL Status Register (LPFLLSTAT)	32	RW	Table 16-3

16.6.1.2 Version ID Register (VERID)

16.6.1.2.1 Offset

Register	Offset
VERID	0h

16.6.1.2.2 Function

Note: Writing to this register will result in a transfer error.

16.6.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	1	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	VERSION															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.2.4 Fields

Field	Function
31-0 VERSION	SCG Version Number

16.6.1.3 Parameter Register (PARAM)

16.6.1.3.1 Offset

Register	Offset
PARAM	4h

16.6.1.3.2 Function

Note: Writing to this register will result in a transfer error.

16.6.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DIVPRES					0										
W																
Reset	1[1]	0	0	0	1	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								CLKPRES							
W																
Reset	0	0	0	0	0	0	0	0	0[2]	0	1	0	1	1	1	0

1. The reset value is controlled by which SCG System Dividers are used by the SoC.
2. The reset value is controlled by which SCG Clock Sources are used by the SoC. Please reference the Reference manual clocking chapter.

16.6.1.3.4 Fields

Field	Function
31-27 DIVPRES	Divider Present Indicates which system clock dividers are present in this instance of SCG. 1_xxxxb - System DIVCORE is present. x_xxx1b - System DIVSLOW is present.
26-16 —	Reserved
15-8 —	Reserved
7-0 CLKPRES	Clock Present 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. 0000_0000b-0000_0001b - Reserved xx1x_xxxxb - Low Power FLL (LPFLL) is present. xxxx_1xxxb - Fast IRC (FIRC) is present. xxxx_x1xxb - Slow IRC (SIRC) is present. xxxx_xx1xb - System OSC (SOSC) is present.

16.6.1.4 Clock Status Register (CSR)

16.6.1.4.1 Offset

Register	Offset
CSR	10h

16.6.1.4.2 Function

The Clock Status Register (CSR) returns the currently configured system clock source and the system clock dividers for the core (DIVCORE) and peripheral interface clock (DIVSLOW). The SCG_CSR reports the configuration set by one of the clock control registers SCG_RCCR, SCG_VCCR, SCG_HCCR.

Note: Writing to the Clock Status Register (CSR) will result in a transfer error.

16.6.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

1. 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

16.6.1.4.4 Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Returns the currently configured clock source generating the system clock. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Low Power FLL (LPFLL_CLK) 0110b - Reserved 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2

Table continues on the next page...

Field	Function
	0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Reserved 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

16.6.1.5 Run Clock Control Register (RCCR)

16.6.1.5.1 Offset

Register	Offset
RCCR	14h

16.6.1.5.2 Function

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 a new clock source is valid.

NOTE

> Switching to new system clock source and system clock dividers must be done after previous RCCR changes have been completed and the Clock Status Register has updated to match the present RCCR setting.

16.6.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					SCS			0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

[1] 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

16.6.1.5.4 Fields

Field	Function
31-27 —	Reserved
26-24 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. **NOTE:** Programming SCS to a different value should only be done after the previous SCS clock switch has finished. 000b - Reserved 001b - System OSC (SOSC_CLK) 010b - Slow IRC (SIRC_CLK)

Table continues on the next page...

Field	Function
	011b - Fast IRC (FIRC_CLK) 100b - Reserved 101b - Low Power FLL (LPFLL_CLK) 110b - Reserved 111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Reserved 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

16.6.1.6 VLPR Clock Control Register (VCCR)

16.6.1.6.1 Offset

Register	Offset
VCCR	18h

16.6.1.6.2 Function

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.

NOTE

Switching to a new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

16.6.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W	▨								▨							
Reset	0	0	0	0	0	0	1	0	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				DIVSLOW			
W	▨				▨				▨							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

1. 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.

16.6.1.6.4 Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Field	Function
—	
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. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Reserved 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12

Field	Function
	1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

16.6.1.7 HSRUN Clock Control Register (HCCR)

16.6.1.7.1 Offset

Register	Offset
HCCR	1Ch

16.6.1.7.2 Function

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.

NOTE

Switching to a new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

16.6.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	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				0				0				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

16.6.1.7.4 Fields

Field	Function
31-28 —	Reserved
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. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Low Power FLL (LPFLL_CLK) 0110b - Reserved 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Reserved 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7

Field	Function
	0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

16.6.1.8 SCG CLKOUT Configuration Register (CLKOUTCNFG)

16.6.1.8.1 Offset

Register	Offset
CLKOUTCNFG	20h

16.6.1.8.2 Function

This register controls which SCG clock source is selected to be ported out to the CLKOUT pin.

16.6.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CLKOUTSEL				0							
W																
Reset	0	0	0	0	0	0	1	1	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															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.8.4 Fields

Field	Function
31-28	Reserved

Table continues on the next page...

Field	Function
—	
27-24 CLKOUTSEL	SCG Clkout Select Selects the SCG system clock. 0000b - SCG SLOW Clock 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Low Power FLL (LPFLL_CLK) 0110b - Reserved 0111b - Reserved 1111b - Reserved
23-0 —	Reserved

16.6.1.9 System OSC Control Status Register (SOSCCSR)

16.6.1.9.1 Offset

Register	Offset
SOSCCSR	100h

16.6.1.9.2 Function

Contains System OSC Clock Error, System OSC Selected, System OSC Valid, Lock Register, System OSC Clock Monitor Reset Enable, System OSC Clock Monitor Enable , System OSC 3V ERCLK Enable, System OSC Low Power Enable, System OSC Stop Enable, System OSC Enable.

16.6.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					SOSCERR	SOSCSEL	SOSCVLD	LK	0					SOSCCMRE	SOSCCM
W						W1C										
Reset	0	0	0	0	0	u[1]	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												SOSCERCLKEN	SOSCLPEN	SOSCSTEN	SOSCEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

[1] This flag is reset on Chip POR only

16.6.1.9.4 Fields

Field	Function
31-27 —	Reserved
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. 0b - System OSC Clock Monitor is disabled or has not detected an error 1b - System OSC Clock Monitor is enabled and detected an error
25 SOSCSEL	System OSC Selected 0b - System OSC is not the system clock source 1b - System OSC is the system clock source
24 SOSCVLD	System OSC Valid The SOSC is considered valid after 4096 xtal counts. 0b - System OSC is not enabled or clock is not valid 1b - System OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - This Control Status Register can be written. 1b - This Control Status Register cannot be written.
22-18 —	Reserved
17	System OSC Clock Monitor Reset Enable

Table continues on the next page...

Field	Function
SOSCCMRE	0b - Clock Monitor generates interrupt when error detected 1b - Clock Monitor generates reset when error detected
16 SOSCCM	System OSC Clock Monitor Enable Enables the clock monitor when SOSCVLD is set. **NOTE:** SOSC clock monitor remains enabled in VLPR and VLPS if SOSCCM is enabled. 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. **NOTE:** The reference clock used to monitor the SOSC is the SIRC. This clock must be programmed to be enabled in order to monitor the SOSC. SIRC is automatically disabled in LLS and VLLS power modes and clock monitor of SOSC will be disabled. 0b - System OSC Clock Monitor is disabled 1b - System OSC Clock Monitor is enabled
15-4 —	Reserved
3 SOSCERCLKEN	System OSC 3V ERCLK Enable SOSCERCLKEN is required for stop modes. 0b - System OSC 3V ERCLK output clock is disabled. 1b - 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. 0b - System OSC is disabled in VLP modes 1b - System OSC is enabled in VLP modes
1 SOSCSTEN	System OSC Stop Enable 0b - System OSC is disabled in Stop modes 1b - System OSC is enabled in Stop modes if SOSCEN=1.
0 SOSCEN	System OSC Enable 0b - System OSC is disabled 1b - System OSC is enabled

16.6.1.10 System OSC Divide Register (SOSCDIV)

16.6.1.10.1 Offset

Register	Offset
SOSCDIV	104h

16.6.1.10.2 Function

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.

16.6.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													Reserved		
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					SOSCDIV2			0					Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.10.4 Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15-11 —	Reserved
10-8 SOSCDIV2	System OSC Clock Divide 2 Clock divider 2 for System OSC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.

16.6.1.11 System Oscillator Configuration Register (SOSCCFG)

16.6.1.11.1 Offset

Register	Offset
SOSCCFG	108h

16.6.1.11.2 Function

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.

16.6.1.11.3 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				0				0		RANGE		HGO	EREFS	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

16.6.1.11.4 Fields

Field	Function
31-12 —	Reserved
11-8 —	Reserved
7-6 —	Reserved
5-4 RANGE	System OSC Range Select Selects the frequency range for the system crystal oscillator (OSC) **NOTE:** See the device data sheet for the crystal frequencies supported on this device.
3 HGO	High Gain Oscillator Select Controls the crystal oscillator power mode of operations.

Table continues on the next page...

Field	Function
	0b - Configure crystal oscillator for low-power operation 1b - 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 0b - External reference clock selected 1b - Internal crystal oscillator of OSC requested.
1-0 —	Reserved

16.6.1.12 Slow IRC Control Status Register (SIRCCSR)

16.6.1.12.1 Offset

Register	Offset
SIRCCSR	200h

16.6.1.12.2 Function

Contains Slow IRC Selected, Slow IRC Valid, Lock Register , Slow IRC Low Power Enable, Slow IRC Stop Enable, Slow IRC Enable.

NOTE

SIRCVLD is reset only by POR.

16.6.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						SIRCSEL	SIRCVLD	LK	Reserved						
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												0	SIRCLPEN	SIRCSTEN	SIRCEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

16.6.1.12.4 Fields

Field	Function
31-26 —	Reserved
25 SIRCSEL	Slow IRC Selected 0b - Slow IRC is not the system clock source 1b - Slow IRC is the system clock source
24 SIRCVLD	Slow IRC Valid 0b - Slow IRC is not enabled or clock is not valid 1b - Slow IRC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. The purpose for this bitfield is to prevent runaway code from changing SIRC clock configurations. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-4 —	This field is reserved and is always has the value 0
3 —	This field is reserved and is always has the value 0
2 SIRCLPEN	Slow IRC Low Power Enable 0b - Slow IRC is disabled in VLP modes 1b - Slow IRC is enabled in VLP modes
1 SIRCSTEN	Slow IRC Stop Enable 0b - Slow IRC is disabled in Stop modes 1b - Slow IRC is enabled in Stop modes
0 SIRCEN	Slow IRC Enable 0b - Slow IRC is disabled 1b - Slow IRC is enabled

16.6.1.13 Slow IRC Divide Register (SIRCDIV)

16.6.1.13.1 Offset

Register	Offset
SIRCDIV	204h

16.6.1.13.2 Function

To prevent glitches to the output divided clock, change SIRDIV when the Slow IRC is disabled.

16.6.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													Reserved		
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					SIRCDIV2			0					Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.13.4 Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15-11 —	Reserved
10-8 SIRCDIV2	Slow IRC Clock Divide 2 Clock divider 2 for Slow IRC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1

Table continues on the next page...

Field	Function
	010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.

16.6.1.14 Slow IRC Configuration Register (SIRCCFG)

16.6.1.14.1 Offset

Register	Offset
SIRCCFG	208h

16.6.1.14.2 Function

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.

16.6.1.14.3 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															RANGE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

16.6.1.14.4 Fields

Field	Function
31-1 —	Reserved
0 RANGE	Frequency Range 0b - Slow IRC low range clock (2 MHz) 1b - Slow IRC high range clock (8 MHz)

16.6.1.15 Fast IRC Control Status Register (FIRCCSR)

16.6.1.15.1 Offset

Register	Offset
FIRCCSR	300h

16.6.1.15.2 Function

Contains Fast IRC status bits.

16.6.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					FIRCERR	FIRCSEL	FIRCVLD	LK	0						
W						W1C										
Reset	0	0	0	0	0	0	1	1	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		0		0	0	FIRCTRUP	FIRCTREN	0				FIRCREGOFF	FIRCLPEN	FIRCSTEN	FIRCEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

16.6.1.15.4 Fields

Field	Function
31-27 —	Reserved
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 0b - Error not detected with the Fast IRC trimming. 1b - Error detected with the Fast IRC trimming.
25 FIRCSEL	Fast IRC Selected status 0b - Fast IRC is not the system clock source 1b - Fast IRC is the system clock source
24 FIRCVLD	Fast IRC Valid status 0b - Fast IRC is not enabled or clock is not valid. 1b - Fast IRC is enabled and output clock is valid. The clock is valid after there is an output clock from the FIRC analog.
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-14 —	Reserved
13-12 —	Reserved
11 —	Reserved
10 —	Reserved
9 FIRCTRUP	Fast IRC Trim Update 0b - Disable Fast IRC trimming updates 1b - Enable Fast IRC trimming updates
8 FIRCTREN	Fast IRC Trim Enable 0b - Disable trimming Fast IRC to an external clock source 1b - Enable trimming Fast IRC to an external clock source
7-4 —	Reserved
3 FIRCREGOFF	Fast IRC Regulator Enable **NOTE:** When Fast IRC is used, FIRCREGOFF must be 0. Fast IRC cannot be operated with FIRCREGOFF=1. 0b - Fast IRC Regulator is enabled. 1b - Fast IRC Regulator is disabled.
2 FIRCLPEN	Fast IRC Low Power Enable 0b - Fast IRC is disabled in VLP modes 1b - Fast IRC is enabled in VLP modes
1 FIRCSTEN	Fast IRC Stop Enable 0b - Fast IRC is disabled in Stop modes. 1b - Fast IRC is enabled in Stop modes

Table continues on the next page...

Field	Function
0 FIRCEN	Fast IRC Enable 0b - Fast IRC is disabled 1b - Fast IRC is enabled

16.6.1.16 Fast IRC Divide Register (FIRCDIV)

16.6.1.16.1 Offset

Register	Offset
FIRCDIV	304h

16.6.1.16.2 Function

Changes to FIRCDIV should be done when FAST IRC is disabled to prevent glitches to output divided clock.

16.6.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													Reserved		
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					FIRCDIV2			0					Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.16.4 Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15-11	Reserved

Table continues on the next page...

Field	Function
—	
10-8 FIRCDIV2	Fast IRC Clock Divide 2 Clock divider 2 for the Fast IRC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.

16.6.1.17 Fast IRC Configuration Register (FIRCCFG)

16.6.1.17.1 Offset

Register	Offset
FIRCCFG	308h

16.6.1.17.2 Function

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.

16.6.1.17.3 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														RANGE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.17.4 Fields

Field	Function
31-2 —	Reserved
1-0 RANGE	Frequency Range 00b - Fast IRC is trimmed to 48 MHz 01b - Fast IRC is trimmed to 52 MHz 10b - Fast IRC is trimmed to 56 MHz 11b - Fast IRC is trimmed to 60 MHz

16.6.1.18 Fast IRC Trim Configuration Register (FIRCTCFG)

16.6.1.18.1 Offset

Register	Offset
FIRCTCFG	30Ch

16.6.1.18.2 Function

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.

16.6.1.18.3 Diagram

Bits	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

Bits	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

16.6.1.18.4 Fields

Field	Function
31-27 —	Reserved
26-16 —	Reserved
15-11 —	Reserved
10-8 TRIMDIV	Fast IRC Trim Predivide Divide the System OSC down for Fast IRC trimming. 000b - Divide by 1 001b - Divide by 128 010b - Divide by 256 011b - Divide by 512 100b - Divide by 1024 101b - Divide by 2048 110b - Reserved. Writing this value will result in Divide by 1. 111b - Reserved. Writing this value will result in a Divide by 1.
7-2 —	Reserved
1-0 TRIMSRC	Trim Source Configures the external clock source to tune the Fast IRC. TRIMSRC must be configured before programming FIRCSTAT register for trim update 00b - Reserved 01b - Reserved 10b - System OSC. This option requires that SOSC be divided using the TRIMDIV field to get a frequency slower than 32kHz. 11b - Reserved

16.6.1.19 Fast IRC Status Register (FIRCSTAT)

16.6.1.19.1 Offset

Register	Offset
FIRCSTAT	318h

16.6.1.19.2 Function

This register is loaded from IFR during reset. This register is 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 are allowed and values written to this register are used to trim FIRC clock.

16.6.1.19.3 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		TRIMCOAR						0	TRIMFINE						
W																
Reset	0	0	u[1]	u	u	u	u	u	0	u[1]	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

16.6.1.19.4 Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved
13-8 TRIMCOAR	Trim Coarse TRIMCOAR bits are used to coarsely 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
6-0 TRIMFINE	Trim Fine 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.

16.6.1.20 Low Power FLL Control Status Register (LPFLLCSR)

16.6.1.20.1 Offset

Register	Offset
LPFLLCSR	500h

16.6.1.20.2 Function

Contains LPFLL Clock Error, LPFLL Selected, LPFLL Valid, Lock Register, LPFLL Clock Monitor Reset Enable, LPFLL Clock Monitor, LPFLL Trim LOCK, LPFLL Trim Update, LPFLL Trim Enable, LPFLL Stop Enable, LPFLL Enable.

16.6.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					LPFLLERR	LPFLLSEL	LPFLLVLD	LK	0					LPFLLCMRE	LPFLLCM
W						W1C										
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					LPFLLTRMLOCK	LPFLLTRUP	LPFLLTREN	0						0	LPFLLEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.20.4 Fields

Field	Function
31-27 —	Reserved
26 LPFLLERR	LPFLL Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one

Table continues on the next page...

Field	Function
	When LPFLLTREN=1 and LPFLLTRUP=1, LPFLLERR=1 if the LPFLL can't lock the reference clock. This occurs when the reference clock is too fast/slow or LPFLL clock is stopped. LPFLLERR indicates a loss of lock or loss of clock. To change the reference clock frequency to re-lock, the LPFLLTREN or LPFLLTRUP bits must also be re-enabled (LPFLLTREN=1 or LPFLLTRUP=1). 0b - Error not detected with the LPFLL trimming. 1b - Error detected with the LPFLL trimming.
25 LPFLLSEL	LPFLL Selected 0b - LPFLL is not the system clock source 1b - LPFLL is the system clock source
24 LPFLLVLD	LPFLL Valid 0b - LPFLL is not enabled or clock is not valid. 1b - LPFLL is enabled and output clock is valid.
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-18 —	Reserved
17 LPFLLCMRE	LPFLL Clock Monitor Reset Enable 0b - Clock Monitor generates interrupt when error detected 1b - Clock Monitor generates reset when error detected
16 LPFLLCM	LPFLL Clock Monitor Enables the clock monitor when LPFLLTREN is set and LPFLL is enabled. The clock monitor is always disabled in low power 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. 0b - LPFLL Clock Monitor is disabled 1b - LPFLL Clock Monitor is enabled
15-11 —	Reserved
10 LPFLLTRMLOCK	LPFLL Trim LOCK Asserts only when LPFLLTREN=1 and LPFLLTRUP=1 and LPFLL has locked to target frequency. In open-loop mode (LPFLLTRUP=0), lock conditions cannot be checked. 0b - LPFLL not locked 1b - LPFLL trimmed and locked
9 LPFLLTRUP	LPFLL Trim Update 0b - Disable LPFLL trimming updates. LPFLL frequency determined by AUTOTRIM written value. 1b - Enable LPFLL trimming updates. LPFLL frequency determined by reference clock multiplication
8 LPFLLTREN	LPFLL Trim Enable 0b - Disable trimming LPFLL to an reference clock source 1b - Enable trimming LPFLL to an reference clock source
7-2 —	Reserved
1 —	Reserved

Table continues on the next page...

Field	Function
0 LPFLLEN	LPFLL Enable 0b - LPFLL is disabled 1b - LPFLL is enabled

16.6.1.21 Low Power FLL Divide Register (LPFLLDIV)

16.6.1.21.1 Offset

Register	Offset
LPFLLDIV	504h

16.6.1.21.2 Function

Changes to LPFLLDIV should be done when LPFLL is disabled to prevent glitches to output divided clock.

16.6.1.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													Reserved		
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					LPFLLDIV2			0					Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.21.4 Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.
15-11	Reserved

Table continues on the next page...

Field	Function
—	
10-8 LPFLLDIV2	LPFLL Clock Divide 2 Clock divider 2 for the LPFLL. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 —	Reserved This bit field is reserved. Software should write 0 to this bit field to maintain compatibility.

16.6.1.22 Low Power FLL Configuration Register (LPFLLCFG)

16.6.1.22.1 Offset

Register	Offset
LPFLLCFG	508h

16.6.1.22.2 Function

The LPFLLCFG register cannot be changed when the LPFLL is enabled. When the LPFLL is enabled, writes to this register are ignored, and there is no transfer error.

16.6.1.22.3 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														FSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.22.4 Fields

Field	Function
31-2 —	Reserved
1-0 FSEL	Frequency Select 00b - LPFLL is trimmed to 48 MHz. 01b - LPFLL is trimmed to 72 MHz. 10b - LPFLL is trimmed to 96 MHz 11b - Reserved

16.6.1.23 Low Power FLL Trim Configuration Register (LPFLLTCFG)

16.6.1.23.1 Offset

Register	Offset
LPFLLTCFG	50Ch

16.6.1.23.2 Function

The LPFLLTCFG register cannot be changed when LPFLL tuning is enabled. When the LPFLL tuning is enabled, writes to this register are ignored, and there is no transfer error.

16.6.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															LOCKW2LSB
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			TRIMDIV					0						TRIMSRC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16.6.1.23.4 Fields

Field	Function
31-17 —	Reserved
16 LOCKW2LSB	Lock LPFLL with 2 LSBS This bitfield is used to control the condition to set LPFLLTRMLOCK: difference between LPFLL actual clock and target clock (48 MHz, 72 MHz, 96 MHz) is within 0.8% or 0.4%; 0b - LPFLL locks within 1LSB (0.4%) 1b - LPFLL locks within 2LSB (0.8%)
15-13 —	Reserved
12-8 TRIMDIV	LPFLL Trim Predivide Use to divide the reference clock down for LPFLL trimming by 1,2,4,8,....31,32. The divided frequency should be either 32.768 KHz or 2 MHz. 00000 Divide by 1 00001 Divide by 2 00010 Divide by 3 .... 11110 Divide by 31 11111 Divide by 32
7-2 —	Reserved
1-0 TRIMSRC	Trim Source Configures the reference clock source to tune the LPFLL. 00b - SIRC 01b - FIRC 10b - System OSC 11b - Reserved

16.6.1.24 Low Power FLL Status Register (LPFLLSTAT)

16.6.1.24.1 Offset

Register	Offset
LPFLLSTAT	514h

16.6.1.24.2 Function

This register is loaded from IFR during reset. These register gets uploaded with the trim values generated by LPFLL auto trimming which is enabled when LPFLL is enabled and LPFLLTREN=1 and LPFLLTRUP=1. When LPFLL auto trimming is enabled and LPFLLTRUP is off, writes to this register are allowed and values written to this register are used to trim LPFLL clock.

16.6.1.24.3 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								AUTOTRIM							
W																
Reset	0	0	0	0	0	0	0	0	u[1]	u	u	u	u	u	u	u

[1] Reset values are loaded out of IFR.

16.6.1.24.4 Fields

Field	Function
31-8 —	Reserved
7-0 AUTOTRIM	Auto Tune Trim Status When LPFLL is enabled and auto trimming is enabled (LPFLLTREN=1 and LPFLLTRUP=1) these register gets uploaded with the trimmed value. When LPFLLTRUP=0, these register is writeable to allow user programming of trim values.

Chapter 17
Peripheral Clock Controller (PCC)

17.1 Chip-specific information for this module

17.1.1 Information of PCC on this device

The clock connection information for this module is as follows.

Clock Source : SCG	Clock Source Descriptions	PCS Clock Names of PCC
SOSCDIV2_CLK	SOSCDIV2 of system OSC clock	OSCCLK
SIRCDIV2_CLK	SIRCDIV2 of slow IRC clock	SCGIRCLK
FIRCDIV2_CLK	FIRCDIV2 of fast IRC clock	SCGFIRCLK
SFLLDIV2_CLK	FLLDIV2 of LPFLL clock	SCGFLLCLK

For PCS bitfield, the following clock source select options are applicable in this device:

- 000 - Clock is off .
- 001 - System Oscillator Bus Clock.
- 010 - Slow IRC Clock.
- 011 - Fast IRC Clock.
- 100 - Reserved.
- 101 - Low-power FLL (LPFLL) clock.
- 110 - Reserved.
- 111 - Reserved.

NOTE

PCC_FLASH[CGC] is always 1 in this device, and writing 0 takes no effect.

17.2 Overview

PCC provides clock control and configuration for on-chip peripherals. Each peripheral has its own clock control and configuration register.

17.2.1 Block diagram

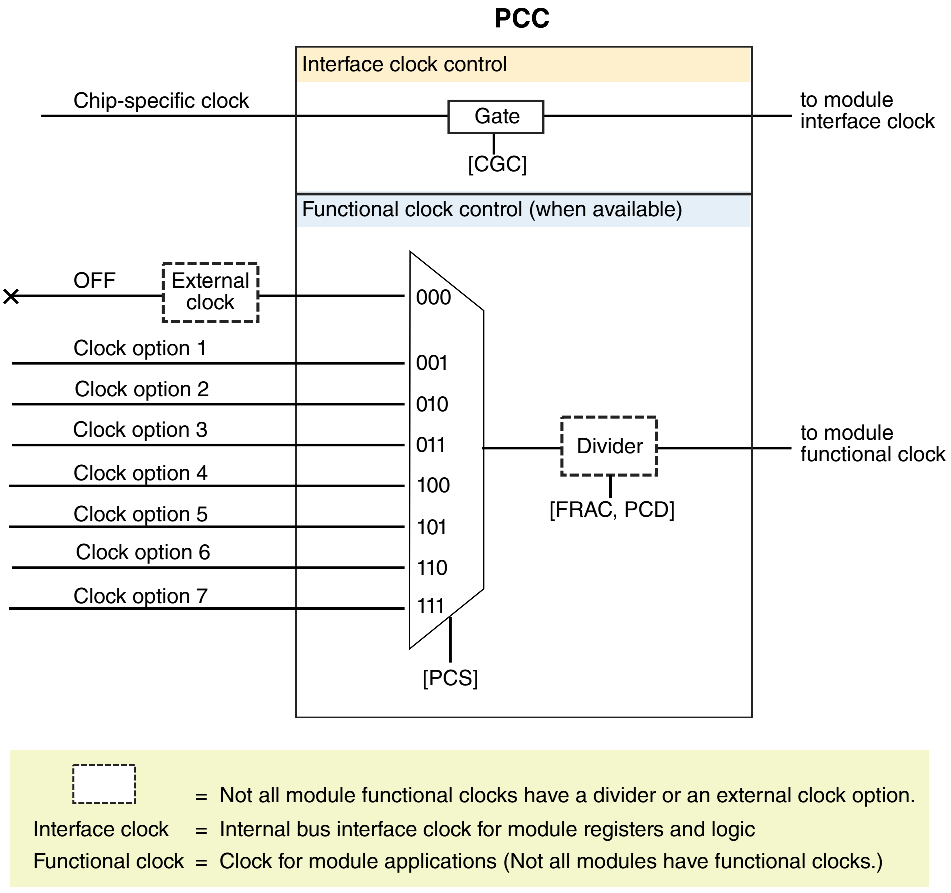

PCC
Interface clock control
Chip-specific clock
Gate
[CGC]
to module interface clock
Functional clock control (when available)
OFF
External clock
Clock option 1
Clock option 2
Clock option 3
Clock option 4
Clock option 5
Clock option 6
Clock option 7
000
001
010
011
100
101
110
111
[PCS]
Divider
[FRAC, PCD]
to module functional clock
= Not all module functional clocks have a divider or an external clock option.
Interface clock = Internal bus interface clock for module registers and logic
Functional clock = Clock for module applications (Not all modules have functional clocks.)

Figure 17-1. PCC block diagram

17.2.2 Features

PCC enables software to configure the following clocking options for each peripheral:

- Interface clock gating
- Functional clock source selection
- Functional clock divide values

17.3 Functional description

PCC provides on-chip peripherals (modules) their own dedicated PCC registers for clock gating and configuration options. Each module's PCC register contains a clock gating control bit (CGC) for the module's interface clock. Before a module can be used, its interface clock must be enabled (CGC = 1) in the module's PCC register.

If a module has a functional clock, its PCC register may provide options for the clock source, selected by programming the Peripheral Clock Select (PCS) field. Optionally, a module may also have a clock divider, selected by programming the Peripheral Clock Divider (PCD) field along with a Fraction (FRAC) field. Before configuring a functional clock, the module's interface clock must be disabled (CGC = 0).

17.3.1 Interrupts

This module has no interrupts.

17.3.2 Clocking

This module has no clocking considerations.

17.4 External signals

This module has no external signals.

17.5 Register descriptions

Each module has its own dedicated PCC register, which controls the clock gating, clock source and divider (when applicable) for that specific module. See each module's PCC register for details.

PCC registers can be written only in supervisor mode using 32-bit accesses.

NOTE

> To configure the clocking options available to a given module or to modify an existing configuration, first disable the module's interface clock by writing 0 to its CGC bit.

17.5.1 PCC register descriptions

17.5.1.1 PCC memory map

PCC base address: 4006_5000h

Offset	Register	Width (In bits)	Access	Reset value
20h	PCC DMA (PCC_DMA)	32	RW	C000_0000h
80h	PCC FLASH (PCC_FLASH)	32	RW	C000_0000h
84h	PCC DMAMUX (PCC_DMAMUX)	32	RW	8000_0000h
B0h	PCC LPSPI0 (PCC_LPSPI0)	32	RW	8000_0000h
B4h	PCC LPSPI1 (PCC_LPSPI1)	32	RW	8000_0000h
C8h	PCC CRC (PCC_CRC)	32	RW	8000_0000h
DCh	PCC LPIT0 (PCC_LPIT0)	32	RW	8000_0000h
E0h	PCC FLEXTMR0 (PCC_FLEXTMR0)	32	RW	8000_0000h
E4h	PCC FLEXTMR1 (PCC_FLEXTMR1)	32	RW	8000_0000h
E8h	PCC FLEXTMR2 (PCC_FLEXTMR2)	32	RW	8000_0000h
ECh	PCC ADC0 (PCC_ADC0)	32	RW	C000_0000h
F4h	PCC RTC (PCC_RTC)	32	RW	8000_0000h
100h	PCC LPTMR0 (PCC_LPTMR0)	32	RW	8000_0000h
114h	PCC TSI0 (PCC_TSI0)	32	RW	8000_0000h
11Ch	PCC TSI1 (PCC_TSI1)	32	RW	8000_0000h
124h	PCC PORTA (PCC_PORTA)	32	RW	8000_0000h
128h	PCC PORTB (PCC_PORTB)	32	RW	8000_0000h
12Ch	PCC PORTC (PCC_PORTC)	32	RW	8000_0000h
130h	PCC PORTD (PCC_PORTD)	32	RW	8000_0000h
134h	PCC PORTE (PCC_PORTE)	32	RW	8000_0000h
158h	PCC PWT (PCC_PWT)	32	RW	8000_0000h
168h	PCC FLEXIO (PCC_FLEXIO)	32	RW	8000_0000h
184h	PCC EWM (PCC_EWM)	32	RW	8000_0000h
188h	PCC FLEXIOTRIG0 (PCC_FLEXIOTRIG0)	32	RW	C000_0000h
18Ch	PCC FLEXIOTRIG1 (PCC_FLEXIOTRIG1)	32	RW	C000_0000h
198h	PCC LPI2C0 (PCC_LPI2C0)	32	RW	8000_0000h
19Ch	PCC LPI2C1 (PCC_LPI2C1)	32	RW	8000_0000h
1A8h	PCC LPUART0 (PCC_LPUART0)	32	RW	8000_0000h
1ACh	PCC LPUART1 (PCC_LPUART1)	32	RW	8000_0000h
1B0h	PCC LPUART2 (PCC_LPUART2)	32	RW	8000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1B4h	PCC SCI0 (PCC_SCI0)	32	RW	8000_0000h
1B8h	PCC SCI1 (PCC_SCI1)	32	RW	8000_0000h
1CCh	PCC CMP0 (PCC_CMP0)	32	RW	8000_0000h

17.5.1.2 PCC DMA (PCC_DMA)

17.5.1.2.1 Offset

Register	Offset
PCC_DMA	20h

17.5.1.2.2 Function

This register is for the DMA module.

17.5.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.2.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present.

Table continues on the next page...

Field	Function
	1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.3 PCC FLASH (PCC_FLASH)

17.5.1.3.1 Offset

Register	Offset
PCC_FLASH	80h

17.5.1.3.2 Function

This register is for the FLASH module.

17.5.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.3.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.4 PCC DMAMUX (PCC_DMAMUX)

17.5.1.4.1 Offset

Register	Offset
PCC_DMAMUX	84h

17.5.1.4.2 Function

This register is for the DMAMUX module.

17.5.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.4.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.5 PCC LPSPI0 (PCC_LPSPI0)

17.5.1.5.1 Offset

Register	Offset
PCC_LPSPI0	B0h

17.5.1.5.2 Function

This register is for the LPSPI0 module.

17.5.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.5.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4	Reserved

Table continues on the next page...

Field	Function
—	
3 —	Reserved
2-0 —	Reserved

17.5.1.6 PCC LPSPI1 (PCC_LPSPI1)

17.5.1.6.1 Offset

Register	Offset
PCC_LPSPI1	B4h

17.5.1.6.2 Function

This register is for the LPSPI1 module.

17.5.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.6.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.7 PCC CRC (PCC_CRC)

17.5.1.7.1 Offset

Register	Offset
PCC_CRC	C8h

17.5.1.7.2 Function

This register is for the CRC module.

17.5.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.7.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.8 PCC LPIT0 (PCC_LPIT0)

17.5.1.8.1 Offset

Register	Offset
PCC_LPIT0	DCh

17.5.1.8.2 Function

This register is for the LPIT0 module.

17.5.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.8.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.9 PCC FLEXTMR0 (PCC_FLEXTMR0)

17.5.1.9.1 Offset

Register	Offset
PCC_FLEXTMR0	E0h

17.5.1.9.2 Function

This register is for the FLEXTMR0 module.

17.5.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.9.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.10 PCC FLEXTMR1 (PCC_FLEXTMR1)

17.5.1.10.1 Offset

Register	Offset
PCC_FLEXTMR1	E4h

17.5.1.10.2 Function

This register is for the FLEXTMR1 module.

17.5.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.10.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.11 PCC FLEXTMR2 (PCC_FLEXTMR2)

17.5.1.11.1 Offset

Register	Offset
PCC_FLEXTMR2	E8h

17.5.1.11.2 Function

This register is for the FLEXTMR2 module.

17.5.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.11.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.12 PCC ADC0 (PCC_ADC0)

17.5.1.12.1 Offset

Register	Offset
PCC_ADC0	ECh

17.5.1.12.2 Function

This register is for the ADC0 module.

17.5.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.12.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.13 PCC RTC (PCC_RTC)

17.5.1.13.1 Offset

Register	Offset
PCC_RTC	F4h

17.5.1.13.2 Function

This register is for the RTC module.

17.5.1.13.3 Diagram

<table>
<tr><td>Bits</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td>PR</td><td rowspan="2">CGC</td><td>Reserved</td><td colspan="2">0</td><td colspan="3">0</td><td colspan="8">0</td></tr>
<tr><td>W</td><td></td><td></td><td colspan="2"></td><td colspan="3"></td><td colspan="8"></td></tr>
<tr><td>Reset</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bits</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="12">0</td><td>0</td><td colspan="3">0</td></tr>
<tr><td>W</td><td colspan="12"></td><td></td><td colspan="3"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

17.5.1.13.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.14 PCC LPTMR0 (PCC_LPTMR0)

17.5.1.14.1 Offset

Register	Offset
PCC_LPTMR0	100h

17.5.1.14.2 Function

This register is for the LPTMR0 module.

17.5.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.14.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.15 PCC TSI0 (PCC_TSI0)

17.5.1.15.1 Offset

Register	Offset
PCC_TSI0	114h

17.5.1.15.2 Function

This register is for the TSI0 module.

17.5.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.15.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.16 PCC TSI1 (PCC_TSI1)

17.5.1.16.1 Offset

Register	Offset
PCC_TSI1	11Ch

17.5.1.16.2 Function

This register is for the TSI1 module.

17.5.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.16.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.17 PCC PORTA (PCC_PORTA)

17.5.1.17.1 Offset

Register	Offset
PCC_PORTA	124h

17.5.1.17.2 Function

This register is for the PORTA module.

17.5.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.17.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.18 PCC PORTB (PCC_PORTB)

17.5.1.18.1 Offset

Register	Offset
PCC_PORTB	128h

17.5.1.18.2 Function

This register is for the PORTB module.

17.5.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.18.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.19 PCC PORTC (PCC_PORTC)

17.5.1.19.1 Offset

Register	Offset
PCC_PORTC	12Ch

17.5.1.19.2 Function

This register is for the PORTC module.

17.5.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.19.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.20 PCC PORTD (PCC_PORTD)

17.5.1.20.1 Offset

Register	Offset
PCC_PORTD	130h

17.5.1.20.2 Function

This register is for the PORTD module.

17.5.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.20.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.21 PCC PORTE (PCC_PORTE)

17.5.1.21.1 Offset

Register	Offset
PCC_PORTE	134h

17.5.1.21.2 Function

This register is for the PORTE module.

17.5.1.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.21.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.22 PCC PWT (PCC_PWT)

17.5.1.22.1 Offset

Register	Offset
PCC_PWT	158h

17.5.1.22.2 Function

This register is for the PWT module.

17.5.1.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.22.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.23 PCC FLEXIO (PCC_FLEXIO)

17.5.1.23.1 Offset

Register	Offset
PCC_FLEXIO	168h

17.5.1.23.2 Function

This register is for the FLEXIO module.

17.5.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.23.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4	Reserved

Table continues on the next page...

Field	Function
—	
3 —	Reserved
2-0 —	Reserved

17.5.1.24 PCC EWM (PCC_EWM)

17.5.1.24.1 Offset

Register	Offset
PCC_EWM	184h

17.5.1.24.2 Function

This register is for the EWM module.

17.5.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.24.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.25 PCC FLEXIOTRIG0 (PCC_FLEXIOTRIG0)

17.5.1.25.1 Offset

Register	Offset
PCC_FLEXIOTRIG0	188h

17.5.1.25.2 Function

This register is for the FLEXIOTRIG0 module.

17.5.1.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							FRAC
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 W	PCD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.25.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-17	Reserved

Table continues on the next page...

Field	Function
—	
16 FRAC	Peripheral Clock Divider Fraction Sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
15-0 PCD	Peripheral Clock Divider Select Is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). 0000_0000_0000_0000b - Divide by 1. 0000_0000_0000_0001b - Divide by 2. 0000_0000_0000_0010b - Divide by 3. 0000_0000_0000_0011b - Divide by 4. 0000_0000_0000_0100b - Divide by 5. 0000_0000_0000_0101b - Divide by 6. 0000_0000_0000_0110b - Divide by 7. 0000_0000_0000_0111b - Divide by 8.

17.5.1.26 PCC FLEXIOTRIG1 (PCC_FLEXIOTRIG1)

17.5.1.26.1 Offset

Register	Offset
PCC_FLEXIOTRIG1	18Ch

17.5.1.26.2 Function

This register is for the FLEXIOTRIG1 module.

17.5.1.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							FRAC
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 W	PCD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.26.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-17	Reserved

Table continues on the next page...

Field	Function
—	
16 FRAC	Peripheral Clock Divider Fraction Sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
15-0 PCD	Peripheral Clock Divider Select Is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). 0000_0000_0000_0000b - Divide by 1. 0000_0000_0000_0001b - Divide by 2. 0000_0000_0000_0010b - Divide by 3. 0000_0000_0000_0011b - Divide by 4. 0000_0000_0000_0100b - Divide by 5. 0000_0000_0000_0101b - Divide by 6. 0000_0000_0000_0110b - Divide by 7. 0000_0000_0000_0111b - Divide by 8.

17.5.1.27 PCC LPI2C0 (PCC_LPI2C0)

17.5.1.27.1 Offset

Register	Offset
PCC_LPI2C0	198h

17.5.1.27.2 Function

This register is for the LPI2C0 module.

17.5.1.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.27.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4	Reserved

Table continues on the next page...

Field	Function
—	
3 —	Reserved
2-0 —	Reserved

17.5.1.28 PCC LPI2C1 (PCC_LPI2C1)

17.5.1.28.1 Offset

Register	Offset
PCC_LPI2C1	19Ch

17.5.1.28.2 Function

This register is for the LPI2C1 module.

17.5.1.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.28.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.29 PCC LPUART0 (PCC_LPUART0)

17.5.1.29.1 Offset

Register	Offset
PCC_LPUART0	1A8h

17.5.1.29.2 Function

This register is for the LPUART0 module.

17.5.1.29.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.29.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7

Table continues on the next page...

Field	Function
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.30 PCC LPUART1 (PCC_LPUART1)

17.5.1.30.1 Offset

Register	Offset
PCC_LPUART1	1ACh

17.5.1.30.2 Function

This register is for the LPUART1 module.

17.5.1.30.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.30.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.31 PCC LPUART2 (PCC_LPUART2)

17.5.1.31.1 Offset

Register	Offset
PCC_LPUART2	1B0h

17.5.1.31.2 Function

This register is for the LPUART2 module.

17.5.1.31.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		PCS			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.31.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 PCS	Peripheral Clock Source Select Is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3

Table continues on the next page...

Field	Function
	100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.32 PCC SCI0 (PCC_SCI0)

17.5.1.32.1 Offset

Register	Offset
PCC_SCI0	1B4h

17.5.1.32.2 Function

This register is for the SCI0 module.

17.5.1.32.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.32.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.33 PCC SCI1 (PCC_SCI1)

17.5.1.33.1 Offset

Register	Offset
PCC_SCI1	1B8h

17.5.1.33.2 Function

This register is for the SCI1 module.

17.5.1.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.33.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified. 1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

17.5.1.34 PCC CMP0 (PCC_CMP0)

17.5.1.34.1 Offset

Register	Offset
PCC_CMP0	1CCh

17.5.1.34.2 Function

This register is for the CMP0 module.

17.5.1.34.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	Reserved	0		0			0							
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	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1.34.4 Fields

Field	Function
31 PR	Present Indicates whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control Specifies whether to enable or disable the interface clock for the peripheral, allowing access to the module's registers. It also specifies whether the clock selection and divider options can be modified. 0b - Disables the clock. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Enables the clock. The current clock selection and divider options are locked and cannot be modified.
29 —	This bit is reserved. This bit can change values but is a don't-care.
28-27 —	Reserved
26-24 —	Reserved
23-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

Chapter 18
Reset and Boot

18.1 Introduction

The following reset sources are supported in this MCU:

Table 18-1. Reset sources

Reset sources	Description
POR reset	• Power-on reset (POR)
System resets	• External pin reset (PIN) • Low voltage detect (LVD) • Software watchdog reset (WDOG) • Clock generator loss of clock (LOC) reset • Clock generator loss of lock (LOL) reset • Stop mode acknowledge error (SACKERR) • Software reset (SW) • Lockup reset (LOCKUP) • MDM DAP system reset
Debug reset	• Debug reset

Each of the reset sources has an associated bit in the system reset status (RCM_SRS) register. Besides immediate reset, the RCM module also supports optional delays of the system resets for a period of time with an interrupt generated. This provides software an option to perform a graceful shutdown. See the Reset Control Module (RCM) chapter for register details.

The MCU exits reset in functional mode where the CPU is executing code. See Boot options for more details.

The following figure shows a block diagram of the reset sources for this device.

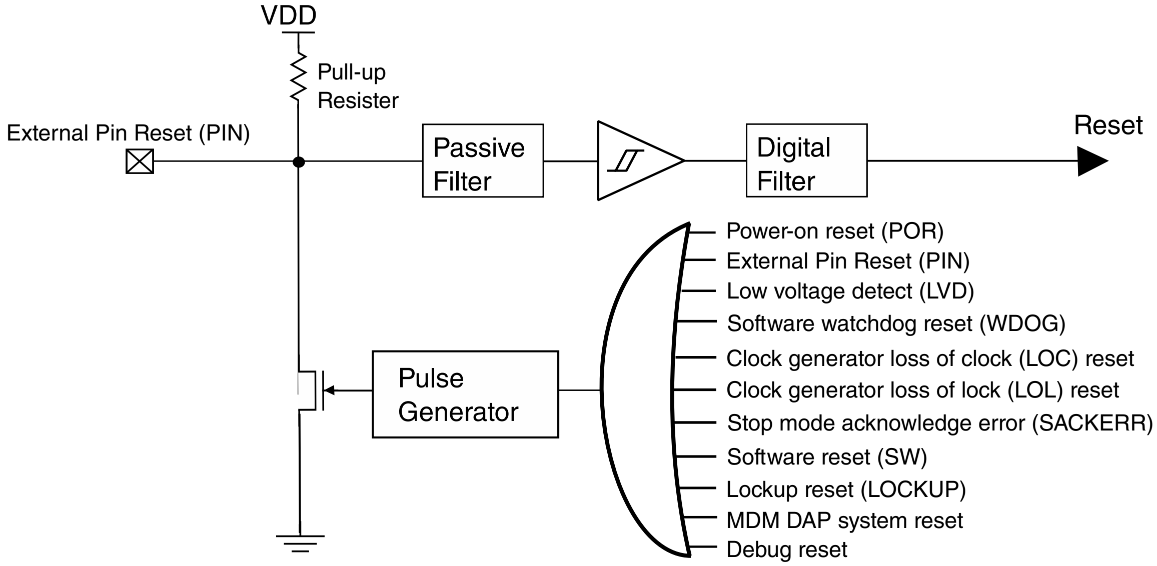


Figure 18-1. Reset Sources

18.2 Reset

This section 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.

18.2.1 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 LVR circuit holds the MCU in reset until the supply has risen above the LVR threshold (V_{LVR}). The POR and LVD bits in RCM_SRS register are set following a POR.

18.2.2 System resets

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 are assigned by default to their analog functions after reset.

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

- SWD_CLK in pull-down (PD)
- SWD_DIO in pull-up (PU)

18.2.2.1 External pin reset (PIN)

On this device, asserting $\overline{\text{RESET}}$ wakes and resets the device from any mode. During a pin reset, RCM_SRS[PIN] is set.

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, 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.

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.

18.2.2.2 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. The LVD system can always be enabled in HSRUN, normal Run, or Wait mode. The LVD system is disabled (LVR active only) when entering VLPx modes or Stop mode.

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting the PMC_LVDSC1[LVDRE] bit to 1. 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. The RCM_SRS[LVD] bit is set following either an LVD reset or POR.

Refer to the "Low-voltage Detect (LVD) System" section in the Power Management Controller (PMC) chapter for more information. For LVR related content, see Low Voltage Reset (LVR) Operation.

18.2.2.3 Watchdog timer (WDOG)

The watchdog timer (WDOG) 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, the watchdog issues a system reset. The reset causes the RCM_SRS[WDOG] bit to set.

18.2.2.4 Clock generator loss-of-clock (LOC)

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

NOTE

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

18.2.2.5 Loss-of-lock (LOL) reset

The SCG module contains a loss-of-lock detector, to indicate a reset has been caused by a loss of lock in the SCG PLL/FLL.

NOTE

This reset source does not cause a reset if the chip is in VLPR/VLPW/VLPS mode.

18.2.2.6 Stop mode acknowledge error (SACKERR)

This reset is generated if the core attempts to enter stop mode, but not all modules acknowledge stop mode within 1025 cycles of the 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.

The RCM_SRS[SACKERR] bit is set to indicate this reset source.

18.2.2.7 Software reset (SW)

The SYSRESETREQ bit in the System Control Block's (SCB) application interrupt and reset control register can be set to force a software reset on the device. (See ARM's Cortex-M user guide 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 the RCM_SRS[SW] bit to set.

18.2.2.8 Lockup reset (LOCKUP)

The LOCKUP gives immediate indication of 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 the RCM_SRS[LOCKUP] bit to set.

18.2.2.9 MDM-AP system reset request

Set the system reset request bit in the MDM-AP control register to initiate a system reset. This is the primary method for resets via the SWD interface. The system reset is held until this bit is cleared.

Set the core hold reset bit in the MDM-AP control register to hold the core in reset as the rest of the chip comes out of system reset.

18.2.3 MCU Resets

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

18.2.3.1 POR Only

The POR Only reset asserts on the POR reset source only. It resets the PMC registers.

The POR Only reset also causes all other reset types to occur.

18.2.3.2 Chip POR

The Chip POR asserts on POR, LVD Wakeup reset sources. It resets the Reset Pin Filter registers and parts of the SIM and SCG .

The Chip POR also causes the Chip Reset (including Early Chip Reset) to occur.

18.2.3.3 Early Chip Reset

The 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).

18.2.3.4 Chip Reset

Chip Reset asserts on all reset sources and only negates after flash initialization has completed and the RESET_b pin has also negated. It resets the remaining modules (the modules not reset by other reset types).

18.2.4 Reset Pin

For all reset sources, 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.

18.3 Boot

This section describes the boot sequence, including sources and options.

18.3.1 Boot options

The Flash Option (FOPT) register in the Flash Memory module (FTFE_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 FTFE_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 and any system reset. For more details on programming the option byte, see the flash memory chapter.

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

Table 18-2. Flash Option Register (FTFE_FOPT) definition

Bit Num	Field	Value	Definition
7	Reserved	Reserved for future expansion	
6	Reserved	Reserved for future expansion	
5-4	Reserved	Reserved for future expansion	
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 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	Reserved	Reserved for future expansion	

Table continues on the next page...

Table 18-2. Flash Option Register (FTFE_FOPT) definition (continued)

Bit Num	Field	Value	Definition
0	LPBOOT	Controls the reset value of clock divider of IRC48M to feed the core clock. Larger divide value selections produce lower average power consumption during POR and reset sequencing and after reset exit. The recovery times are also extended .	
		0	Low-power boot: Core and system clock divider (DIVCORE) is 0x1 (divide by 2).
		1	Normal boot: Core and system clock divider (DIVCORE) is 0x0 (divide by 1).

This device supports cold booting from either internal flash.

When the device boots from internal flash, the reset vectors are located at address 0x0 (initial SP_main) and 0x4 (initial PC).

The device also supports relocating the exception vector table to RAM. This is implemented through a programmable Vector Table Offset Register (VTOR) in SCB module.

18.3.2 Boot sequence

At power up, the on-chip regulator holds the system in a POR state until the input supply is above 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 LVR. The Mode Controller reset logic then controls a sequence to exit reset.

1. A system reset is held on internal logic, the RESET_b pin is driven out low, and the SCG is enabled in its default clocking mode.
2. Required clocks are enabled (Core Clock, 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_b pin out low.
4. Early in reset sequencing the NVM option byte is read and stored to the Flash Memory module's FOPT register.
5. When Flash Initialization completes, the RESET_b pin is released. If RESET_b continues to be asserted (an indication of a slow rise time on the RESET_b pin or external drive in low), the system continues to be held in reset. Once the RESET_b 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. What happens next depends on the NMI input and the FOPT[NMI_DIS] field in the Flash Memory module:

- If the NMI input is high or the NMI function is disabled in the NMI_DIS field, the CPU begins execution at the PC location.
- If the NMI input is low and the NMI function is enabled in the NMI_DIS field, 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.

Subsequent system resets follow this same reset flow.

The following figure shows the boot sequence.

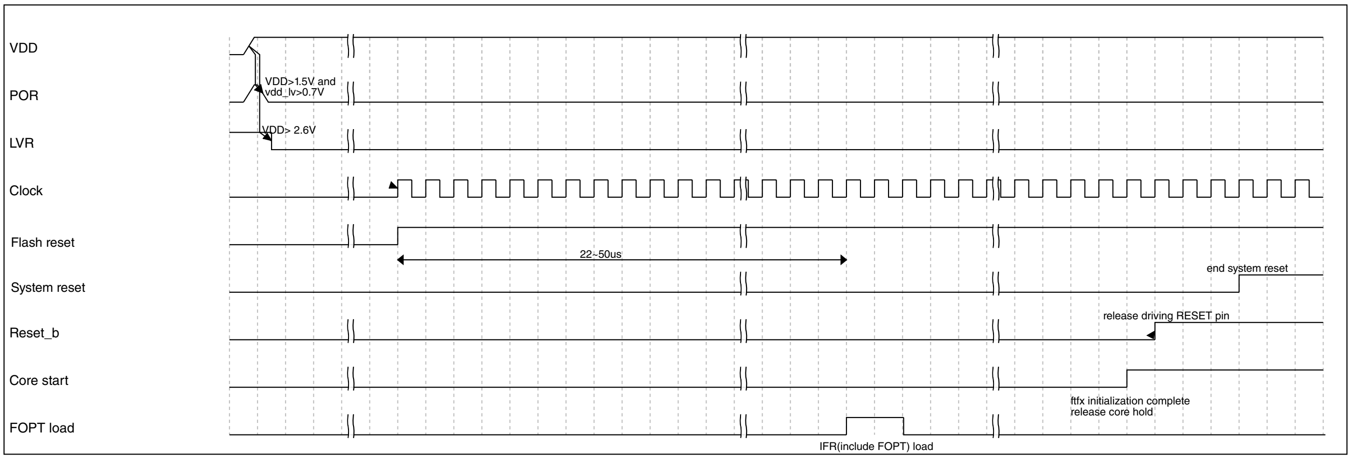


Figure 18-2. Boot Sequence

Chapter 19
Kinetis Flashloader

19.1 Chip-specific information for this module

19.1.1 Flashloader Configuration

This device has various peripherals supported by the Kinetis Flashloader. The pinmux table for peripherals supported is shown as follows.

Peripheral	Instance	Signal	GPIO	ALT
LPUART	2	LPUART2_TX	PTE12	3
		LPUART2_RX	PTD17	3
LPSPI	0	LPSPI0_PCS2	PTE6	2
		LPSPI0_SOUT	PTE2	2
		LPSPI0_SIN	PTE1	2
		LPSPI0_SCK	PTE0	2
LPI2C	0	LPI2C0_SCL	PTA3	3
		LPI2C0_SDA	PTA2	3

19.2 Introduction

The Kinetis devices *that do not have an on-chip ROM* are shipped with the pre-programmed Kinetis Flashloader in the on-chip flash memory, for one-time, in-system factory programming. The Kinetis Flashloader's main task is to load a customer firmware image into the flash memory. The image on the flash has 2 programs: flashloader_loader and flashloader. After a device reset, the flashloader_loader program starts its execution first. The flashloader_loader program copies the contents of flashloader image from the flash to the on-chip RAM; the device then switches execution to the flashloader program to execute from RAM.

For this device, the Kinetis Flashloader can interface with LPUART, LPI2C, and LPSPI 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 Flashloader. Regardless of the host/master (PC or embedded host), the Kinetis Flashloader always uses a command protocol to communicate with that host/master. Commands are provided to write to memory (flash or RAM), erase flash, and get/set flashloader options and property values. The host application can query the set of available commands.

This chapter describes Kinetis Flashloader features, functionality, command structure and which peripherals are supported.

Features supported by the Kinetis Flashloader :

- Supports LPUART, LPI2C, and LPSPI peripheral interfaces
- Automatic detection of the active peripheral
- LPUART peripheral with autobaud
- Common packet-based protocol for all peripherals
- Packet error detection and retransmission
- Protection of RAM used by the flashloader while it is running
- Provides command to read properties of the device, such as flash and RAM size

Table 19-1. Commands supported by the Kinetis Flashloader

Command	Description	When flash security is enabled, then this command is
Execute	Run user application code that never returns control to the flashloader	Not supported
FlashEraseAll	Erase the entire flash array	Not supported
FlashEraseRegion	Erase a range of sectors in flash	Not supported
WriteMemory	Write data to memory	Not supported
ReadMemory	Read data from memory	Not supported
GetProperty	Get the current value of a property	Supported
Reset	Reset the chip	Supported
FlashEraseAllUnsecure	Erase the entire flash array, including protected sectors	Supported

19.3 Functional Description

The following sub-sections describe the Kinetis Flashloader functionality.

19.3.1 Memory Maps

While executing, the Kinetis Flashloader uses RAM memory.

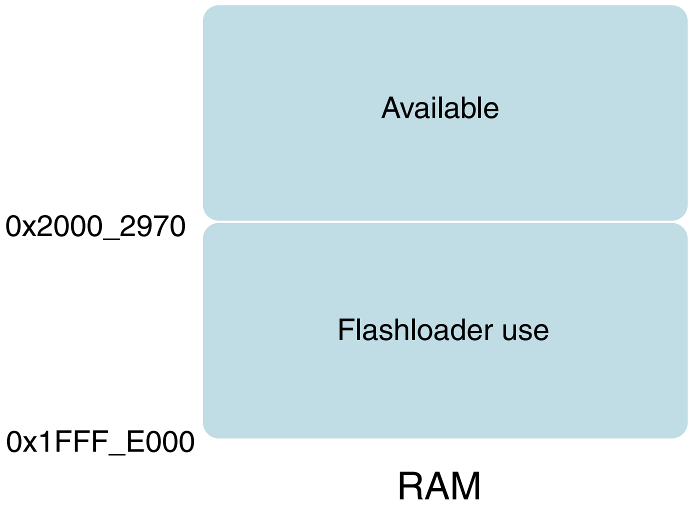


Figure 19-1. Kinetis Flashloader RAM Memory Map

NOTE

The Kinetis Flashloader requires a minimum memory space of 32 KB of RAM. For Kinetis devices with less than this amount of on-chip RAM, the Kinetis Flashloader is not available.

19.3.2 Start-up Process

As the Kinetis Flashloader begins executing, flashloader operations begin:

1. The flashloader’s temporary working area in RAM is initialized.
2. All supported peripherals are initialized.
3. The flashloader waits for communication to begin on a peripheral.
    - There is no timeout for the active peripheral detection process.
    - If communication is detected, then all inactive peripherals are shut down, and the command phase is entered.

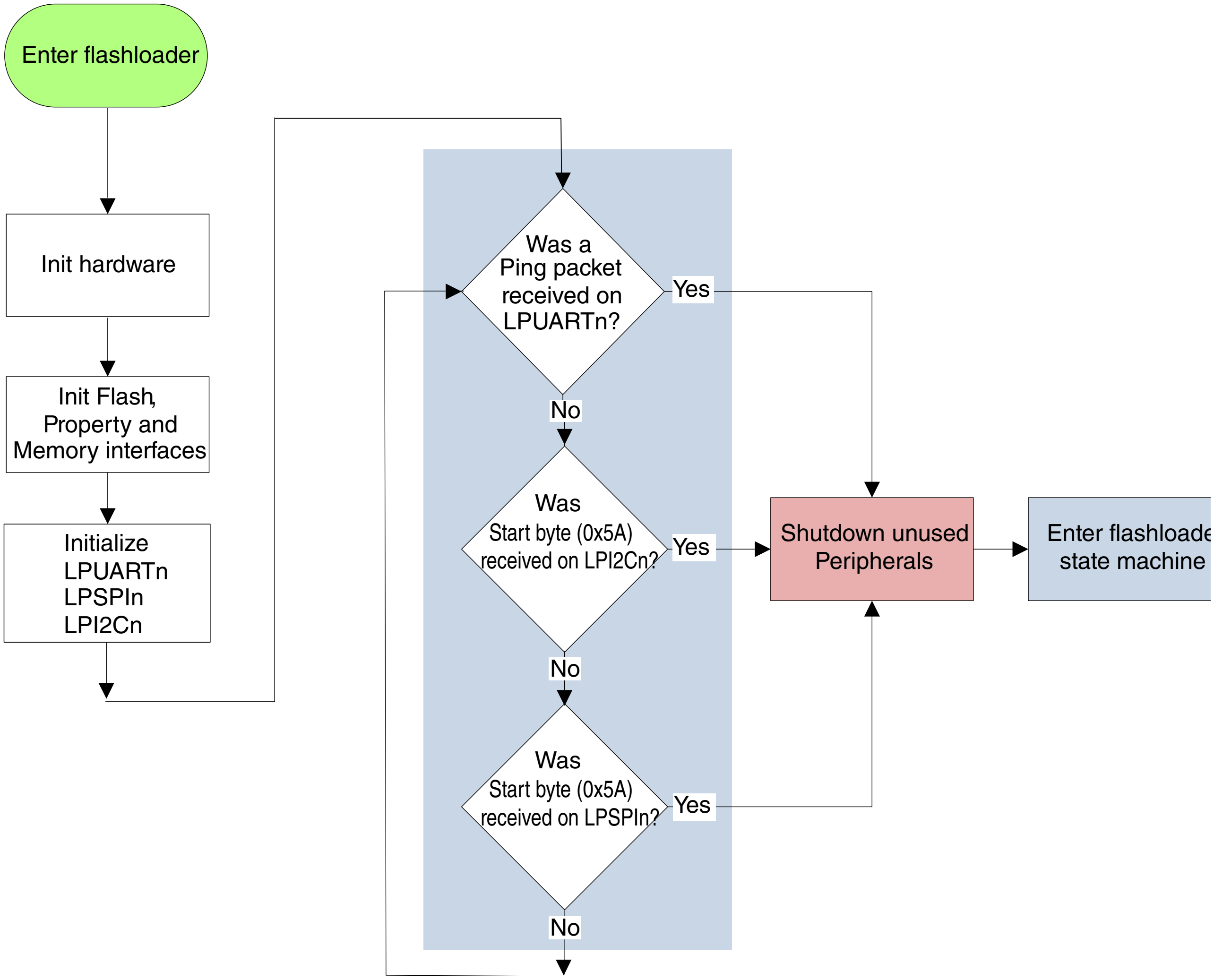

Enter flashloader
Init hardware
Init Flash, Property and Memory interfaces
Initialize LPUARTn LPSPIn LPI2Cn
Was a Ping packet received on LPUARTn?
Yes
No
Was Start byte (0x5A) received on LPI2Cn?
Yes
No
Was Start byte (0x5A) received on LPSPIn?
Yes
No
Shutdown unused Peripherals
Enter flashloade state machine

Figure 19-2. Kinetis Flashloader Start-up Flowchart

19.3.3 Clock Configuration

The Flashloader uses the default clocks.

19.3.4 Flashloader Protocol

This section explains the general protocol for the packet transfers between the host and the Kinetis Flashloader. 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 flashloader), then the data phase is part of the **original command**.
- If the data phase is **outgoing** (from flashloader 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.

19.3.4.1 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)

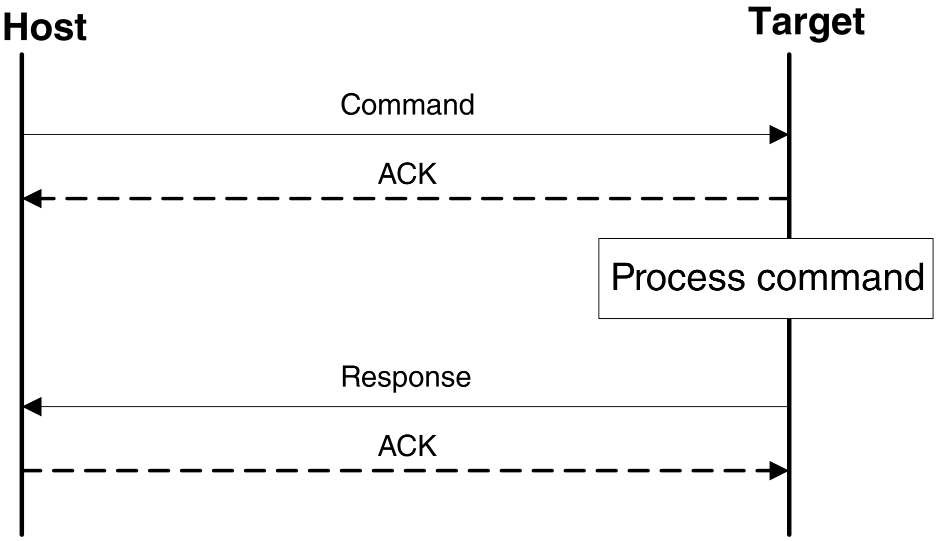


Figure 19-3. Command with No Data Phase

19.3.4.2 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)

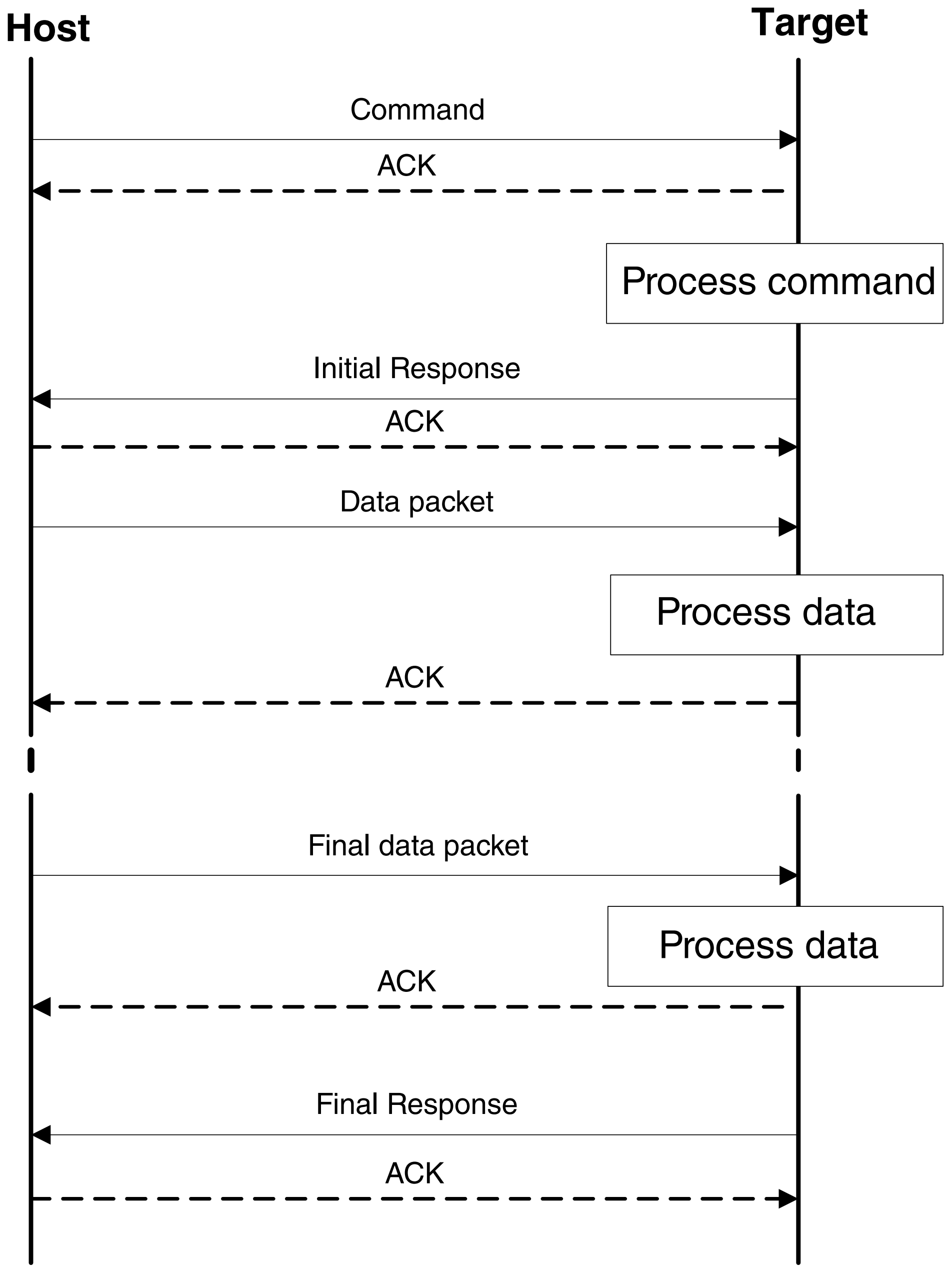


Figure 19-4. Command with incoming data phase

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.

19.3.4.3 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)

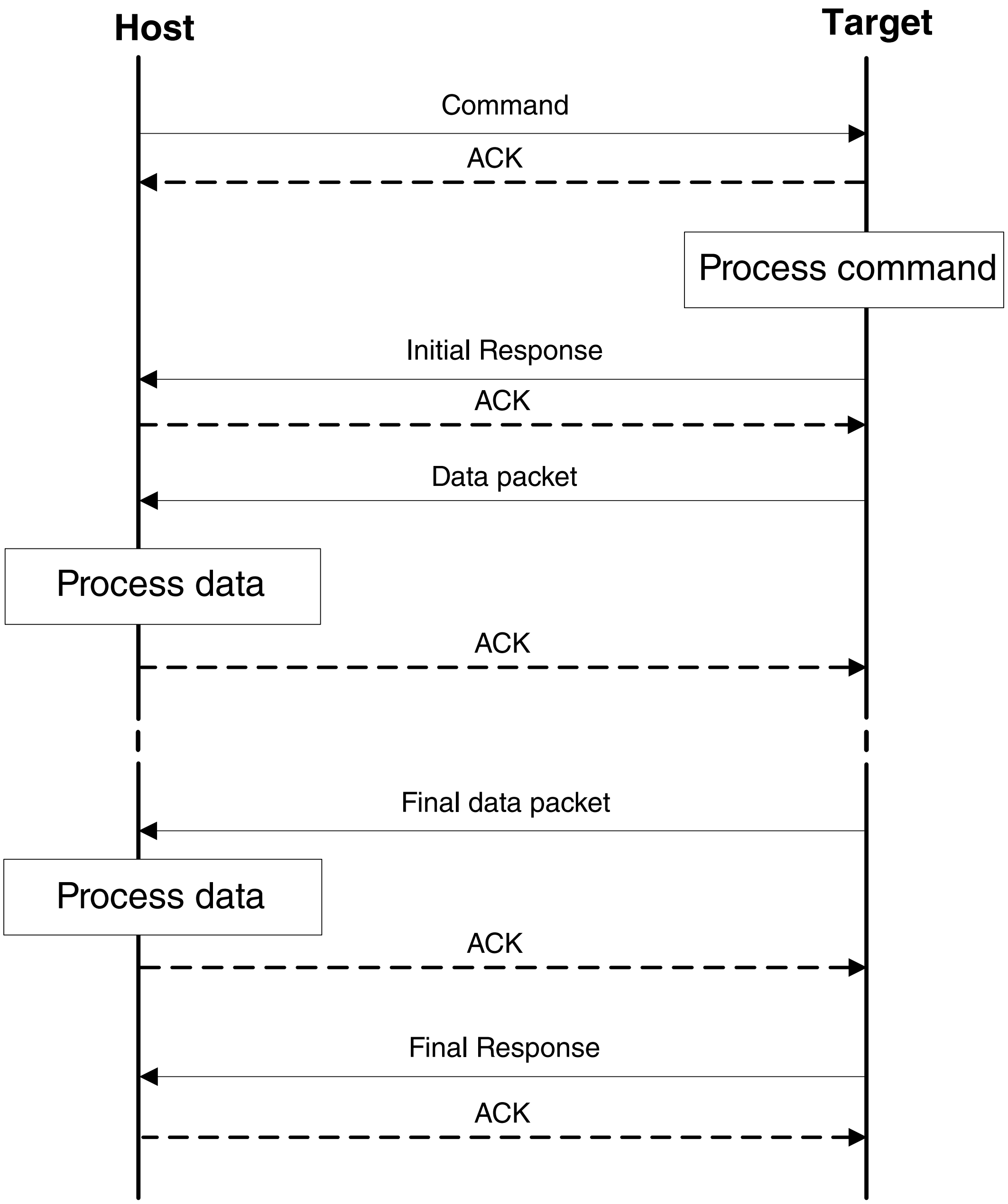


Figure 19-5. Command with outgoing data phase

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.

19.3.5 Flashloader Packet Types

The Kinetis Flashloader device works in slave mode. All data communication is initiated by a host, which is either a PC or an embedded host . The Kinetis Flashloader 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 Flashloader 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.

19.3.5.1 Ping packet

The Ping packet is the first packet sent from a host to the target (Kinetis Flashloader), to establish a connection on a selected peripheral. For a UART 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 19-2. Ping Packet Format

Byte #	Value	Name
0	0x5A	start byte
1	0xA6	ping

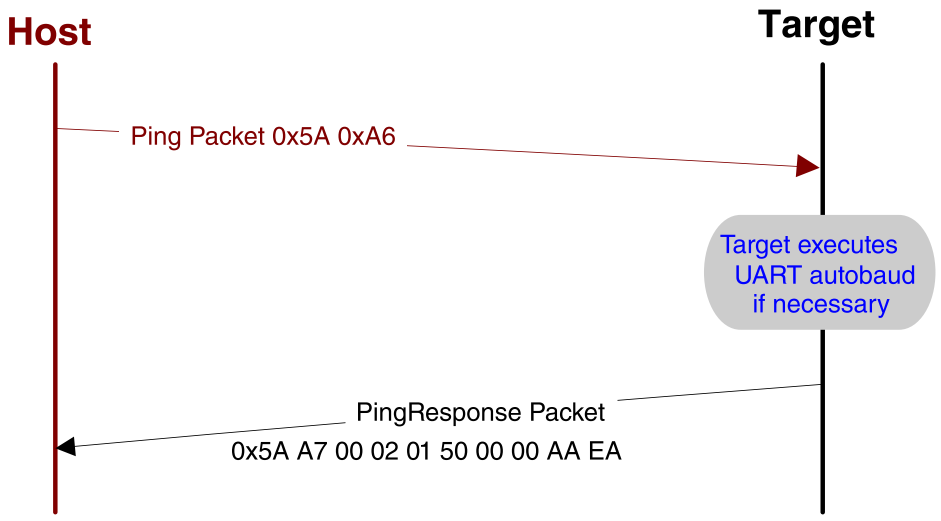

Host
Target
Ping Packet 0x5A 0xA6
Target executes UART autobaud if necessary
PingResponse Packet
0x5A A7 00 02 01 50 00 00 AA EA

Figure 19-6. Ping Packet Protocol Sequence

19.3.5.2 Ping Response Packet

The target (Kinetis Flashloader) sends a Ping Response packet back to the host after receiving a Ping packet. If communication is over a UART 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 Flashloader).

Table 19-3. 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

19.3.5.3 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 UART, I2C, SPI.

Table 19-4. 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 19-5. 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 19-6. 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.
0xA6	kFramingPacketType_Ping	Sent to verify the other side is alive. Also used for UART autobaud.

Table continues on the next page...

Table 19-6. packetType Field (continued)

packetType	Name	Description
0xA7	kFramingPacketType_PingResponse	A response to Ping; contains the framing protocol version number and options.

This device uses the Cyclic Redundancy Check module (CRC) to perform the CRC algorithm. See the CRC chapter for more details.

19.3.5.4 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 19-7. 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 19-8. 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 19-9. Commands that are supported

Command	Name
0x01	FlashEraseAll
0x02	FlashEraseRegion
0x03	ReadMemory

Table continues on the next page...

Table 19-9. Commands that are supported (continued)

Command	Name
0x04	WriteMemory
0x05	Reserved
0x06	Reserved
0x07	GetProperty
0x08	Reserved
0x09	Execute
0x0A	Reserved
0x0B	Reset
0x0C	Reserved
0x0D	FlashEraseAllUnsecure
0x0E	Reserved
0x0F	Reserved
0x10	Reserved
0x11	Reserved
0x12	Reserved

Table 19-10. 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)

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.

19.3.5.5 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.

19.3.5.6 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses include:

- GenericResponse
- GetPropertyResponse
- ReadMemoryResponse

GenericResponse: After the Kinetis Flashloader has processed a command, the flashloader 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 19-11. 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 Flashloader). If a command succeeds, then a kStatus_Success code is returned. Table 19-34, Kinetis Flashloader Status Error Codes, lists the status codes returned to the host by the Kinetis Flashloader.
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 19-12. GetPropertyResponse Parameters

Byte #	Value	Parameter
0 - 3		Status code
4 - 7		Property value
. . .		. . .
		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 19-13. 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.

19.3.6 Flashloader Command API

All Kinetis Flashloader 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 Flashloader, see Table 19-1, Commands supported.
- For a list of status codes returned by the Kinetis Flashloader, see Table 19-34, Kinetis Flashloader Status Error Codes.

NOTE

All the examples in this section depict byte traffic on serial peripherals that use framing packets.

19.3.6.1 GetProperty command

The GetProperty command is used to query the flashloader 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 Flashloader, see Table 19-30.

The 32-bit property tag is the only parameter required for GetProperty command.

Table 19-14. Parameters for GetProperty Command

Byte #	Command
0 - 3	Property tag

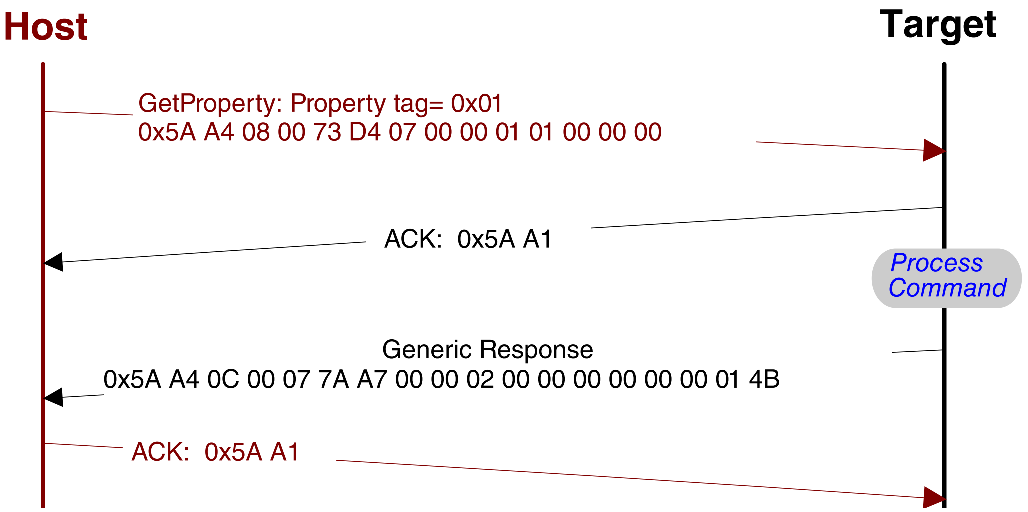

Host
Target
GetProperty: Property tag= 0x01
0x5A A4 08 00 73 D4 07 00 00 01 01 00 00 00
ACK: 0x5A A1
Process
Command
Generic Response
0x5A A4 0C 00 07 7A A7 00 00 02 00 00 00 00 00 00 01 4B
ACK: 0x5A A1

Figure 19-7. Protocol Sequence for GetProperty Command

Table 19-15. GetProperty Command Packet Format (Example)

GetProperty	Parameter	Value
Framing packet	start byte	0x5A

Table continues on the next page...

Table 19-15. GetProperty Command Packet Format (Example) (continued)

GetProperty	Parameter	Value
	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 19-16. 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

19.3.6.2 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 FTFE_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 1 parameter: memoryId.

Table 19-17. Parameters for FlashEraseAll command

Bytes	Parameter
0 - 3	MemoryId • 0x00 - Internal PFlash • 0x01 - QuadSPI memory

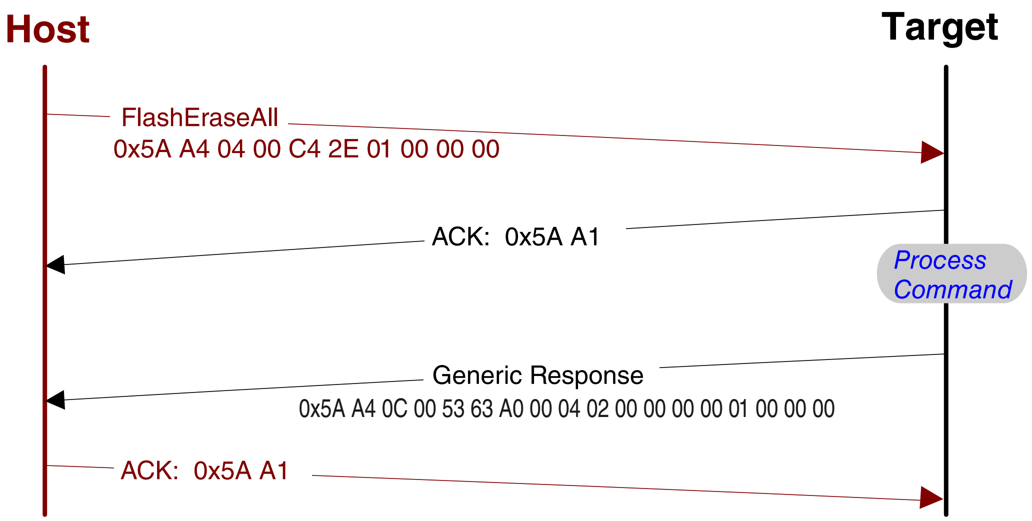

Host
Target
FlashEraseAll
0x5A A4 04 00 C4 2E 01 00 00 00
ACK: 0x5A A1
Process Command
Generic Response
0x5A A4 0C 00 53 63 A0 00 04 02 00 00 00 00 01 00 00 00
ACK: 0x5A A1

Figure 19-8. Protocol Sequence for FlashEraseAll Command

Table 19-18. 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
	MemoryID	• If MemoryID = 0x00h, then internal flash. • If MemoryID = 0x01h, then QSPI0 memory.

The FlashEraseAll command has no data phase.

Response: The target (Kinetis Flashloader) 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.

19.3.6.3 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 4-byte aligned ([1:0] = 00), 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 19-19. Parameters for FlashEraseRegion Command

Byte #	Parameter
0 - 3	Start address
4 - 7	Byte count

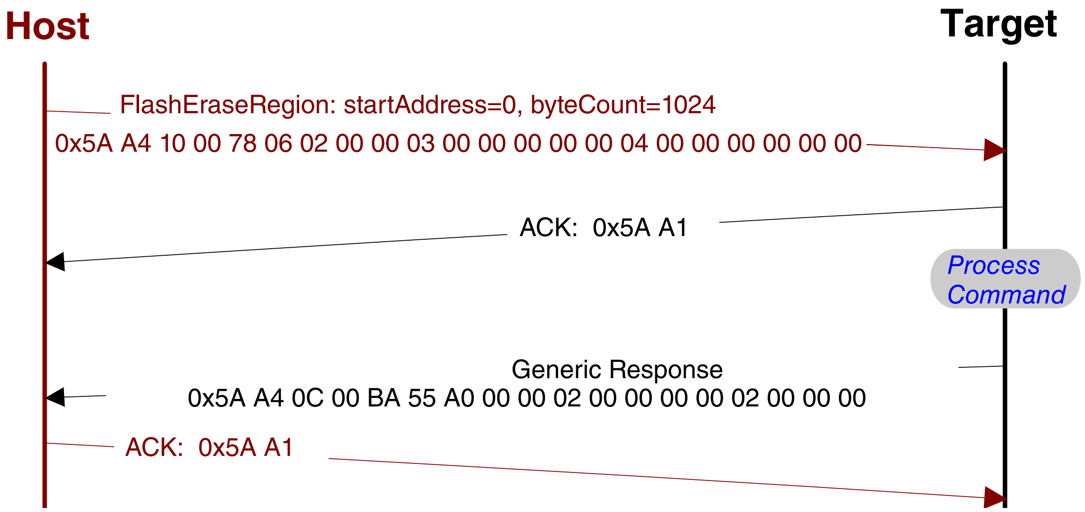

Host
Target
FlashEraseRegion: startAddress=0, byteCount=1024
0x5A A4 10 00 78 06 02 00 00 03 00 00 00 00 00 04 00 00 00 00 00 00
ACK: 0x5A A1
Process Command
Generic Response
0x5A A4 0C 00 BA 55 A0 00 00 02 00 00 00 00 02 00 00 00
ACK: 0x5A A1

Figure 19-9. Protocol Sequence for FlashEraseRegion Command

Table 19-20. FlashEraseRegion Command Packet Format (Example)

FlashEraseRegion	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0x78 0x06
Command packet	commandTag	0x02, kCommandTag_FlashEraseRegion
	flags	0x00
	reserved	0x00
	parameterCount	0x03
	startAddress	0x00 0x00 0x00 0x00 (0x0000_0000)
	byte count	0x00 0x04 0x00 0x00 (0x400)
	memory_id	0x00 0x00 0x00 0x00 (internal flash)

The FlashEraseRegion command has no data phase.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with one of following error status codes.

Table 19-21. FlashEraseRegion Response Status Codes

Status Code	Description
kStatus_FLASH_Success (0)	Executed successfully
kStatusMemoryRangeInvalid (10200)	Memory range is invalid
kStatus_FLASH_InvalidArgument (4)	Invalid argument
kStatus_FLASH_AlignmentError (101)	Parameter is not aligned with the specified baseline
kStatus_FLASH_AddressError (102)	Address is out of range
kStatus_FLASH_EraseKeyError (107)	Erase key is invalid
kStatus_FLASH_AccessError (103)	Invalid instruction codes and out-of bound addresses
kStatus_FTFx_ProtectionViolation (104)	The program/erase operation is requested to execute on protected areas
kStatus_FLASH_CommandFailure (105)	Run-time error during the command execution

19.3.6.4 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.

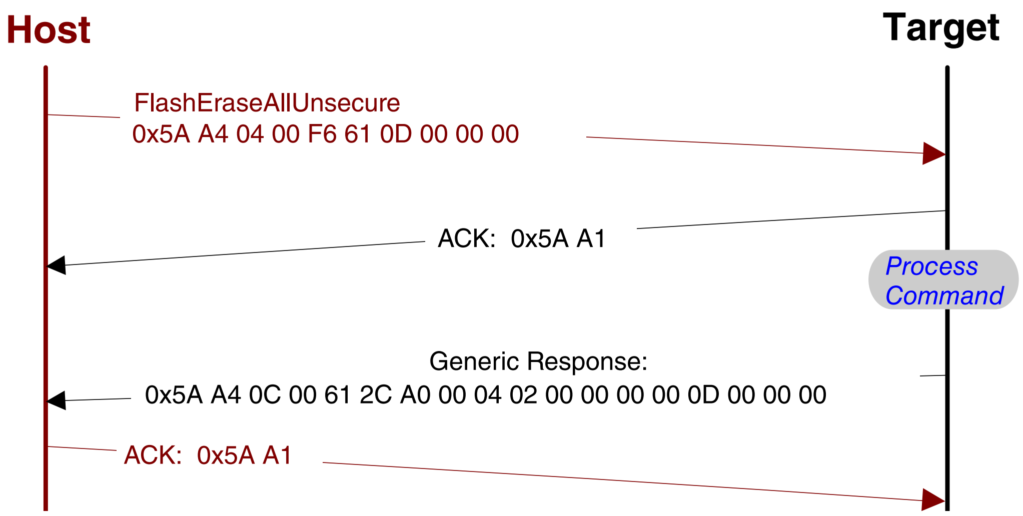


Figure 19-10. Protocol Sequence for FlashEraseAllUnsecure Command

Table 19-22. 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 Flashloader) 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.

19.3.6.5 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 or FlashEraseRegion command.
- Writing to flash requires the start address to be 4-byte aligned ([1:0] = 00).
- 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 19-23. Parameters for WriteMemory Command

Byte #	Command
0 - 3	Start address
4 - 7	Byte count

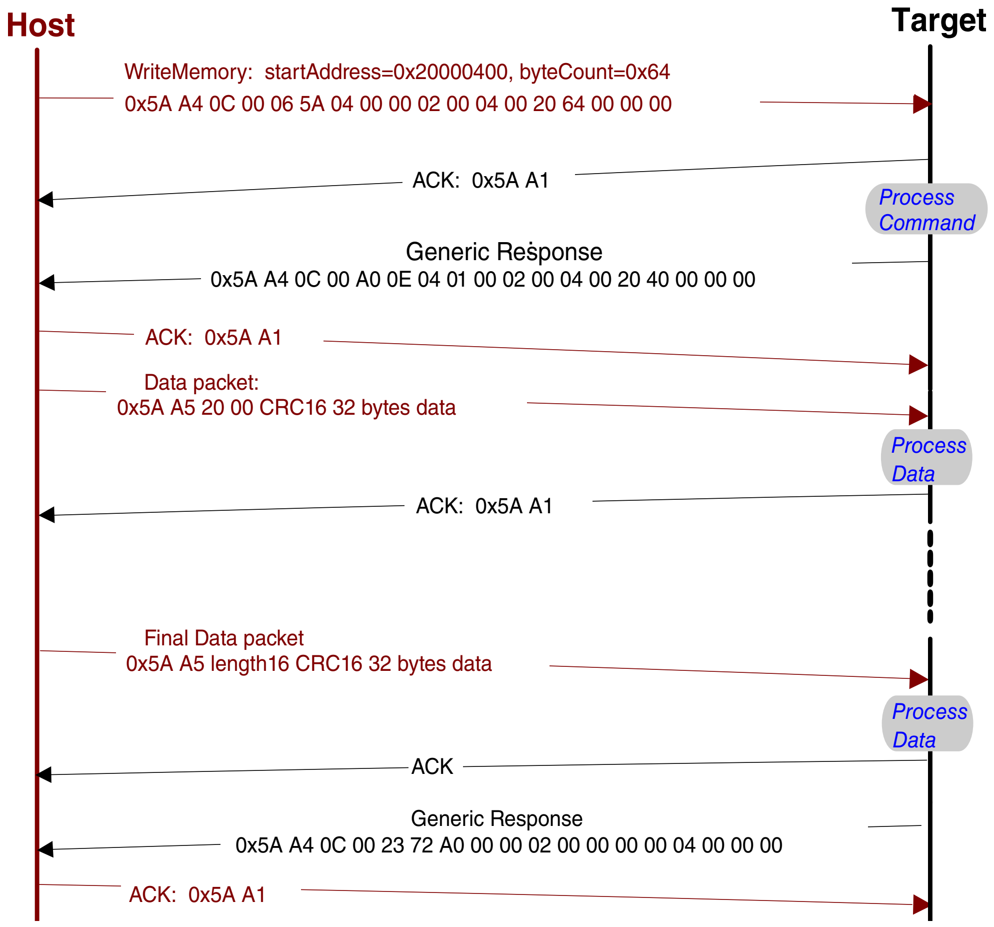

Host
Target
WriteMemory: startAddress=0x20000400, byteCount=0x64
0x5A A4 0C 00 06 5A 04 00 00 02 00 04 00 20 64 00 00 00
ACK: 0x5A A1
Process Command
Generic Response
0x5A A4 0C 00 A0 0E 04 01 00 02 00 04 00 20 40 00 00 00
ACK: 0x5A A1
Data packet:
0x5A A5 20 00 CRC16 32 bytes data
Process Data
ACK: 0x5A A1
Final Data packet
0x5A A5 length16 CRC16 32 bytes data
Process Data
ACK
Generic Response
0x5A A4 0C 00 23 72 A0 00 00 02 00 00 00 00 04 00 00 00
ACK: 0x5A A1

Figure 19-11. Protocol Sequence for WriteMemory Command

Table 19-24. 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 Flashloader) 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.

19.3.6.6 ReadMemory 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 Flash memory, SRAM_L and SRAM_U 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 19-25. Parameters for ReadMemory 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

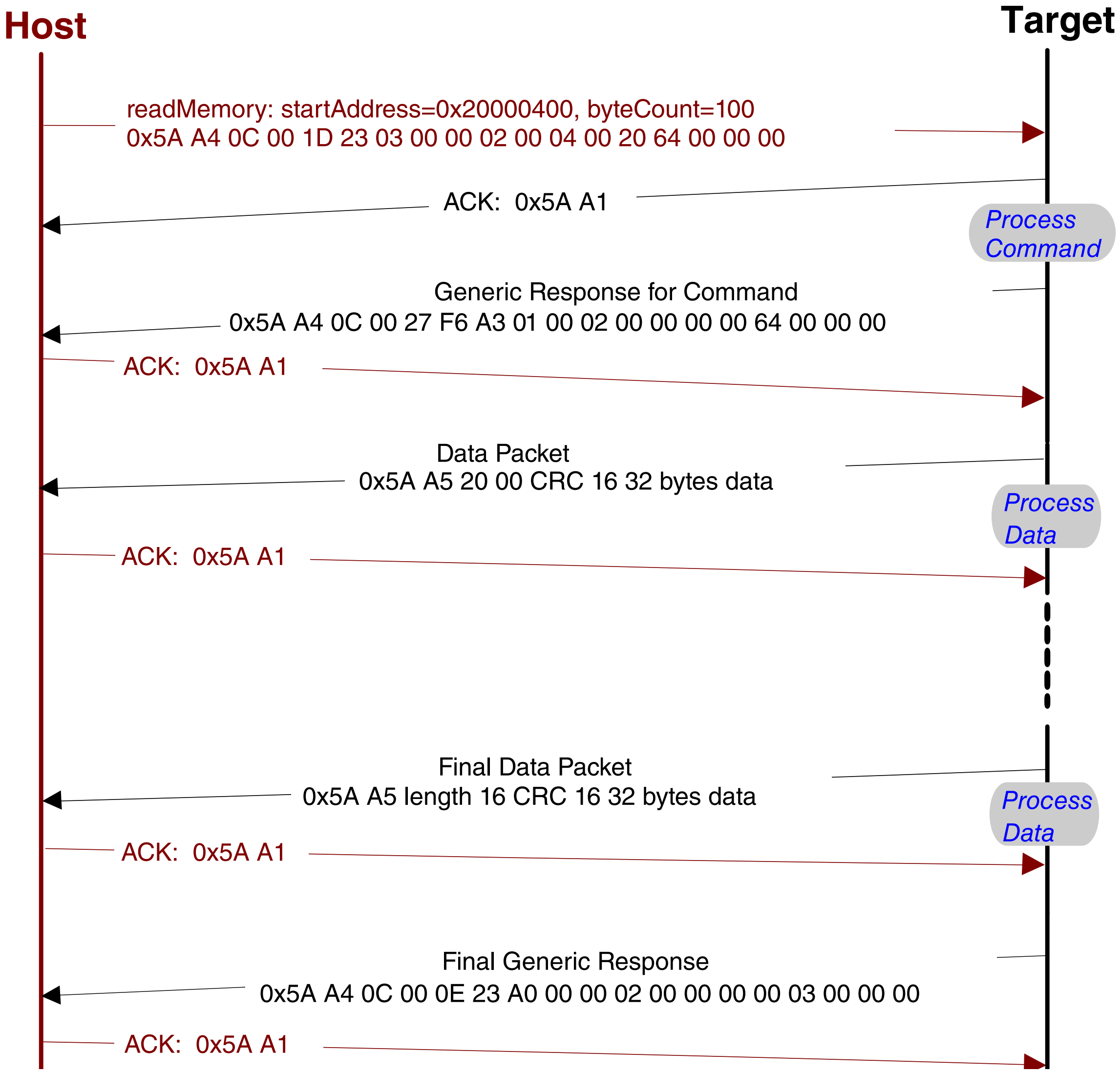

Host
Target
readMemory: startAddress=0x20000400, byteCount=100
0x5A A4 0C 00 1D 23 03 00 00 02 00 04 00 20 64 00 00 00
ACK: 0x5A A1
Process Command
Generic Response for Command
0x5A A4 0C 00 27 F6 A3 01 00 02 00 00 00 00 64 00 00 00
ACK: 0x5A A1
Data Packet
0x5A A5 20 00 CRC 16 32 bytes data
Process Data
ACK: 0x5A A1
Final Data Packet
0x5A A5 length 16 CRC 16 32 bytes data
Process Data
ACK: 0x5A A1
Final Generic Response
0x5A A4 0C 00 0E 23 A0 00 00 02 00 00 00 00 03 00 00 00
ACK: 0x5A A1

Figure 19-12. Command sequence for ReadMemory

Table 19-26. ReadMemory Command Packet Format (Example)

ReadMemory	Parameter	Value
Framing packet	Start byte	0x5A 0xA4
	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 Flashloader) 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 Flashloader) 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.

Table 19-27. ReadMemory Response Status Codes

Status Code	Description
kStatus_FLASH_Success (0)	Executed successfully
kStatus_OutOfRange (3)	Address is out of memory range
kStatusMemoryRangeInvalid (10200)	Memory range is invalid

19.3.6.7 Execute command

The execute command results in the flashloader 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 19-28. 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 Flashloader) will validate the parameters and return a GenericResponse packet with a status code either set to kStatus_Success or an appropriate error status code.

19.3.6.8 Reset command

The Reset command will result in flashloader resetting the chip.

The Reset command requires no parameters.

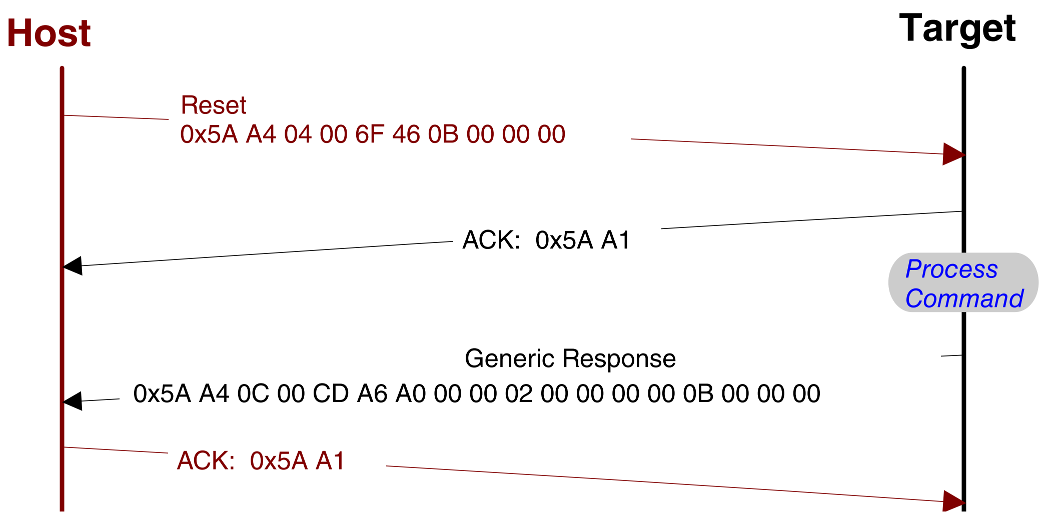

Host
Target
Reset
0x5A A4 04 00 6F 46 0B 00 00 00
ACK: 0x5A A1
Process Command
Generic Response
0x5A A4 0C 00 CD A6 A0 00 00 02 00 00 00 00 0B 00 00 00
ACK: 0x5A A1

Figure 19-13. Protocol Sequence for Reset Command

Table 19-29. 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 Flashloader) will return a GenericResponse packet with status code set to kStatus_Success, before resetting the chip.

19.4 Peripherals Supported

This section describes the peripherals supported by the Kinetis Flashloader.

19.4.1 LPI2C Peripheral

The Kinetis Flashloader 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.

The Kinetis Flashloader uses 0x10 as the LPI2C slave address, and supports 400 kbps as the LPI2C baud rate.

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.

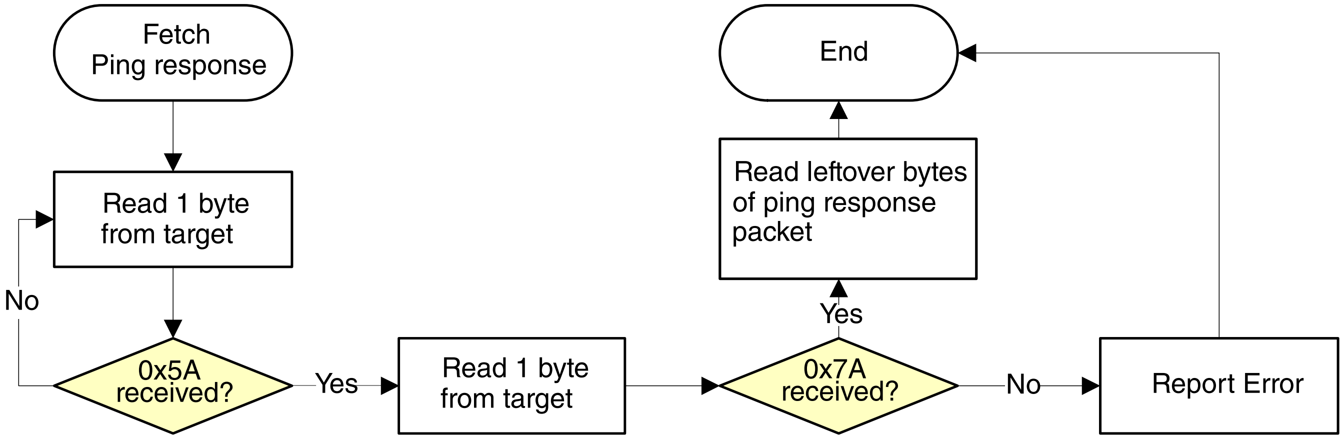


Figure 19-14. Host reads ping response from target via LPI2C

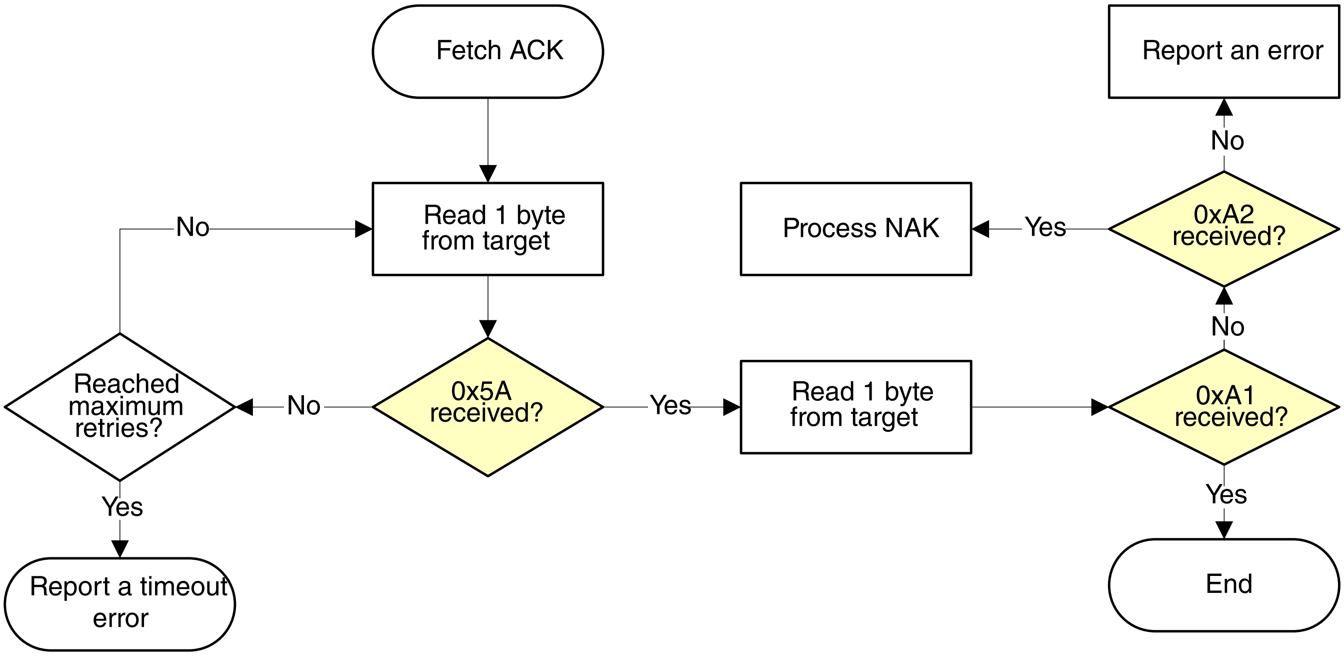


Figure 19-15. Host reads ACK packet from target via LPI2C

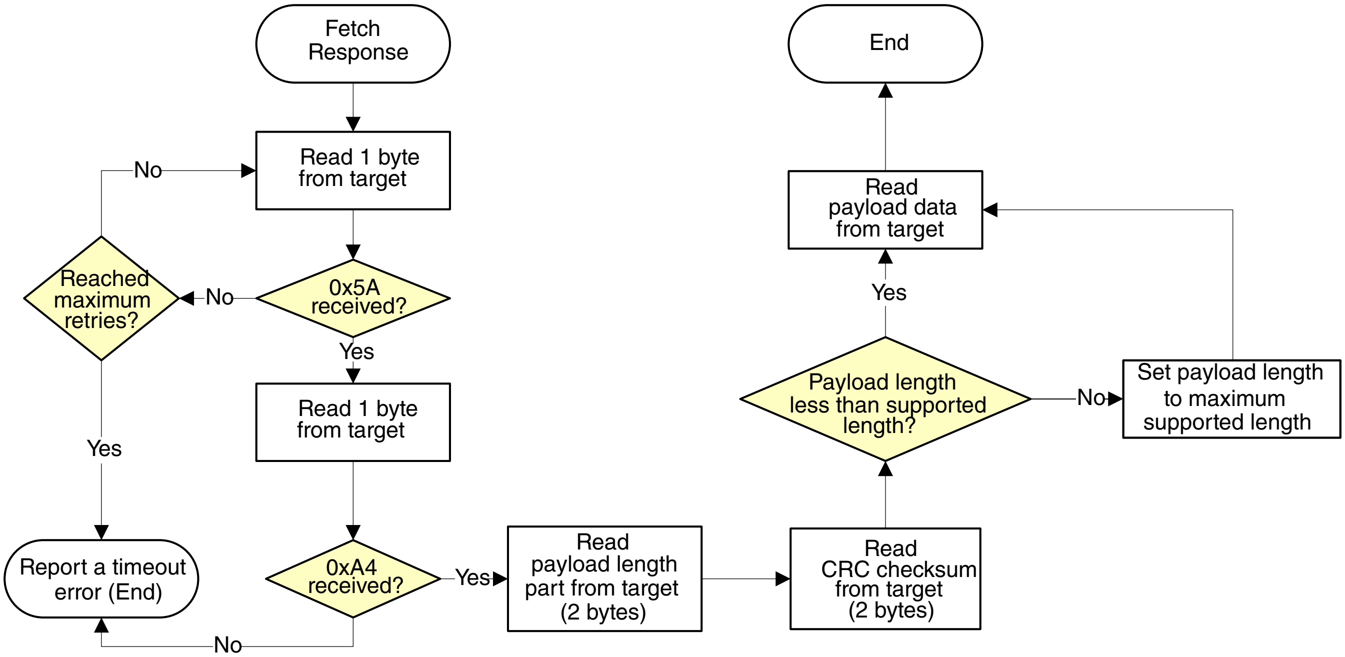

Fetch Response
Read 1 byte from target
0x5A received?
No
Reached maximum retries?
No
Yes
Report a timeout error (End)
Yes
Read 1 byte from target
0xA4 received?
No
Yes
Read payload length part from target (2 bytes)
Read CRC checksum from target (2 bytes)
Payload length less than supported length?
No
Set payload length to maximum supported length
Yes
Read payload data from target
End

Figure 19-16. Host reads response from target via LPI2C

19.4.2 LPSPI Peripheral

The Kinetis Flashloader supports loading data into flash via the LPSPI peripheral, where the LPSPI peripheral serves as a LPSPI slave.

The Kinetis Flashloader supports 400 kbps as the LPSPI baud rate.

The LPSPI peripheral uses the following bus attributes:

- Clock Phase = 1 (Second Edge)
- Clock Polarity = 1 (Active Low)

Because the LPSPI peripheral serves as a SPI 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.

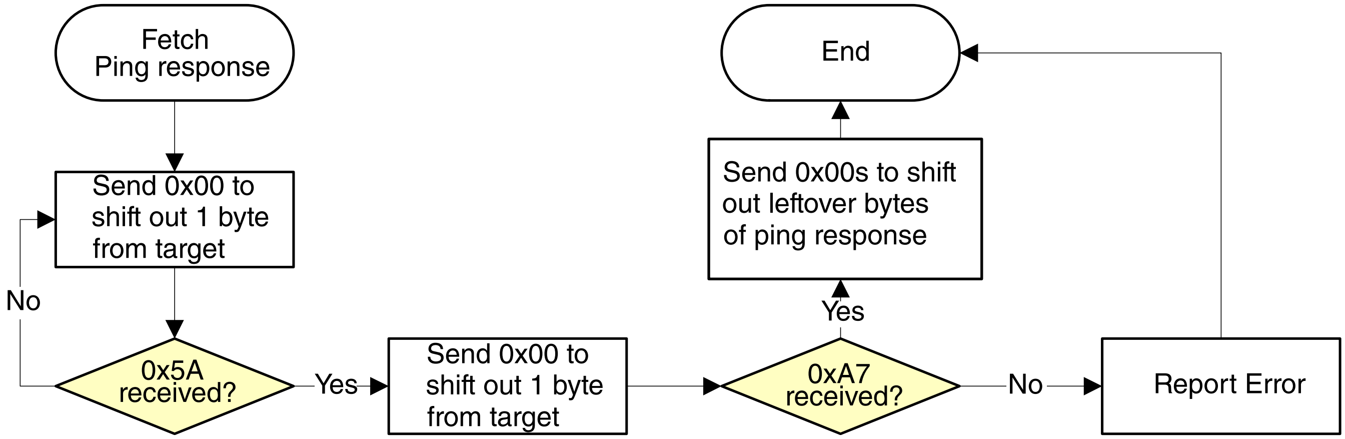


Figure 19-17. Host reads ping packet from target via LPSPI

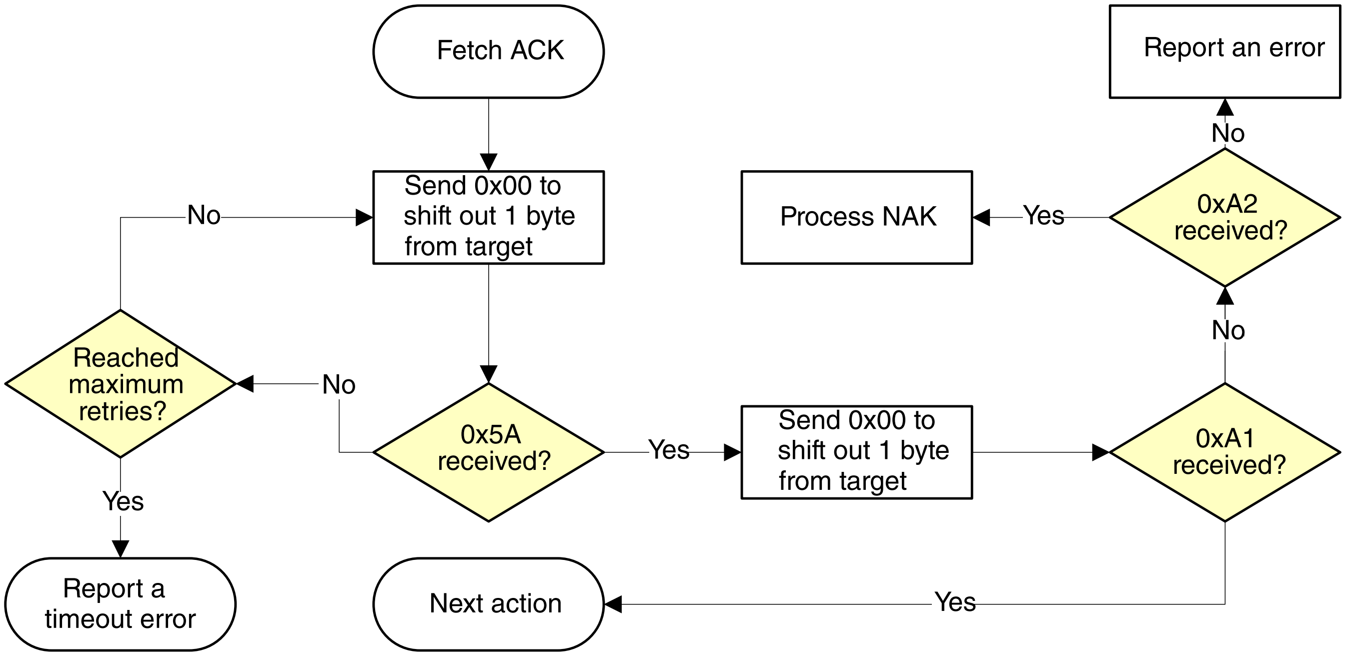


Figure 19-18. Host reads ACK from target via LPSPI

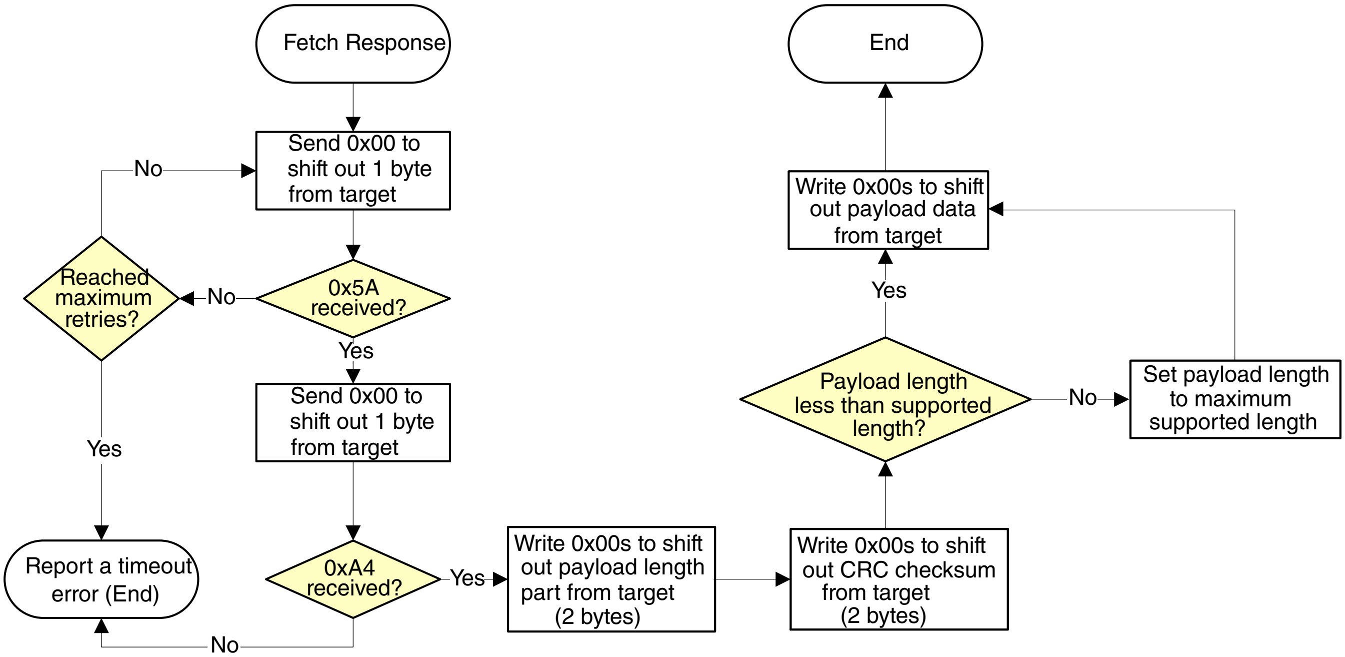

Fetch Response
Send 0x00 to shift out 1 byte from target
No
Reached maximum retries?
No
0x5A received?
Yes
Send 0x00 to shift out 1 byte from target
Yes
Report a timeout error (End)
0xA4 received?
No
Yes
Write 0x00s to shift out payload length part from target (2 bytes)
Write 0x00s to shift out CRC checksum from target (2 bytes)
Payload length less than supported length?
No
Set payload length to maximum supported length
Yes
Write 0x00s to shift out payload data from target
End

Figure 19-19. Host reads response from target via LPSPI

19.4.3 LPUART Peripheral

The Kinetis Flashloader 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 flashloader, 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 flashloader detects the ping packet (0x5A 0xA6) on LPUART*n*_RX, the flashloader firmware executes the autobaud sequence. If the baudrate is successfully detected, then the flashloader 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 Flashloader then enters a loop, waiting for flashloader 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.

Packet transfer: After autobaud detection succeeds, flashloader 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

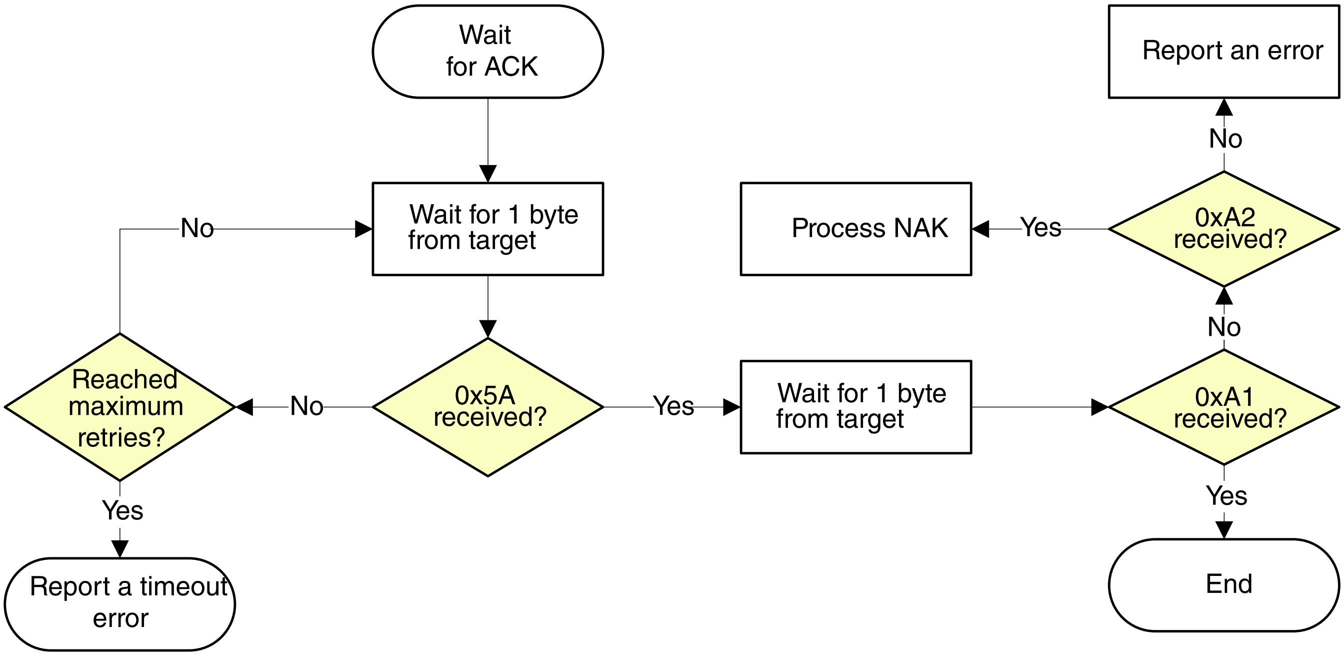


Figure 19-20. Host reads an ACK from target via LPUART

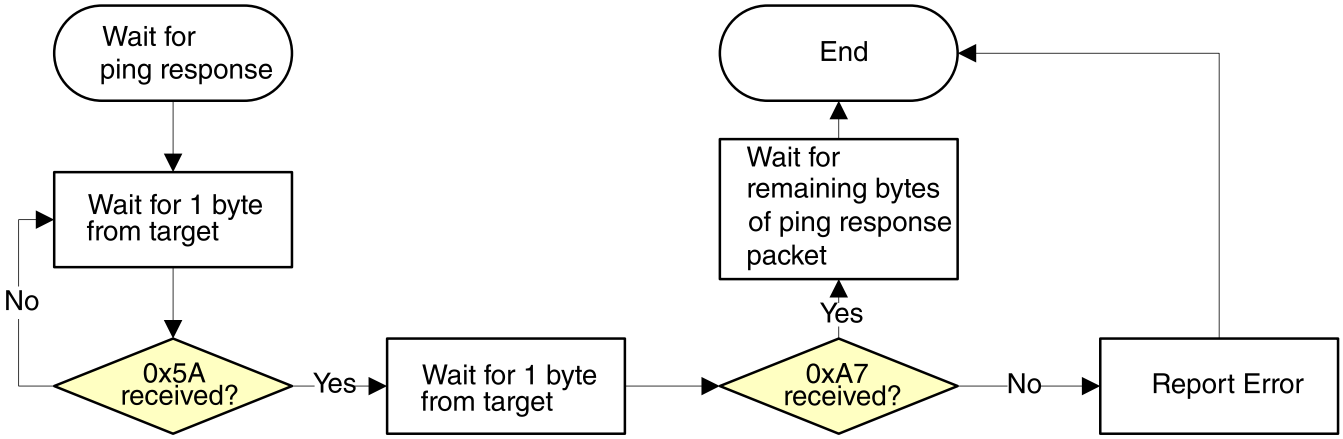


Figure 19-21. Host reads a ping response from target via LPUART

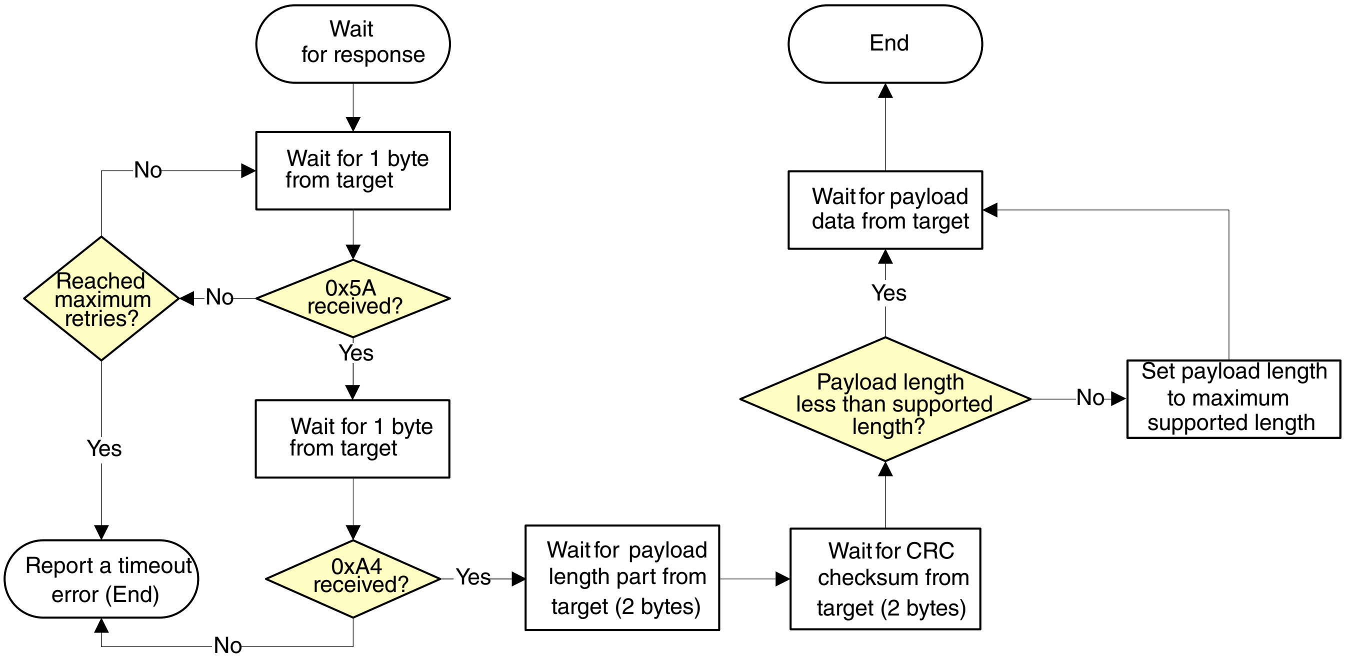


Figure 19-22. Host reads a command response from target via LPUART

19.5 GetProperty Command Properties

This section lists the properties of the GetProperty commands.

Table 19-30. Properties used by GetProperty Commands, sorted by Value

Property	Writable	Tag Value	Size	Description
CurrentVersion	No	01h	4	Current flashloader version.
AvailablePeripherals	No	02h	4	The set of peripherals supported on this chip.
FlashSizeInBytes	No	04h	4	Size in bytes of program flash.
AvailableCommands	No	07h	4	The set of commands supported by the flashloader.
MaxPacketSize	No	0Bh	4	Maximum supported packet size for the currently active peripheral interface.
RAMSizeInBytes	No	0Fh	4	Size in bytes of RAM segment. The first parameter to GetProperty command identifies the segment. See the device specific memory map for number of RAM segments the device contains.

19.5.1 Property Definitions

Get/Set property definitions are provided in this section.

19.5.1.1 CurrentVersion Property

The value of this property is a 4-byte structure containing the current version of the flashloader.

Table 19-31. Fields of CurrentVersion property:

Bits	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'K' (0x4B)	Major version	Minor version	Bugfix version

19.5.1.2 AvailablePeripherals Property

The value of this property is a bitfield that lists the peripherals supported by the flashloader and the hardware on which it is running.

Table 19-32. Peripheral bits:

Bit	[31:7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Peripheral	Reserved	Reserved	Reserved	Reserved	Reserved	SPI 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.

19.5.1.3 AvailableCommands Property

This property value is a bitfield with set bits indicating the commands enabled in the flashloader. 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 19-33. 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	Reserved	Reserved	Reserved	FlashEraseAllUnsecure	Reserved	Reset	Reserved	Execute	Reserved	GetProperty	Reserved	Reserved	WriteMemory	ReadMemory	FlashEraseRegion	FlashEraseAll

19.6 Kinetis Flashloader Status Error Codes

This section describes the status error codes that the Kinetis Flashloader returns to the host.

Table 19-34. Kinetis Flashloader 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 FTFE_FSTAT[ACCERR] bit is set.
kStatus_FlashProtectionViolation	104	The FTFE_FSTAT[FPVIOL] bit is set.
kStatus_FlashCommandFailure	105	The FTFE_FSTAT[MGSTAT0] bit is set.
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.

Table continues on the next page...

Table 19-34. Kinetis Flashloader Status Error Codes, sorted by Value (continued)

Error Code	Value	Description
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.
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.
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 20
Reset Control Module (RCM)

20.1 Chip-specific information for this module

20.1.1 Instantiation Information

20.1.1.1 Information of RCM on this device

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. A bus error will generate a hard fault interrupt on this device.

NOTE

High-Voltage Detect (HVD) is not supported on this device. Therefore, HVD related descriptions are not applicable in RCM_SRS[LVD].

20.2 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.

20.3 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	20.3.1/448
4007_F008	System Reset Status Register (RCM_SRS)	32	R	0000_0082h	20.3.2/449
4007_F00C	Reset Pin Control register (RCM_RPC)	32	R/W	0000_0000h	20.3.3/451
4007_F018	Sticky System Reset Status Register (RCM_SSRS)	32	R/W	0000_0082h	20.3.4/453
4007_F01C	System Reset Interrupt Enable Register (RCM_SRIE)	32	R/W	0000_0000h	20.3.5/455

20.3.1 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.

Table continues on the next page...

RCM_VERID field descriptions (continued)

Field	Description
	0x0003 Standard feature set.

20.3.2 System Reset Status Register (RCM_SRS)

This register includes read-only status flags to indicate the source of the most recent reset. Note that multiple flags can be set if multiple reset events occur at the same time. 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
- 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	0
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. 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.

Table continues on the next page...

RCM_SRS field descriptions (continued)

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

20.3.3 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

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			RSTFLTSEL					0					RSTFLTSS	RSTFLTSRW	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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. 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

20.3.4 Sticky System Reset Status Register (RCM_SSRS)

This register includes status flags to indicate all reset sources since the last POR or LVD 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

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	SSACKERR	0	SMDM_AP	SSW	SLOCKUP	0	SPOR	SPIN	SWDOG	0	SLOL	SLOC	SLVD	0
W			w1c		w1c	w1c	w1c		w1c	w1c	w1c		w1c	w1c	w1c	
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0

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.
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

Table continues on the next page...

RCM_SSRS field descriptions (continued)

Field	Description
	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 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.

Table continues on the next page...

RCM_SSRS field descriptions (continued)

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

20.3.5 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. The SRS updates only after the system reset occurs.

NOTE

This register is reset on Chip POR only, it is unaffected by other reset types.

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.
10 SW	Software Interrupt 0 Interrupt disabled. 1 Interrupt enabled.
9 LOCKUP	Core Lockup Interrupt **NOTE:** The LOCKUP bit is useful only in devices with more than one core processor. 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. Note that the individual interrupt-enable bits still need to be set to generate interrupts.
6 PIN	External Reset Pin Interrupt 0 Reset not caused by external reset pin 1 Reset caused by external reset pin

Table continues on the next page...

RCM_SRIE field descriptions (continued)

Field	Description
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 21
Power Management

21.1 Introduction

This chapter describes the various chip power modes and functionality of the individual modules in these modes. Following stated are general power modes, which are supported additionally by certain clocking mode options. Clock gating technique is used for general power modes and for the additional clocking mode options.

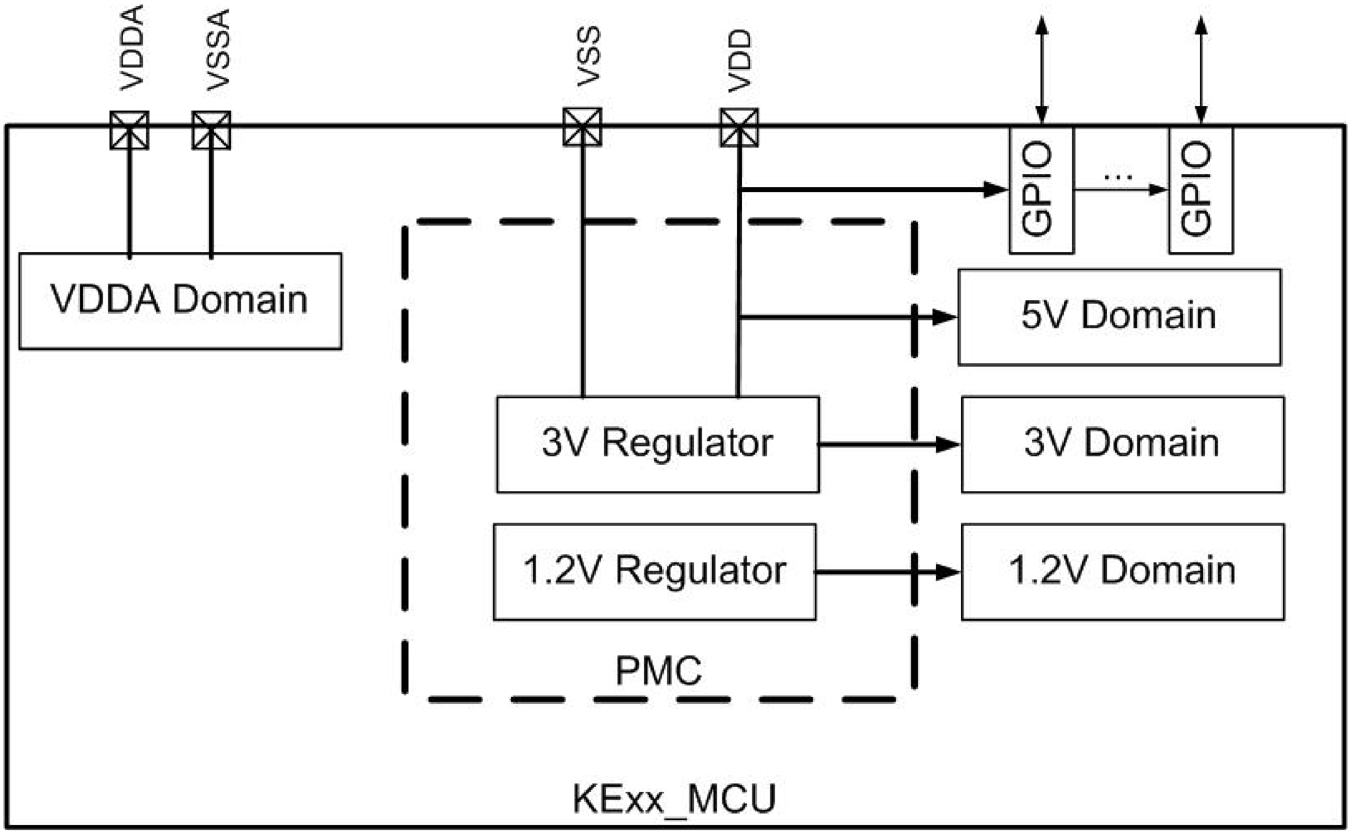


Figure 21-1. Power Infrastructure

21.2 Power Modes Description

The power management controller (PMC) 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 Run and VLPR 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.

Stop mode entry is not supported directly from HSRUN and requires transition to Run prior to an attempt to enter a stop mode.

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 21-1. Chip power modes

Chip mode	Description	Core mode	Normal recovery method
Normal Run	Default mode out of reset; on-chip voltage regulator is on.	Run	-
High Speed Run	Allows maximum performance of chip. In this state, the MCU is able to operate at a faster frequency compared to normal run mode.	Run	-
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
Normal Stop - via WFI	Places chip in static state. On-chip voltage regulator is in a low power mode. LVD is off while maintaining LVR and POR protection. NVIC is disabled; AWIC is used to wake up from interrupt; Peripheral clocks are stopped. All SRAM is operating (content retained and I/O state held). ADC and CMP are optional on.	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 (); LVD off; internal oscillator provides a low power MHz source for the core, the bus and the peripheral clocks.	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

Table continues on the next page...

Table 21-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
VLPS (Very Low Power Stop)-via WFI	Same as Stop mode, but PMC_REGSC register provides options to gate off unused modules and further reduce power in low power mode.	Sleep Deep	Interrupt

21.2.1 Run mode

Run mode is the default mode after reset, and refers to any mode in which CPU execution is possible. Depending on the on-chip regulator settings, Run mode has the following configurations:

- HSRUN mode — The on-chip regulator voltage output is slightly elevated. The 1.2V domain is powered by 1.4V instead. This allows the MCU digital modules to operate at a faster frequency.
- Normal RUN mode — The on-chip regulator voltage output is normal. The 1.2V domain is powered by 1.2V. This allows the MCU digital modules to operate at a normal frequency.
- Very Low Power RUN mode — The on-chip regulator voltage is in Low Power mode. The MCU digital modules should operate at a limited frequency but with much lower power.

Run mode configurations can be selected by configuring SMC_PMCTRL.

The following sections describe optimizing power in Run modes.

21.2.1.1 Clock Gating

To conserve power, the clocks to most modules can be turned off using CGC bit of the peripheral control registers in the PCC module. These bits are cleared after any reset, which disables the clock to the corresponding module. Prior to initializing a module, set the corresponding bit in the PCC peripheral control register to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution and PCC chapters.

21.2.1.2 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 Run mode, HSRUN mode, or VLPR 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 VLPR mode, module functionality for bus masters and slaves is the equivalent of VLPS mode. 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 bus (PPB) remains accessible during Compute Operation, including the MCM, System Control Space (SCS) (for NVIC), and SysTick. Although access to the GPIO registers 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, which is only accessible to the CPU. Setting or clearing the CPOREQ bit in the MCM 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 bit indicates when entry has completed. When exiting Compute Operation in Run mode, the CPOACK status bit negates immediately. When exiting Compute Operation in VLPR mode, the exit is delayed to allow the PMC to handle the change in power consumption. This delay means the CPOACK bit is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

The DMA wakeup is also supported during Compute Operation and causes the CPOACK status bit to clear and the AIPS peripheral space to be accessible for the duration of the DMA wakeup. At the completion of the DMA wakeup, the device transitions back into Compute Operation.

21.2.2 Wait mode

Wait mode refers to a power modes in which the CPU execution is halted. The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate.

Depending on the on-chip regulator settings, Wait mode has the following configurations:

- Normal Wait mode — The on-chip regulator voltage output is normal. The 1.2V domain is powered by 1.2V. This allows the MCU digital modules to operate at a normal frequency.
- Very Low Power Wait mode — The on-chip regulator voltage is in Low Power mode. The MCU digital modules must operate at a limited frequency but with much lower power.

After the CPU executes the WFI/WFE instruction, VLPW mode is entered when MCU is in VLPR mode and Normal Wait mode is entered when MCU is in Normal Run mode. Run mode configurations can be selected by configuring SMC_PMCTRL.

Clock gating can be used to optimize the power in Wait mode. Any interrupt can be used as a wake up source from the Wait mode. See the "Interrupt vector assignments" table in Interrupts chapter for all the available interrupt sources.

21.2.3 Stop mode

Stop mode refers to power modes in which the CPU and most peripherals are static. The SRAM and all registers are retained. The core clock, system clock, and the bus clock are gated off. NVIC is disabled; AWIC is used to wake up from interrupt. In the Stop mode, some peripherals can remain operational with asynchronous clock and can wake up the MCU as needed.

Stop mode configurations can be selected by configuring SMC_PMCTRL.

In Stop mode, the bus clock is gated as core clock and system clock. This device supports a partial Stop mode that permits peripherals to run with the bus clock.

21.2.3.1 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.

Any AWIC interrupt can be used as a wake up source from Stop (normal Stop and VLPS) mode. See Table 21-5 for all the available wake up source. Besides waking up the CPU from Stop mode, the DMA can perform data transfer while retaining the CPU in Low Power mode.

21.2.3.2 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, 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.

21.2.4 Power domains

The following table describe the power domain of this device.

Table 21-2. Power domain summary

Domain name	Description
5V	5V domain is powered by VDD/VSS directly. It contains GPIO and PMC.
VDDA	Analog domain is powered by VDDA/VSSA. It contains analog modules such as ADC and CMP.

Table continues on the next page...

Table 21-2. Power domain summary (continued)

Domain name	Description
3V	3V domain is powered by the PMC 3V regulator. It contains TSI, OSC, and Flash memory.
1.2V	1.2V domain is powered by the PMC 1.2V regulator. It contains all digital logics and SRAM.

Table 21-3. Module power domain summary

VDD (5V)	
PMC	GPIOx (all ports)
VDDA	
ADC	CMP
3V CORE	
TSI*x*	OSC
Flash Memory	
1.2V	
Cortex-M0+ Core	DMAMUX
SRAM	EWM
SCG	WDOG
PCC	CRC
AXBS-Lite	FlexIO
SCI	LPI2C
SIM	LPSPI
RCM	LPUARTx
MCM	LPIT
MTB	FTMx
AIPS-Lite	LPTMRx
AWIC	PORTx
eDMA	TRGMUX*x*

21.2.5 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. The "Interrupt vector assignments" table in the Interrupts chapter describes interrupt operation and what peripherals can cause interrupts.

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.

21.3 Power mode transitions

The following figure shows the power mode transitions. Any reset always brings the chip back to the normal run state. In run, wait, and stop modes active power regulation is enabled. The VLPx modes offer a lower power operating mode than normal modes. VLPR and VLPW are limited in frequency.

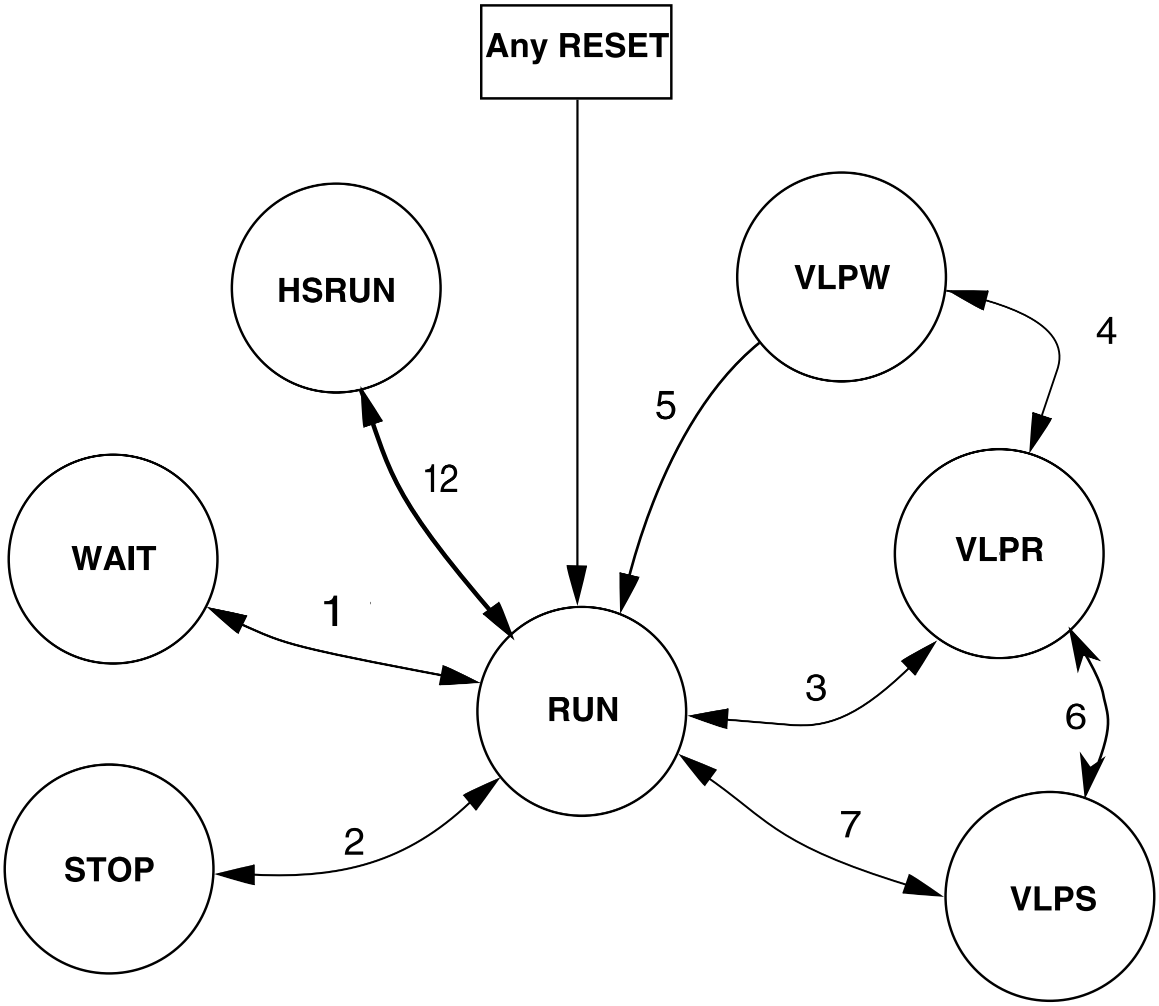


Figure 21-2. Power mode state transition diagram

NOTE

See Table 22-1 in the SMC chapter for more detailed mode transition conditions.

21.4 Power modes shutdown sequencing

When entering stop or other low-power modes, the clocks are shut off in an orderly sequence to safely place the chip in the targeted low-power state. All low-power entry sequences are initiated by the core executing an WFI instruction. The ARM core's outputs, SLEEPDEEP and SLEEPING, trigger entry to the various low-power modes:

- System level wait and VLPW modes equate to: SLEEPING & $\overline{\text{SLEEPDEEP}}$
- All other low power modes equate to: SLEEPING & SLEEPDEEP

When entering the non-wait modes, the chip performs the following sequence:

- Shuts off Core Clock and System Clock to the ARM Cortex-M core immediately.
- Polls stop acknowledge indications from the non-core crossbar masters (DMA), supporting peripherals (SPI, PIT) and the Flash Controller for indications that System Clocks, Bus Clock and/or Flash Clock need to be left enabled to complete a previously initiated operation, effectively stalling entry to the targeted low power mode. When all acknowledges are detected, System Clock, Bus Clock and Flash Clock are turned off at the same time.
- SCG and Mode Controller shut off clock sources and/or the internal supplies driven from the on-chip regulator as defined for the targeted low power mode.

In wait modes, most of the system clocks are not affected by the low power mode entry. The Core Clock to the ARM Cortex-M core is shut off. Some modules support stop-in-wait functionality and have their clocks disabled under these configurations.

The debugger modules support a transition from stop, wait, VLPS, and VLPW back to a halted state when the debugger is enabled. This transition is initiated by setting the Debug Request bit in MDM-AP control register. As part of this transition, system clocking is re-established and is equivalent to normal run/VLPR mode clocking configuration.

21.5 Module operation in low power modes

The following table 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 (PSTOP2).

Debug modules are discussed separately, see "Debug in low power modes" in the Debug chapter. Number ratings (such as 2 MHz and 1 Mbit/s) represent the maximum frequencies or maximum data rates per mode. Also, these terms are used:

- 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 operation = Fully functional with alternate clock source, provided the selected clock source remains enabled
- static = Module register states and associated memories are retained.
- powered = Memory is powered to retain contents.
- low power = Memory is powered to retain contents in a lower power state
- OFF = Modules are powered off; module is in reset state upon wakeup. For clocks, OFF means disabled.
- wakeup = Modules can serve as a wakeup source for the chip.

Table 21-4. Module operation in low power modes

Modules	VLPR	VLPW	Stop	VLPS
Core modules				
NVIC	FF	FF	static	static
System modules				
System Mode Controller	FF	FF	FF	FF
Regulator	low power	low power	low power	low power
LVD/LVR	disabled (LVR active only)	disabled (LVR active only)	disabled (LVR active only)	disabled (LVR active only)
POR (Brown-out Detection)	FF	FF	FF	FF
DMA	FF Async operation in CPO	FF	Async operation	Async operation
Watchdog	FF	FF	FF	FF
EWM	FF static in CPO	static	static FF in PSTOP2	static
Clocks				
128 kHz LPO	FF	FF	FF	FF
System oscillator (SOSC)	SOSC_CLK optional ON	SOSC_CLK optional ON	SOSC_CLK optional ON	SOSC_CLK optional ON
SCG	SOSC, SIRC, FIRC, LPFLL optional ON	SOSC, SIRC, FIRC, LPFLL optional ON	SOSC, SIRC, FIRC optional ON	SOSC, SIRC, FIRC optional ON
Core clock	4 MHz max	OFF	OFF	OFF
System clock	4 MHz max OFF in CPO	4 MHz max	OFF	OFF
Bus clock	4 MHz max OFF in CPO	4 MHz max	OFF FF in PSTOP2	OFF
Memory and memory interfaces				
Flash	1 MHz max access - no program/erase	low power	low power	low power

Table continues on the next page...

Table 21-4. Module operation in low power modes (continued)

Modules	VLPR	VLPW	Stop	VLPS
	No register access in CPO			
System RAM (SRAM_U and SRAM_L)	low power [1]	low power	low power	low power
Communication interfaces				
LPUART	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
SCI	FF static in CPO	FF	static	static
LPSPI	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
LPI^{2}C	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
FlexIO	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
Security				
CRC	FF static in CPO	FF	static FF in PSTOP2	static
Timers				
FTM	FF static in CPO	FF	static	static
LPIT	FF static in CPO	FF	Async operation FF in PSTOP2	Async operation
PWT	FF static in CPO	FF	static FF in PSTOP2	static
LPTMR	FF	FF	Async operation FF in PSTOP2	Async operation
RTC	FF Async operation in CPO	FF	Async operation FF in PSTOP2	Async operation
Analog				
12-bit ADC	FF SIRC, FIRC and SOSC clocks only	FF SIRC, FIRC and SOSC clocks only	FF SIRC, FIRC and SOSC clocks only	FF SIRC, FIRC and SOSC clocks only
CMP [2]	LS compare only	LS compare only	LS compare FF in PSTOP2	LS compare only
Human-machine interfaces				
GPIO	FF IOPORT write only in CPO	FF	static output, wakeup input FF in PSTOP2	static output, wakeup input
TSI	FF	FF	Async operation	Async operation

Table 21-4. Module operation in low power modes

Modules	VLPR	VLPW	Stop	VLPS
	Async operation in CPO		FF in PSTOP2	

1. SRAM is writable and readable in VLPR mode.
2. CMP in stop or VLPS supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled and filtered modes of operation are not available while in stop or VLPS modes.

NOTE

- The load current should only change slowly (by peripheral modules being turned on/off, and clock speed being changed) in low power modes.
- Before entering low power modes, the peripheral clock frequencies should be set to desired values, for the modules working in 1.2 V power domain (see Table 21-2, e.g. FlexIO, LPIT, LPTMR and communication modules).

21.5.1 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.

21.6 Low-power wake-up sources

Table 21-5. AWIC Stop and VLPS Wake-up Sources

Wake-up source	Description
Available system resets	RESET pin, WDOG, loss of clock(LOC) reset and loss of lock (LOL) reset
Pin interrupts	Port Control Module - Any enabled pin interrupt is capable of waking the system
ADC	ADC is optional functional with clock source from SIRC, FIRC or OSC
CMP	Functional in Stop/VLPS modes
LPI2C	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPUART	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPSPI	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPIT	Functional in Stop/VLPS modes with clock source from SIRC or OSC
FlexIO	Functional in Stop/VLPS modes with clock source from SIRC or OSC
LPTMR	Functional in Stop/VLPS modes
RTC	Functional in Stop/VLPS modes
SCG	Functional in Stop mode
RCM	Reset wakeup
TSI	Touch sense wakeup
SCI	SCI asynchronous interrupt can wake up the low power mode
NMI	Non-maskable interrupt

21.7 Power supply supervisor

This device integrates the following power supervisor circuits:

- Power-on reset (POR)
- Low voltage detection (LVD)

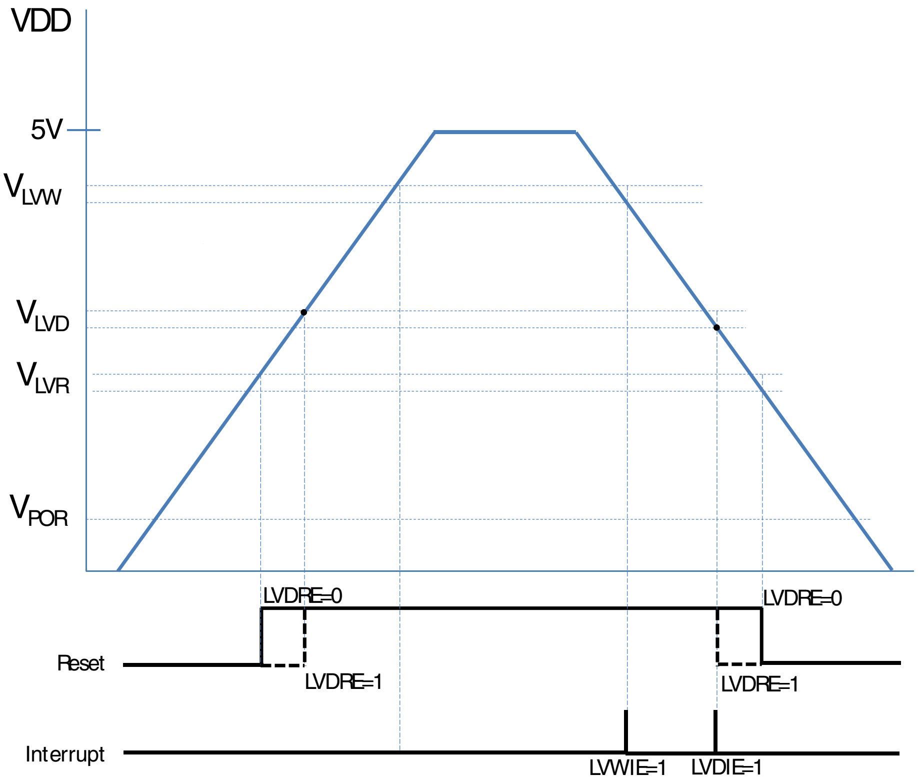


Figure 21-3. Power Supply Supervisor

NOTE

When VDD ramps up above V_{LVR} , the RESET pin is released (indicating MCU quits POR reset state), and it starts to run code immediately.

If LVDRE bit is not operated in code, the LVDRE bit is 0 after POR reset by default, then LVD does not take effect.

If LVDRE bit is configured as 1 in user's code, and the current VDD still below V_{LVD}, then LVD takes effect, and holds MCU in system reset state, keeps the RESET pin low until VDD increases above V_{LVD}.

During power-on, the POR keeps the device under reset until the supply voltage V_{DD} reaches the specified threshold. When V_{DD} is above the threshold, the device reset is released and the system can start.

The LVD circuit can be used to monitor the power supply voltage by comparing it to a configurable threshold. User can choose to generate LVD reset or LVW interrupt when power supply voltage drops below the threshold. See PMC chapters for details.

For more details on the POR/LVD reset and the LVW interrupt thresholds, see the electrical characteristics (LVR, LVD and POR) section in the Data Sheet.

Chapter 22
System Mode Controller (SMC)

22.1 Overview

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.

22.2 Functional description

22.2.1 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.

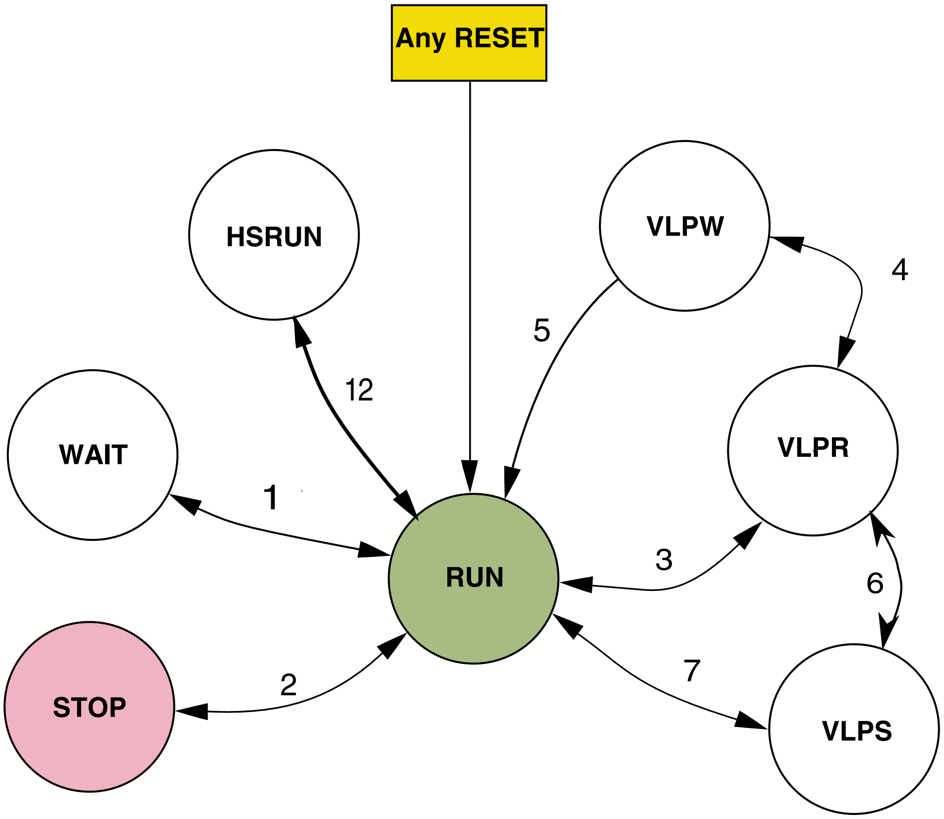


Figure 22-1. Power mode state diagram

The following table defines triggers for the various state transitions shown in the previous figure.

Table 22-1. 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] 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.
	VLPR	RUN	Set PMCTRL[RUNM]=00 or

Table continues on the next page...

Table 22-1. Power mode transition triggers (continued)

Transition #	From	To	Trigger conditions
			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
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

22.2.2 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.

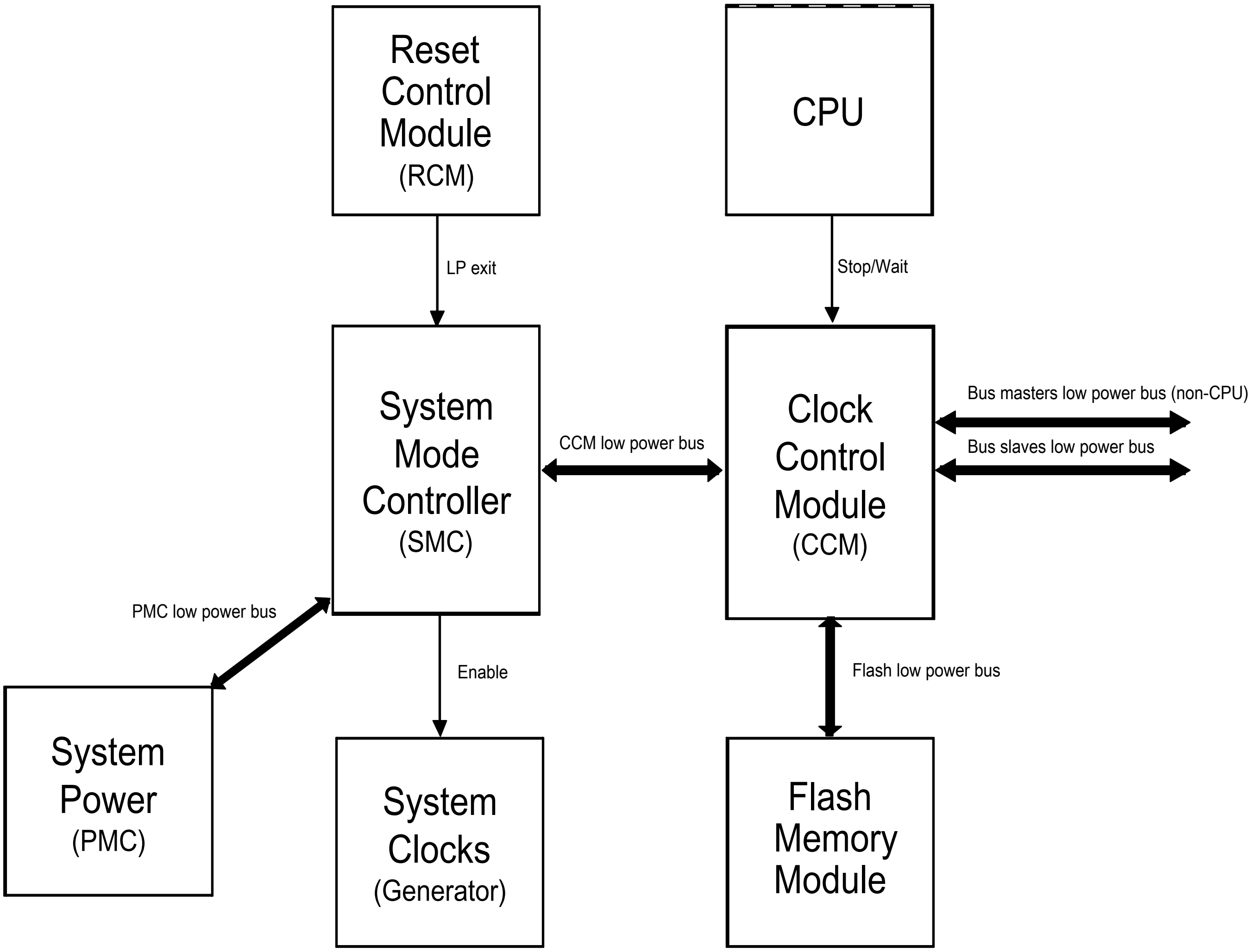

Reset Control Module (RCM)
CPU
LP exit
Stop/Wait
System Mode Controller (SMC)
CCM low power bus
Clock Control Module (CCM)
Bus masters low power bus (non-CPU)
Bus slaves low power bus
PMC low power bus
Enable
Flash low power bus
System Power (PMC)
System Clocks (Generator)
Flash Memory Module

Figure 22-2. Low-power system components and connections

22.2.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS) 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.

22.2.2.2 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.

22.2.2.3 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.

22.2.2.4 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.

22.2.2.5 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.

22.2.3 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 22-2. 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.

Table continues on the next page...

Table 22-2. Power modes (continued)

Mode	Description
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.
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.

22.2.4 Run modes

The run modes supported by this device can be found here.

- Run (RUN)
- Very Low-Power Run (VLPR)
- High Speed Run (HSRUN)

22.2.4.1 RUN mode

This is the normal operating mode for the device.

This mode is selected after any reset. 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.

22.2.4.2 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.

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.

22.2.4.3 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.

22.2.5 Wait modes

This device contains two different wait modes which are listed here.

- Wait
- Very-Low Power Wait (VLPW)

22.2.5.1 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.

22.2.5.2 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.

22.2.6 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)

22.2.6.1 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.

22.2.6.2 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.

22.2.7 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 VLPW 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.

22.2.8 Clocking

Table 22-3. SMC clocks

Clock name	Description
Peripheral bus clock	Clock that controls the registers

22.2.9 Interrupts

This module has no interrupts.

22.3 External signals

This module has no external signals.

22.4 Initialization

This module does not require initialization.

22.5 Application information

If any very-low-power mode (VLPR, VLPW, and VLPS) is needed, you need to write 1 to SMC_PMPROT[AVLP] field.

22.6 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	22.6.1/487
4007_E004	SMC Parameter Register (SMC_PARAM)	32	R	See section	22.6.2/488
4007_E008	Power Mode Protection register (SMC_PMPROT)	32	R/W	0000_0000h	22.6.3/489
4007_E00C	Power Mode Control register (SMC_PMCTRL)	32	R/W	0000_0000h	22.6.4/490
4007_E010	Stop Control Register (SMC_STOPCTRL)	32	R/W	0000_0003h	22.6.5/492
4007_E014	Power Mode Status register (SMC_PMSTAT)	32	R	0000_0001h	22.6.6/494

22.6.1 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

22.6.2 SMC Parameter Register (SMC_PARAM)

Address: 4007_E000h base + 4h offset = 4007_E004h

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	EVLLS0	ELLS2	0	ELLS	0		EHSRUN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

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.

22.6.3 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 and by reset types that trigger Chip Reset. It is unaffected by reset types that do not trigger Chip Reset. 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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								AHSRUN	0	AVLP	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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 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.

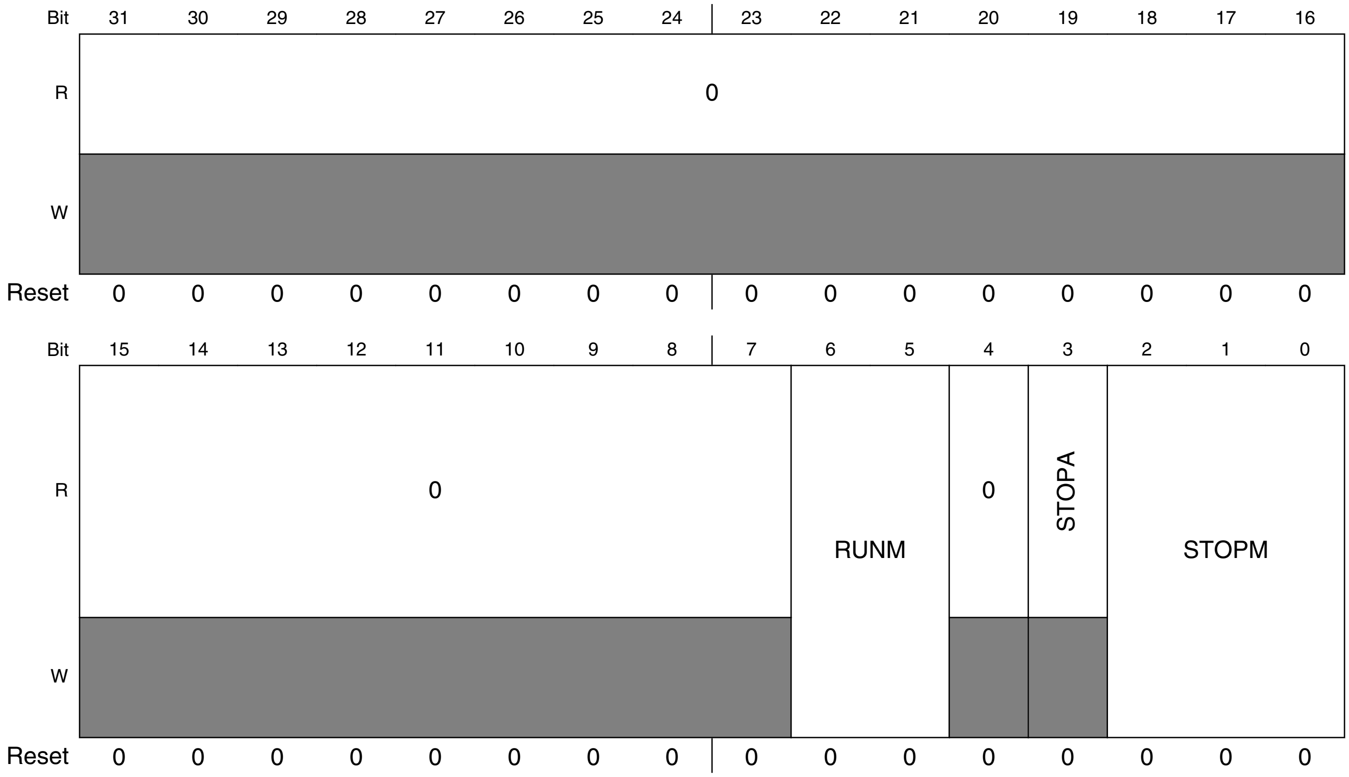
22.6.4 Power Mode Control register (SMC_PMCTRL)

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 and by reset types that trigger Chip POR. It is unaffected by reset types that do not trigger Chip POR. See the Reset section details for more information.

Address: 4007_E000h base + Ch offset = 4007_E00Ch



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 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)

Table continues on the next page...

SMC_PMCTRL field descriptions (continued)

Field	Description
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 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 Reserved 101 Reserved 110 Reseved 111 Reserved

22.6.5 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 and by reset types that trigger Chip POR. It is unaffected by reset types that do not trigger Chip POR. See the Reset section details for more information.

Address: 4007_E000h base + 10h offset = 4007_E010h

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								PSTOPO		0	0	Reserved	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

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 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
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 bit is reserved for future expansion. Software should write 0 to this bit to maintain compatibility.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

22.6.6 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 and by reset types that trigger Chip POR. It is unaffected by reset types that do not trigger Chip POR. 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	0	0	0	0	0	0	0	0	0	0	1

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 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 Reserved 0100_0000 Reserved 1000_0000 Current power mode is HSRUN

Chapter 23
Power Management Controller (PMC)

23.1 Chip-specific Information for this Module

NOTE

If needed in some case, PMC_REGSC[CLKBIASDIS] should be set manually before entering STOP or VLPS mode. See CLKBIASDIS for more information. In the bitfield description, "RPM" is an alias of Low Power Mode (LPM).

23.2 Overview

The PMC contains the internal voltage regulator, power on reset (POR) and the low voltage detect (LVD) system.

23.2.1 Features

The PMC features include:

- Internal voltage regulator offering a variety of power modes
- Active POR providing brown-out detect
- Low voltage reset (LVR)
- Low voltage detect supporting two low voltage trip points and interrupt
- Low power oscillator (LPO) with a typical frequency of 128 kHz

23.3 Functional description

The following sections describe functional details of the PMC module.

23.3.1 Modes of Operation

23.3.1.1 Full Performance Mode (FPM)

For the following Chip Power Modes, the internal voltage regulator is in full performance mode: HSRUN, RUN, STOP1, STOP2.

23.3.1.2 Low Power Mode (LPM)

For the following Chip Power Modes, the internal voltage regulator is in low power mode: VLPR and VLPS.

23.3.2 Low Voltage Detect (LVD) System

NOTE

The low voltage detect system (Low voltage detect flag, Low voltage warning flag and Low voltage detect reset generation) is disabled in low power mode.

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 two trip points. The LVD is disabled upon entering low power mode.

Two flags are available to indicate the status of the low voltage detect system:

- The low voltage detect flag (LVDF) operates in a level sensitive manner. The LVDF bit is set when the supply voltage falls below the trip point (V_{LVD}). The LVDF bit is cleared by writing one to the LVDACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVDF bit remains set. This flag gets cleared on reset. The flag is only valid after the device has come out of the reset, at which point the flag will be set accordingly to the voltage level. If supply level is higher than LVD threshold then this flag stay cleared, else this flag gets set.
- The low voltage warning flag (LVWF) operates in a level sensitive manner. The LVWF bit is set when the supply voltage falls below the selected monitor trip point (V_{LVW}). The LVWF bit is cleared by writing one to the LVWACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVWF bit remains set. This flag gets cleared on reset. The flag is only valid after the device has come

out of the reset, at which point the flag will be set accordingly to the voltage level. If supply level is higher than LVW threshold then this flag stay cleared, else this flag gets set.

23.3.2.1 Low Voltage Reset (LVR) Operation

If the supply voltage falls below the reset trip point (V_{LVR}), a system reset will be generated.

NOTE

This device includes a system to guard against low voltage conditions. This protects memory contents and controls MCU system states during supply voltage variations.

If PMC_LVDSC1[LVDRE] is set and the supply voltage falls below V_{LVD}, a system reset will be generated.

PMC_LVDSC1[LVDF] will be cleared by system reset, so after recovery PMC_LVDSC1[LVDF] will read zero. Usage of PMC_LVDSC1[LVDF] is intended for LVD interrupt opteration only (for example, PMC_LVDSC1[LVDIE] = 1 and PMC_LVDSC1[LVDRE] = 0).

23.3.2.2 LVD Interrupt Operation

By configuring the LVD circuit for interrupt operation (LVDIE set), PMC_LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. The LVDF bit is cleared by writing one to the PMC_LVDSC1[LVDACK] bit, when the supply returns to above the trip point.

23.3.2.3 Low-voltage warning (LVW) interrupt operation

The LVD system contains a low-voltage warning flag (LVWF) 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 the PMC_LVDSC2[LVWIE] bit. If enabled, an LVW interrupt request occurs when the LVWF is set. LVWF is cleared by writing one to the PMC_LVDSC2[LVWACK] bit, when the supply returns to above the trip point.

23.4 Memory Map and Register Definition

This sections provides the detailed information of all registers for the PMC module.

23.4.1 PMC register descriptions

NOTE

Different portions of PMC registers are reset only by particular reset types. Each register's description provides details.

NOTE

The PMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

23.4.1.1 PMC memory map

PMC base address: 4007_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Low Voltage Detect Status and Control 1 (LVDSC1)	8	RW	Table 23-
1h	Low Voltage Detect Status and Control 2 (LVDSC2)	8	RW	00h
2h	Regulator Status and Control (REGSC)	8	RW	Table 23-
4h	Low Power Oscillator Trim (LPOTRIM)	8	RW	Table 23-1

23.4.1.2 Low Voltage Detect Status and Control 1 (LVDSC1)

23.4.1.2.1 Offset

Register	Offset
LVDSC1	0h

23.4.1.2.2 Function

Indicates status and configures control bits to support the low voltage detect function.

NOTE

When the internal voltage regulator is in lowe power mode, the LVD system is disabled, regardless of the PMC_LVDSC1 settings.

23.4.1.2.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	LVDF		LVDIE	LVDRE	0			
W		LVDACK						
Reset	0	0	0	u[1]	0	0	0	0

1. LVDRE = 0b after POR. Unaffected by reset.

23.4.1.2.4 Fields

Field	Function
7 LVDF	Low Voltage Detect Flag Indicates a low-voltage detect event. The threshold voltage is V_{LVD}. 0b - Low-voltage event not detected 1b - Low-voltage event detected
6 LVDACK	Low Voltage Detect Acknowledge Acknowledges low voltage detection errors. Write 1 to clear LVDF. Read always return 0.
5 LVDIE	Low Voltage Detect Interrupt Enable Enables hardware interrupt requests for LVDF. 0b - Hardware interrupt disabled (use polling) 1b - Request a hardware interrupt when LVDF = 1
4 LVDRE	Low Voltage Detect Reset Enable Enables the low voltage detect events to generate a system reset. 0b - No system resets on low voltage detect events. 1b - If the supply voltage falls below VLVD, a system reset will be generated.
3-0 —	Reserved

23.4.1.3 Low Voltage Detect Status and Control 2 (LVDSC2)

23.4.1.3.1 Offset

Register	Offset
LVDSC2	1h

23.4.1.3.2 Function

Indicates status and configures control bits to support the low voltage warning (LVW) function.

NOTE

When the internal voltage regulator is in low power mode, the LVD system is disabled regardless of the PMC_LVDSC2 settings.

23.4.1.3.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	LVWF		LVWIE	0				
W		LVWACK						
Reset	0	0	0	0	0	0	0	0

23.4.1.3.4 Fields

Field	Function
7 LVWF	Low-Voltage Warning Flag Indicates a low-voltage detect event. The threshold voltage is V_{LVW}. 0b - Low-voltage warning event not detected 1b - Low-voltage warning event detected
6 LVWACK	Low-Voltage Warning Acknowledge Acknowledges 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. 0b - Disables hardware interrupt (use polling).

Table continues on the next page...

Field	Function
	1b - Enables a hardware interrupt when LVWF = 1.
4-0 —	Reserved

23.4.1.4 Regulator Status and Control (REGSC)

23.4.1.4.1 Offset

Register	Offset
REGSC	2h

23.4.1.4.2 Function

Indicates status and configures control bits for the regulator and LPO.

23.4.1.4.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	LPODIS	LPOSTAT	0			REGFPM	CLKBIASDIS	BIASEN
W								
Reset	0[1]	u	0	0	0	1	0	0

[1] Cleared on power on reset, unaffected by other reset.

23.4.1.4.4 Fields

Field	Function
7 LPODIS	LPO Disable Disables the low power oscillator. **NOTE:** After disabling the LPO a time of 2 LPO clock cycles is required before it is allowed to enable it again. Violating this waiting time of 2 cycles can result in malfunction of the LPO. 0b - Enable 1b - Disable
6 LPOSTAT	LPO Status

Table continues on the next page...

Field	Function
	Indicates the status of the LPO clock to be either in high phase (logic 1) or low phase (logic 0) of the clock period. Software can poll this status bit to measure actual LPO clock frequency and eventually use the LPOTRIM[4:0] register to change the LPO frequency. 0b - Low phase 1b - High phase
5-3 —	Reserved
2 REGFPM	Regulator in Full Performance Mode Status Indicates the current status of the internal voltage regulator. 0b - Regulator is in low power mode. 1b - Regulator is in full performance mode.
1 CLKBIASDIS	Clock Bias Disable Disables the bias currents and reference voltages for some clock modules in order to further reduce power consumption in STOP or VLPS mode. **NOTE:** While using this bit, it must be ensured that respective clock modules are disabled in STOP or VLPS mode. Otherwise, severe malfunction of clock modules will happen. 0b - No effect 1b - Disables the bias currents and reference voltages for SIRC, FIRC, and PLL clock modules (if available on device) in STOP or VLPS mode.
0 BIASEN	Bias Enable Enables source and well biasing for the core logic in low power mode. In full performance mode this bit has no effect. This is useful to further reduce MCU power consumption in low power mode. **NOTE:** Enabling this bit at high temperatures can significantly reduce MCU power consumption. 0b - Disables biasing, core logic can run in full performance. 1b - Enables biasing. Core logic is slower and there are restrictions in allowed system clock speed (see Data Sheet for details).

23.4.1.5 Low Power Oscillator Trim (LPOTRIM)

23.4.1.5.1 Offset

Register	Offset
LPOTRIM	4h

23.4.1.5.2 Function

Configures the period trimming bits for the low power oscillator.

Table 23-1. Trimming effect of LPOTRIM[4:0]

LPOTRIM[4:0]	Decimal	Period of LPO clock
10000	–16	lowest
10001	–15	increasing
...	...	
11110	–2	
11111	–1	
00000	0	typical 128 kHz
00001	+1	increasing
...	...	
01110	+14	
01111	+15	highest

NOTE

The LPO trimming bits represent signed values. Starting from -16 the period of the LPO clock will increase monotonically (for example, frequency decreases monotonically).

23.4.1.5.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0			LPOTRIM				
W								
Reset	0	0	0	u[1]	u	u	u	u

[1] Automatically loaded from flash memory IFR after any reset.

23.4.1.5.4 Fields

Field	Function
7-5 —	Reserved
4-0 LPOTRIM	LPO Trimming Trims the frequency of the low power oscillator.

Chapter 24
Integrity Functions Overview

24.1 Introduction

This chapter summarizes all security related features of this device.

24.2 Flash security feature summary

The flash security features supported by this MCU are summarized here.

24.2.1 Flash security byte

Security state can be enabled via programming Flash security byte (FSEC at 0x0000 040E) in the flash configuration field (a 16 Byte region start from 0x0000 0400). User can program the FSEC byte using the flash program phrase commands in Flash Memory Module. The FSEC byte will be loaded into the FSEC register during boot sequence after chip reset. This FSEC register is read-only.

The SEC bit of FSEC byte controls the chip security status. After enabling device security, the debug port (SWD) cannot access the memory resources of the MCU.

The flash security byte (FSEC) also allow user to enable the flash backdoor key access feature by configuring the KEYEN bits. When backdoor Key is enabled, the software can unsecure the MCU after presenting the correct backdoor key with Verify Backdoor Access Key command.

The MEEN bit of FSEC byte can be used to disable the mass erase capability from debug port .

The FSLACC bit of FSEC byte can be used to disable the NXP failure analysis. The FSLACC bit permits the user to disable all special or test mode which is only accessible by NXP. This feature help user to achieve a highest level to control the access of MCU on chip data.

Please refer to FSEC sections of the Flash Memory Module chapter for more details.

From debug port point of view, user can only disable the secure mode by the external mass erase bit from SWD. But if Mass Erase is disabled, the debug port can no longer unsecure the MCU. Please refer to the "Debug and security" section in the Debug chapter for more details.

24.3 Security hardware accelerators

24.3.1 CRC

This device contain one cyclic redundancy check (CRC) module which can generates 16/32-bit CRC code for error detection.

24.4 General security features

24.4.1 Unique ID

This device features 128-bit unique identification number, which programmed in factory and load to SIM register after power on reset. This unique ID permits the software to build a trusted device. This Unique ID generated based on the wafer lot and die series number of factory. The ID is unique for each device and it is accessible from SIM_UIDH, SIM_UIDMH, SIM_UIDML and SIM_UIDL registers. Please refer to the SIM chapter for more details.

24.4.2 Program Once Field

This device also contains 96 bytes Program Once Field in the program flash 0 IFR. User can program specific data into this field by Program Once command (in Flash Memory Module) with index 0x00 ~ 0x07. The data can no longer be erased nor modified after programming. The Program Once Field can be read through Read Once commands. Please refer to the "Program Once field" section in Flash Memory Module chapter for more details.

Chapter 25
External Watchdog Monitor (EWM)

25.1 Overview

For safety purposes, a redundant watchdog system, EWM, is designed to monitor external circuits and the MCU software flow. This provides a backup mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The internal watchdog is used to monitor the flow and execution of the embedded software within the MCU. It consists of a counter that, if allowed to overflow, forces an internal, asynchronous reset to all on-chip peripherals. The counter also optionally asserts the RESET_B pin to reset external devices and circuits. The watchdog counter must not overflow if the software code works well and services the watchdog to restart the actual counter.

The EWM does not reset the MCU's CPU and peripherals, making it different from internal watchdog. The EWM module provides an independent ewm_out_b signal that, when asserted, resets or places an external circuit into a safe mode. The ewm_out_b signal asserts upon EWM counter timeout. An optional external input, ewm_in, allows additional control when asserting the ewm_out_b signal.

25.1.1 Block diagram

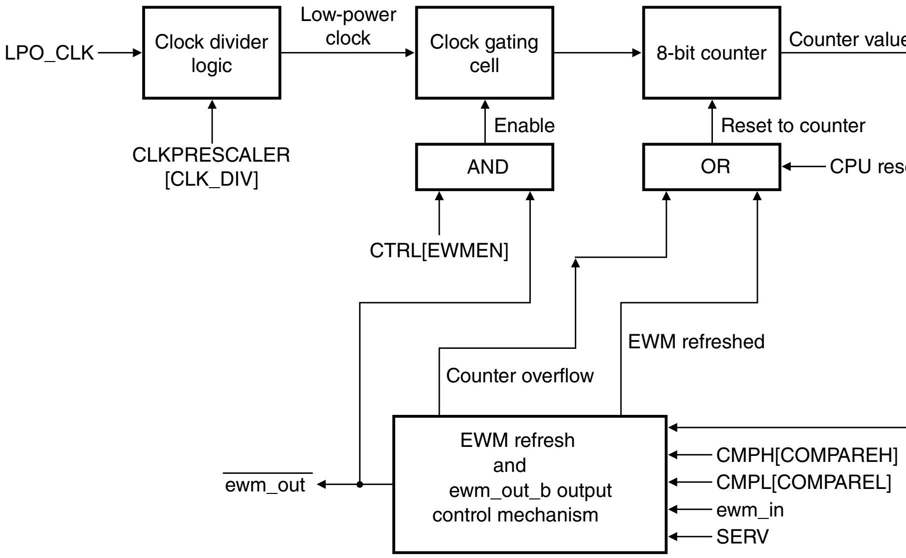


Figure 25-1. Block diagram

25.1.2 Features

- Independent LPO_CLK source
- Programmable timeout period, specified in terms of the number of EWM LPO_CLK cycles
- Windowed refresh option that provides:
  - A robust check to confirm that the program flow is faster than expected.
  - A programmable window.
  - Refresh outside the window, leading to assertion of the ewm_out_b signal.
- Robust refresh mechanism:
  - Write values of B4h and 2Ch to Service (SERV) within 8'd63 peripheral bus clock cycles.
- One output port, ewm_out_b, which when asserted is used to reset or place the external circuit into Safe mode
- One input port, ewm_in, which allows an external circuit to control the assertion of the ewm_out_b signal

25.2 Functional description

The following sections discuss these aspects of EWM:

- Functional details
- Operating modes

NOTE

If the BUS_CLK is lost, EWM does not generate the ewm_out_b signal and no refresh operation is possible.

25.2.1 Modes of operation

25.2.1.1 Stop mode

When EWM is in Stop mode, the CPU cannot refresh EWM. After entering Stop mode, the EWM counter freezes.

Following are the possible ways to exit Stop mode:

- Through a reset: EWM remains disabled in this case.
- Through an interrupt: EWM is re-enabled and the counter continues to be clocked from the same value as prior to Stop mode entry.

NOTE

Consider the following if EWM enters Stop mode during the CPU refresh mechanism:

- While exiting Stop mode through an interrupt, the refresh mechanism starts from the previous state. That is, if you write the refresh command correctly and EWM enters Stop mode immediately, you must write the next command in *8'd63* peripheral bus clocks after exiting Stop mode.
- You must mask all interrupts before executing the EWM refresh instructions.

25.2.1.2 Wait mode

The EWM module treats the Stop and Wait modes as the same. EWM functionality remains the same in both modes.

25.2.1.3 Debug mode

EWM remains unimpacted when entering Debug mode:

- If EWM is enabled before entering Debug mode, it remains enabled.
- If EWM is disabled before entering Debug mode, it remains disabled.

25.2.2 Using the EWM counter

EWM uses an 8-bit ripple counter that is fed by a clock source independent of the peripheral bus clock source. As the preferred timeout is between 1 ms and 100 ms, the actual clock source must be in the kHz range.

The counter is reset to 0 in these conditions:

- After CPU reset
- After the EWM refresh action completes
- At counter overflow

The CPU cannot access the counter value.

25.2.3 Using compare registers

You can write to Compare Low (CMPL) and Compare High (CMPH) only once after a CPU reset and you cannot modify them until another CPU reset occurs. These registers are used to create a refresh window for the EWM module.

You cannot program Compare Low (CMPL) and Compare High (CMPH) with the same value. In case of any attempt, the ewm_out_b signal asserts as soon as the counter reaches the value of Compare Low (CMPL) + 1.

Note

- You must update Compare Low (CMPL) and Compare High (CMPH) before enabling EWM. Therefore, you must provide a reasonable time after POR for the external monitoring circuit to stabilize. You must also ensure that the ewm_in pin is deasserted.
- Service should be requested after 1 clock period of slowest clock frequency while updating Compare Low (CMPL) register.

25.2.4 Using the refresh mechanism

Other than the initial configuration of EWM, the CPU can access EWM only through Service (SERV). The CPU must access this register by correctly writing unique data within the windowed time frame, as determined by Compare Low (CMPL) and Compare High (CMPH) for the correct EWM refresh operation. The following table describes conditions that exist and the refresh mechanisms that apply to those conditions.

Table 25-1. Refresh mechanisms

Condition	Mechanism
The EWM refresh action completes when the value of Compare Low (CMPL) ≤ the counter value ≤ the value of Compare High (CMPH).	The software behaves as expected and the EWM counter resets to 0. The ewm_out_b output signal remains in Deasserted state if, during the EWM refresh action, the ewm_in input is in Deasserted state.
The EWM refresh action completes when the counter value < the value of Compare Low (CMPL).	The software refreshes EWM before the windowed time frame, the counter resets to 0, and the ewm_out_b output signal asserts no matter what the value of the ewm_in input signal is.
The counter value becomes greater than the value of Compare High (CMPH) prior to completion of the EWM refresh action.	The software does not refresh EWM. The EWM counter resets to 0 and the ewm_out_b output signal asserts no matter what the value of the ewm_in input signal is.

See Service (SERV) for more on the refresh mechanism.

25.2.5 Interrupt

When the ewm_out_b signal asserts, an interrupt request can be generated to indicate the assertion of the EWM reset out signal. The interrupt is enabled when CTRL[INTEN] = 1. Writing 0 to this field clears the interrupt request but does not affect the ewm_out_b signal, which can be deasserted only by forcing a system reset.

25.2.6 Clocking

The following table shows EWM clocks.

Table 25-2. EWM Clocks

Clock	Description
IPG_CLK	This is the system clock and should be turned on for EWM to be able to work properly. During low power modes in which the core is powered down, this clock is disabled,
IPG_CLK_S	This is the IPS clock and is synchronous with IPG_CLK. It is disabled except during IPS write accesses. it is enabled with EWM's IPS_MODULE_EN

Table continues on the next page...

Table 25-2. EWM Clocks (continued)

Clock	Description
LPO_CLOCK	This is a low power clock used for running EWM counter. This clock is gated when EWM is disabled or when ewm_out_b is asserted.

25.2.7 Using the counter clock prescaler

You can program CLKPRESCALER[CLK_DIV] to divide the EWM counter clock source. This divided clock is used to run the EWM counter.

NOTE

The divided clock used to run the EWM counter must not exceed half the frequency of the bus clock.

25.3 External signals

EWM includes external signals, as shown in the following table.

NOTE

All active-low signals are represented with the suffix "_b" throughout the chapter.

Table 25-3. Signal descriptions

Signal	Description	I/O
ewm_in	EWM's input for the safety status of external safety circuits. You can program the polarity of ewm_in by using CTRL[ASSIN]. The default polarity is active-low.	I
ewm_out_b	EWM's reset out signal	O

25.3.1 Using the ewm_out_b signal

The ewm_out_b signal is a digital output signal used to gate an external circuit (application-specific) that controls critical safety functions. For example, EWM_out must be connected to the high-voltage transistor circuits that control an AC motor in a large appliance.

The ewm_out_b signal remains deasserted when the CPU regularly refreshes EWM within the programmable refresh window, indicating that the application code is executing as expected.

The ewm_out_b signal asserts in any of the following conditions:

- An EWM refresh action occurs when the counter value is less than the value of Compare Low (CMPL).
- The EWM counter value becomes greater than the value of Compare High (CMPH) and no EWM refresh occurs.
- The functionality of the ewm_in pin is enabled and the ewm_in pin asserts when refreshing EWM.
- After any reset.

The ewm_out_b signal asserts after any reset by the virtue of the external pulldown mechanism on the ewm_out_b pin. To deassert the ewm_out_b signal, write 1 to CTRL[EWMEN] to enable EWM.

If the ewm_out_b signal shares its pad with a digital I/O pin, this actual pad defers to being an input signal on reset. The ewm_out_b signal controls the pad state only after CTRL[EWMEN] enables EWM.

Note

> The ewm_out_b pad must be in Pulldown state when the EWM functionality is being used and EWM is under reset.

25.3.2 Using the ewm_out_b pin state in low-power modes

During Wait, Stop and Power Down modes, ewm_out_b pin preserves its state before entering Wait or Stop mode. When the CPU enters Run mode from Wait or Stop mode recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode after exiting Power-Down mode, the pin returns to its reset state.

25.3.3 Using the ewm_in signal

The ewm_in signal is a digital input signal for the safety status of external safety circuits. This signal allows an external circuit to control the assertion of the ewm_out_b signal. For example, in the application, an external circuit monitors a critical safety function, and if there is a fault with the safety function, the external circuit can actively initiate the ewm_out_b signal, which controls the gating circuit.

The ewm_in signal is ignored if EWM is disabled, or if CTRL[INEN] = 0 after any reset.

After you enable EWM (by writing 1 to CTRL[EWMEN]) and the ewm_in functionality (by writing 1 to CTRL[INEN]), the ewm_in signal must be in Deasserted state before the CPU starts refreshing EWM. This ensures that the ewm_out_b signal stays in Deasserted state; otherwise, the ewm_out_b output signal asserts.

25.4 Memory map and register definitions

This section contains the module memory map and registers.

NOTE

EWM supports only 8-bit register accesses; 16-bit and 32-bit accesses are not supported.

25.4.1 EWM register descriptions

25.4.1.1 EWM memory map

EWM base address: 4006_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	8	RW	00h
1h	Service (SERV)	8	W	00h
2h	Compare Low (CMPL)	8	RW	00h
3h	Compare High (CMPH)	8	RW	FFh
5h	Clock Prescaler (CLKPRESCALER)	8	RW	00h

25.4.1.2 Control (CTRL)

25.4.1.2.1 Offset

Register	Offset
CTRL	0h

25.4.1.2.2 Function

Controls the functionality of EWM.

NOTE

You can write to CTRL[INEN], CTRL[ASSIN], and CTRL[EWMEN] only once after a CPU reset. Modifying these fields more than once generates a bus transfer error.

25.4.1.2.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0				INTEN	INEN	ASSIN	EWMEN
W								
Reset	0	0	0	0	0	0	0	0

25.4.1.2.4 Fields

Field	Function
7-4 —	Reserved
3 INTEN	Interrupt Enable Enables interrupt request generation. If this field = 1 and the ewm_out_b signal is asserted, an interrupt request is generated. To deassert interrupt requests, write 0 to this field. 0b - Deasserts interrupt requests 1b - Generates interrupt requests
2 INEN	Input Enable Enables the ewm_in port. When this field = 1, it enables the ewm_in port. 0b - Disables

Table continues on the next page...

Field	Function
	1b - Enables
1 ASSIN	Assertion State Select Specifies the asserted state of the ewm_in signal. By default, the asserted state of the ewm_in signal is logic 0 (active-low), which is when this field = 0. When this field = 1, the ewm_in asserted state is logic 1 (active-high). You can use this field to change the expected polarity of the ewm_in signal. 0b - Logic 0 1b - Logic 1
0 EWMEN	EWM Enable Enables the EWM module. If this field = 1, it enables the EWM module, and if the field = 0, it disables the EWM module. You cannot re-enable this field until the next reset because of its write-once nature. 0b - Disables 1b - Enables

25.4.1.3 Service (SERV)

25.4.1.3.1 Offset

Register	Offset
SERV	1h

25.4.1.3.2 Function

Provides an interface from the CPU to the EWM module.

Attempted reads of this register return 0.

25.4.1.3.3 Diagram

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

25.4.1.3.4 Fields

Field	Function
7-0 SERVICE	Service Provides an interface from the CPU to the EWM module. The EWM refresh mechanism requires the CPU to write these values to this field: a first data byte of B4h, followed by a second data byte of 2Ch. The EWM refresh action is invalid if either of the following conditions is true: • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte, known as EWM_refresh_time. The number of peripheral bus clock cycles required for EWM_refresh_time is 8'd63.

25.4.1.4 Compare Low (CMPL)

25.4.1.4.1 Offset

Register	Offset
CMPL	2h

25.4.1.4.2 Function

Determines the lower value of the windowed time frame for the correct EWM refresh operation.

This register is reset to 0 after a CPU reset. This provides no minimum time for the CPU to refresh the EWM counter.

NOTE

You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error.

25.4.1.4.3 Diagram

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

25.4.1.4.4 Fields

Field	Function
7-0 COMPAREL	Compare Low Configures the minimum counter value when refreshes are allowed. If a refresh is attempted while the counter value is lower than COMPAREL, then the ewm_out_b signal is asserted. To prevent runaway code from changing the value of this field, you must write to this field after a CPU reset even if the (default) minimum refresh time is required.

25.4.1.5 Compare High (CMPH)

25.4.1.5.1 Offset

Register	Offset
CMPH	3h

25.4.1.5.2 Function

Determines the higher value of the windowed time frame for the correct EWM refresh operation.

This register is reset to FFh after a CPU reset. This provides a maximum time of up to 256 clocks for the CPU to refresh the EWM counter.

NOTE

> You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error.
>
> The valid values for this register are up to FEh because the EWM counter never expires when the value of COMPAREH = FFh. The expiration happens only if the EWM counter is greater than the value of COMPAREH.

25.4.1.5.3 Diagram

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

25.4.1.5.4 Fields

Field	Function
7-0 COMPAREH	Compare High Configures the maximum counter value till when refreshes are allowed. If a refresh is not attempted while the counter value is greater than COMPAREH, then the ewm_out_b signal is asserted. To prevent runaway code from changing the value of this field, you must write to this field after a CPU reset.

25.4.1.6 Clock Prescaler (CLKPRESCALER)

25.4.1.6.1 Offset

Register	Offset
CLKPRESCALER	5h

25.4.1.6.2 Function

Prescales the EWM counter clock source by a clock divider.

This register is reset to 00h after a CPU reset.

NOTE

> You can write to this register only once after a CPU reset. Writing to the register more than once generates a bus transfer error. You must write the required prescaler value before enabling EWM.

25.4.1.6.3 Diagram

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

25.4.1.6.4 Fields

Field	Function
7-0 CLK_DIV	Clock Divider Prescales the selected low-power clock source for running the EWM counter: Prescaled clock frequency = low-power clock source frequency ÷ (1 + the value of CLK_DIV) See chip-specific information for low-power clock source frequency used in your device.

25.5 Usage Guide

25.5.1 EWM low-power modes

This table shows the EWM low-power modes and the corresponding chip low-power modes.

Table 25-4. EWM low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS

25.5.2 $\overline{\text{EWM\_out}}$ pin state in low power modes

During Wait, Stop, and Power Down modes the $\overline{\text{EWM\_out}}$ pin preserve its state before entering Wait or Stop mode. When the CPU enters a Run mode from Wait or Stop recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode from Power Down, the pin returns to its reset state.

25.5.3 Example code

25.5.3.1 Initializing the EWM

The following code segment shows the initialize sequence of the EWM module. It enables EWM_in pin input with assert state logic zero, enables interrupt when EWM_out is assert. The compare value is also set into CMPL/H register before enabling EWM.

```
// Initialize the EWM module
EWM_CMPL = compareValue & 0xFF;
EWM_CMPH = (compareValue >> 8) 0xFF;
EWM_CTRL = EWM_CTRL_INEN(1) | EWM_CTRL_ASSIN(0) |
           EWM_CTRL_INTEN(1) | EWM_CTRL_EWMEN(1);
```

25.5.3.2 Refreshing the EWM

The following code segment shows the refresh write sequence of the EWM module.

```
// Refresh EWM
DisableInterrupts; // disable global interrupt
EWM_SERV= 0xB4; // write the 1st refresh words
EWM_SERV= 0x2C; // write the 2nd refresh words
EnableInterrupts; // enable global interrupt
```

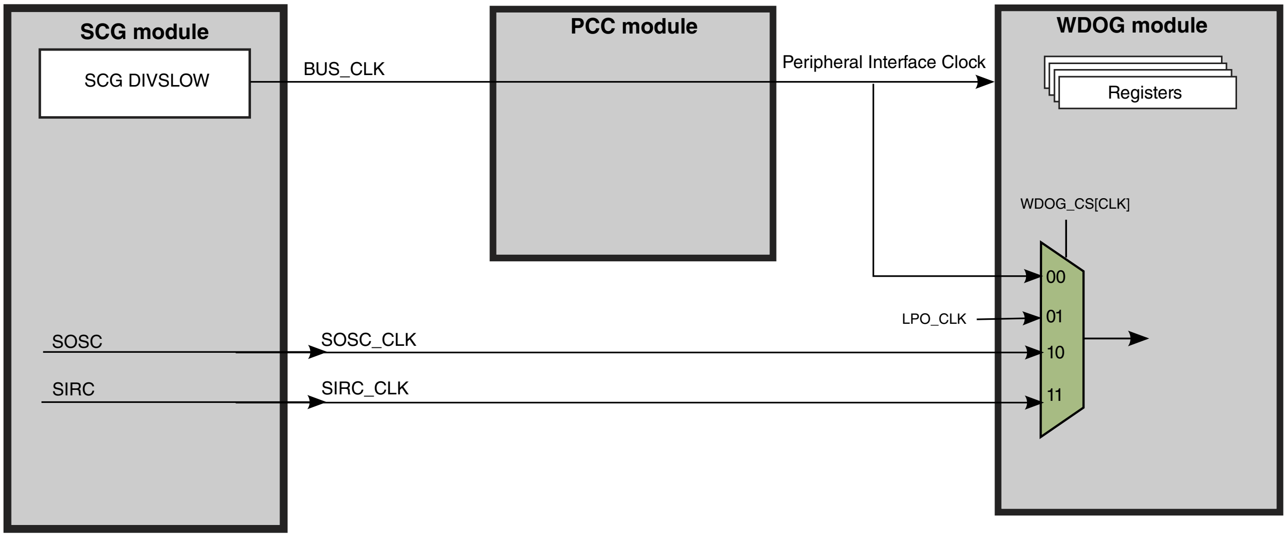
Chapter 26
Watchdog timer (WDOG)

26.1 Chip-specific information for this module

26.1.1 WDOG Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - WDOG



26.1.2 WDOG low-power modes

This table shows the WDOG low-power modes and the corresponding chip low-power modes.

Table 26-1. WDOG low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS

26.2 Overview

WDOG is an independent timer that is available for system use. It provides a safety feature to ensure that the software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If WDOG is not serviced (refreshed) within a certain period, it resets the MCU.

26.2.1 Block diagram

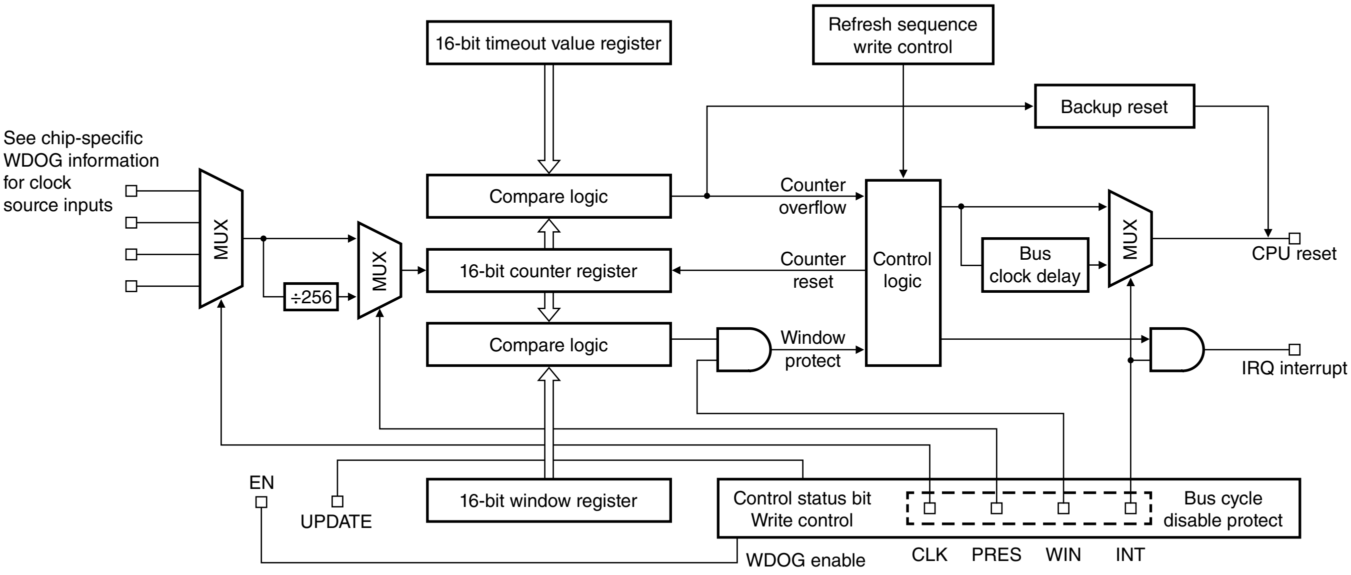


Figure 26-1. Block diagram

26.2.2 Features

- Configurable clock source inputs independent of the bus clock
- Programmable timeout period

  - Programmable 16-bit timeout value
  - Optional, fixed 256 clock prescaler when longer timeout periods are needed
- Robust write sequences for counter refresh: provision to refresh the sequence of writing to WDOG Counter (CNT)
- Window mode option for the refresh mechanism
  - Programmable 16-bit window value
  - Robust check to ensure that program flow is faster than expected
  - Early refresh attempts that trigger a reset
- Optional timeout interrupt to allow post-processing diagnostics
  - Interrupt request to CPU with an interrupt vector for an interrupt service routine (ISR)
  - Forced reset that occurs 128 bus clocks after the interrupt vector fetch
- Write-once-after-reset configuration fields to ensure that WDOG configuration is not altered mistakenly
- Robust write sequence for unlocking write-once configuration fields
  - Unlock sequence of writing to WDOG Counter (CNT), for allowing updates to write-once configuration fields
  - You must make updates within 8'd128 bus clocks after unlocking and before WDOG closing unlock window

26.3 Functional description

WDOG provides a fail-safe mechanism to ensure that you can reset the system to a known state of operation in case of system failure, such as the CPU clock stopping or there being a runaway condition in the software code. The WDOG counter runs continuously off a selectable clock source and expects to be serviced (refreshed) periodically. If it is not refreshed, it generates a reset triggering event.

26.3.1 Refresh mechanism

WDOG resets the MCU if the WDOG counter is not refreshed. A robust refresh mechanism makes it very unlikely for a runaway code to refresh WDOG.

To refresh the WDOG counter, you must execute a refresh write sequence before the timeout period expires. In addition, in case of Window mode, you must not start the refresh sequence until you set the time value in Watchdog Window (WIN). See the following figure for more.

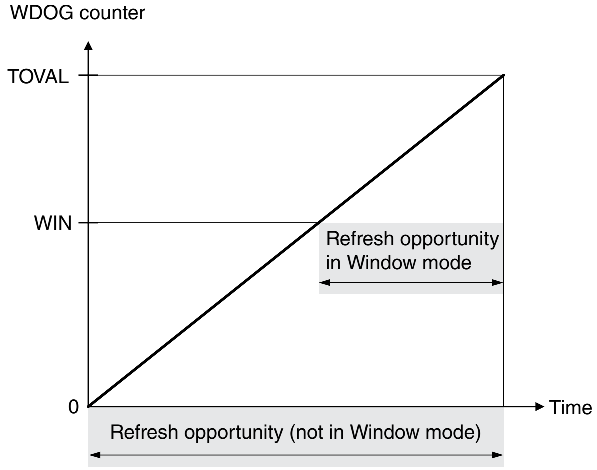


Figure 26-2. Refresh opportunity for the WDOG counter

26.3.1.1 Using 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, you can program WDOG to force a reset when refresh attempts are early.

When Window mode is enabled, you must refresh WDOG after the counter has reached a minimum expected time value; otherwise, WDOG resets the MCU. The minimum expected time value is specified in Watchdog Window (WIN). Writing 1 to WIN enables Window mode.

26.3.1.2 Refreshing WDOG

The refresh write sequence is based on the following methods:

- Either two 16-bit writes (A602h, B480h) or four 8-bit writes (A6h, 02h, B4h, 80h) to WDOG Counter (CNT) if CMD32EN = 0
- One 32-bit write (B480_A602h) to WDOG Counter (CNT) if CMD32EN = 1

You must apply these methods before WDOG times out; otherwise, it resets the MCU.

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, restore the global interrupt control state.

See Application information for example code.

26.3.2 Configuring WDOG

26.3.2.1 Configuring WDOG once

All WDOG control fields, timeout value, and window value are write-once after reset. This means that after a write has occurred, they cannot be changed unless a reset occurs. You can ensure this by configuring the window and timeout values first, followed by the other control fields, when UPDATE = 0.

This provides a robust mechanism to configure WDOG and ensure that a runaway condition cannot mistakenly disable or modify the WDOG configuration, after the module is configured.

The new configuration takes effect only after you write to all registers except WDOG Counter (CNT) after reset. Otherwise, WDOG uses the reset values by default. If you do not use Window mode (WIN), you do not need to write to Watchdog Window (WIN) to bring the new configuration into effect.

26.3.2.2 Reconfiguring WDOG

In some cases (for example, when supporting a bootloader function), you may want to reconfigure or disable WDOG, without forcing a reset first:

- By writing 1 to UPDATE on the initial configuration of WDOG after a reset, you can reconfigure WDOG at any time by executing an unlock sequence.
- Conversely, if UPDATE remains 0, the only way to reconfigure WDOG is by initiating a reset.

The unlock sequence is similar to the refresh sequence but uses different values.

26.3.2.3 Unlocking WDOG

The unlock sequence is based on the following two methods:

- Either two 16-bit writes (C520h, D928h) or four 8-bit writes (C5h, 20h, D9h, 28h) to WDOG Counter (CNT), after WDOG is configured, if CMD32EN = 0
- One 32-bit write (D928_C520h) to WDOG Counter (CNT), after WDOG is configured, if CMD32EN = 1

An improper unlock sequence causes WDOG to reset. On completing the unlock sequence, you must reconfigure WDOG within 8'd128 bus clocks; otherwise, WDOG closes the unlock window.

NOTE

Because it requires 8'd128 bus clocks to reconfigure WDOG, you must insert some delays before executing stop or wait instructions after reconfiguring WDOG. This ensures that WDOG's new configuration takes effect before the MCU enters Low-Power mode. Otherwise, the MCU may not wake up from Low-Power mode.

26.3.3 Functionality in Debug and Low-Power modes

By default, WDOG is not functional in Debug, Wait, or Stop modes. However, it can remain functional in these modes as follows:

- For Debug mode**,** write 1 to DBG (this way WDOG is functional in Debug mode even when Debug mode holds the CPU).
- For Wait mode, write 1 to WAIT.
- For Stop mode, write 1 to STOP and WAIT, and ensure that the clock source is active in Stop mode.

NOTE

WDOG can generate an interrupt in Stop mode.

Don't write 1 to INT in Stop mode. Otherwise, WDOG reset after a delay of 128 bus clocks (lose bus clock) will not occur, but backup reset will take effect.

For Debug and Stop modes, in addition to the aforementioned configuration, you must use a clock source other than the bus clock as the reference clock for the counter; otherwise, WDOG cannot function.

26.3.4 Fast testing of WDOG

Before executing the application code in safety-critical applications, you must test that WDOG 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 k clocks).

To help minimize the startup delay for application code after reset, WDOG implements a feature that tests its functioning more quickly by splitting the counter into its constituent byte-wide stages. The low and high bytes are run independently and tested for a timeout against the corresponding byte of WDOG Timeout Value (TOVAL). For a 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 POR, the POR field in the system reset register becomes 1, indicating that you must perform the WDOG fast test.

26.3.4.1 Testing each byte of the counter

Perform this procedure to test each byte of the counter:

1. Program the preferred WDOG timeout value in WDOG Timeout Value (TOVAL) during the WDOG configuration period.
2. Select a byte of the counter to test by using the configuration TST = 10b for the low byte, and TST = 11b for the high byte.
3. Wait for WDOG to timeout. Optionally, in the idle loop, increment RAM locations as a parallel software counter for later comparison. Because RAM is not affected by a WDOG reset, the timeout period of the WDOG counter can be compared with the software counter to verify whether the timeout period occurred as expected.

   The WDOG counter times out and forces a reset.

4. Confirm that the WDOG flag in the system reset register is set, indicating that WDOG caused the reset (the POR flag remains clear).
5. Confirm that TST showing a test (10b or 11b) was performed.

If confirmed, the count and compare functions work for the selected byte. Repeat the procedure, and then select the next byte in step 2.

NOTE

Only a POR writes 0 to TST, which is not affected by other resets.

26.3.4.2 Entering User mode

After successfully testing the low and high bytes of the WDOG counter, you can configure TST to 01b to indicate that WDOG is ready for use in application User mode. Therefore, if a reset occurs again, you can recognize the reset trigger as a real WDOG reset caused by runaway or faulty application code.

As an ongoing test when using the default clock source, you can periodically read WDOG Counter (CNT) to ensure that the counter is being incremented.

26.3.5 Clocking

You can program CLK to select clock source options in the WDOG counter. See the chip-specific WDOG information for available clock inputs and the default option for this chip.

The option allows you to select a clock source that is 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 WDOG counter continues to run if the bus clock is somehow halted (see Backup reset).

For WDOG to function properly, you must enable the default WDOG clock source after its functional reset is deasserted.

An optional fixed prescaler for all clock sources allows longer timeout periods. When PRES = 1, the clock source is prescaled by 256 before clocking the WDOG counter.

The following table summarizes examples of the different available WDOG timeout periods. In the table, RCP means "reference clock period".

Table 26-2. WDOG timeout availability

Reference clock	Prescaler	WDOG timeout availability
REF_CLK	Pass through	1 × RCP to 65535 × RCP
	Enable	256 × RCP to 16776960 × RCP

NOTE

When you switch clock sources during reconfiguration, WDOG 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 period (128 bus clocks) ends. This delay ensures a smooth transition before restarting the counter with the new configuration.

26.3.6 Backup reset

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 WDOG counter overflows twice in succession (without an intervening reset), the backup reset function takes effect and generates a reset.

The backup reset becomes valid when an interrupt is enabled and the WDOG clock is not from a bus clock. If an interrupt is enabled after the bus clock is cut off before exiting an interrupt routine, the normal WDOG reset is blocked. In this case, the second overflow causes a backup reset directly.

NOTE

You must use a clock source other than the bus clock as a reference clock for the counter; otherwise, the backup reset function becomes unavailable.

26.3.7 Interrupts

WDOG can generate an interrupt request to delay resets.

When interrupts are enabled (INT = 1), and after a reset-triggering event (such as a counter timeout or invalid refresh attempt), WDOG:

1. Generates an interrupt request.
2. Waits 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event).
3. Forces a reset.

This process allows the ISR to perform tasks such as analyzing the stack to debug code.

When interrupts are disabled (INT = 0), WDOG does not wait before forcing a reset.

26.4 External signals

This module has no external signals.

26.5 Initialization

See Configuring WDOG once.

26.6 Application information

You must disable WDOG or reconfigure it before the first WDOG timeout. Disabling or reconfiguring WDOG must occur at the very beginning of the software code, for example, at the beginning of the startup or main function.

NOTE

- After you configure WDOG, it needs at least 2.5 periods of WDOG clock to take effect. This means you must have a gap of at least 2.5 clocks between two configurations.
- When the chip starts from boot ROM and then jumps to flash memory, you must disable WDOG at the beginning of the bootloader and enable it after the bootloader exits. To reconfigure WDOG using the flash memory program, it also needs the interval of at least 2.5 WDOG clocks after the bootloader exits.

To disable or reconfigure WDOG without forcing a reset, you must write 1 to UPDATE during the initial WDOG configuration. You can use the unlock sequence at any time, within the timeout limit, to reconfigure WDOG.

26.6.1 Disabling WDOG

To disable WDOG, you must first perform the unlock sequence. Then, write 0 to EN. The following code snippet shows an example of a 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
WDOG_CS &= ~WDOG_CS_EN_MASK; //disable watchdog
EnableInterrupts; //enable global interrupt
```

26.6.2 Disabling WDOG after reset

All WDOG registers are unlocked by reset. Therefore, an unlock sequence is unnecessary but it needs to be written to all WDOG registers to make the new configuration take effect. The following code snippet shows an example of disabling WDOG after reset.

```
DisableInterrupts; // disable global interrupt
WDOG_CS &= ~WDOG_CS_EN_MASK; // disable watchdog
WDOG_TOVAL= 0xFFFF;
while(WDOG_CS[ULK]);  // waiting for lock
while(~WDOG_CS[RCS]); // waiting for new configuration to take effect
EnableInterrupts; // enable global interrupt
```

26.6.3 Configuring WDOG

You can write 0 to UPDATE to configure WDOG. After that, you cannot reconfigure WDOG until a reset. To reconfigure without forcing a reset, write 1 to UPDATE when configuring WDOG. The following example code shows how to configure WDOG without Window mode, clock source as LPO, interrupt enabled, and timeout value to 256 clocks. The following code snippet shows an example of a 32-bit write.

26.6.3.1 Configuring once

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(0);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

26.6.3.2 Configuring for Reconfigurable mode

```
DisableInterrupts; //disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(1);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

26.6.4 Refreshing WDOG

To refresh WDOG and reset the WDOG counter to zero, you require a refresh sequence. The following code snippet shows an example of a 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xB480A602; // refresh watchdog
EnableInterrupts; // enable global interrupt
```

26.7 Memory map and register definition

26.7.1 WDOG register descriptions

26.7.1.1 WDOG memory map

WDOG base address: 4005_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	WDOG Control and Status (CS)	32	RW	0000_2980h
4h	WDOG Counter (CNT)	32	RW	0000_0000h
8h	WDOG Timeout Value (TOVAL)	32	RW	0000_0400h
Ch	Watchdog Window (WIN)	32	RW	0000_0000h

26.7.1.2 WDOG Control and Status (CS)

26.7.1.2.1 Offset

Register	Offset
CS	0h

26.7.1.2.2 Function

Describes watchdog control and status.

NOTE

TST is cleared (0:0) on POR only. Any other reset does not affect the value of this field.

26.7.1.2.3 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	WIN	FLG	CMD32EN	PRES	ULK	RCS	CLK		EN	INT	UPDATE	TST		DBG	WAIT	STOP
W		W1C														
Reset	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0

26.7.1.2.4 Fields

Field	Function
31-16 —	Reserved
15 WIN	WDOG Window Enables Window mode. You can write to this field only once. 0b - Disable 1b - Enable
14 FLG	WDOG Interrupt Flag Acts as an interrupt indicator when INT = 1. 0b - No interrupt occurred 1b - An interrupt occurred
13 CMD32EN	Command 32 Enable Enables or disables WDOG support for 32-bit (otherwise 16-bit or 8-bit) refresh or unlock command write words. If this field = 0, it disables support for 32-bit refresh or unlock command write words. Only a 16-bit or 8-bit write is supported. If this field = 1, it enables support for 32-bit refresh or unlock command write words. A 16-bit or 8-bit write is not supported. This is a write-once field, and you must unlock WDOG after writing to this field for reconfiguration. 0b - Disable 1b - Enable
12 PRES	WDOG Prescaler Enables a fixed 256 prescaling of the WDOG counter reference clock. See Block diagram that shows the clock divider option. This is a write-once field. 0b - Disable 1b - Enable

Table continues on the next page...

Field	Function
11 ULK	Unlock Status Indicates whether WDOG is unlocked. 0b - Locked 1b - Unlocked
10 RCS	Reconfiguration Success Indicates whether the reconfiguration is successful. This field becomes 1 when new configuration takes effect, and becomes 0 with a successful unlock command. 0b - Unsuccessful 1b - Successful
9-8 CLK	WDOG Clock Selects the clock source that feeds the WDOG counter. You can write to this field only once. **NOTE:** See chip-specific WDOG information for details of bit field settings. 00b - IPG 01b - LPO 10b - INT 11b - EXT
7 EN	WDOG Enable Enables the WDOG counter to start counting. You can write to this field only once. 0b - Disable 1b - Enable
6 INT	WDOG Interrupt Configures WDOG to immediately generate an interrupt request upon a reset-triggering event (timeout or illegal write to WDOG), before forcing a reset. After the interrupt vector fetch (that takes place after the reset-triggering event), the reset occurs after a delay of 128 bus clocks. If this field = 0, it disables WDOG interrupts (WDOG resets are not delayed). If this field = 1, it enables WDOG interrupts (WDOG interrupts are delayed by 128 bus clocks from the interrupt vector fetch). You can write to this field only once. 0b - Disable 1b - Enable
5 UPDATE	Updates Allowed Allows you to reconfigure WDOG without a reset. If this field = 0, you cannot update WDOG after the initial configuration, without forcing a reset. If this field = 1, you can modify the WDOG configuration registers within 1024 bus clocks after performing the unlock write sequence. You can write to this field only once. 0b - Updates not allowed 1b - Updates allowed
4-3 TST	WDOG Test Enables Fast Test mode, which allows you to exercise all bits of the counter to demonstrate that WDOG is functioning properly. See Fast testing of WDOG for more information. If this field = 0, it disables WDOG Test mode. If this field = 1, it enables WDOG User mode and disables WDOG Test mode. After testing WDOG, you must use this setting to indicate that WDOG is functioning normally in User mode. If this field = 10, it enables WDOG Test mode; only the low byte is used. CNTLOW is compared with TOVALLOW. If this field = 11, it enables WDOG Test mode; only the high byte is used. CNTHIGH is compared with TOVALHIGH.

Table continues on the next page...

Field	Function
	This write-once field is cleared (0:0) on POR only. Any other reset does not affect the value of this field. 00b - Disable WDOG Test mode 01b - Enable WDOG User mode 10b-11b - Enable WDOG Test mode
2 DBG	Debug Enable Enables WDOG to operate when the chip is in Debug mode. You can write to this field only once. 0b - Disable 1b - Enable
1 WAIT	Wait Enable Enables WDOG to operate when the chip is in Wait mode. You can write to this field only once. 0b - Disable 1b - Enable
0 STOP	Stop Enable Enables WDOG to operate when the chip is in Stop mode. You can write to this field only once. 0b - Disable 1b - Enable

26.7.1.3 WDOG Counter (CNT)

26.7.1.3.1 Offset

Register	Offset
CNT	4h

26.7.1.3.2 Function

Provides access to the value of the free-running WDOG counter. You can read the counter register at any time but cannot write directly to it. However, the following write sequences to this register have special functions:

1. The refresh sequence resets WDOG counter to 0000h. See Refreshing WDOG for more information.
2. The unlock sequence allows WDOG to be reconfigured without forcing a reset (when UPDATE = 1). See Configuring for Reconfigurable mode for more information.

NOTE

All other writes to this register are illegal and force a reset.

26.7.1.3.3 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 W	CNTHIGH								CNTLOW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.7.1.3.4 Fields

Field	Function
31-16 —	Reserved
15-8 CNTHIGH	Counter Low Byte Contains the WDOG low byte.
7-0 CNTLOW	Counter High Byte Contains the WDOG high byte.

26.7.1.4 WDOG Timeout Value (TOVAL)

26.7.1.4.1 Offset

Register	Offset
TOVAL	8h

26.7.1.4.2 Function

Contains the 16-bit value used to set the timeout period of WDOG.

The WDOG counter (CNT) is continuously compared with the timeout value (TOVAL). If the counter reaches the timeout value, WDOG forces a reset triggering event.

NOTE

Do not write 0 to this register (if TST = 11b, then TOVALHIGH cannot be written as 0; if TST = 10b, then TOVALLOW cannot be 0); otherwise, WDOG always generates a reset.

26.7.1.4.3 Diagram

<table>
<tr><td>Bits</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td colspan="16">0</td></tr>
<tr><td>W</td><td colspan="16"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bits</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R
W</td><td colspan="8">TOVALHIGH</td><td colspan="8">TOVALLOW</td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

26.7.1.4.4 Fields

Field	Function
31-16 —	Reserved
15-8 TOVALHIGH	Timeout Value High Contains the high byte of the timeout value.
7-0 TOVALLOW	Timeout Value Low Contains the low byte of the timeout value.

26.7.1.5 Watchdog Window (WIN)

26.7.1.5.1 Offset

Register	Offset
WIN	Ch

26.7.1.5.2 Function

Determines the earliest time that a refresh sequence is considered valid, if WIN = 1. See Refresh mechanism for more information.

The WIN register value must be less than the TOVAL register value.

26.7.1.5.3 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 W	WINHIGH								WINLOW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.7.1.5.4 Fields

Field	Function
31-16 —	Reserved
15-8 WINHIGH	High Byte Contains the high byte of the WDOG window.
7-0 WINLOW	Low Byte Contains the low byte of the WDOG window.

Chapter 27
Cyclic Redundancy Check (CRC)

27.1 Overview

CRC generates 16-bit or 32-bit CRC codes output for error detection. You can calculate these codes up to 32 bits input data at a time.

CRC provides a programmable polynomial and other parameters that you require to meet the 16-bit or 32-bit CRC standards.

27.1.1 Block diagram

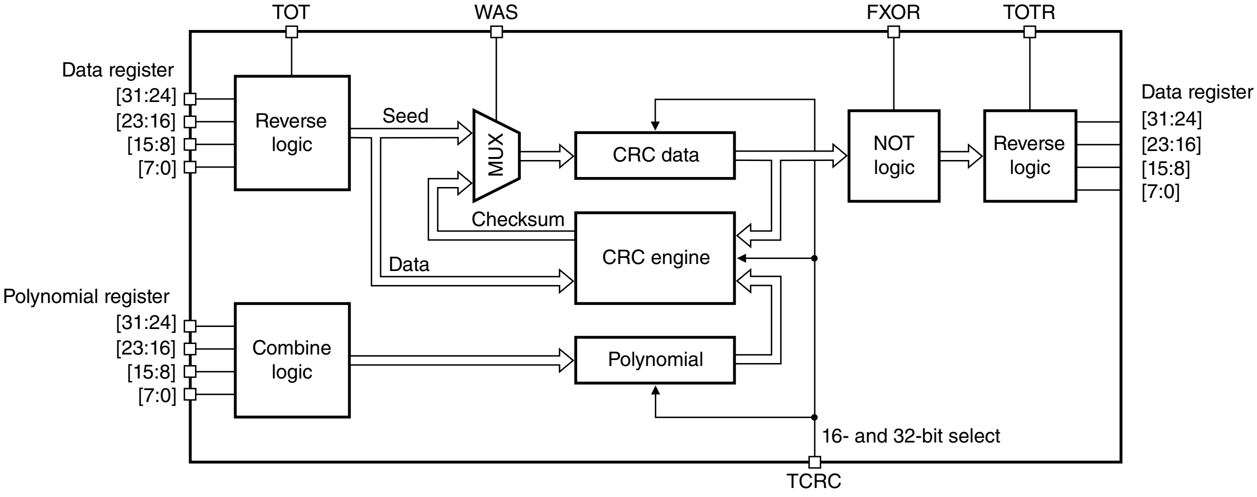


Figure 27-1. Block diagram

27.1.2 Features

CRC has the following features:

- Hardware CRC generator circuit using 16-bit or 32-bit programmable shift registers
- Programmable initial seed value and polynomial
- Transpose of input or output data (CRC result) in bitwise or bytewise (this option is required for certain CRC standards. You cannot perform a bytewise transpose operation when accessing DATA via 8-bit access.)
- Invert final CRC result
- 32-bit CPU register programming interface

27.2 Functional description

27.2.1 Modes of operation

The following sections describe various modes of operation that affect the functionality of CRC: Run mode and Low power mode.

27.2.1.1 Run mode

Run mode is the basic mode of operation.

27.2.1.2 Low power mode

When the chip enters the lower power mode, the CRC module clock (ipg_clk and ipg_clk_s) is disabled and the in-progress CRC calculation stops. The calculation resumes after the CRC module clock is enabled or the chip exits low power mode via system reset.

27.2.2 CRC calculations

In 16-bit and 32-bit CRC modes, you can program data values as 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes lead to an incorrect CRC calculation.

27.2.2.1 Calculating a 16-bit CRC

Perform these steps to calculate a 16-bit CRC:

1. Write 0 to CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in Data (DATA) as required for the CRC calculation.
3. Write a 16-bit polynomial to GPOLY[LOW].

   GPOLY[HIGH] is not usable in 16-bit CRC mode.

4. Write 1 to CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to DATA[LU] and DATA[LL].

   DATA[HU] and DATA[HL] are not used.

6. Write 0 to CTRL[WAS] to start writing data values.
7. Write data values into DATA[LU], and DATA[LL]

   CRC is calculated on every data value write and the intermediate CRC result is stored back into DATA[LU] and DATA[LL]

8. After writing all the data values, read the final CRC result from DATA[LU] and DATA[LL].

CRC is calculated bytewise and two clocks are required to complete one CRC calculation.

The transpose and complement operations are performed on-the-fly when reading or writing values. See Transpose feature and Result complement for details.

27.2.2.2 Calculating a 32-bit CRC

Perform these steps to calculate a 32-bit CRC:

1. Write 1 to CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in Control (CTRL) as required for CRC calculation. See Transpose feature and Result complement for details.
3. Write a 32-bit polynomial to GPOLY[HIGH] and GPOLY[LOW] .
4. Write 1 to CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to DATA[HU], DATA[HL], DATA[LU], and DATA[LL].
6. Write 0 to CTRL[WAS] to start writing data values.
7. Write data values into DATA[HU], DATA[HL], DATA[LU], and DATA[LL]

   CRC is computed on every data value write and the intermediate CRC result is stored back into DATA[LU] and DATA[LL]

8. After writing all the values, read the final CRC result from DATA[HU], DATA[HL], DATA[LU], and DATA[LL].

CRC is calculated bytewise and two clocks are required to complete one CRC calculation.

The transpose and complement operations are performed on-the-fly when reading or writing values. See Transpose feature and Result complement for details.

27.2.3 Transpose feature

Transpose is not enabled by default. However, CRC requires input data and/or final checksum to be transposed. You have an option to configure each transpose operation separately to meet CRC standards. The data is transposed on-the-fly while being read or written.

Some protocols use the little-endian format for data stream to calculateCRC. In this case, transpose flips bits.

27.2.3.1 Types of transpose

CRC provides several types of transpose to flip bits and/or bytes for both writing input data and reading result separately using the CTRL[TOT] and CTRL[TOTR] according to the CRC calculation being used.

The following types of transpose are available for writing to and reading from DATA.

1. CTRL[TOT] or CTRL[TOTR] is 0.

   No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 1.

   Bits in a byte are transposed when bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}.

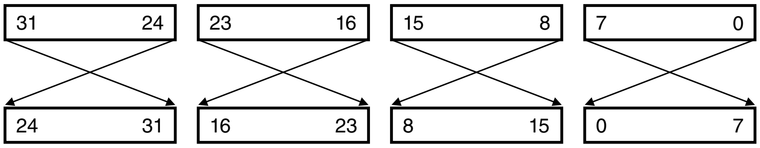


Figure 27-2. Transpose type 1b

3. CTRL[TOT] or CTRL[TOTR] is 10b.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes {reg[0:7], reg[8:15], reg[16:23], reg[24:31]}.

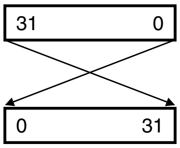


Figure 27-3. Transpose type 10b

4. CTRL[TOT] or CTRL[TOTR] is 11b.

   Bytes are transposed but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}.

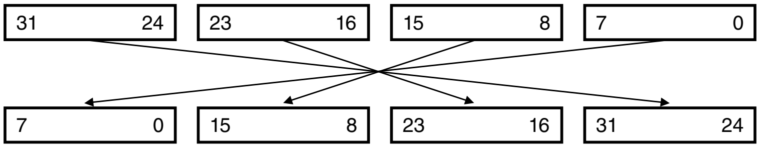


Figure 27-4. Transpose type 11b

NOTE

For 8-bit and 16-bit write accesses to Data (DATA), the data is transposed with 0s on the unused byte or bytes (taking 32 bits as a whole), but CRC is calculated on the valid byte(s) only. When reading the Data (DATA) for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in DATA[HU] and DATA[HL]. You must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

27.2.4 Result complement

When CTRL[FXOR] = 1, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in Data (DATA) every time Data (DATA) is read. When CTRL[FXOR] = 0,, reading Data (DATA) accesses the raw checksum value.

27.2.5 Clocking

Table 27-1. CRC clocks

Type of clock	Description
Bus clock (ipg_clk/ ipg_clk_s)	ipg_clk_s controls the access to the CRC registers. ipg_clk and ipg_clk_s function the CRC module.

27.2.6 Interrupts

This module has no interrupts.

27.3 Use cases

The following tables use the little-endian format.

27.3.1 CTRL programming

The following table shows Control (CTRL) programming for 16-bit CRC.

Table 27-2. CTRL programming for 16-bit CRC

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-16_CCITT_FALSE	1021h	FFFFh	0	0	0000h	0h	0h	0h
CRC-16_ARC	8005h	0000h	1	1	0000h	1h	2h	0h
CRC-16_AUG_CCITT	1021h	1D0Fh	0	0	0000h	0h	0h	0h
CRC-16_BUYPASS	8005h	0000h	0	0	0000h	0h	0h	0h
CRC-16_CCITT_ZERO	1021h	0000h	0	0	0000h	0h	0h	0h
CRC-16_CDMA2000	C867h	FFFFh	0	0	0000h	0h	0h	0h
CRC-16_DDS_110	8005h	800Dh	0	0	0000h	0h	0h	0h
CRC-16_DECT_X	589h	0000h	0	0	0000h	0h	0h	0h
CRC-16_DNP	3D65h	0000h	1	1	FFFFh	1h	2h	1h
CRC-16_EN_13757	3D65h	0000h	0	0	FFFFh	0h	0h	1h
CRC-16_GENIBUS	1021h	FFFFh	0	0	FFFFh	0h	0h	1h
CRC-16_MAXIM	8005h	0000h	1	1	FFFFh	1h	2h	1h
CRC-16_MCRF4XX	1021h	FFFFh	1	1	0000h	1h	2h	0h
CRC-16_RIELLO	1021h	B2AAh	1	1	0000h	1h	2h	0h
CRC-16_T10_DIF	8BB7h	0000h	0	0	0000h	0h	0h	0h
CRC-16_TELEDISK	A097h	0000h	0	0	0000h	0h	0h	0h
CRC-16_TMS37157	1021h	89ECh	1	1	0000h	1h	2h	0h
CRC-16_USB	8005h	FFFFh	1	1	FFFFh	1h	2h	1h
CRC-16_A	1021h	C6C6h	1	1	0000h	1h	2h	0h
CRC-16_KERMIT	1021h	0000h	1	1	0000h	1h	2h	0h
CRC-16_MODBUS	8005h	FFFFh	1	1	0000h	1h	2h	0h
CRC-16_X_25	1021h	FFFFh	1	1	FFFFh	1h	2h	1h
CRC-16_XMODEM	1021h	0000h	0	0	0000h	0h	0h	0h

The following table shows Control (CTRL) programming for 32-bit CRC.

Table 27-3. CTRL programming for 32-bit CRC

Algorithm	Polynomial	Seed	Ref in	Ref out	XOR out	CTRL[TOT]	CTRL[TOTR]	CTRL[FXOR]
CRC-32	04C11DB7h	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32_BZIP2	04C11DB7h	FFFFFFFFh	0	0	FFFF_FFFFh	0h	0h	1h
CRC-32C	1EDC6F41h	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32D	A833982Bh	FFFFFFFFh	1	1	FFFF_FFFFh	1h	2h	1h
CRC-32_MPEG-2	04C11DB7h	FFFFFFFFh	0	0	0000_0000h	0h	0h	0h
CRC-32_POSIX	04C11DB7h	00000000h	0	0	FFFF_FFFFh	0h	0h	1h
CRC-32Q	814141ABh	00000000h	0	0	0000_0000h	0h	0h	0h
CRC-32_JAMCRC	04C11DB7h	FFFFFFFFh	1	1	0000_0000h	1h	2h	0h
CRC-32_XFER	000000AFh	00000000h	0	0	0000_0000h	0h	0h	0h

27.3.2 Expected read data fields

The following table shows the expected read data fields for 16-bit CRC.

Table 27-4. Expected read data fields for 16-bit CRC

Algorithm	Data (DATA)
CRC16_CCITT_FALSE	[31:16] = Unknown[15:0] = Valid data
CRC16_ARC	[31:16] = Valid data [15:0] = Unknown
CRC16_AUG_CCITT	[31:16] = Unknown [15:0] = Valid data
CRC16_BUYPASS	[31:16] = Unknown [15:0] = Valid data
CRC16_CCITT_ZERO	[31:16] = Unknown [15:0] = Valid data
CRC16_CDMA2000	[31:16] = Unknown [15:0] = Valid data
CRC16_DDS_110	[31:16] = Unknown [15:0] = Valid data
CRC16_DECT_X	[31:16] = Unknown [15:0] = Valid data
CRC16_DNP	[31:16] = Valid data [15:0] = Unknown
CRC-16_EN_13757	[31:16] = Unknown [15:0] = Valid data
CRC-16_GENIBUS	[31:16] = Unknown [15:0] = Valid data
CRC-16_MAXIM	[31:16] = Valid data [15:0] = Unknown
CRC-16_MCRF4XX	[31:16] = Valid data [15:0] = Unknown
CRC-16_RIELLO	[31:16] = Valid data [15:0] = Unknown
CRC-16_T10_DIF	[31:16] = Unknown [15:0] = Valid data
CRC-16_TELEDISK	[31:16] = Unknown [15:0] = Valid data
CRC-16_TMS37157	[31:16] = Valid data [15:0] = Unknown
CRC-16_USB	[31:16] = Valid data [15:0] = Unknown
CRC-16_A	[31:16] = Valid data [15:0] = Unknown

Table continues on the next page...

Table 27-4. Expected read data fields for 16-bit CRC (continued)

Algorithm	Data (DATA)
CRC-16_KERMIT	[31:16] = Valid data [15:0] = Unknown
CRC-16_MODBUS	[31:16] = Valid data [15:0] = Unknown
CRC-16_X_25	[31:16] = Valid data [15:0] = Unknown
CRC-16_XMODEM	[31:16] = Unknown [15:0] = Valid data

The following table shows the expected read data fields for 32-bit CRC.

Table 27-5. Expected read data fields for 32-bit CRC

Algorithm	Data (DATA)
CRC-32	[31:0] = Valid data
CRC-32_BZIP2	[31:0] = Valid data
CRC-32C	[31:0] = Valid data
CRC-32D	[31:0] = Valid data
CRC-32_MPEG-2	[31:0] = Valid data
CRC-32_POSIX	[31:0] = Valid data
CRC-32Q	[31:0] = Valid data
CRC-32_JAMCRC	[31:0] = Valid data
CRC-32_XFER	[31:0] = Valid data

27.4 External signals

There is no CRC signal that connects off chip.

27.5 Initialization

To enable CRC calculation, you must program:

- CTRL[WAS] .
- Polynomial (GPOLY).
- Parameters for transposition and CRC result inversion in the applicable registers.

Writing 1 to CTRL[WAS] enables you to program the seed value into Data (DATA).

After a CRC calculation completes, you can reinitialize the module for a new CRC computation by again writing 1 to CTRL[WAS] and programming a new, or previously used, seed value. You must set all other parameters before programming the seed value and subsequent data values.

27.6 Memory map and register descriptions

NOTE

> You must reconfigure the CRC registers in case a transfer error occurs at the register programming interface.

The CRC module generates a transfer error in the following cases:

- Write accesses to the register addresses that are not mapped to the peripheral but included in the address space of the peripheral.
- Any read/write operation different from byte/halfword/word (free byte enables or other operations) in each register.

27.6.1 CRC register descriptions

27.6.1.1 CRC memory map

CRC base address: 4003_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Data (DATA)	32	RW	FFFF_FFFFh
4h	Polynomial (GPOLY)	32	RW	0000_1021h
8h	Control (CTRL)	32	RW	0000_0000h

27.6.1.2 Data (DATA)

27.6.1.2.1 Offset

Register	Offset
DATA	0h

27.6.1.2.2 Function

Configures the value of seed, data, and checksum. When CTRL[WAS] = 1, any write to this register is regarded as the seed value. When CTRL[WAS] becomes 0, any write to this register is regarded as data for general CRC calculation.

In 16-bit CRC mode, DATA[HU] and DATA[HL] 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, you can program to write 8 bits, 16 bits, or 32 bits in big endian order, provided all bytes are contiguous.

After writing all data values, you can read the CRC result from DATA register. In 16-bit CRC mode, the CRC result is available in DATA[LU] and DATA[LL]. In 32-bit CRC mode, all fields contain the result. Reads of this register, at any time, return the intermediate CRC value, if the CRC module is configured.

27.6.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HU								HL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LU								LL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

27.6.1.2.4 Fields

Field	Function
31-24 HU	Upper Part of High Byte Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 0. • In 16-bit CRC mode (CTRL[TCRC] = 0), this field is not used for programming a seed value. • In 32-bit CRC mode (CTRL[TCRC] = 1), the values written to this field are part of the seed value when CTRL[WAS] = 1.
23-16 HL	Lower Part of High Byte Generates CRC checksum in both 16-bit and 32-bit CRC modes if CTRL[WAS] = 1. • In 16-bit CRC mode (CTRL[TCRC] = 0), this field is not used for programming a seed value. • In 32-bit CRC mode (CTRL[TCRC] = 1), the values written to this field are part of the seed value when CTRL[WAS] = 1.

Table continues on the next page...

Field	Function
15-8 LU	Upper Part of Low Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 1,the values written to this field are part of the seed value.
7-0 LL	Lower Part of Low Byte Generates CRC checksum when CTRL[WAS] = 0. When CTRL[WAS] = 0, the values written to this field are part of the seed value.

27.6.1.3 Polynomial (GPOLY)

27.6.1.3.1 Offset

Register	Offset
GPOLY	4h

27.6.1.3.2 Function

Configures polynomial value for CRC calculation.

- Sets the upper 16 bits of polynomial that are used only in 32-bit CRC mode. Writes to this field are ignored in 16-bit CRC mode.
- Sets the lower 16 bits of polynomial that are used in both 16-bit and 32-bit CRC modes.

27.6.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HIGH															
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	LOW															
Reset	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1

27.6.1.3.4 Fields

Field	Function
31-16 HIGH	High Half-Word Writable and readable in 32-bit CRC mode (CTRL[TCRC] = 1). You cannot write to this field in 16-bit CRC mode (CTRL[TCRC] = 0).
15-0 LOW	Low Half-Word Writable and readable in both 16-bit and 32-bit CRC modes.

27.6.1.4 Control (CTRL)

27.6.1.4.1 Offset

Register	Offset
CTRL	8h

27.6.1.4.2 Function

Sets control for CRC. You must write 1 to the appropriate fields of this register before starting a new CRC calculation, which you can initialize by writing 1 to CTRL[WAS] and then writing the seed into DATA.

27.6.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TOT		TOTR		0	FXOR	WAS	TCRC	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															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

27.6.1.4.4 Fields

Field	Function
31-30	Transpose Type for Write

Table continues on the next page...

Field	Function
TOT	Sets transpose type for the values written to DATA. See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed, but bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed, no bits in a byte are transposed.
29-28 TOTR	Transpose Type for Read Sets transpose type for the values read from DATA . See Transpose feature for the available transpose options. 00b - No transposition 01b - Bits in bytes are transposed, but bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed, no bits in a byte are transposed.
27 —	Reserved
26 FXOR	Complement Read of CRC Data Register Enables on-the-fly complementing of read data. Some CRC protocols require the final checksum to be XORed with FFFFFFFFh or FFFFh. 0b - Disables XOR on reading data. 1b - Inverts or complements the read value of the CRC Data.
25 WAS	Write as Seed Specifies whether writes to DATA are data values or seed values. When this field = 1, the value that you write to is considered as seed value. When this field = 0, the value that you write to is considered as data for CRC calculation. 0b - Data values 1b - Seed values
24 TCRC	TCRC Defines the width of CRC. 0b - 16 bits 1b - 32 bits
23-0 —	Reserved

27.7 Usage Guide

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. The DATA register is written with MSB of data value first, thus the application with little-endian configured, the data write bytes transpose should be enabled when writing a 32bit value from variable to DATA register. After all data values are written, the CRC result can be read from this data register. For a 16-bit CRC result, if transpose options 10 and 11 is used, the resulting value after transposition resides in the CRC[HU:HL] fields.

This section shows two examples of using CRC module to implement typical CRC algorithms, including both 32-bit and 16-bit algorithms.

27.7.1 32-bit POSIX CRC

CRC-32/POSIX: width=32 poly=0x04c11db7 init=0x00000000 refin=false refout=false xorout=0xffffffff check=0x765e7680

```
uint32_t checksum32, dataSize;
uint8_t data[] = "123456789";
uint32_t *data32;

// Transport Bytes for data write, as the CRC_DATA requires MSB write first
// No transport for checksum read, enable complement read as xorout not zero
CRC_CTRL = CRC_CTRL_TOT(3) | CRC_CTRL_TOTR(0) | CRC_CTRL_FXOR(1) |
           CRC_CTRL_TCRC(1) | CRC_CTRL_WAS(0);
// write polynomial register
CRC_GPOLY = 0x04c11bd7;
// write pre-computed control register value along with WAS to start checksum computation
CRC_CTRL |= CRC_CTRL_WAS(1);
// write seed (initial checksum)
CRC_DATA = 0;
// deassert WAS by writing pre-computed CRC control register value
CRC_CTRL &= ~CRC_CTRL_WAS(1);

// write data
dataSize = sizeof(data);
// 8-bit reads and writes till source address is aligned 4 bytes */
while ((data) && ((uint32_t)data & 3U))
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// use 32-bit reads and writes as long as possible
data32 = (uint32_t *)data;
while (dataSize >= sizeof(uint32_t))
{
    CRC_DATA = *data32;
    data32++;
    dataSize -= sizeof(uint32_t);
}

data = (uint8_t *)data32;

// 8-bit reads and writes till end of data buffer
while (dataSize)
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// read 32bit checksum result
checksum32 = CRC_DATA;
```

27.7.2 16-bit KERMIT CRC

CRC-16/KERMIT: width=16 poly=0x1021 init=0x0000 refin=true refout=true xorout=0x0000 check=0x2189

```
uint32_t checksum16, dataSize;
uint8_t data[] = "123456789";
uint32_t *data32;

// Transport Bytes and Bits for both data write and read
// Bytes transport is because of the CRC_DATA requires MSB write first
// Bits transport is because of the KERMIT algorithm requirement
// No complement for checksum result
CRC_CTRL = CRC_CTRL_TOT(2) | CRC_CTRL_TOTR(2) | CRC_CTRL_FXOR(0) |
           CRC_CTRL_TCRC(0) | CRC_CTRL_WAS(0);
// write polynomial register
CRC_GPOLY = 0x1021;
// write pre-computed control register value along with WAS to start checksum computation
CRC_CTRL |= CRC_CTRL_WAS(1);
// write seed (initial checksum)
CRC_DATA = 0;
// deassert WAS by writing pre-computed CRC control register value
CRC_CTRL &= ~CRC_CTRL_WAS(1);

// write data
dataSize = sizeof(data);
// 8-bit reads and writes till source address is aligned 4 bytes */
while ((data) && ((uint32_t)data & 3U))
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// use 32-bit reads and writes as long as possible
data32 = (uint32_t *)data;
while (dataSize >= sizeof(uint32_t))
{
    CRC_DATA = *data32;
    data32++;
    dataSize -= sizeof(uint32_t);
}

data = (uint8_t *)data32;

// 8-bit reads and writes till end of data buffer
while (dataSize)
{
    CRC_DATA = *data;
    data++;
    dataSize--;
}

// due to the transport option TOTR >= 2
// read 16bit checksum result from CRC_DATA[HU:HL]
// otherwise, read checksum from CRC_DATA[LU:LL]
checksum16 = (CRC_DATA & 0xFFFF0000) >> 16;
```

Chapter 28
Debug

28.1 Introduction

This device's debug 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.

It 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 Basic BranchBuffer (BBB) capability to provide simple program trace.

This device supports only one debug interface, Serial Wire Debug (SWD).

28.2 Debug port pin descriptions

The debug port pins default to their SWD functionality after power-on-reset (POR).

Table 28-1. 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.
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.

28.3 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 Figure 28-1. 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.

A miscellaneous debug module (MDM) is implemented on this device, which contains the DAP control and status registers. 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 the table below.

Table 28-2. 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

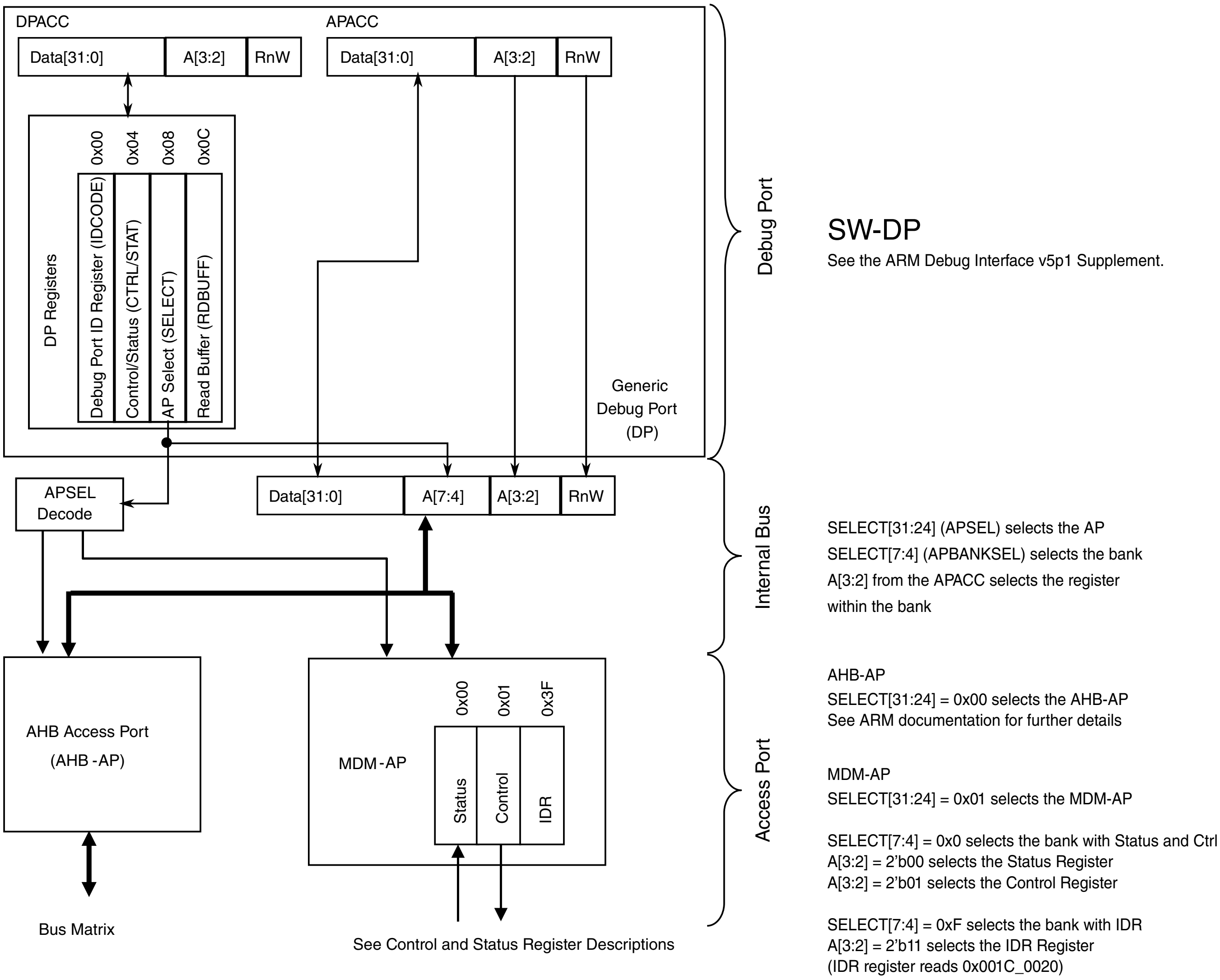

DPACC
Data[31:0]
A[3:2]
RnW
APACC
Data[31:0]
A[3:2]
RnW
DP Registers
0x00
0x04
0x08
0x0C
Debug Port ID Register (IDCODE)
Control/Status (CTRL/STAT)
AP Select (SELECT)
Read Buffer (RDBUFF)
Generic
Debug Port
(DP)
Debug Port
SW-DP
See the ARM Debug Interface v5p1 Supplement.
APSEL
Decode
Data[31:0]
A[7:4]
A[3:2]
RnW
Internal Bus
SELECT[31:24] (APSEL) selects the AP
SELECT[7:4] (APBANKSEL) selects the bank
A[3:2] from the APACC selects the register
within the bank
AHB Access Port
(AHB -AP)
MDM-AP
0x00
0x01
0x3F
Status
Control
IDR
Access Port
AHB-AP
SELECT[31:24] = 0x00 selects the AHB-AP
See ARM documentation for further details
MDM-AP
SELECT[31:24] = 0x01 selects the MDM-AP
SELECT[7:4] = 0x0 selects the bank with Status and Ctrl
A[3:2] = 2'b00 selects the Status Register
A[3:2] = 2'b01 selects the Control Register
SELECT[7:4] = 0xF selects the bank with IDR
A[3:2] = 2'b11 selects the IDR Register
(IDR register reads 0x001C_0020)
Bus Matrix
See Control and Status Register Descriptions

Figure 28-1. MDM AP addressing

28.3.1 MDM-AP status register

Table 28-3. MDM-AP status register assignments

Bit	Name	Description
0	Flash Mass Erase Acknowledge	The Flash Mass Erase Acknowledge field is cleared after POR reset. The field is also cleared at launch of a mass erase command due to write of Flash Mass Erase in Progress field in MDM AP Control Register. The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation.
1	Flash Ready	Indicates that flash memory has been initialized and debugger can be configured even if system is continuing to be held in reset via the debugger. 0 Flash is under initialization.

Table continues on the next page...

Table 28-3. MDM-AP status register assignments (continued)

Bit	Name	Description
		1 Flash is ready.
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 field indicates when the part is locked and no system bus access is possible. **NOTE:** This bit is not valid until Flash Ready bit set. 0 Device is unsecured. 1 Device is secured.
3	System Reset	Indicates the system reset state. 0 System is in reset. 1 System is not in reset.
4	Reserved	
5 – 15	Reserved for future use	Always read 0.
16	Core Halted	Indicates the core has entered Debug Halt mode 0 Core is not halted. 1 Core is halted.
17	Core SLEEPDEEP	SLEEPDEEP=1 indicates the core has entered Stop mode.
18	Core SLEEPING	SLEEPING=1 indicates the core has entered Wait mode.
19 – 31	Reserved for future use	Always reads 0.

28.3.2 MDM-AP Control register

Table 28-4. MDM-AP Control register assignments

Bit	Name	Secure[1]	Description
0	Flash Mass Erase in Progress	Y	Set to cause mass erase. Cleared by hardware after mass erase operation completes.
1	Debug Disable	N	Set to disable debug. Clear to allow debug operation. When set, it overrides the C_DEBUGEN field within the DHCSR[2] and forces to disable Debug logic.
2	Debug Request	N	Set to force the core to halt. If the core is in Stop or Wait mode, this field 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 this field is cleared. When this bit is set, $\overline{\text{RESET}}$ pin does not reflect the status of system reset and does not keep low.
4	Core Hold	N	Configuration field to control core operation at the end of system reset sequencing. 0 Normal operation—release the core from reset along with the rest of the system at the end of system reset sequencing.

Table continues on the next page...

Table 28-4. MDM-AP Control register assignments (continued)

Bit	Name	Secure[1]	Description
			1 Suspend operation—hold the core in reset at the end of reset sequencing. Once the system enters this suspended state, clearing this control bit immediately releases the core from reset and CPU operation begins.
5– 31	Reserved for future use	N	

[1] Command available in secure mode

[2] DHCSR: refer to the Debug Halting Control and Status Register in the ARMv6-M Architecture Reference Mannual.

28.4 Debug resets

The debug system receives the following sources of reset:

- 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.
- Writing 1 to the 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.

28.5 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 also be 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.

28.6 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.

- If 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.
- If the debugger logic is powered off, the debugger is reset on recovery and must be reconfigured once the low-power mode is exited.

The active debug will prevent the chip from entering low-power mode. In case the chip is already in low-power mode, a debug request from MDM-AP control register will wake the chip from low-power mode.

28.7 Debug and security

When flash security is enabled, 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 status register and can determine the current security state of the device. In the case of a secure device, the debugger has the capability of performing only a mass erase operation.

Chapter 29
Micro Trace Buffer (MTB)

29.1 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 and DWT capabilities.

29.1.1 Overview

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown as follows:

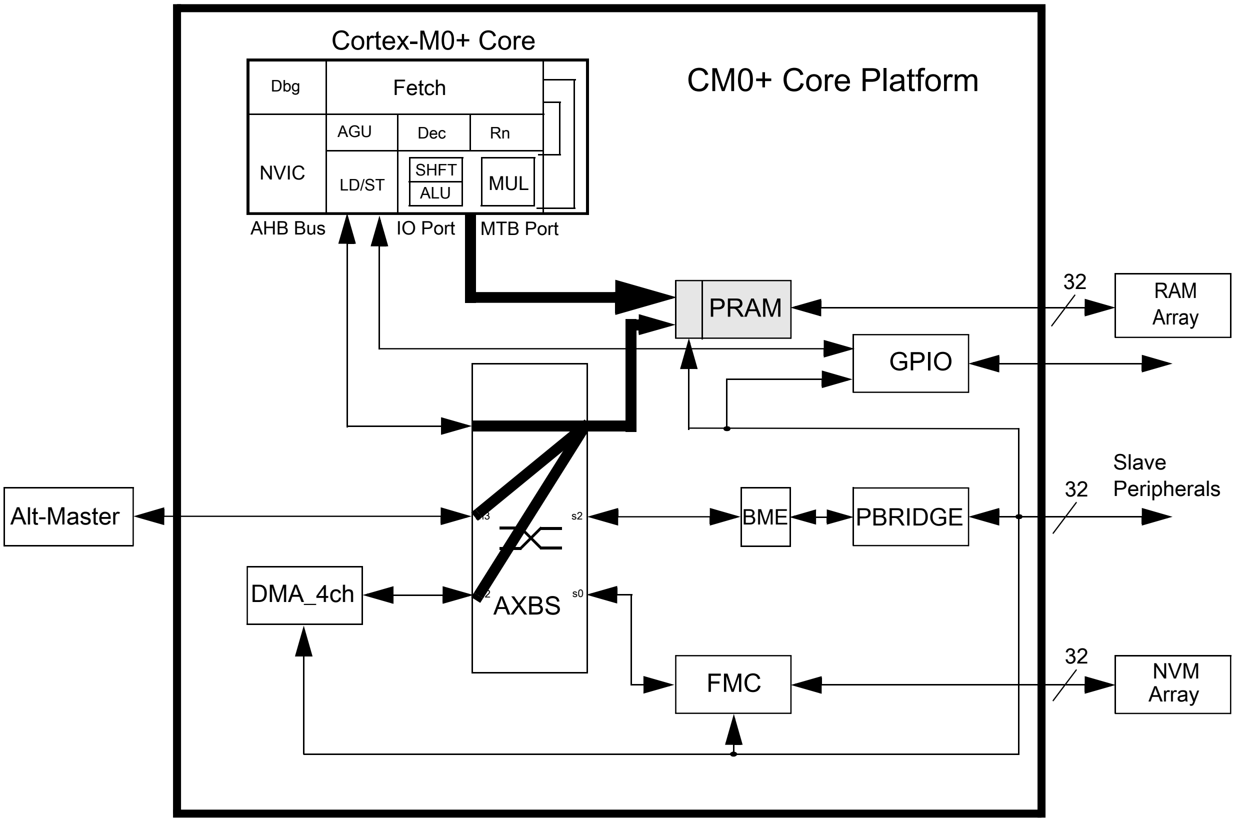


Figure 29-1. Generic Cortex-M0+ core platform block diagram

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 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.

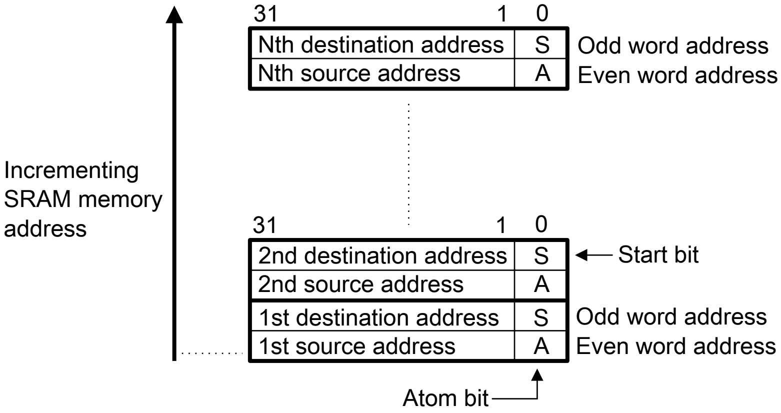


Figure 29-2. MTB execution trace storage format

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 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.

29.1.2 Features

The key features of the MTB and 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

29.1.3 Modes of operation

The MTB and DWT functions do not support any special modes of operation. The MTB 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 and DWT modules are memory-mapped, so their programming models can be accessed.

All functionality associated with the MTB and DWT modules resides in the core platform's clock domain; this includes its connections with the RAM array.

29.2 External signal description

The MTB and 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 controller.

Table 29-1. Private execution trace port from the core to MTB

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 DWT module and driven to the MTB 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.

29.3 Memory map and register definition

The MTB and 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

29.3.1 MTB register descriptions

29.3.1.1 MTB memory map

MTB base address: F000_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	MTB Position Register (POSITION)	32	RW	Table 29-1
4h	MTB Master Register (MASTER)	32	RW	Table 29-1
8h	MTB Flow Register (FLOW)	32	RW	Table 29-1
Ch	MTB Base Register (BASE)	32	R	Table 29-1

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
F00h	Integration Mode Control Register (MODECTRL)	32	R	0000_0000h
FA0h	Claim TAG Set Register (TAGSET)	32	R	0000_0000h
FA4h	Claim TAG Clear Register (TAGCLEAR)	32	R	0000_0000h
FB0h	Lock Access Register (LOCKACCESS)	32	R	0000_0000h
FB4h	Lock Status Register (LOCKSTAT)	32	R	0000_0000h
FB8h	Authentication Status Register (AUTHSTAT)	32	R	0000_0000h
FBCh	Device Architecture Register (DEVICEARCH)	32	R	4770_0A31h
FC8h	Device Configuration Register (DEVICECFG)	32	R	0000_0000h
FCCh	Device Type Identifier Register (DEVICETYPID)	32	R	0000_0031h
FD0h	Peripheral ID Register (PERIPHID4)	32	R	Table 29-1
FD4h	Peripheral ID Register (PERIPHID5)	32	R	Table 29-1
FD8h	Peripheral ID Register (PERIPHID6)	32	R	Table 29-1
FDCh	Peripheral ID Register (PERIPHID7)	32	R	Table 29-1
FE0h	Peripheral ID Register (PERIPHID0)	32	R	Table 29-1
FE4h	Peripheral ID Register (PERIPHID1)	32	R	Table 29-1
FE8h	Peripheral ID Register (PERIPHID2)	32	R	Table 29-1
FECh	Peripheral ID Register (PERIPHID3)	32	R	Table 29-1
FF0h - FFCh	Component ID Register (COMPID0 - COMPID3)	32	R	Table 29-1

29.3.1.2 MTB Position Register (POSITION)

29.3.1.2.1 Offset

Register	Offset
POSITION	0h

29.3.1.2.2 Function

The 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.

29.3.1.2.3 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	POINTER													WRAP	0	
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	0	0

29.3.1.2.4 Fields

Field	Function
31-16 —	Reserved These bits must be treated as UNK/SBZP (unknown on reads, should-be-zero-or-preserved on writes).
15-3 POINTER	Trace Packet Address Pointer 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: systemAddress = BASE+(POSITION & 0xFFFF_FFF8); **NOTE:** The size of the RAM is parameterized and the most significant bits of the POINTER field are RAZ/WI. For these devices, POSITION[31:16] == POSITION[POINTER[28:13] are RAZ/WI. Therefore, the active bits in this field are POSITION[15:3] == POSITION[POINTER[12:0]].
2 WRAP	WRAP This field is set to 1 automatically when the POINTER value wraps (as determined by the 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.
1-0 —	Reserved These bits must be treated as UNK/SBZP (unknown on reads, should-be-zero-or-preserved on writes).

29.3.1.3 MTB Master Register (MASTER)

29.3.1.3.1 Offset

Register	Offset
MASTER	4h

29.3.1.3.2 Function

The MASTER register contains the main program trace enable plus other trace controls. This register can be modified by the explicit programming model writes. MASTER[EN] and MASTER[HALTREQ] fields are also automatically updated by the MTB hardware.

Before MASTER[EN] or MASTER[TSTARTEN] are set to 1, the software must initialize the POSITION and FLOW registers.

If FLOW[WATERMARK] is used to stop tracing or to halt the processor, MASTER[MASK] must still be set to a value that prevents POSITION[POINTER] from wrapping before it reaches the FLOW[WATERMARK] value.

NOTE

> The format of this mask field is different than DWT_MASKn[MASK].

29.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EN	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						HALTREQ	RAMPRIV	SFRWPRIV	TSTOPEN	TSTARTEN	MASK				
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	u	u	u	u	u

29.3.1.3.4 Fields

Field	Function
31 EN	Main Trace Enable

Table continues on the next page...

Field	Function
	When this field is 1, trace data is written into the RAM memory location addressed by POSITION[POINTER]. The POSITION[POINTER] value auto increments after the trace data packet is written. EN can be automatically set to 0 using the FLOW[WATERMARK] field and the 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 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 FLOW[WATERMARK] or POSITION[POINTER] value is changed by software.
30-10 —	Reserved These bits must be treated as UNK/SBZP (unknown on reads, should-be-zero-or-preserved on writes).
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 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.
7 SFRWPRIV	Special Function Register Write Privilege If this field is 0, then user or privileged AHB read and write accesses to the MTB 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 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.
4-0 MASK	Mask This value determines the maximum size of the trace buffer in RAM. It specifies the most-significant bit of the POSITION[POINTER] field that can be updated by automatic increment. If the trace tries to advance past this power of 2, the POSITION[WRAP] bit is set to 1, the POSITION[MASK+3:3] == POSITION[POINTER[MASK:0]] bits are set to 0, and the POSITION[14:MASK+3] == 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 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.

29.3.1.4 MTB Flow Register (FLOW)

29.3.1.4.1 Offset

Register	Offset
FLOW	8h

29.3.1.4.2 Function

The 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 POSITION[POINTER] field value to point to the beginning of the trace buffer, or
- Setting FLOW[AUTOSTOP] = 0.

A debug agent can use FLOW[AUTOSTOP] to fill the trace buffer once only without halting the processor.A debug agent can use 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 FLOW[WATERMARK] field must be set below the final entry in the trace buffer region.

29.3.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WATERMARK															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WATERMARK													0	AUTOHALT	AUTOSTOP
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	0	u	u

29.3.1.4.4 Fields

Field	Function
31-3	WATERMARK[28:0]

Table continues on the next page...

Field	Function
WATERMARK	This field contains an address in the same format as the POSITION[POINTER] field. When POSITION[POINTER] matches the WATERMARK field value, actions defined by the AUTOHALT and AUTOSTOP bits are performed.
2 —	Reserved These bits must be treated as UNK/SBZP (unknown on reads, should-be-zero-or-preserved on writes).
1 AUTOHALT	AUTOHALT If this field is 1 and WATERMARK is equal to POSITION[POINTER], then 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 POSITION[POINTER], then MASTER[EN] is automatically set to 0. This stops tracing.

29.3.1.5 MTB Base Register (BASE)

29.3.1.5.1 Offset

Register	Offset
BASE	Ch

29.3.1.5.2 Function

The read-only 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.

29.3.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BASEADDR															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BASEADDR															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.1.5.4 Fields

Field	Function
31-0 BASEADDR	BASEADDR This value is defined (hardwired): 0x1FFF8000

29.3.1.6 Integration Mode Control Register (MODECTRL)

29.3.1.6.1 Offset

Register	Offset
MODECTRL	F00h

29.3.1.6.2 Function

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.

29.3.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MODECTRL															
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	MODECTRL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.6.4 Fields

Field	Function
31-0 MODECTRL	MODECTRL Hardwired to 0x0000_0000

29.3.1.7 Claim TAG Set Register (TAGSET)

29.3.1.7.1 Offset

Register	Offset
TAGSET	FA0h

29.3.1.7.2 Function

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.

29.3.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAGSET															
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	TAGSET															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.7.4 Fields

Field	Function
31-0 TAGSET	TAGSET Hardwired to 0x0000_0000

29.3.1.8 Claim TAG Clear Register (TAGCLEAR)

29.3.1.8.1 Offset

Register	Offset
TAGCLEAR	FA4h

29.3.1.8.2 Function

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.

29.3.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAGCLEAR															
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	TAGCLEAR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.8.4 Fields

Field	Function
31-0 TAGCLEAR	TAGCLEAR Hardwired to 0x0000_0000

29.3.1.9 Lock Access Register (LOCKACCESS)

29.3.1.9.1 Offset

Register	Offset
LOCKACCESS	FB0h

29.3.1.9.2 Function

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.

29.3.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCKACCESS															
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	LOCKACCESS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.9.4 Fields

Field	Function
31-0 LOCKACCESS	LOCKACCESS Hardwired to 0x0000_0000

29.3.1.10 Lock Status Register (LOCKSTAT)

29.3.1.10.1 Offset

Register	Offset
LOCKSTAT	FB4h

29.3.1.10.2 Function

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.

29.3.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCKSTAT															
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	LOCKSTAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.10.4 Fields

Field	Function
31-0 LOCKSTAT	LOCKSTAT Hardwired to 0x0000_0000

29.3.1.11 Authentication Status Register (AUTHSTAT)

29.3.1.11.1 Offset

Register	Offset
AUTHSTAT	FB8h

29.3.1.11.2 Function

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.

AUTHSTAT[3:2] indicates if nonsecure, noninvasive debug is enabled or disabled, while 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.

29.3.1.11.3 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												1	BIT2	1	BIT0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.11.4 Fields

Field	Function
31-4 —	Reserved This field is hardwired to 0x0000000.
3 BIT3	BIT3 Hardwired to 1.
2 BIT2	BIT2 Connected to NIDEN or DBGEN signal.
1 BIT1	BIT1 Hardwired to 1.
0 BIT0	BIT0 Connected to DBGEN.

29.3.1.12 Device Architecture Register (DEVICEARCH)

29.3.1.12.1 Offset

Register	Offset
DEVICEARCH	FBCh

29.3.1.12.2 Function

This register indicates the device architecture. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVICEARCH															
W																
Reset	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DEVICEARCH															
W																
Reset	0	0	0	0	1	0	1	0	0	0	1	1	0	0	0	1

29.3.1.12.4 Fields

Field	Function
31-0 DEVICEARCH	DEVICEARCH Hardwired to 0x4770_0A31.

29.3.1.13 Device Configuration Register (DEVICECFG)

29.3.1.13.1 Offset

Register	Offset
DEVICECFG	FC8h

29.3.1.13.2 Function

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVICECFG															
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	DEVICECFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.1.13.4 Fields

Field	Function
31-0 DEVICECFG	DEVICECFG Hardwired to 0x0000_0000.

29.3.1.14 Device Type Identifier Register (DEVICETYPID)

29.3.1.14.1 Offset

Register	Offset
DEVICETYPID	FCCh

29.3.1.14.2 Function

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVICETYPID															
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	DEVICETYPID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1

29.3.1.14.4 Fields

Field	Function
31-0 DEVICETYPID	DEVICETYPID Hardwired to 0x0000_0031.

29.3.1.15 Peripheral ID Register (PERIPHID0 - PERIPHID7)

29.3.1.15.1 Offset

Register	Offset
PERIPHID4	FD0h
PERIPHID5	FD4h
PERIPHID6	FD8h
PERIPHID7	FDCh
PERIPHID0	FE0h
PERIPHID1	FE4h
PERIPHID2	FE8h
PERIPHID3	FECh

29.3.1.15.2 Function

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIPHID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIPHID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.1.15.4 Fields

Field	Function
31-0 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.

29.3.1.16 Component ID Register (COMPID0 - COMPID3)

29.3.1.16.1 Offset

Register	Offset
COMPID0	FF0h
COMPID1	FF4h
COMPID2	FF8h
COMPID3	FFCh

29.3.1.16.2 Function

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COMPID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMPID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.1.16.4 Fields

Field	Function
31-0 COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

29.3.2 DWT register descriptions

29.3.2.1 DWT memory map

The DWT programming model supports a very simplified subset of the v7M debug architecture and follows the standard ARM DWT definition.

MTBDWT base address: F000_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DWT Control Register (CTRL)	32	R	2F00_0000h
20h	DWT Comparator Register (COMP0)	32	RW	0000_0000h
24h	DWT Comparator Mask Register (MASK0)	32	RW	0000_0000h
28h	DWT Comparator Function Register 0 (FCT0)	32	RW	0000_0000h
30h	DWT Comparator Register (COMP1)	32	RW	0000_0000h
34h	DWT Comparator Mask Register (MASK1)	32	RW	0000_0000h
38h	DWT Comparator Function Register 1 (FCT1)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
200h	DWT Trace Buffer Control Register (TBCTRL)	32	RW	2000_0000h
FC8h	Device Configuration Register (DEVICECFG)	32	R	0000_0000h
FCCh	Device Type Identifier Register (DEVICETYPID)	32	R	0000_0004h
FD0h	Peripheral ID Register (PERIPHID4)	32	R	Table 29-1
FD4h	Peripheral ID Register (PERIPHID5)	32	R	Table 29-1
FD8h	Peripheral ID Register (PERIPHID6)	32	R	Table 29-1
FDCh	Peripheral ID Register (PERIPHID7)	32	R	Table 29-1
FE0h	Peripheral ID Register (PERIPHID0)	32	R	Table 29-1
FE4h	Peripheral ID Register (PERIPHID1)	32	R	Table 29-1
FE8h	Peripheral ID Register (PERIPHID2)	32	R	Table 29-1
FECh	Peripheral ID Register (PERIPHID3)	32	R	Table 29-1
FF0h - FFCh	Component ID Register (COMPID0 - COMPID3)	32	R	Table 29-1

29.3.2.2 DWT Control Register (CTRL)

29.3.2.2.1 Offset

Register	Offset
CTRL	0h

29.3.2.2.2 Function

This register provides read-only information on the watchpoint configuration for the DWT.

29.3.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NUMCMP				DWTCFGCTRL											
W																
Reset	0	0	1	0	1	1	1	1	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	DWTCFGCTRL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.2.4 Fields

Field	Function
31-28 NUMCMP	Number of comparators The DWT implements two comparators.
27-0 DWTCFGCTRL	DWT configuration controls This field is hardwired to 0xF00_0000, disabling all the remaining DWT functionality. The specific fields and their state are: CTRL[27] = NOTRCPKT = 1, trace sample and exception trace is not supported CTRL[26] = NOEXTTRIG = 1, external match signals are not supported CTRL[25] = NOCYCCNT = 1, cycle counter is not supported CTRL[24] = NOPRFCNT = 1, profiling counters are not supported CTRL[22] = CYCEBTENA = 0, no POSTCNT underflow packets generated CTRL[21] = FOLDEVTENA = 0, no folded instruction counter overflow events CTRL[20] = LSUEVTENA = 0, no LSU counter overflow events CTRL[19] = SLEEPEVTENA = 0, no sleep counter overflow events CTRL[18] = EXCEVTENA = 0, no exception overhead counter events CTRL[17] = CPIEVTENA = 0, no CPI counter overflow events CTRL[16] = EXCTRCENA = 0, generation of exception trace disabled CTRL[12] = PCSAMPLENA = 0, no periodic PC sample packets generated CTRL[11:10] = SYNCTAP = 0, no synchronization packets CTRL[9] = CYCTAP = 0, cycle counter is not supported CTRL[8:5] = POSTINIT = 0, cycle counter is not supported CTRL[4:1] = POSTPRESET = 0, cycle counter is not supported CTRL[0] = CYCCNTENA = 0, cycle counter is not supported

29.3.2.3 DWT Comparator Register (COMP0 - COMP1)

29.3.2.3.1 Offset

Register	Offset
COMP0	20h
COMP1	30h

29.3.2.3.2 Function

The COMPn registers provide the reference value for comparator n.

29.3.2.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMP															
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	COMP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.3.4 Fields

Field	Function
31-0 COMP	Reference value for comparison If 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".

29.3.2.4 DWT Comparator Mask Register (MASK0 - MASK1)

29.3.2.4.1 Offset

Register	Offset
MASK0	24h
MASK1	34h

29.3.2.4.2 Function

The 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].

29.3.2.4.3 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											MASK				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.4.4 Fields

Field	Function
31-5 —	Reserved
4-0 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. 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 DWT hardware to 24. If COMP0 is used as a data value comparator, then MASK0 should be programmed to zero.

29.3.2.5 DWT Comparator Function Register 0 (FCT0)

29.3.2.5.1 Offset

Register	Offset
FCT0	28h

29.3.2.5.2 Function

The FCTn registers control the operation of comparator n.

29.3.2.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MATCHED	0				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	DATAVADDR0				DATAVSIZE		0	DATAVMATCH	0				FUNCTION			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.5.4 Fields

Field	Function
31-25 —	Reserved
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. 0b - No match. 1b - Match occurred.
23-20 —	Reserved
19-16 —	Reserved Data Value Address 1. Reserved, RAZ/WI.
15-12 DATAVADDR0	Data Value Address 0 Since the 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 COMP0 is used as a data watchpoint and 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. 00b - Byte. 01b - Halfword. 10b - Word. 11b - Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

Table continues on the next page...

Field	Function
9 —	Reserved Data Value Address 1. Reserved, RAZ/WI.
8 DATAVMATCH	Data Value Match When this field is 1, it enables data value comparison. For this implementation, COMP0 supports address or data value comparisons; COMP1 only supports address comparisons. 0b - Perform address comparison. 1b - Perform data value comparison.
7-4 —	Reserved Data Value Address 1. Reserved, RAZ/WI.
3-0 FUNCTION	Function Selects the action taken on a comparator match. If COMP0 is used for a data value and COMP1 for an address value, then FCT1[FUNCTION] must be set to zero. For this configuration, MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses. 0000b - Disabled. 0001b-0011b - Reserved. Any attempts to use this value results in UNPREDICTABLE behavior. 0100b - Instruction fetch. 0101b - Data operand read. 0110b - Data operand write. 0111b - Data operand (read + write). 1000b-1111b - Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

29.3.2.6 DWT Comparator Function Register 1 (FCT1)

29.3.2.6.1 Offset

Register	Offset
FCT1	38h

29.3.2.6.2 Function

The FCTn registers control the operation of comparator n. Since the DWT only supports data value comparisons on comparator 0, there are several fields in the FCT1 register that are RAZ/WI (bits 12, 11:10, 8).

29.3.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MATCHED	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												FUNCTION			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.6.4 Fields

Field	Function
31-25 —	Reserved
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. 0b - No match. 1b - Match occurred.
23-4 —	Reserved
3-0 FUNCTION	Function Selects the action taken on a comparator match. If COMP0 is used for a data value and COMP1 for an address value, then FCT1[FUNCTION] must be set to zero. For this configuration, MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses. 0000b - Disabled. 0001b-0011b - Reserved. Any attempts to use this value results in UNPREDICTABLE behavior. 0100b - Instruction fetch. 0101b - Data operand read. 0110b - Data operand write. 0111b - Data operand (read + write). 1000b-1111b - Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.

29.3.2.7 DWT Trace Buffer Control Register (TBCTRL)

29.3.2.7.1 Offset

Register	Offset
TBCTRL	200h

29.3.2.7.2 Function

The 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.

29.3.2.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NUMCOMP				0											
W																
Reset	0	0	1	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														ACOMP1	ACOMP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.7.4 Fields

Field	Function
31-28 NUMCOMP	Number of Comparators This read-only field specifies the number of comparators in the DWT. This implementation includes two registers.
27-2 —	Reserved RAZ/WI
1 ACOMP1	Action based on Comparator 1 match When the FCT1[MATCHED] is set, it indicates COMP1 address compare has triggered and the trace buffer's recording state is changed.

Table continues on the next page...

Field	Function
	0b - Trigger TSTOP based on the assertion of FCT1[MATCHED]. 1b - Trigger TSTART based on the assertion of FCT1[MATCHED].
0 ACOMP0	Action based on Comparator 0 match When the FCT0[MATCHED] is set, it indicates COMP0 address compare has triggered and the trace buffer's recording state is changed. The assertion of FCT0[MATCHED] is caused by the following conditions: • Address match in COMP0 when FCT0[DATAVMATCH] = 0 • Data match in COMP0 when FCT0[DATAVMATCH, DATAVADDR0] = {1,0} • Data match in COMP0 and address match in COMP1 when FCT0[DATAVMATCH, DATAVADDR0] = {1,1} 0b - Trigger TSTOP based on the assertion of FCT0[MATCHED]. 1b - Trigger TSTART based on the assertion of FCT0[MATCHED].

29.3.2.8 Device Configuration Register (DEVICECFG)

29.3.2.8.1 Offset

Register	Offset
DEVICECFG	FC8h

29.3.2.8.2 Function

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.2.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVICECFG															
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	DEVICECFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.2.8.4 Fields

Field	Function
31-0 DEVICECFG	DEVICECFG Hardwired to 0x0000_0000.

29.3.2.9 Device Type Identifier Register (DEVICETYPID)

29.3.2.9.1 Offset

Register	Offset
DEVICETYPID	FCCh

29.3.2.9.2 Function

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.2.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVICETYPID															
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	DEVICETYPID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

29.3.2.9.4 Fields

Field	Function
31-0 DEVICETYPID	DEVICETYPID Hardwired to 0x0000_0004.

29.3.2.10 Peripheral ID Register (PERIPHID0 - PERIPHID7)

29.3.2.10.1 Offset

Register	Offset
PERIPHID4	FD0h
PERIPHID5	FD4h
PERIPHID6	FD8h
PERIPHID7	FDCh
PERIPHID0	FE0h
PERIPHID1	FE4h
PERIPHID2	FE8h
PERIPHID3	FECh

29.3.2.10.2 Function

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.2.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIPHID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIPHID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.2.10.4 Fields

Field	Function
31-0 PERIPHID	PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000.

29.3.2.11 Component ID Register (COMPID0 - COMPID3)

29.3.2.11.1 Offset

Register	Offset
COMPID0	FF0h
COMPID1	FF4h
COMPID2	FF8h
COMPID3	FFCh

29.3.2.11.2 Function

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.2.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COMPID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMPID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.2.11.4 Fields

Field	Function
31-0 COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

29.3.3 ROM register descriptions

29.3.3.1 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.

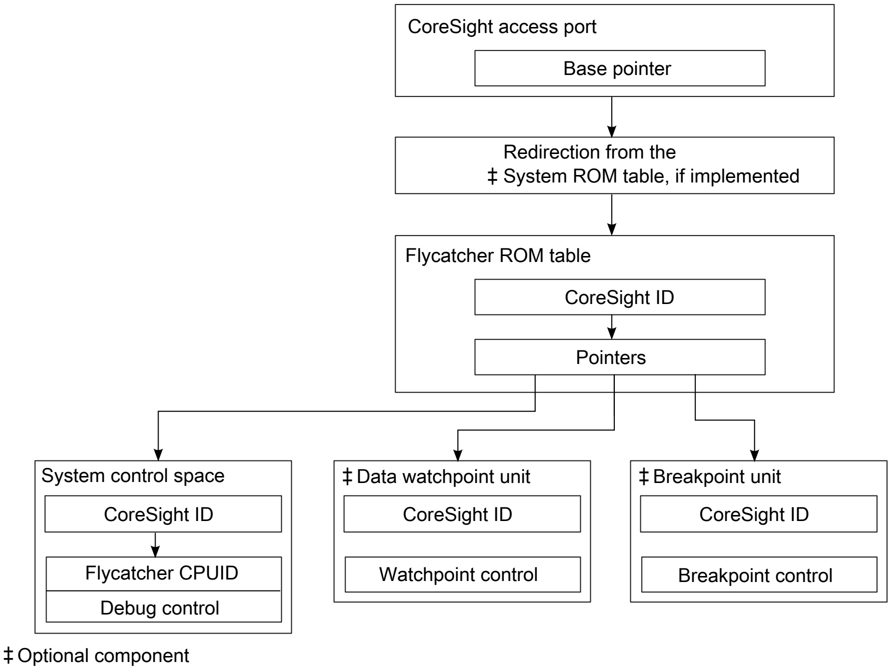


Figure 29-3. CoreSight discovery process

MTBDWTROM base address: F000_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Ch	Entry (ENTRY0 - ENTRY3)	32	R	Table 29-1

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
10h	End of Table Marker Register (TABLEMARK)	32	R	0000_0000h
FCCh	System Access Register (SYSACCESS)	32	R	0000_0001h
FD0h	Peripheral ID Register (PERIPHID4)	32	R	Table 29-1
FD4h	Peripheral ID Register (PERIPHID5)	32	R	Table 29-1
FD8h	Peripheral ID Register (PERIPHID6)	32	R	Table 29-1
FDCh	Peripheral ID Register (PERIPHID7)	32	R	Table 29-1
FE0h	Peripheral ID Register (PERIPHID0)	32	R	Table 29-1
FE4h	Peripheral ID Register (PERIPHID1)	32	R	Table 29-1
FE8h	Peripheral ID Register (PERIPHID2)	32	R	Table 29-1
FECh	Peripheral ID Register (PERIPHID3)	32	R	Table 29-1
FF0h - FFCh	Component ID Register (COMPID0 - COMPID3)	32	R	Table 29-1

29.3.3.2 Entry (ENTRY0 - ENTRY3)

29.3.3.2.1 Offset

Register	Offset
ENTRY0	0h
ENTRY1	4h
ENTRY2	8h
ENTRY3	Ch

29.3.3.2.2 Function

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.

29.3.3.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENTRY															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENTRY															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.3.2.4 Fields

Field	Function
31-0 ENTRY	ENTRY Entry*n* values are hardwired: Entry 0 (MTB) is hardwired to 0xFFFF_E003, Entry 1 (DWT) to 0xFFFF_F003, Entry 2 (CM0PCTI) to 0x0000_4003, Entry 3 (CM0+ ROM Table) to 0xF00F_D003

29.3.3.3 End of Table Marker Register (TABLEMARK)

29.3.3.3.1 Offset

Register	Offset
TABLEMARK	10h

29.3.3.3.2 Function

This register indicates end of table marker. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.3.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MARK															
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	MARK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

29.3.3.3.4 Fields

Field	Function
31-0 MARK	MARK Hardwired to 0x0000_0000

29.3.3.4 System Access Register (SYSACCESS)

29.3.3.4.1 Offset

Register	Offset
SYSACCESS	FCCh

29.3.3.4.2 Function

This register indicates system access. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.3.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SYSACCESS															
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	SYSACCESS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

29.3.3.4.4 Fields

Field	Function
31-0 SYSACCESS	SYSACCESS Hardwired to 0x0000_0001

29.3.3.5 Peripheral ID Register (PERIPHID0 - PERIPHID7)

29.3.3.5.1 Offset

Register	Offset
PERIPHID4	FD0h
PERIPHID5	FD4h
PERIPHID6	FD8h
PERIPHID7	FDCh
PERIPHID0	FE0h
PERIPHID1	FE4h
PERIPHID2	FE8h
PERIPHID3	FECh

29.3.3.5.2 Function

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.3.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIPHID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIPHID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.3.5.4 Fields

Field	Function
31-0 PERIPHID	PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000.

29.3.3.6 Component ID Register (COMPID0 - COMPID3)

29.3.3.6.1 Offset

Register	Offset
COMPID0	FF0h
COMPID1	FF4h
COMPID2	FF8h
COMPID3	FFCh

29.3.3.6.2 Function

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

29.3.3.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COMPID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMPID															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

29.3.3.6.4 Fields

Field	Function
31-0 COMPID	Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0010; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.

29.4 Usage Guide

29.4.1 ARM reference

For more information about MTB, please refer to the ARM document ARM Debug Interface Architecture Specification .

Chapter 30
Port Control and Interrupts (PORT)

30.1 Chip-specific information for this module

30.1.1 I/O pin structure

The following figure shows the structure of normal I/O pin.

See the "Pin properties" section in DataSheet for properties on each pin.

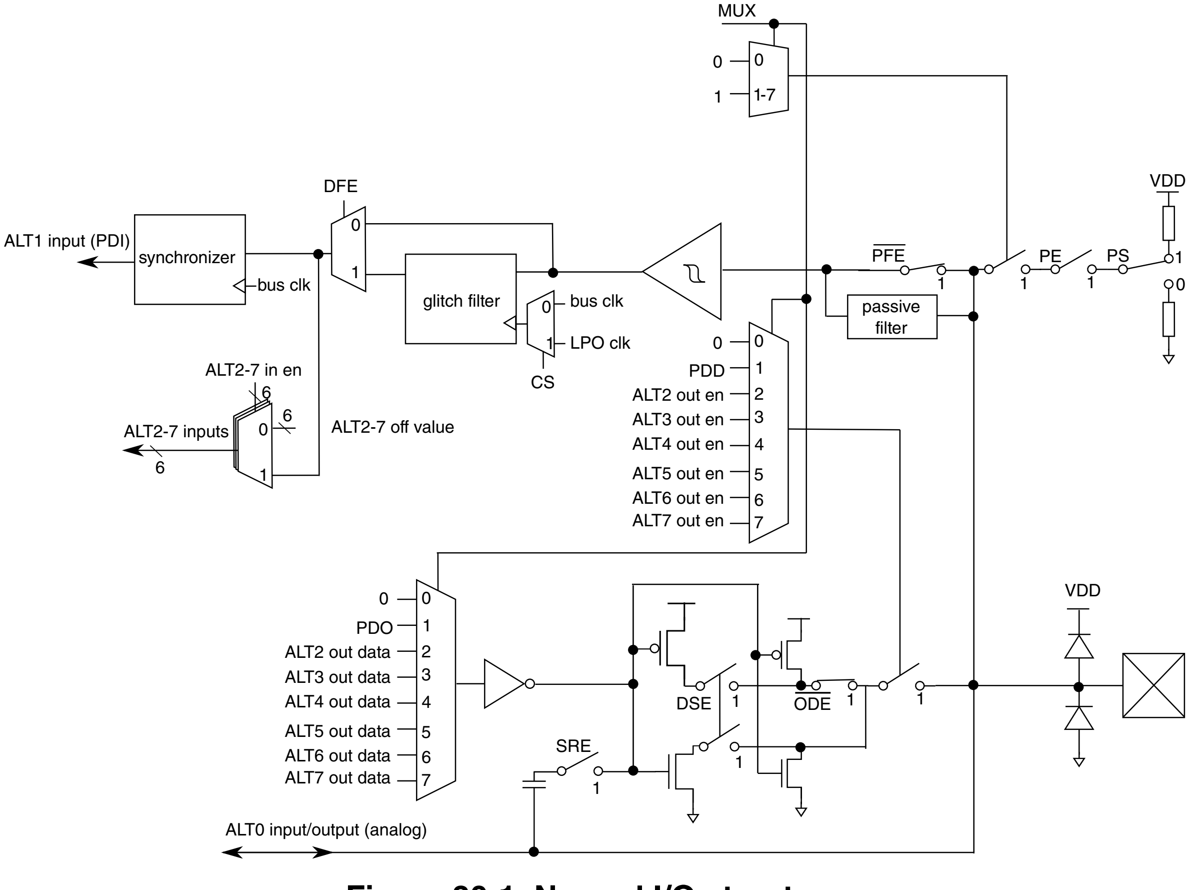


Figure 30-1. Normal I/O structure

30.1.2 Port control and interrupt module features

- 32-pin ports

NOTE

Not all pins are available on the device. See the "GPIO Signal Descriptions" table in DataSheet, and the following section for details.

- Each 32-pin port is assigned one interrupt.

Table 30-1. Ports summary

Feature	Port A	Port B	Port C	Port D	Port E
Pull select control	Yes	Yes	Yes	Yes	Yes
Pull select at reset	PTA4/PTA5=Pull up, Others=No	No	PTC4=Pull down, Others=No	PTD3=Pull up, Others=No	No
Pull enable control	Yes	Yes	Yes	Yes	Yes
Pull enable at reset	PTA4/ PTA5=Enabled; Others=Disabled	Disabled	PTC4=Enabled; Others=Disabled	PTD3=Enabled; Others=Disabled	Disabled
Passive filter enable control	PTA5=Yes; Others=No	No	No	PTD3=Yes; Others=No	No
Passive filter enable at reset	PTA5=Enabled; Others=Disabled	Disabled	Disabled	Disabled	Disabled
Open drain enable control	I2C and UART Tx=Enabled; Others=Disabled	I2C and UART Tx=Enabled; Others=Disabled	I2C and UART Tx=Enabled; Others=Disabled	I2C and UART Tx=Enabled; Others=Disabled	I2C and UART Tx=Enabled; Others=Disabled
Open drain enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Drive strength enable control	No	PTB4/PTB5 only	No	PTD0/PTD1/ PTD15/PTD16 only	PTE0/PTE1 only
Drive strength enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Pin mux control	Yes	Yes	Yes	Yes	Yes
Pin mux at reset	PTA4/PTA5=ALT7; Others=ALT0	ALT0	PTC4=ALT7; Others=ALT0	PTD3=ALT7; Others=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	Yes

30.1.3 Application-related Information

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.
3. The clock to the port control module can be gated on and off using the PCC_PORTx register. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing the corresponding module, set PCC_PORTx[CGC] to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution chapter.

30.2 Introduction

30.2.1 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.

30.2.2 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
- Digital input filter
  - 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
  - Individual drive strength field supporting high and low drive strength
  - Individual input passive filter field supporting enable and disable of the individual input passive filter
  - 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.

30.2.3 Modes of operation

30.2.3.1 Run mode

In Run mode, the PORT operates normally.

30.2.3.2 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.

30.2.3.3 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.

30.2.3.4 Debug mode

In Debug mode, PORT operates normally.

30.3 External signal description

The table found here describes the PORT external signal.

Table 30-2. 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.

30.4 Detailed signal description

The table found here contains the detailed signal description for the PORT interface.

Table 30-3. 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.

30.5 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
4004_9000	Pin Control Register n (PORTA_PCR0)	32	R/W	See section	30.5.1/618
4004_9004	Pin Control Register n (PORTA_PCR1)	32	R/W	See section	30.5.1/618
4004_9008	Pin Control Register n (PORTA_PCR2)	32	R/W	See section	30.5.1/618

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_900C	Pin Control Register n (PORTA_PCR3)	32	R/W	See section	30.5.1/618
4004_9010	Pin Control Register n (PORTA_PCR4)	32	R/W	See section	30.5.1/618
4004_9014	Pin Control Register n (PORTA_PCR5)	32	R/W	See section	30.5.1/618
4004_9018	Pin Control Register n (PORTA_PCR6)	32	R/W	See section	30.5.1/618
4004_901C	Pin Control Register n (PORTA_PCR7)	32	R/W	See section	30.5.1/618
4004_9020	Pin Control Register n (PORTA_PCR8)	32	R/W	See section	30.5.1/618
4004_9024	Pin Control Register n (PORTA_PCR9)	32	R/W	See section	30.5.1/618
4004_9028	Pin Control Register n (PORTA_PCR10)	32	R/W	See section	30.5.1/618
4004_902C	Pin Control Register n (PORTA_PCR11)	32	R/W	See section	30.5.1/618
4004_9030	Pin Control Register n (PORTA_PCR12)	32	R/W	See section	30.5.1/618
4004_9034	Pin Control Register n (PORTA_PCR13)	32	R/W	See section	30.5.1/618
4004_9038	Pin Control Register n (PORTA_PCR14)	32	R/W	See section	30.5.1/618
4004_903C	Pin Control Register n (PORTA_PCR15)	32	R/W	See section	30.5.1/618
4004_9040	Pin Control Register n (PORTA_PCR16)	32	R/W	See section	30.5.1/618
4004_9044	Pin Control Register n (PORTA_PCR17)	32	R/W	See section	30.5.1/618
4004_9048	Pin Control Register n (PORTA_PCR18)	32	R/W	See section	30.5.1/618
4004_904C	Pin Control Register n (PORTA_PCR19)	32	R/W	See section	30.5.1/618
4004_9050	Pin Control Register n (PORTA_PCR20)	32	R/W	See section	30.5.1/618
4004_9054	Pin Control Register n (PORTA_PCR21)	32	R/W	See section	30.5.1/618
4004_9058	Pin Control Register n (PORTA_PCR22)	32	R/W	See section	30.5.1/618
4004_905C	Pin Control Register n (PORTA_PCR23)	32	R/W	See section	30.5.1/618
4004_9060	Pin Control Register n (PORTA_PCR24)	32	R/W	See section	30.5.1/618
4004_9064	Pin Control Register n (PORTA_PCR25)	32	R/W	See section	30.5.1/618
4004_9068	Pin Control Register n (PORTA_PCR26)	32	R/W	See section	30.5.1/618
4004_906C	Pin Control Register n (PORTA_PCR27)	32	R/W	See section	30.5.1/618
4004_9070	Pin Control Register n (PORTA_PCR28)	32	R/W	See section	30.5.1/618
4004_9074	Pin Control Register n (PORTA_PCR29)	32	R/W	See section	30.5.1/618
4004_9078	Pin Control Register n (PORTA_PCR30)	32	R/W	See section	30.5.1/618
4004_907C	Pin Control Register n (PORTA_PCR31)	32	R/W	See section	30.5.1/618
4004_9080	Global Pin Control Low Register (PORTA_GPCLR)	32	W (always reads 0)	0000_0000h	30.5.2/621
4004_9084	Global Pin Control High Register (PORTA_GPCHR)	32	W (always reads 0)	0000_0000h	30.5.3/621
4004_90A0	Interrupt Status Flag Register (PORTA_ISFR)	32	w1c	0000_0000h	30.5.4/622
4004_90C0	Digital Filter Enable Register (PORTA_DFER)	32	R/W	0000_0000h	30.5.5/622
4004_90C4	Digital Filter Clock Register (PORTA_DFCR)	32	R/W	0000_0000h	30.5.6/623
4004_90C8	Digital Filter Width Register (PORTA_DFWR)	32	R/W	0000_0000h	30.5.7/623

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A000	Pin Control Register n (PORTB_PCR0)	32	R/W	See section	30.5.1/618
4004_A004	Pin Control Register n (PORTB_PCR1)	32	R/W	See section	30.5.1/618
4004_A008	Pin Control Register n (PORTB_PCR2)	32	R/W	See section	30.5.1/618
4004_A00C	Pin Control Register n (PORTB_PCR3)	32	R/W	See section	30.5.1/618
4004_A010	Pin Control Register n (PORTB_PCR4)	32	R/W	See section	30.5.1/618
4004_A014	Pin Control Register n (PORTB_PCR5)	32	R/W	See section	30.5.1/618
4004_A018	Pin Control Register n (PORTB_PCR6)	32	R/W	See section	30.5.1/618
4004_A01C	Pin Control Register n (PORTB_PCR7)	32	R/W	See section	30.5.1/618
4004_A020	Pin Control Register n (PORTB_PCR8)	32	R/W	See section	30.5.1/618
4004_A024	Pin Control Register n (PORTB_PCR9)	32	R/W	See section	30.5.1/618
4004_A028	Pin Control Register n (PORTB_PCR10)	32	R/W	See section	30.5.1/618
4004_A02C	Pin Control Register n (PORTB_PCR11)	32	R/W	See section	30.5.1/618
4004_A030	Pin Control Register n (PORTB_PCR12)	32	R/W	See section	30.5.1/618
4004_A034	Pin Control Register n (PORTB_PCR13)	32	R/W	See section	30.5.1/618
4004_A038	Pin Control Register n (PORTB_PCR14)	32	R/W	See section	30.5.1/618
4004_A03C	Pin Control Register n (PORTB_PCR15)	32	R/W	See section	30.5.1/618
4004_A040	Pin Control Register n (PORTB_PCR16)	32	R/W	See section	30.5.1/618
4004_A044	Pin Control Register n (PORTB_PCR17)	32	R/W	See section	30.5.1/618
4004_A048	Pin Control Register n (PORTB_PCR18)	32	R/W	See section	30.5.1/618
4004_A04C	Pin Control Register n (PORTB_PCR19)	32	R/W	See section	30.5.1/618
4004_A050	Pin Control Register n (PORTB_PCR20)	32	R/W	See section	30.5.1/618
4004_A054	Pin Control Register n (PORTB_PCR21)	32	R/W	See section	30.5.1/618
4004_A058	Pin Control Register n (PORTB_PCR22)	32	R/W	See section	30.5.1/618
4004_A05C	Pin Control Register n (PORTB_PCR23)	32	R/W	See section	30.5.1/618
4004_A060	Pin Control Register n (PORTB_PCR24)	32	R/W	See section	30.5.1/618
4004_A064	Pin Control Register n (PORTB_PCR25)	32	R/W	See section	30.5.1/618
4004_A068	Pin Control Register n (PORTB_PCR26)	32	R/W	See section	30.5.1/618
4004_A06C	Pin Control Register n (PORTB_PCR27)	32	R/W	See section	30.5.1/618
4004_A070	Pin Control Register n (PORTB_PCR28)	32	R/W	See section	30.5.1/618
4004_A074	Pin Control Register n (PORTB_PCR29)	32	R/W	See section	30.5.1/618
4004_A078	Pin Control Register n (PORTB_PCR30)	32	R/W	See section	30.5.1/618
4004_A07C	Pin Control Register n (PORTB_PCR31)	32	R/W	See section	30.5.1/618
4004_A080	Global Pin Control Low Register (PORTB_GPCLR)	32	W (always reads 0)	0000_0000h	30.5.2/621
4004_A084	Global Pin Control High Register (PORTB_GPCHR)	32	W (always reads 0)	0000_0000h	30.5.3/621
4004_A0A0	Interrupt Status Flag Register (PORTB_ISFR)	32	w1c	0000_0000h	30.5.4/622

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A0C0	Digital Filter Enable Register (PORTB_DFER)	32	R/W	0000_0000h	30.5.5/622
4004_A0C4	Digital Filter Clock Register (PORTB_DFCR)	32	R/W	0000_0000h	30.5.6/623
4004_A0C8	Digital Filter Width Register (PORTB_DFWR)	32	R/W	0000_0000h	30.5.7/623
4004_B000	Pin Control Register n (PORTC_PCR0)	32	R/W	See section	30.5.1/618
4004_B004	Pin Control Register n (PORTC_PCR1)	32	R/W	See section	30.5.1/618
4004_B008	Pin Control Register n (PORTC_PCR2)	32	R/W	See section	30.5.1/618
4004_B00C	Pin Control Register n (PORTC_PCR3)	32	R/W	See section	30.5.1/618
4004_B010	Pin Control Register n (PORTC_PCR4)	32	R/W	See section	30.5.1/618
4004_B014	Pin Control Register n (PORTC_PCR5)	32	R/W	See section	30.5.1/618
4004_B018	Pin Control Register n (PORTC_PCR6)	32	R/W	See section	30.5.1/618
4004_B01C	Pin Control Register n (PORTC_PCR7)	32	R/W	See section	30.5.1/618
4004_B020	Pin Control Register n (PORTC_PCR8)	32	R/W	See section	30.5.1/618
4004_B024	Pin Control Register n (PORTC_PCR9)	32	R/W	See section	30.5.1/618
4004_B028	Pin Control Register n (PORTC_PCR10)	32	R/W	See section	30.5.1/618
4004_B02C	Pin Control Register n (PORTC_PCR11)	32	R/W	See section	30.5.1/618
4004_B030	Pin Control Register n (PORTC_PCR12)	32	R/W	See section	30.5.1/618
4004_B034	Pin Control Register n (PORTC_PCR13)	32	R/W	See section	30.5.1/618
4004_B038	Pin Control Register n (PORTC_PCR14)	32	R/W	See section	30.5.1/618
4004_B03C	Pin Control Register n (PORTC_PCR15)	32	R/W	See section	30.5.1/618
4004_B040	Pin Control Register n (PORTC_PCR16)	32	R/W	See section	30.5.1/618
4004_B044	Pin Control Register n (PORTC_PCR17)	32	R/W	See section	30.5.1/618
4004_B048	Pin Control Register n (PORTC_PCR18)	32	R/W	See section	30.5.1/618
4004_B04C	Pin Control Register n (PORTC_PCR19)	32	R/W	See section	30.5.1/618
4004_B050	Pin Control Register n (PORTC_PCR20)	32	R/W	See section	30.5.1/618
4004_B054	Pin Control Register n (PORTC_PCR21)	32	R/W	See section	30.5.1/618
4004_B058	Pin Control Register n (PORTC_PCR22)	32	R/W	See section	30.5.1/618
4004_B05C	Pin Control Register n (PORTC_PCR23)	32	R/W	See section	30.5.1/618
4004_B060	Pin Control Register n (PORTC_PCR24)	32	R/W	See section	30.5.1/618
4004_B064	Pin Control Register n (PORTC_PCR25)	32	R/W	See section	30.5.1/618
4004_B068	Pin Control Register n (PORTC_PCR26)	32	R/W	See section	30.5.1/618
4004_B06C	Pin Control Register n (PORTC_PCR27)	32	R/W	See section	30.5.1/618
4004_B070	Pin Control Register n (PORTC_PCR28)	32	R/W	See section	30.5.1/618
4004_B074	Pin Control Register n (PORTC_PCR29)	32	R/W	See section	30.5.1/618
4004_B078	Pin Control Register n (PORTC_PCR30)	32	R/W	See section	30.5.1/618
4004_B07C	Pin Control Register n (PORTC_PCR31)	32	R/W	See section	30.5.1/618
4004_B080	Global Pin Control Low Register (PORTC_GPCLR)	32	W (always reads 0)	0000_0000h	30.5.2/621

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_B084	Global Pin Control High Register (PORTC_GPCHR)	32	W (always reads 0)	0000_0000h	30.5.3/621
4004_B0A0	Interrupt Status Flag Register (PORTC_ISFR)	32	w1c	0000_0000h	30.5.4/622
4004_B0C0	Digital Filter Enable Register (PORTC_DFER)	32	R/W	0000_0000h	30.5.5/622
4004_B0C4	Digital Filter Clock Register (PORTC_DFCR)	32	R/W	0000_0000h	30.5.6/623
4004_B0C8	Digital Filter Width Register (PORTC_DFWR)	32	R/W	0000_0000h	30.5.7/623
4004_C000	Pin Control Register n (PORTD_PCR0)	32	R/W	See section	30.5.1/618
4004_C004	Pin Control Register n (PORTD_PCR1)	32	R/W	See section	30.5.1/618
4004_C008	Pin Control Register n (PORTD_PCR2)	32	R/W	See section	30.5.1/618
4004_C00C	Pin Control Register n (PORTD_PCR3)	32	R/W	See section	30.5.1/618
4004_C010	Pin Control Register n (PORTD_PCR4)	32	R/W	See section	30.5.1/618
4004_C014	Pin Control Register n (PORTD_PCR5)	32	R/W	See section	30.5.1/618
4004_C018	Pin Control Register n (PORTD_PCR6)	32	R/W	See section	30.5.1/618
4004_C01C	Pin Control Register n (PORTD_PCR7)	32	R/W	See section	30.5.1/618
4004_C020	Pin Control Register n (PORTD_PCR8)	32	R/W	See section	30.5.1/618
4004_C024	Pin Control Register n (PORTD_PCR9)	32	R/W	See section	30.5.1/618
4004_C028	Pin Control Register n (PORTD_PCR10)	32	R/W	See section	30.5.1/618
4004_C02C	Pin Control Register n (PORTD_PCR11)	32	R/W	See section	30.5.1/618
4004_C030	Pin Control Register n (PORTD_PCR12)	32	R/W	See section	30.5.1/618
4004_C034	Pin Control Register n (PORTD_PCR13)	32	R/W	See section	30.5.1/618
4004_C038	Pin Control Register n (PORTD_PCR14)	32	R/W	See section	30.5.1/618
4004_C03C	Pin Control Register n (PORTD_PCR15)	32	R/W	See section	30.5.1/618
4004_C040	Pin Control Register n (PORTD_PCR16)	32	R/W	See section	30.5.1/618
4004_C044	Pin Control Register n (PORTD_PCR17)	32	R/W	See section	30.5.1/618
4004_C048	Pin Control Register n (PORTD_PCR18)	32	R/W	See section	30.5.1/618
4004_C04C	Pin Control Register n (PORTD_PCR19)	32	R/W	See section	30.5.1/618
4004_C050	Pin Control Register n (PORTD_PCR20)	32	R/W	See section	30.5.1/618
4004_C054	Pin Control Register n (PORTD_PCR21)	32	R/W	See section	30.5.1/618
4004_C058	Pin Control Register n (PORTD_PCR22)	32	R/W	See section	30.5.1/618
4004_C05C	Pin Control Register n (PORTD_PCR23)	32	R/W	See section	30.5.1/618
4004_C060	Pin Control Register n (PORTD_PCR24)	32	R/W	See section	30.5.1/618
4004_C064	Pin Control Register n (PORTD_PCR25)	32	R/W	See section	30.5.1/618
4004_C068	Pin Control Register n (PORTD_PCR26)	32	R/W	See section	30.5.1/618
4004_C06C	Pin Control Register n (PORTD_PCR27)	32	R/W	See section	30.5.1/618
4004_C070	Pin Control Register n (PORTD_PCR28)	32	R/W	See section	30.5.1/618
4004_C074	Pin Control Register n (PORTD_PCR29)	32	R/W	See section	30.5.1/618
4004_C078	Pin Control Register n (PORTD_PCR30)	32	R/W	See section	30.5.1/618

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_C07C	Pin Control Register n (PORTD_PCR31)	32	R/W	See section	30.5.1/618
4004_C080	Global Pin Control Low Register (PORTD_GPCLR)	32	W (always reads 0)	0000_0000h	30.5.2/621
4004_C084	Global Pin Control High Register (PORTD_GPCHR)	32	W (always reads 0)	0000_0000h	30.5.3/621
4004_C0A0	Interrupt Status Flag Register (PORTD_ISFR)	32	w1c	0000_0000h	30.5.4/622
4004_C0C0	Digital Filter Enable Register (PORTD_DFER)	32	R/W	0000_0000h	30.5.5/622
4004_C0C4	Digital Filter Clock Register (PORTD_DFCR)	32	R/W	0000_0000h	30.5.6/623
4004_C0C8	Digital Filter Width Register (PORTD_DFWR)	32	R/W	0000_0000h	30.5.7/623
4004_D000	Pin Control Register n (PORTE_PCR0)	32	R/W	See section	30.5.1/618
4004_D004	Pin Control Register n (PORTE_PCR1)	32	R/W	See section	30.5.1/618
4004_D008	Pin Control Register n (PORTE_PCR2)	32	R/W	See section	30.5.1/618
4004_D00C	Pin Control Register n (PORTE_PCR3)	32	R/W	See section	30.5.1/618
4004_D010	Pin Control Register n (PORTE_PCR4)	32	R/W	See section	30.5.1/618
4004_D014	Pin Control Register n (PORTE_PCR5)	32	R/W	See section	30.5.1/618
4004_D018	Pin Control Register n (PORTE_PCR6)	32	R/W	See section	30.5.1/618
4004_D01C	Pin Control Register n (PORTE_PCR7)	32	R/W	See section	30.5.1/618
4004_D020	Pin Control Register n (PORTE_PCR8)	32	R/W	See section	30.5.1/618
4004_D024	Pin Control Register n (PORTE_PCR9)	32	R/W	See section	30.5.1/618
4004_D028	Pin Control Register n (PORTE_PCR10)	32	R/W	See section	30.5.1/618
4004_D02C	Pin Control Register n (PORTE_PCR11)	32	R/W	See section	30.5.1/618
4004_D030	Pin Control Register n (PORTE_PCR12)	32	R/W	See section	30.5.1/618
4004_D034	Pin Control Register n (PORTE_PCR13)	32	R/W	See section	30.5.1/618
4004_D038	Pin Control Register n (PORTE_PCR14)	32	R/W	See section	30.5.1/618
4004_D03C	Pin Control Register n (PORTE_PCR15)	32	R/W	See section	30.5.1/618
4004_D040	Pin Control Register n (PORTE_PCR16)	32	R/W	See section	30.5.1/618
4004_D044	Pin Control Register n (PORTE_PCR17)	32	R/W	See section	30.5.1/618
4004_D048	Pin Control Register n (PORTE_PCR18)	32	R/W	See section	30.5.1/618
4004_D04C	Pin Control Register n (PORTE_PCR19)	32	R/W	See section	30.5.1/618
4004_D050	Pin Control Register n (PORTE_PCR20)	32	R/W	See section	30.5.1/618
4004_D054	Pin Control Register n (PORTE_PCR21)	32	R/W	See section	30.5.1/618
4004_D058	Pin Control Register n (PORTE_PCR22)	32	R/W	See section	30.5.1/618
4004_D05C	Pin Control Register n (PORTE_PCR23)	32	R/W	See section	30.5.1/618
4004_D060	Pin Control Register n (PORTE_PCR24)	32	R/W	See section	30.5.1/618
4004_D064	Pin Control Register n (PORTE_PCR25)	32	R/W	See section	30.5.1/618
4004_D068	Pin Control Register n (PORTE_PCR26)	32	R/W	See section	30.5.1/618
4004_D06C	Pin Control Register n (PORTE_PCR27)	32	R/W	See section	30.5.1/618

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_D070	Pin Control Register n (PORTE_PCR28)	32	R/W	See section	30.5.1/618
4004_D074	Pin Control Register n (PORTE_PCR29)	32	R/W	See section	30.5.1/618
4004_D078	Pin Control Register n (PORTE_PCR30)	32	R/W	See section	30.5.1/618
4004_D07C	Pin Control Register n (PORTE_PCR31)	32	R/W	See section	30.5.1/618
4004_D080	Global Pin Control Low Register (PORTE_GPCLR)	32	W (always reads 0)	0000_0000h	30.5.2/621
4004_D084	Global Pin Control High Register (PORTE_GPCHR)	32	W (always reads 0)	0000_0000h	30.5.3/621
4004_D0A0	Interrupt Status Flag Register (PORTE_ISFR)	32	w1c	0000_0000h	30.5.4/622
4004_D0C0	Digital Filter Enable Register (PORTE_DFER)	32	R/W	0000_0000h	30.5.5/622
4004_D0C4	Digital Filter Clock Register (PORTE_DFCR)	32	R/W	0000_0000h	30.5.6/623
4004_D0C8	Digital Filter Width Register (PORTE_DFWR)	32	R/W	0000_0000h	30.5.7/623

30.5.1 Pin Control Register n (PORTx_PCR*n*)

NOTE

> See the GPIO Configuration section for details on the available functions for each pin.
>
> Do not modify pin configuration registers associated with pins that are not available in a reduced-pin package offering. Unbonded pins not available in a package are disabled by default to prevent them from consuming power.

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	Reserved	PFE	0	Reserved	PE	PS
W																
Reset	0	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 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.
- 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 Reserved. 0110 Reserved. 0111 Reserved. 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 Reserved. 1110 Reserved. 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). 001 Alternative 1 (GPIO). 010 Alternative 2 (chip-specific).

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	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 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 Reserved	This field is reserved.
4 PFE	Passive Filter Enable 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 Reserved	This field is reserved.
1 PE	Pull Enable 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.
0 PS	Pull Select 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.

30.5.2 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.

30.5.3 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.

30.5.4 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.

30.5.5 Digital Filter Enable Register (PORTx_DFER)

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.

PORTx_DFER field descriptions (continued)

Field	Description
	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.

30.5.6 Digital Filter Clock Register (PORTx_DFCR)

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.

30.5.7 Digital Filter Width Register (PORTx_DFWR)

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.

30.6 Functional description

30.6.1 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
- Drive strength
- Passive input filter enable
- 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^{2}C function is enabled on a pin, that does not override the pullup 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.

30.6.2 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.

30.6.3 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
- 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.

30.6.4 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 31
General-Purpose Input/Output (GPIO)

31.1 Chip-specific information for this module

31.1.1 Instantiation Information

The number of GPIO signals available on the devices covered by this document are detailed in the "Ordering information" section and the "GPIO Signal Descriptions" table of the Data Sheet.

See "Pin properties" in the Data Sheet for features of each pins.

Port control and interrupt module features are supported, each 32-pin port will support a single interrupt.

31.1.2 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 at 0x400F_F000 and at an aliased slot (15) at address 0x4000_F000.

31.2 Introduction

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.

31.2.1 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.

31.2.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 31-1. 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.

31.2.3 GPIO signal descriptions

Table 31-2. 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

Table continues on the next page...

Table 31-2. GPIO signal descriptions (continued)

GPIO signal descriptions	Description	I/O
PORTE31–PORTE0	General-purpose input/output	I/O

NOTE

Not all pins within each port are implemented on each device. See the "Signal Multiplexing" section and the "GPIO Signal Descriptions" table in DataSheet, for the number of GPIO ports available in the device.

31.2.3.1 Detailed signal description

Table 31-3. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
PORTA31–PORTA0 PORTB31–PORTB0 PORTC31–PORTC0 PORTD31–PORTD0 PORTE31–PORTE0	I/O	General-purpose input/output	
		State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.
		Timing	Assertion: 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. 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.

31.3 Memory map and register definition

The GPIO module has two address slots on AIPS-Lite peripheral bridge to keep software compatibility between product portfolios. All of the GPIO registers could be accessed either by using base address of 0x400F_F000 or 0x4000_F000. It is recommended to use 0x400F_F000 as the base address of GPIO module, and the register memory map of this chapter is also based on the base address of 0x400F_F000.

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
400F_F000	Port Data Output Register (GPIOA_PDOR)	32	R/W	0000_0000h	31.3.1/633
400F_F004	Port Set Output Register (GPIOA_PSOR)	32	W (always reads 0)	0000_0000h	31.3.2/634
400F_F008	Port Clear Output Register (GPIOA_PCOR)	32	W (always reads 0)	0000_0000h	31.3.3/634
400F_F00C	Port Toggle Output Register (GPIOA_PTOR)	32	W (always reads 0)	0000_0000h	31.3.4/635
400F_F010	Port Data Input Register (GPIOA_PDIR)	32	R	0000_0000h	31.3.5/635
400F_F014	Port Data Direction Register (GPIOA_PDDR)	32	R/W	0000_0000h	31.3.6/636
400F_F040	Port Data Output Register (GPIOB_PDOR)	32	R/W	0000_0000h	31.3.1/633
400F_F044	Port Set Output Register (GPIOB_PSOR)	32	W (always reads 0)	0000_0000h	31.3.2/634
400F_F048	Port Clear Output Register (GPIOB_PCOR)	32	W (always reads 0)	0000_0000h	31.3.3/634
400F_F04C	Port Toggle Output Register (GPIOB_PTOR)	32	W (always reads 0)	0000_0000h	31.3.4/635
400F_F050	Port Data Input Register (GPIOB_PDIR)	32	R	0000_0000h	31.3.5/635
400F_F054	Port Data Direction Register (GPIOB_PDDR)	32	R/W	0000_0000h	31.3.6/636
400F_F080	Port Data Output Register (GPIOC_PDOR)	32	R/W	0000_0000h	31.3.1/633
400F_F084	Port Set Output Register (GPIOC_PSOR)	32	W (always reads 0)	0000_0000h	31.3.2/634
400F_F088	Port Clear Output Register (GPIOC_PCOR)	32	W (always reads 0)	0000_0000h	31.3.3/634
400F_F08C	Port Toggle Output Register (GPIOC_PTOR)	32	W (always reads 0)	0000_0000h	31.3.4/635
400F_F090	Port Data Input Register (GPIOC_PDIR)	32	R	0000_0000h	31.3.5/635
400F_F094	Port Data Direction Register (GPIOC_PDDR)	32	R/W	0000_0000h	31.3.6/636
400F_F0C0	Port Data Output Register (GPIOD_PDOR)	32	R/W	0000_0000h	31.3.1/633
400F_F0C4	Port Set Output Register (GPIOD_PSOR)	32	W (always reads 0)	0000_0000h	31.3.2/634
400F_F0C8	Port Clear Output Register (GPIOD_PCOR)	32	W (always reads 0)	0000_0000h	31.3.3/634

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F0CC	Port Toggle Output Register (GPIOD_PTOR)	32	W (always reads 0)	0000_0000h	31.3.4/635
400F_F0D0	Port Data Input Register (GPIOD_PDIR)	32	R	0000_0000h	31.3.5/635
400F_F0D4	Port Data Direction Register (GPIOD_PDDR)	32	R/W	0000_0000h	31.3.6/636
400F_F100	Port Data Output Register (GPIOE_PDOR)	32	R/W	0000_0000h	31.3.1/633
400F_F104	Port Set Output Register (GPIOE_PSOR)	32	W (always reads 0)	0000_0000h	31.3.2/634
400F_F108	Port Clear Output Register (GPIOE_PCOR)	32	W (always reads 0)	0000_0000h	31.3.3/634
400F_F10C	Port Toggle Output Register (GPIOE_PTOR)	32	W (always reads 0)	0000_0000h	31.3.4/635
400F_F110	Port Data Input Register (GPIOE_PDIR)	32	R	0000_0000h	31.3.5/635
400F_F114	Port Data Direction Register (GPIOE_PDDR)	32	R/W	0000_0000h	31.3.6/636

31.3.1 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.

31.3.2 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.

31.3.3 Port Clear Output Register (GPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

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																															
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

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.

31.3.4 Port Toggle Output Register (GPIOx_PTOR)

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																															
W	PTTO																															
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_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.

31.3.5 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.

31.3.6 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. The pin will be high-Z if the port input is disabled in GPIOx_PIDR register. 1 Pin is configured as general-purpose output, for the GPIO function.

31.4 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	31.4.1/638
F800_0004	Port Set Output Register (FGPIOA_PSOR)	32	W (always reads 0)	0000_0000h	31.4.2/638
F800_0008	Port Clear Output Register (FGPIOA_PCOR)	32	W (always reads 0)	0000_0000h	31.4.3/639

Table continues on the next page...

FGPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F800_000C	Port Toggle Output Register (FGPIOA_PTOR)	32	W (always reads 0)	0000_0000h	31.4.4/639
F800_0010	Port Data Input Register (FGPIOA_PDIR)	32	R	0000_0000h	31.4.5/640
F800_0014	Port Data Direction Register (FGPIOA_PDDR)	32	R/W	0000_0000h	31.4.6/640
F800_0040	Port Data Output Register (FGPIOB_PDOR)	32	R/W	0000_0000h	31.4.1/638
F800_0044	Port Set Output Register (FGPIOB_PSOR)	32	W (always reads 0)	0000_0000h	31.4.2/638
F800_0048	Port Clear Output Register (FGPIOB_PCOR)	32	W (always reads 0)	0000_0000h	31.4.3/639
F800_004C	Port Toggle Output Register (FGPIOB_PTOR)	32	W (always reads 0)	0000_0000h	31.4.4/639
F800_0050	Port Data Input Register (FGPIOB_PDIR)	32	R	0000_0000h	31.4.5/640
F800_0054	Port Data Direction Register (FGPIOB_PDDR)	32	R/W	0000_0000h	31.4.6/640
F800_0080	Port Data Output Register (FGPIOC_PDOR)	32	R/W	0000_0000h	31.4.1/638
F800_0084	Port Set Output Register (FGPIOC_PSOR)	32	W (always reads 0)	0000_0000h	31.4.2/638
F800_0088	Port Clear Output Register (FGPIOC_PCOR)	32	W (always reads 0)	0000_0000h	31.4.3/639
F800_008C	Port Toggle Output Register (FGPIOC_PTOR)	32	W (always reads 0)	0000_0000h	31.4.4/639
F800_0090	Port Data Input Register (FGPIOC_PDIR)	32	R	0000_0000h	31.4.5/640
F800_0094	Port Data Direction Register (FGPIOC_PDDR)	32	R/W	0000_0000h	31.4.6/640
F800_00C0	Port Data Output Register (FGPIOD_PDOR)	32	R/W	0000_0000h	31.4.1/638
F800_00C4	Port Set Output Register (FGPIOD_PSOR)	32	W (always reads 0)	0000_0000h	31.4.2/638
F800_00C8	Port Clear Output Register (FGPIOD_PCOR)	32	W (always reads 0)	0000_0000h	31.4.3/639
F800_00CC	Port Toggle Output Register (FGPIOD_PTOR)	32	W (always reads 0)	0000_0000h	31.4.4/639
F800_00D0	Port Data Input Register (FGPIOD_PDIR)	32	R	0000_0000h	31.4.5/640
F800_00D4	Port Data Direction Register (FGPIOD_PDDR)	32	R/W	0000_0000h	31.4.6/640
F800_0100	Port Data Output Register (FGPIOE_PDOR)	32	R/W	0000_0000h	31.4.1/638

Table continues on the next page...

FGPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F800_0104	Port Set Output Register (FGPIOE_PSOR)	32	W (always reads 0)	0000_0000h	31.4.2/638
F800_0108	Port Clear Output Register (FGPIOE_PCOR)	32	W (always reads 0)	0000_0000h	31.4.3/639
F800_010C	Port Toggle Output Register (FGPIOE_PTOR)	32	W (always reads 0)	0000_0000h	31.4.4/639
F800_0110	Port Data Input Register (FGPIOE_PDIR)	32	R	0000_0000h	31.4.5/640
F800_0114	Port Data Direction Register (FGPIOE_PDDR)	32	R/W	0000_0000h	31.4.6/640

31.4.1 Port Data Output Register (FGPIOx_PDOR)

This register configures the logic levels that are driven on each general-purpose output pins.

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

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.

31.4.2 Port Set Output Register (FGPIOx_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

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.

31.4.3 Port Clear Output Register (FGPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

Address: Base address + 8h offset

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="32">0</td></tr>
<tr><td>W</td><td colspan="32">PTCO</td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

FGPIOx_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.

31.4.4 Port Toggle Output Register (FGPIOx_PTOR)

Address: Base address + Ch offset

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="32">0</td></tr>
<tr><td>W</td><td colspan="32">PTTO</td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

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:

FGPIOx_PTOR field descriptions (continued)

Field	Description
	0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

31.4.5 Port Data Input Register (FGPIOx_PDIR)

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

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.

31.4.6 Port Data Direction Register (FGPIOx_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

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. The pin will be high-Z if the port input is disabled in FPIOx_PIDR register. 1 Pin is configured as general-purpose output, for the GPIO function.

31.5 Functional description

31.5.1 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.

The Port Data Input registers return the synchronized pin state after any enabled digital filter in the Port Control and Interrupt module. The input pin synchronizers are shared with the Port Control and Interrupt module, so that if the corresponding Port Control and Interrupt module is disabled, then synchronizers are also disabled. This reduces power consumption when a port is not required for general-purpose input functionality.

31.5.2 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.

31.5.3 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.

Chapter 32
Analog-to-Digital Converter (ADC)

32.1 Chip-specific information for this module

32.1.1 Instantiation information

Number of ADC	1
Number of result registers per ADC	4

32.1.1.1 Number of ADC channels

Each SAR ADC supports up to 24 external analog input channels. See the following table for the exact ADC channel number present on the devices with different packages.

For details regarding a specific ADC channel available on a particular package, refer to the Pinout section in the device data sheet.

Table 32-1. ADC external channels per package

ADC Module	100LQFP	64LQFP
ADC0	24	24

32.1.1.2 ADC Connections/Channel Assignment

32.1.1.2.1 ADC0 channel assignment

The ADC0 channel assignments for the device are shown in following table. Reserved channels convert to an unknown value.

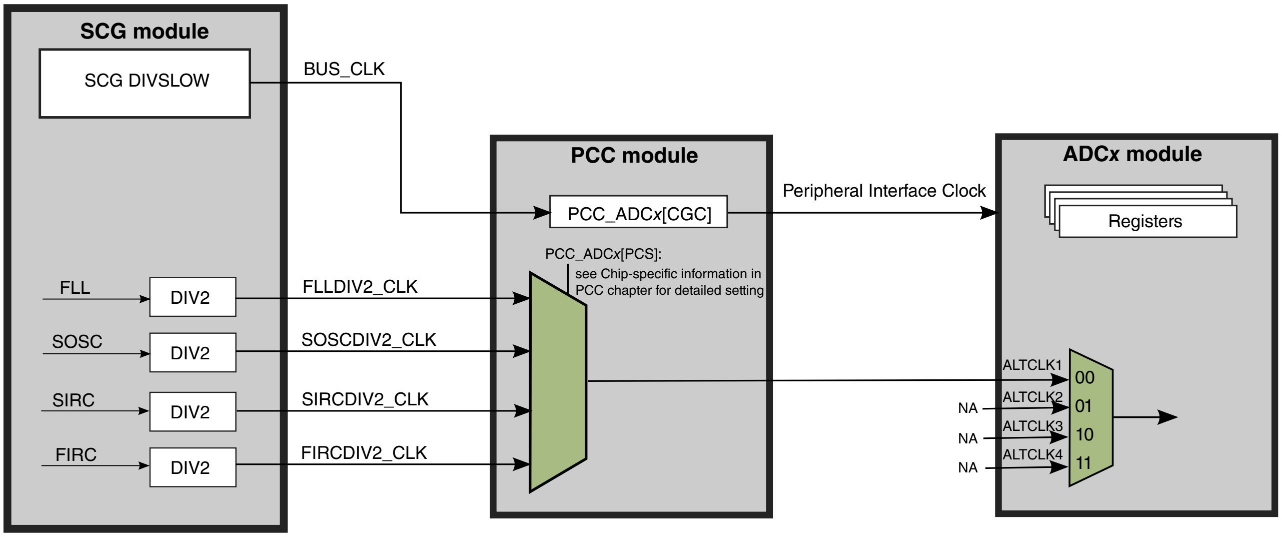
Table 32-2. ADC0 channel assignment

ADCH Value	Channel	Input
000000	AD0	PTE9/ADC0_SE0
000001	AD1	PTE8/ADC0_SE1
000010	AD2	PTD15/ADC0_SE2
000011	AD3	PTB5/ADC0_SE3
000100	AD4	PTD16/ADC0_SE4
000101	AD5	PTB4/ADC0_SE5
000110	AD6	PTE3/ADC0_SE6
000111	AD7	PTC3/ADC0_SE7
001000	AD8	PTC1/ADC0_SE8
001001	AD9	PTD5/ADC0_SE9
001010	AD10	PTC0/ADC0_SE10
001011	AD11	PTD6/ADC0_SE11
001100	AD12	PTC17/ADC0_SE12
001101	AD13	PTD7/ADC0_SE13
001110	AD14	PTC16/ADC0_SE14
001111	AD15	PTC2/ADC0_SE15
100000	AD16	PTC15/ADC0_SE16
100001	AD17	PTC14/ADC0_SE17
100010	AD18	PTB3/ADC0_SE18
100011	AD19	PTB2/ADC0_SE19
100100	AD20	PTB1/ADC0_SE20
100101	AD21	PTB0/ADC0_SE21
100110	AD22	PTC9/ADC0_SE22
010111	AD23	CMP0 8-bit DAC out
100111	AD24	PTC8/ADC0_SE24
011001	AD25	Reserved
011010	AD26	Temperature Sensor
011011	AD27	Bandgap (1V reference voltage)
011100	AD28	Reserved
011101	AD29	VREFH
011110	AD30	VREFL
011111	Module disabled	None

32.1.2 ADC Clocking Information

The following figure shows the input clock sources available for this module.

Peripheral Clocking - ADC

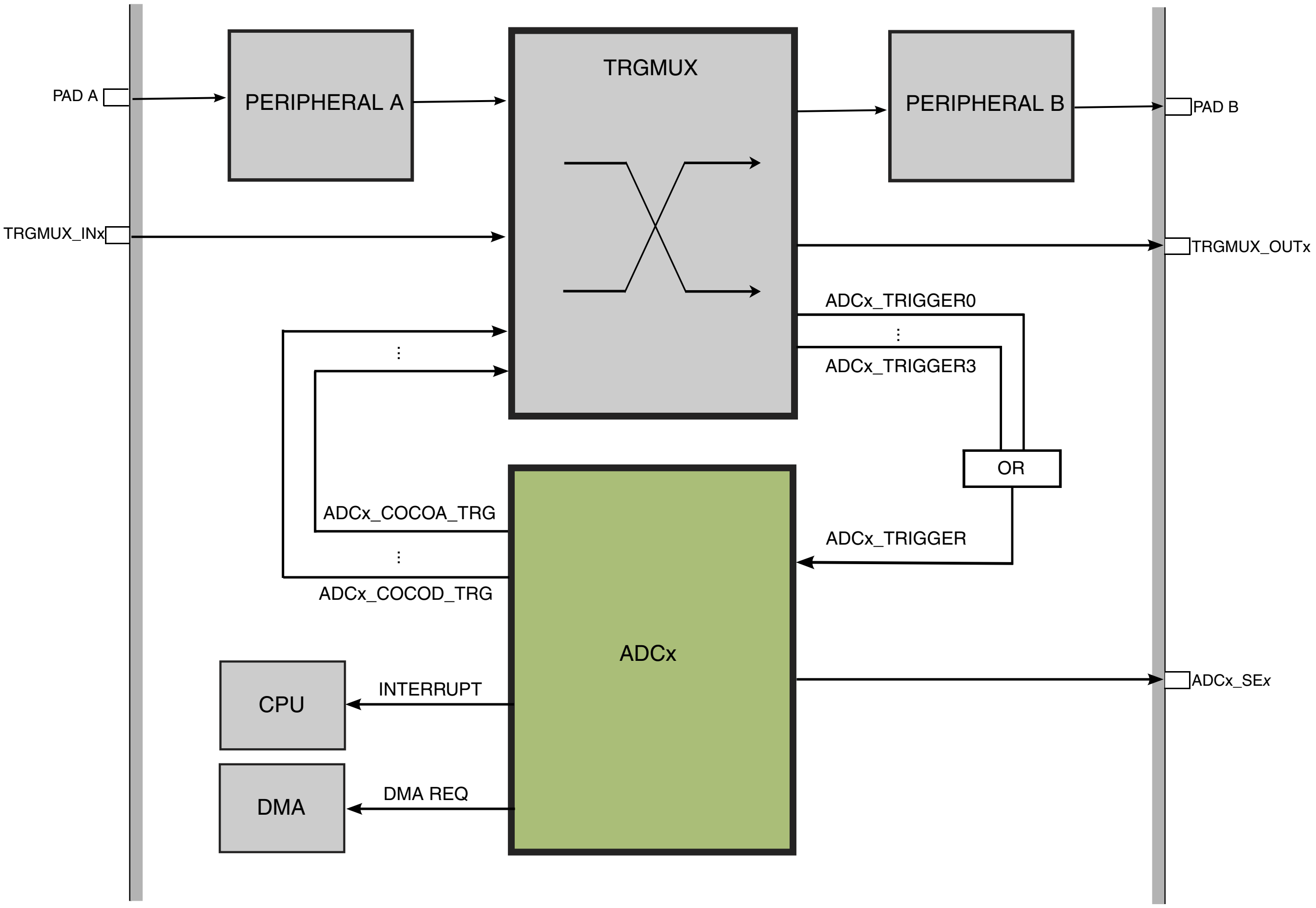


NOTE

ALTCLK2~4 are not connected on this chip.

32.1.3 Inter-connectivity Information

The ADC inter-connectivity is shown in following diagram.



32.1.4 Application-related Information

32.1.4.1 ADC Reference Options

The ADC supports the following references:

- VREFH/VREFL - connected as the primary reference option

NOTE

VREFH pin on the PCB should use 3 bypass capacitors in the range: 1 μF, 100 nF and 1 nF. Capacitors should be placed to the VREFH pin as close as possible.

- Bandgap from PMC is connected within the ADC module as ADC channel 27

ADCx_SC2[REFSEL] bit selects the voltage reference sources for ADC. Refer to REFSEL description in ADC chapter for more details.

NOTE

> V_{ALTH} is connected to the V_{DDA} pin in this device.

32.1.4.2 ADC Trigger Sources

The ADC support multiple trigger sources. There is two kinds of trigger: pre-trigger and trigger. The pre-trigger precondition the ADC block and selects the specific data result register, before the ADC trigger is asserted. The trigger initiate the ADC conversion as soon as it's asserted. The trigger and pre-trigger sources are described as following:

- Hardware pre-triggers/triggers are connected through LPIT and TRGMUX. The pre-triggers can also be controlled by software to provide flexible trigger schemes (by controlling SIM_ADCOPT[ADCxSWPRETRG] registers). Besides the hardware triggers through ADHWT, the ADC module itself also supports software trigger mode by setting SC2[ADTRG]=0. Following a write to SC1A register, a conversion is initiated.
- TRGMUX can provide triggers for each ADC, while the pre-triggers need to be controlled by software to determine relative priority. It should not trigger the ADC again before a single conversion has not completed.

The following triggers are via the TRGMUX:

- CMP out to trigger each ADC
- RTC capable to trigger each ADC
- LPTMR capable to trigger each ADC
- Software trigger capable to trigger each ADC

NOTE

> The software trigger/pre-trigger through TRGMUX, the ADC's own software trigger mode and the software pre-trigger controlled by SIM are different concepts.

TRGMUX triggering scheme:

TRGMUX supports many trigger sources, here we take LPIT as an example (typical), but the trigger source can also be others which mentioned above. LPIT supports up to 4 channels, each channel have a trigger and pre-trigger.

- Set SIM_ADCOPT[ADC*x*TRGSEL]=1. TRGMUX out is selected as ADC trigger source.
- Configure TRGMUX to select LPIT triggers as ADC trigger and pre-trigger source.

- Set SIM_ADCOPT[ADCxPRETRGSEL]=01. LPIT pre-triggers will connect directly to ADC0 ADHWTS ports to control the channels.
- ADC COCO is not required in this case. Software need to take care of the intermission time between each ADC conversion.

NOTE

For other trigger sources other than LPIT, software engagement is required to configure ADC pre-trigger selection. That means it must select pre-trigger source from software (it is required SIM_ADCOPT[ADCxPRETRGSEL] is set to 10 in this case, to make sure that software pre-triggers connect directly to ADC0 ADHWTS ports), and which ADC channel to use (by setting ADCxSWPRETRG).

Software triggering scheme:

It also supports to configure ADC pre-trigger/trigger by software.

- By setting SC2[ADTRG]=0, ADC software trigger mode is selected. A conversion is initiated following a write to SC1A register.

NOTE

ADC software trigger mode only support SC1A and data register A.

- Configure SC2[ADTRG]=1, ADC is in hardware triggering mode. By setting SIM_ADCOPT[ADCxSWPRETRG], the pre-trigger for ADC is selected. The software trigger trough TRGMUX can trigger the ADC conversion. This mechanism supports multiple data registers.

32.2 Overview

The 12-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.

32.2.1 Block diagram

The following figure is the ADC module block diagram.

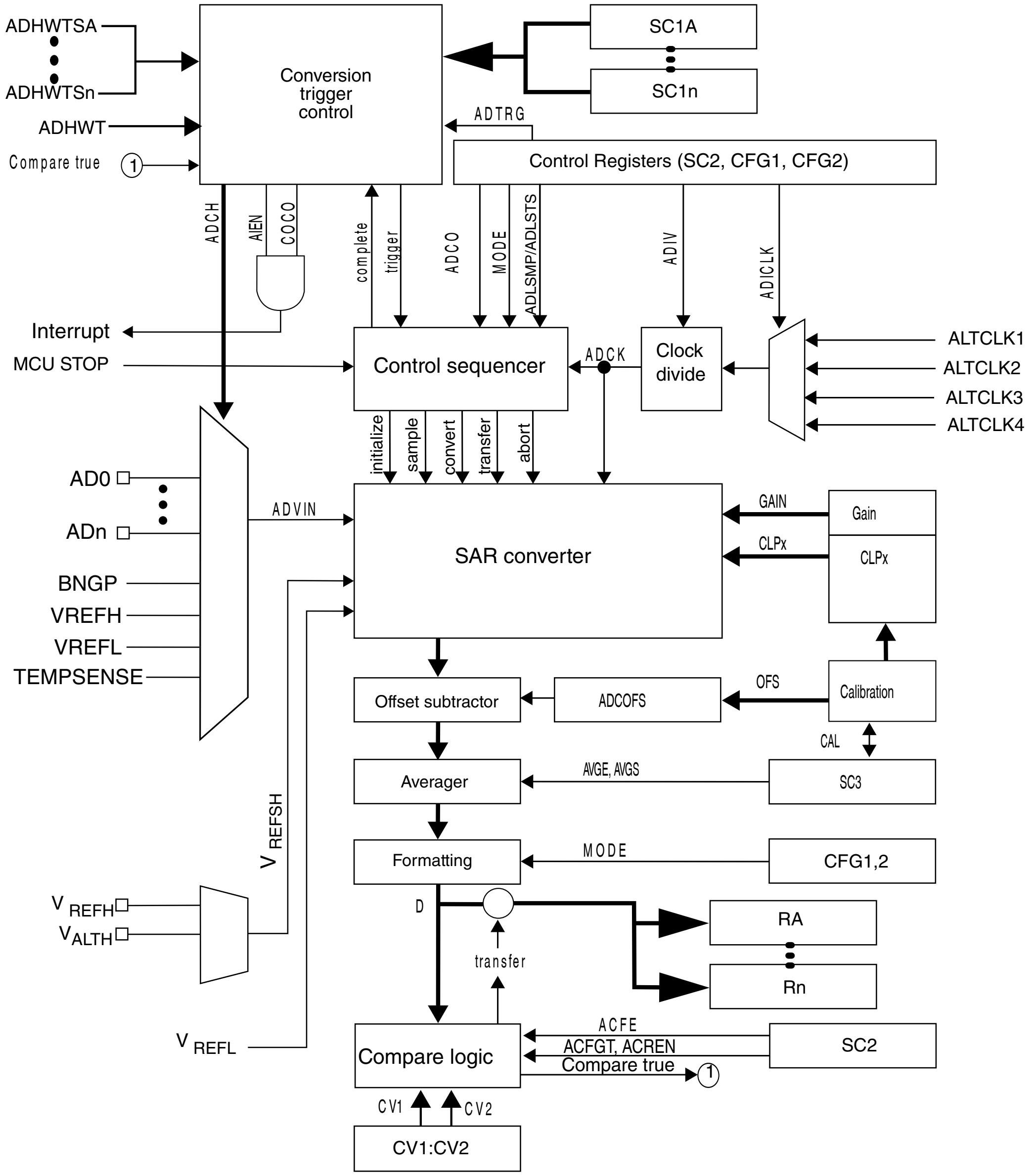


Figure 32-1. ADC block diagram

32.2.2 Features

Following are the features of the ADC module:

- Linear successive approximation algorithm with up to 12-bit resolution
- Up to 4 single-ended external analog inputs
- Single-ended 12-bit, 10-bit, and 8-bit output modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion modes
- 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
- 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

32.3 Functional description

The ADC module is disabled during reset, or when SC1*n*[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 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 SC1*n*[COCO] is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, or when SC1*n*[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 chip.

32.3.1 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 Alternate Clock sources 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 SC1*n*[COCO] and generates an ADC interrupt to wake the MCU from Wait mode if the respective ADC interrupt is enabled, that is, when SC1*n*[AIEN]=1. If the hardware averaging function is enabled, SC1*n*[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1*n*[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.

32.3.2 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.

32.3.2.1 Normal Stop mode with Alternate clock sources enabled

If Alternate clock source selected for the conversion clock is enabled, 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 SC1*n*[COCO] and generates an ADC interrupt to wake the MCU from Normal Stop mode if the respective ADC interrupt is enabled, that is, when SC1*n*[AIEN]=1. The result register, R*n*, will contain the data from the first completed conversion that occurred during Normal Stop mode. If the hardware averaging function is enabled, SC1*n*[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1*n*[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.

32.3.3 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}). These voltage references are selected using SC2[REFSEL]. The alternate V_{ALTH} voltage reference may select additional external pin or internal source depending on MCU configuration. See the chip configuration information for the voltage references specific to this MCU.

32.3.4 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, ADHWTS*n*, has occurred. This source is not available on all MCUs. See the chip-specific ADC information for information on the ADHWT source and the ADHWTS*n* 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, ADHWTS*n*, has occurred. If a conversion is in progress when a rising edge of a trigger occurs, the rising edge is ignored. In continuous conversionn 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, ADHWTS*n*, 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 an 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.
- ADHWTS*n* active selects SC1*n*.

When the conversion is completed, the result is placed in the R*n* registers associated with the ADHWTS*n* received. For example:

- ADHWTSA active selects RA register
- ADHWTS*n* active selects R*n* register

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

32.3.5 Conversion control

Conversion mode is selected by configuring 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

32.3.5.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A, 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, ADHWTS*n*, has occurred. The channel and status fields that are selected depend on the active trigger select signal:

- ADHWTSA active selects SC1A.
- ADHWTS*n* active selects SC1*n*.
- if neither is active, the off condition is selected

Note

Selecting more than one ADHWTS*n* prior to a conversion completion will cause unpredictable results. To avoid this, select only one ADHWTS*n* 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.

32.3.5.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, R*n*, as indicated in the following table.

Table 32-3. Indication of conversion completion

Compare functions	Hardware averaging	Conversion status	Is SC1*n*[COCO] set to 1, and is the conversion result transferred into the data result registers?
Disabled	Disabled	Not completed	No
Disabled	Disabled	Completed	Yes
Disabled	Enabled	Not completed	No
Disabled	Enabled	Completed	Yes, if the last of the selected number of conversions is completed
Enabled	Disabled	Not completed	No
Enabled	Disabled	Completed	Yes, if the compare condition is true

Table continues on the next page...

Table 32-3. Indication of conversion completion (continued)

Compare functions	Hardware averaging	Conversion status	Is SC1*n*[COCO] set to 1, and is the conversion result transferred into the data result registers?
Enabled	Enabled	Not completed	No
Enabled	Enabled	Completed	Yes, if [(the last of the selected number of conversions is completed) AND (the compare condition is true)]

An interrupt is generated if the respective SC1*n*[AIEN] is high at the time that the respective SC1*n*[COCO] is set.

32.3.5.3 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-SC1*n* registers while that specific SC1B-SC1*n* register is actively controlling a conversion aborts the current conversion. The SC1B-SC1*n* registers are not used for software trigger operation and therefore writes to the SC1B-SC1*n* registers do not initiate a new conversion.
- A write to any ADC register besides the SC1A-SC1*n* registers occurs. This indicates that a change in mode of operation has occurred and the current conversion is therefore invalid.
- The MCU is reset.

When a conversion is aborted, the contents of the data registers, R*n*, 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, RA and R*n* return to their reset states.

32.3.5.4 Power control

The ADC module remains in its Idle state until a conversion is initiated. The Idle state implies that ADC conversion routine is held in reset.

32.3.5.5 Sample time and total conversion time

The total conversion time depends upon:

- The sample time as determined by CFG2[SMPLTS]
- The MCU bus frequency
- The conversion mode, as determined by CFG1[MODE]
- The frequency of the conversion clock, that is, f_{ADCK}.

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

1. CFG2[SMPLTS] selects 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.

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]. To calculate total conversion time the following formula is applied:

ADC TOTAL CONVERSION TIME = Sample Phase Time (set by SMPLTS + 1) + Hold Phase (1 ADC Cycle) + Compare Phase Time (8-bit Mode = 20 ADC Cycles, 10-bit Mode = 24 ADC Cycles, 12-bit Mode = 28 ADC Cycles) + Single or First continuous time adder (5 ADC cycles + 5 bus clock cycles)

32.3.5.6 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 after the selected number of conversions have been completed. When hardware averaging is selected, the completion of a single conversion will not set SC1*n*[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, R*n*, and SC1*n*[COCO] is set. An ADC interrupt is generated upon the setting of SC1*n*[COCO] if the respective ADC interrupt is enabled, that is, SC1*n*[AIEN] = 1.

Note

> The hardware average function can perform conversions on a channel while the MCU is in Wait or Normal Stop mode. The ADC interrupt wakes the MCU when the hardware average is completed if SC1*n*[AIEN] is set.

32.3.6 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 32-4. Compare modes

SC2[ACFGT]	SC2[ACREN]	CV1 relative to CV2	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, SC1*n*[COCO] is set.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, SC1*n*[COCO] is not set and the conversion result data will not be transferred to the result register, R*n*. 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 SC1*n*[COCO] is set and the respective ADC interrupt is enabled, that is, SC1*n*[AIEN] = 1.

Note

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

32.3.7 Calibration function

The ADC is equipped with a calibration mechanism to provide high accuracy as specified in the data sheet.

NOTE

It is mandatory to calibrate the ADC after power up or reset. Not doing this can result in ADC conversion results with lower than specified accuracy.

In order to calibrate the ADC correctly, the following has to be done:

- On startup, wait until the reference voltage (VREFH) has stabilized.
- ADC has to be recalibrated after each system reset.
- Calibrate only one ADC instance at a time. So, when calibrating instance ADC0, the instances ADC1, ADC2, and so on, are required to be idle.
- You must set ADCK (ADC clock) to a value less than or equal to half of the maximum specified frequency.
- Before starting calibration, the calibration registers (CLPS, CLP3, CLP2, CLP1, CLP0, CLPX, and CLP9) must be cleared by writing 0000_0000h into each of them.
- Start ADC calibration by writing SC3[CAL] = 1, SC3[AVGE] = 1, and SC3[AVGS] = 11b.
- Wait for calibration to finish. This will be indicated by SE3[CAL] set to 0 only after conversion complete flag (SC1*n*[COCO] = 1).
- Now you can run ADC conversions with high accuracy in your application. Please make sure to reconfigure the ADCK clock speed and reconfigure AVGE and AVGS to your desired settings. (Maximum clock speed and no use of hardware averaging is possible.)

The total calibration conversion time is: 12 × (# of AVERAGE × [Sample time (sample + 1) + 1 cycle for hold + 34 cycles for compare phase]) + 1st conversion synchronization (~5 ADC cycles + 5 module clocks).

For high accuracy of the ADC (as specified in data sheet) on your application board (PCB), the following requirements should be met:

- Bypass caps between VREFH and VREFL. Suggested cap sizes: 1 nF, 100 nF, 10 μF.
- Place caps on PCB as close as possible to the device pins VREFH and VREFL.
- Bypass caps between VDDA and VSSA. Suggested cap sizes: 1 nF, 100 nF, 10 μF.
- Place caps on PCB as close as possible to the device pins VDDA and VSSA.
- Routing of VDDA, VSSA, VREFH, and VREFL on PCB:
    - Low impedance between the bypass caps and the MCU pins.
    - Keep routing distant from noisy signal routes like switching I/Os.

32.3.8 User-defined offset function

OFS is a two's-complement, left-justified register that contains the calibration-generated offset error correction value.

The value in OFS is subtracted from the conversion and the result is transferred into the result registers, R*n*. 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 OFS is different from the data result register, R*n*, 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.

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

Note

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

You can use the offset calibration function to remove application offsets or DC bias values. USR_OFS may be written with a number in two'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 two'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 0000h.

32.3.9 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 by configuring CFG1[ADICLK]. ALTCLK*x*, as defined for this MCU. See the chip configuration information. Conversions are possible using ALTCLK*x* as the input clock source while the MCU is in Normal Stop mode. ALTCLK1 is the default selection following reset.

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 as in the 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 divided by 1, 2, 4, or 8. The ADC bus clock frequency must be greater than or equal to the ADC ALT clock frequency. Please refer to the device *Data Sheet* for ADC specifications.

32.4 ADC signal descriptions

Each ADC module supports up to 4 single-ended inputs.

The ADC also requires four supply/reference/ground connections.

NOTE

> For the number of channels supported on this device, see the chip-specific ADC information.

The ADC does not produce any output signals.

Table 32-5. ADC input signal descriptions

Signal	Description
AD*n*	Single-Ended Analog Channel Inputs
V_{REFSH}	Voltage Reference Select High
V_{REFSL}	Voltage Reference Select Low

Table continues on the next page...

Table 32-5. ADC input signal descriptions (continued)

Signal	Description
V_{DDA}	Analog Power Supply
V_{SSA}	Analog Ground

32.4.1 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.

32.4.2 Analog Ground (V_{SSA})

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

32.4.3 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 the voltage reference pairs for V_{REFSH} and V_{REFSL} by configuring V_{REFSH} as V_{REFH} or V_{ALTH}. 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}) alternate (V_{ALTLH} and V_{REFL}). These voltage references are selected by configuring SC2[REFSEL]. The alternate voltage reference, V_{ALTH} may select additional external pin or internal source depending on MCU configuration. See the chip configuration information on the Voltage References specific to this MCU.

In some packages, V_{REFH} is internally connected 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 V_{REFH} and the V_{DDA} potential. V_{REFH} must never exceed V_{DDA}. Connect the ground references to the same voltage potential as V_{SSA}.

32.4.4 Analog Channel Inputs (ADx)

The ADC module supports up to 4 analog inputs. An analog input is selected for conversion through the SC1[ADCH] channel select field.

32.5 ADC register descriptions

This section describes the ADC registers. All ADC registers support 8-bit, 16-bit, and 32-bit reads, but only 32-bit writes are supported. Executing an 8-bit or a 16-bit write will result in a transfer error.

32.5.1 ADC memory map

ADC0 base address: 4003_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Ch	ADC Status and Control Register 1 (SC1A - SC1D)	32	RW	0000_003Fh
40h	ADC Configuration Register 1 (CFG1)	32	RW	0000_0000h
44h	ADC Configuration Register 2 (CFG2)	32	RW	0000_000Ch
48h - 54h	ADC Data Result Registers (RA - RD)	32	R	0000_0000h
88h - 8Ch	Compare Value Registers (CV1 - CV2)	32	RW	0000_0000h
90h	Status and Control Register 2 (SC2)	32	RW	0000_0000h
94h	Status and Control Register 3 (SC3)	32	RW	0000_0000h
98h	BASE Offset Register (BASE_OFS)	32	RW	0000_0040h
9Ch	ADC Offset Correction Register (OFS)	32	RW	Table 32-6
A0h	USER Offset Correction Register (USR_OFS)	32	RW	0000_0000h
A4h	ADC X Offset Correction Register (XOFS)	32	RW	0000_0030h
A8h	ADC Y Offset Correction Register (YOFS)	32	RW	0000_0037h
ACh	ADC Gain Register (GAIN)	32	RW	Table 32-6
B0h	ADC User Gain Register (UG)	32	RW	0000_0004h
B4h	ADC General Calibration Value Register S (CLPS)	32	RW	Table 32-6
B8h	ADC Plus-Side General Calibration Value Register 3 (CLP3)	32	RW	Table 32-6
BCh	ADC Plus-Side General Calibration Value Register 2 (CLP2)	32	RW	Table 32-6
C0h	ADC Plus-Side General Calibration Value Register 1 (CLP1)	32	RW	Table 32-6
C4h	ADC Plus-Side General Calibration Value Register 0 (CLP0)	32	RW	Table 32-6
C8h	ADC Plus-Side General Calibration Value Register X (CLPX)	32	RW	Table 32-6
CCh	ADC Plus-Side General Calibration Value Register 9 (CLP9)	32	RW	Table 32-6

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
D0h	ADC General Calibration Offset Value Register S (CLPS_OFS)	32	RW	0000_0000h
D4h	ADC Plus-Side General Calibration Offset Value Register 3 (CLP3_OFS)	32	RW	0000_0000h
D8h	ADC Plus-Side General Calibration Offset Value Register 2 (CLP2_OFS)	32	RW	0000_0000h
DCh	ADC Plus-Side General Calibration Offset Value Register 1 (CLP1_OFS)	32	RW	0000_0000h
E0h	ADC Plus-Side General Calibration Offset Value Register 0 (CLP0_OFS)	32	RW	0000_0000h
E4h	ADC Plus-Side General Calibration Offset Value Register X (CLPX_OFS)	32	RW	0000_0440h
E8h	ADC Plus-Side General Calibration Offset Value Register 9 (CLP9_OFS)	32	RW	0000_0240h

32.5.2 ADC Status and Control Register 1 (SC1A - SC1D)

32.5.2.1 Offset

Register	Offset
SC1A	0h
SC1B	4h
SC1C	8h
SC1D	Ch

32.5.2.2 Function

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

At any one point in time, only one of the SC1*n* registers is actively controlling ADC sequential conversions. Updating SC1A while SC1*n* is actively controlling a conversion is allowed, and vice versa for any of the SC1*n* registers specific to this MCU.

Writing to the SC1A register while SC1A is actively controlling a conversion aborts the current conversion. In Software Trigger mode (when SC2[ADTRG]=0), writes to SC1A initiate a new conversion. This is valid for all values of SC1A[ADCH] other than all 1s (module disabled).

Writing any of the SC1*n* registers while that specific SC1*n* register is actively controlling a conversion aborts the current conversion. None of the SC1B-SC1*n* registers are used for software trigger operation and therefore writes to the SC1B-SC1*n* registers do not initiate a new conversion.

32.5.2.3 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								COCO	AIEN	ADCH					
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

32.5.2.4 Fields

Field	Function
31-8 —	Reserved
7 COCO	Conversion Complete Flag This is a read-only field that is set each time a conversion is completed when one or more of the following is true: • The compare function is disabled • SC2[ACFE]=0 and the hardware average function is disabled • SC3[AVGE]=0 If the compare result is true, then COCO is set upon completion of a conversion if one or more of the following is true: • The compare function is enabled • SC2[ACFE]=1 COCO is set upon completion of the selected number of conversions (determined by AVGS) if one or more of the following is true: • The hardware average function is enabled • SC3[AVGE]=1 COCO in SC1A is also set at the completion of a calibration sequence. COCO is cleared when one of the following is true: • The respective SC1n register is written • The respective Rn register is read

Table continues on the next page...

Field	Function
	0b - Conversion is not complete. 1b - Conversion is complete.
6 AIEN	Interrupt Enable Enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted. 0b - Conversion complete interrupt is disabled. 1b - Conversion complete interrupt is enabled.
5-0 ADCH	Input channel select Selects one of the input channels. **NOTE:** Some of the input channel options in the bitfield-setting descriptions might not be available for your chip. For the actual ADC channel assignments for your device, see the chip-specific information. The successive approximation converter subsystem is turned off when the channel bits are all set (i.e. ADCH set to all 1s). 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. 00_0000b - External channel 0 is selected as input. 00_0001b - External channel 1 is selected as input. 00_0010b - External channel 2 is selected as input. 00_0011b - External channel 3 is selected as input. 00_0100b - External channel 4 is selected as input. 00_0101b - External channel 5 is selected as input. 00_0110b - External channel 6 is selected as input. 00_0111b - External channel 7 is selected as input. 00_1000b - External channel 8 is selected as input. 00_1001b - External channel 9 is selected as input. 00_1010b - External channel 10 is selected as input. 00_1011b - External channel 11 is selected as input. 00_1100b - External channel 12 is selected as input. 00_1101b - External channel 13 is selected as input. 00_1110b - External channel 14 is selected as input. 00_1111b - External channel 15 is selected as input. 01_0000b - Reserved 01_0001b - Reserved 01_0010b - Reserved 01_0011b - Reserved 01_0100b - Reserved 01_0101b - Internal channel 0 is selected as input. 01_0110b - Internal channel 1 is selected as input. 01_0111b - Internal channel 2 is selected as input. 01_1000b - Reserved 01_1001b - Reserved 01_1010b - Temp Sensor 01_1011b - Band Gap 01_1100b - Internal channel 3 is selected as input. 01_1101b - V_{REFSH} is selected as input. Voltage reference selected is determined by SC2[REFSEL]. 01_1110b - V_{REFSL} is selected as input. Voltage reference selected is determined by SC2[REFSEL]. 01_1111b - Reserved 10_0000b - External channel 16 is selected as input. 10_0001b - External channel 17 is selected as input.

Field	Function
	10_0010b - External channel 18 is selected as input. 10_0011b - External channel 19 is selected as input. 10_0100b - External channel 20 is selected as input. 10_0101b - External channel 21 is selected as input. 10_0110b - External channel 22 is selected as input. 10_0111b - External channel 23 is selected as input. 10_1000b - External channel 24 is selected as input. 10_1001b - External channel 25 is selected as input. 10_1010b - External channel 26 is selected as input. 10_1011b - External channel 27 is selected as input. 10_1100b - External channel 28 is selected as input. 10_1101b - External channel 29 is selected as input. 10_1110b - External channel 30 is selected as input. 10_1111b - External channel 31 is selected as input. 11_0000b - Reserved 11_0001b - Reserved 11_0010b - Reserved 11_0011b - Reserved 11_0100b - Reserved 11_0101b - Reserved 11_0110b - Reserved 11_0111b - Reserved 11_1000b - Reserved 11_1001b - Reserved 11_1010b - Reserved 11_1011b - Reserved 11_1100b - Reserved 11_1101b - Reserved 11_1110b - Reserved 11_1111b - Reserved

32.5.3 ADC Configuration Register 1 (CFG1)

32.5.3.1 Offset

Register	Offset
CFG1	40h

32.5.3.2 Function

Configuration Register 1 (CFG1) selects the mode of operation, clock source, clock divide.

32.5.3.3 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							0	0	ADIV		0	MODE		ADICLK	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.3.4 Fields

Field	Function
31-9 —	Reserved
8 —	Reserved
7 —	Reserved
6-5 ADIV	Clock Divide Select Selects the divide ratio used by the ADC to generate the internal clock ADCK. 00b - The divide ratio is 1 and the clock rate is input clock. 01b - The divide ratio is 2 and the clock rate is (input clock)/2. 10b - The divide ratio is 4 and the clock rate is (input clock)/4. 11b - The divide ratio is 8 and the clock rate is (input clock)/8.
4 —	Reserved
3-2 MODE	Conversion mode selection Selects the ADC resolution. 00b - 8-bit conversion. 01b - 12-bit conversion. 10b - 10-bit conversion. 11b - Reserved
1-0 ADICLK	Input Clock Select Selects the input clock source to generate the internal clock, ADCK. See the clock distribution/clocking chapter of your device for details on which alternate clocks are supported. 00b - Alternate clock 1 (ALTCLK1) 01b - Alternate clock 2 (ALTCLK2) 10b - Alternate clock 3 (ALTCLK3) 11b - Alternate clock 4 (ALTCLK4)

32.5.4 ADC Configuration Register 2 (CFG2)

32.5.4.1 Offset

Register	Offset
CFG2	44h

32.5.4.2 Function

Configuration Register 2 (CFG2) selects the long sample time duration during long sample mode.

NOTE

Writing 0 is not supported on this register.

32.5.4.3 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								SMPLTS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

32.5.4.4 Fields

Field	Function
31-8 —	Reserved
7-0 SMPLTS	Sample Time Select

Field	Function
	Selects a sample time of 2 to 256 ADCK clock cycles. The value written to this register field is the desired sample time minus 1. A sample time of 1 is not supported. Allows higher impedance inputs to be accurately sampled or conversion speed to be maximized 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.

32.5.5 ADC Data Result Registers (RA - RD)

32.5.5.1 Offset

Register	Offset
RA	48h
RB	4Ch
RC	50h
RD	54h

32.5.5.2 Function

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.

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

Table 32-6. Data result register description

Conversion mode	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
12-bit single-ended	D												Unsigned right-justified
10-bit single-ended	0		D										
8-bit single-ended	0				D								

D: Data. The data result registers are read-only; writing to these registers generates a transfer error.

32.5.5.3 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				D											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.5.4 Fields

Field	Function
31-12 —	Reserved
11-0 D	Data result

32.5.6 Compare Value Registers (CV1 - CV2)

32.5.6.1 Offset

Register	Offset
CV1	88h
CV2	8Ch

32.5.6.2 Function

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. Therefore, the compare function uses only the CVn fields that are related to the ADC mode of operation. CV2 is used only when the compare range function is enabled, that is, SC2[ACREN]=1.

32.5.6.3 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 W	CV															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.6.4 Fields

Field	Function
31-16 —	Reserved
15-0 CV	Compare Value.

32.5.7 Status and Control Register 2 (SC2)

32.5.7.1 Offset

Register	Offset
SC2	90h

32.5.7.2 Function

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.

32.5.7.3 Diagram

Bits	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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			0					ADACT	ADTRG	ACFE	ACFGT	ACREN	DMAEN	REFSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.7.4 Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-13 —	Reserved
12-8 —	Reserved
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. 0b - Conversion not in progress. 1b - Conversion in progress.
6 ADTRG	Conversion Trigger Select Selects the type of trigger used for initiating a conversion. Two types of triggers can be selected: • 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. 0b - Software trigger selected. 1b - Hardware trigger selected.
5 ACFE	Compare Function Enable Enables the compare function. 0b - Compare function disabled.

Table continues on the next page...

Field	Function
	1b - Compare function enabled.
4 ACFGT	Compare Function Greater Than Enable Configures the compare function to check the conversion result relative to CV1 and CV2 based upon the value of ACREN. ACFE must be set for ACFGT to have any effect. See Table 32-4 "Compare modes" for further details.
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. See Table 32-4 "Compare modes" for further details.
2 DMAEN	DMA Enable 0b - DMA is disabled. 1b - DMA is enabled and will assert the ADC DMA request during an ADC conversion complete event , which is indicated when any SC1n[COCO] flag is asserted.
1-0 REFSEL	Voltage Reference Selection Selects the voltage reference source used for conversions. 00b - Default voltage reference pin pair, that is, external pins V_{REFH} and V_{REFL} 01b - Alternate reference voltage, that is, V_{ALTH}. This voltage may be additional external pin or internal source depending on the MCU configuration. See the chip configuration information for details specific to this MCU. 10b - Reserved 11b - Reserved

32.5.8 Status and Control Register 3 (SC3)

32.5.8.1 Offset

Register	Offset
SC3	94h

32.5.8.2 Function

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

32.5.8.3 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								CAL		0		ADCO	AVGE	AVGS	
W										0						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.8.4 Fields

Field	Function
31-8 —	Reserved
7 CAL	Calibration When CAL=1, the ADC begins the calibration sequence. This field stays set while the calibration is in progress and is cleared when the calibration sequence is completed. After it is started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid. Setting CAL will abort any current conversion. **NOTE:** For calibration, it is mandatory to use averaging and average number 32. **NOTE:** If several ADCs are on a device, they should be calibrated sequentially. No parallel calibrations of ADCs are allowed because they will disturb each other.
6 —	Reserved
5-4 —	Reserved
3 ADCO	Continuous Conversion Enable Enables continuous conversions. 0b - One conversion will be performed (or one set of conversions, if AVGE is set) after a conversion is initiated. 1b - Continuous conversions will be performed (or continuous sets of conversions, if AVGE is set) after a conversion is initiated.
2 AVGE	Hardware Average Enable Enables the hardware average function of the ADC. 0b - Hardware average function disabled. 1b - Hardware average function enabled.
1-0 AVGS	Hardware Average Select Determines how many ADC conversions will be averaged to create the ADC average result.

Field	Function
	00b - 4 samples averaged. 01b - 8 samples averaged. 10b - 16 samples averaged. 11b - 32 samples averaged.

32.5.9 BASE Offset Register (BASE_OFS)

32.5.9.1 Offset

Register	Offset
BASE_OFS	98h

32.5.9.2 Function

The BASE Offset Register (BASE_OFS) contains the offset value used by the calibration algorithm to determine the Offset Calibration Value (OFS).

32.5.9.3 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								BA_OFS							
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

32.5.9.4 Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Field	Function
—	
7-0 BA_OFS	Base Offset Error Correction Value

32.5.10 ADC Offset Correction Register (OFS)

32.5.10.1 Offset

Register	Offset
OFS	9Ch

32.5.10.2 Function

The ADC Offset Correction Register (OFS) contains the calibration-generated offset error correction value (OFS). The value in BA_OFS is used in the calibration algorithm to calculate the offset correction value that gets stored in the OFS register. 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.

NOTE

If offset register is set to a negative value and it is lower than or equal to 0xFFF8, the ADC will not result code 0. If offset register is set to a negative value and it is lower than or equal to 0xFFF0, the ADC will not result code 1.

32.5.10.3 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 W	OFS															
Reset	u[1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

[1] Reset values are loaded out of IFR.

32.5.10.4 Fields

Field	Function
31-16 —	Reserved
15-0 OFS	Offset Error Correction Value

32.5.11 USER Offset Correction Register (USR_OFS)

32.5.11.1 Offset

Register	Offset
USR_OFS	A0h

32.5.11.2 Function

The ADC USER Offset Correction Register (USR_OFS) contains the user defined offset error correction value used in the conversion result error correction algorithm.

32.5.11.3 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								USR_OFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.11.4 Fields

Field	Function
31-8 —	Reserved
7-0 USR_OFS	USER Offset Error Correction Value

32.5.12 ADC X Offset Correction Register (XOFS)

32.5.12.1 Offset

Register	Offset
XOFS	A4h

32.5.12.2 Function

The ADC X Offset Correction Register (XOFS) contains the X offset used in the conversion result error correction algorithm.

32.5.12.3 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										XOFS					
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

32.5.12.4 Fields

Field	Function
31-6 —	Reserved
5-0 XOFS	X offset error correction value

32.5.13 ADC Y Offset Correction Register (YOFS)

32.5.13.1 Offset

Register	Offset
YOFS	A8h

32.5.13.2 Function

The ADC Y Offset Correction Register (YOFS) contains the Y offset used in the conversion result error correction algorithm.

32.5.13.3 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								YOFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1

32.5.13.4 Fields

Field	Function
31-8 —	Reserved
7-0 YOFS	Y offset error correction value

32.5.14 ADC Gain Register (GAIN)

32.5.14.1 Offset

Register	Offset
GAIN	ACh

32.5.14.2 Function

The Gain Register (GAIN) contains the gain error correction for the overall conversion. GAIN, a 11-bit real number in binary format, is the gain adjustment factor. This register value is determined and uploaded by the calibration algorithm.

32.5.14.3 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					GAIN										
W																
Reset	0	0	0	0	0	u[1]	u	u	u	u	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

32.5.14.4 Fields

Field	Function
31-11 —	Reserved
10-0 GAIN	GAIN Gain error adjustment factor for the overall conversion

32.5.15 ADC User Gain Register (UG)

32.5.15.1 Offset

Register	Offset
UG	B0h

32.5.15.2 Function

The User Gain Register (UG) contains the user gain error correction. It allows you to adjust the final calibration gain value. This register must be written before calibrating the ADC.

32.5.15.3 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						UG									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

32.5.15.4 Fields

Field	Function
31-10 —	Reserved
9-0 UG	UG User gain error correction value

32.5.16 ADC General Calibration Value Register S (CLPS)

32.5.16.1 Offset

Register	Offset
CLPS	B4h

32.5.16.2 Function

The General Calibration Value Registers (CLPx) contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths. If these registers are written by the user after calibration, the linearity error specifications may not be met.

32.5.16.3 Diagram

<table>
<tr><td>Bits</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td colspan="16">0</td></tr>
<tr><td>W</td><td colspan="16"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bits</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="9">0</td><td colspan="7" rowspan="2">CLPS</td></tr>
<tr><td>W</td><td colspan="9"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>u[1]</td><td>u</td><td>u</td><td>u</td><td>u</td><td>u</td><td>u</td></tr>
</table>

[1] Reset values are loaded out of IFR.

32.5.16.4 Fields

Field	Function
31-7 —	Reserved
6-0 CLPS	CLPS Calibration Value

32.5.17 ADC Plus-Side General Calibration Value Register 3 (CLP3)

32.5.17.1 Offset

Register	Offset
CLP3	B8h

32.5.17.2 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						CLP3									
W																
Reset	0	0	0	0	0	0	u[1]	u	u	u	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

32.5.17.3 Fields

Field	Function
31-10 —	Reserved
9-0 CLP3	CLP3 Calibration Value

32.5.18 ADC Plus-Side General Calibration Value Register 2 (CLP2)

32.5.18.1 Offset

Register	Offset
CLP2	BCh

32.5.18.2 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						CLP2									
W																
Reset	0	0	0	0	0	0	u[1]	u	u	u	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

32.5.18.3 Fields

Field	Function
31-10 —	Reserved
9-0 CLP2	CLP2 Calibration Value

32.5.19 ADC Plus-Side General Calibration Value Register 1 (CLP1)

32.5.19.1 Offset

Register	Offset
CLP1	C0h

32.5.19.2 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							CLP1								
W																
Reset	0	0	0	0	0	0	0	u[1]	u	u	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

32.5.19.3 Fields

Field	Function
31-9 —	Reserved
8-0 CLP1	CLP1 Calibration Value

32.5.20 ADC Plus-Side General Calibration Value Register 0 (CLP0)

32.5.20.1 Offset

Register	Offset
CLP0	C4h

32.5.20.2 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								CLP0							
W																
Reset	0	0	0	0	0	0	0	0	u[1]	u	u	u	u	u	u	u

[1] Reset values are loaded out of IFR.

32.5.20.3 Fields

Field	Function
31-8 —	Reserved
7-0 CLP0	CLP0 Calibration Value

32.5.21 ADC Plus-Side General Calibration Value Register X (CLPX)

32.5.21.1 Offset

Register	Offset
CLPX	C8h

32.5.21.2 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								Reserved	CLPX						
W																
Reset	0	0	0	0	0	0	0	0	0	u[1]	u	u	u	u	u	u

[1] Reset values are loaded out of IFR.

32.5.21.3 Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6-0 CLPX	CLPX Calibration Value (signed 2's complement)

32.5.22 ADC Plus-Side General Calibration Value Register 9 (CLP9)

32.5.22.1 Offset

Register	Offset
CLP9	CCh

32.5.22.2 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								Reserved	CLP9						
W																
Reset	0	0	0	0	0	0	0	0	0	u[1]	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

32.5.22.3 Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6-0 CLP9	CLP9 Calibration Value (signed 2's complement)

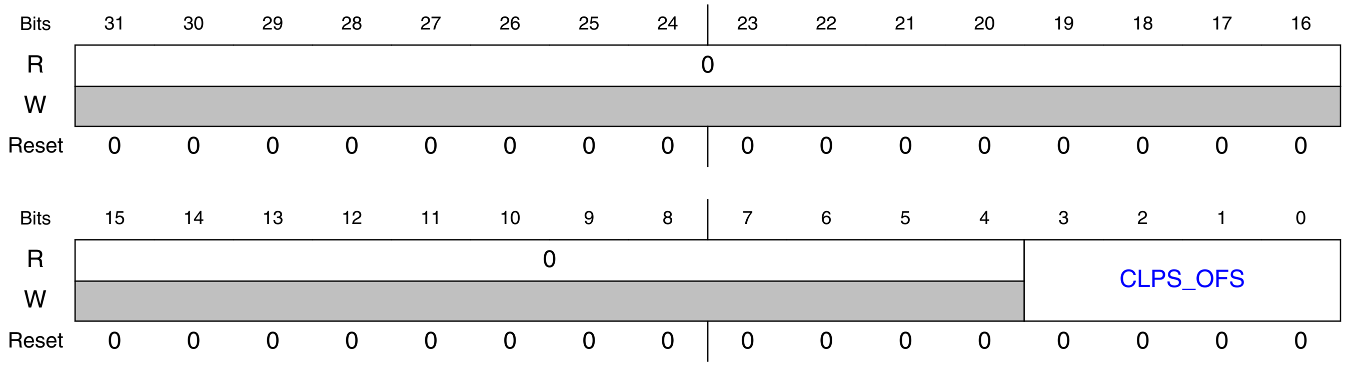
32.5.23 ADC General Calibration Offset Value Register S (CLPS_OFS)

32.5.23.1 Offset

Register	Offset
CLPS_OFS	D0h

32.5.23.2 Function

32.5.23.3 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												CLPS_OFS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.23.4 Fields

Field	Function
31-4 —	Reserved
3-0 CLPS_OFS	CLPS Offset Capacitor offset correction value

32.5.24 ADC Plus-Side General Calibration Offset Value Register 3 (CLP3_OFS)

32.5.24.1 Offset

Register	Offset
CLP3_OFS	D4h

32.5.24.2 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												CLP3_OFS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.24.3 Fields

Field	Function
31-4 —	Reserved
3-0 CLP3_OFS	CLP3 Offset Capacitor offset correction value

32.5.25 ADC Plus-Side General Calibration Offset Value Register 2 (CLP2_OFS)

32.5.25.1 Offset

Register	Offset
CLP2_OFS	D8h

32.5.25.2 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												CLP2_OFS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.25.3 Fields

Field	Function
31-4 —	Reserved
3-0 CLP2_OFS	CLP2 Offset Capacitor offset correction value

32.5.26 ADC Plus-Side General Calibration Offset Value Register 1 (CLP1_OFS)

32.5.26.1 Offset

Register	Offset
CLP1_OFS	DCh

32.5.26.2 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												CLP1_OFS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.26.3 Fields

Field	Function
31-4 —	Reserved
3-0 CLP1_OFS	CLP1 Offset Capacitor offset correction value

32.5.27 ADC Plus-Side General Calibration Offset Value Register 0 (CLP0_OFS)

32.5.27.1 Offset

Register	Offset
CLP0_OFS	E0h

32.5.27.2 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												CLP0_OFS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.5.27.3 Fields

Field	Function
31-4 —	Reserved
3-0 CLP0_OFS	CLP0 Offset Capacitor offset correction value

32.5.28 ADC Plus-Side General Calibration Offset Value Register X (CLPX_OFS)

32.5.28.1 Offset

Register	Offset
CLPX_OFS	E4h

32.5.28.2 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				CLPX_OFS											
W																
Reset	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0

32.5.28.3 Fields

Field	Function
31-12 —	Reserved
11-0 CLPX_OFS	CLPX Offset Capacitor offset correction value

32.5.29 ADC Plus-Side General Calibration Offset Value Register 9 (CLP9_OFS)

32.5.29.1 Offset

Register	Offset
CLP9_OFS	E8h

32.5.29.2 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				CLP9_OFS											
W																
Reset	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0

32.5.29.3 Fields

Field	Function
31-12 —	Reserved
11-0 CLP9_OFS	CLP9 Offset Capacitor offset correction value

32.6 Usage Guide

32.6.1 ADC module initialization sequence

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

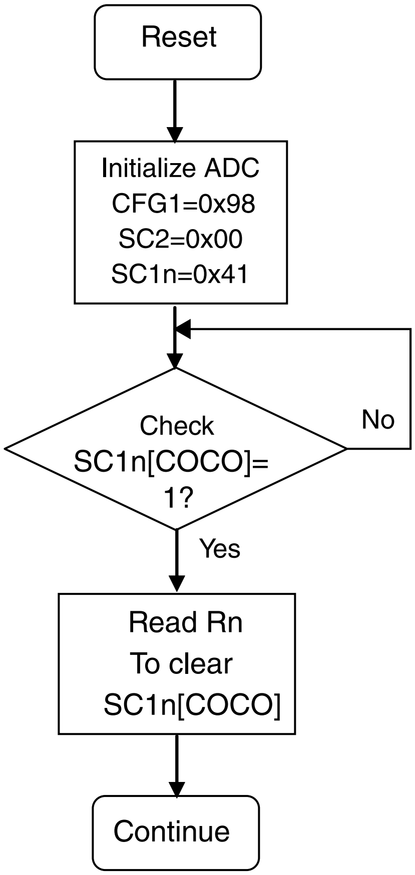
1. Calibrate the ADC by following the calibration instructions in Calibration function.
2. Update CFG to select the input clock source and the divide ratio used to generate ADCK.
3. Update SC2 to select the conversion trigger, hardware or software, and compare function options, if enabled.
4. Update SC3 to select whether conversions will be continuous or completed only once (ADCO) and whether to perform hardware averaging.
5. Update SC1:SC1*n* registers to enable or disable conversion complete interrupts.

Also, select the input channel which can be used to perform conversions.

32.6.2 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.

```
ADC_CFG1 = ADC_CFG1_ADLPC_MASK |
ADC_CFG1_ADLSMP_MASK | ADC_CFG1_MODE(0x10);
// 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.
// Bit 1:0 ADICLK 00 Selects the bus clock.
ADC_SC2 = 0x00;
// 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 ).
ADC_SC1A = ADC_SC1_AIEN_MASK | ADC_SC1_ADCH(0x1);
// Bit 7 COCO 0 Read-only flag which is set when a conversion completes.
// Bit 6 AIEN 1 Conversion complete interrupt enabled.
// Bit 4:0 ADCH 00001 Input channel 1 selected as ADC input channel.
ADC_RA = 0xxx
// Holds results of conversion.
ADC_CV = 0xxx
// Holds compare value when compare function enabled.
```



32.6.3 Calibration

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. Not doing this can result in ADC conversion results with lower than specified accuracy.

In order to calibrate ADC correctly the following steps have to be done:

- On startup, wait until reference voltage (VREFH/VREFL) has stabilized, use 3 bypass capacitance in the range: 1 µF, 100 nF and 1 nF.
- Calibrate only one ADC instance at a time, no parallel calibration of ADCs because they will disturb each other.
- Set ADCK (ADC clock) to half the maximum specified frequency, e.g. 25 MHz.
- Start ADC calibration by writing ADC_SC3 register with: CAL=1, AVGE=1, AVGS=11.

- Wait for calibration to finish. This will be indicated by conversion complete flag (COCO in ADC_SC1A).
- Run ADC conversions with high accuracy in your application. Make sure to re-configure ADCK clock speed and to re-configure AVGE and AVGS to the desired settings.

For more detailed information about calibration guidelines, refer to the application note AN5314: ADC Calibration on Kinetis E+ Microcontrollers.

NOTE

In the OFS, CLPX and CLP9 registers, the calibration values are signed numbers (in 2's complement format).

32.6.4 Application hints

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, refer to the application note AN5250: How to Increase the Analog-to-Digital Converter Accuracy in an Application.

32.6.5 DMA Support on ADC

Applications may require continuous sampling of the ADC (4K samples/sec) that may have considerable load on the CPU. Though using PDB to trigger ADC may reduce some CPU load, the ADC supports DMA request functionality for higher performance when the ADC is sampled at a very high rate or cases where PDB is bypassed. The ADC can trigger the DMA (via DMA req) on conversion completion.

For most cases, the DMA request can be directly triggered from ADC conversion completion. The device also support another way to trigger DMA via TRGMUX module. The TRGMUX will provide user a more flexible DMA triggering scheme using software based on different application requirements, for example, the DMA can be triggered after multiple ADC conversion completion instead of every ADC conversion completion.

32.6.6 ADC low-power modes

The ADC will be available in STOP, VLPR, VLPW, and VLPS mode.

NOTE

When in VLPx mode, the ADC clock source is only limited to OSC and SIRC.

32.6.7 ADC self-test and calibration scheme

ADC calibration needs to be initiated by setting the ADCx_SC3[CAL] bit.

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. Not doing this can result in ADC conversion results with lower than specified accuracy. Calibration needs to be initiated manually by setting the CAL bit. For more details, please refer to "Calibration" section.

Chapter 33
Comparator (CMP)

33.1 Chip-specific information for this module

33.1.1 Instantiation information

Number of CMP	1
8-bit DAC sub-block	Each CMP has its own independent 8-bit DAC.
Analog inputs	Each CMP supports up to 6 analog inputs from external pins.
Internal reference	Each CMP is able to convert an internal reference from the bandgap (1 V reference voltage).
Round-robin mode	Each CMP supports the round-robin sampling scheme.[1]

[1] In summary, this allows the CMP to operate independently in STOP and VLPS mode, whilst being triggered periodically to sample up to 6 inputs. Only if an input changes state, a full wake-up is generated.

33.1.1.1 CMP input connections

The following table shows the input connections to the CMP.

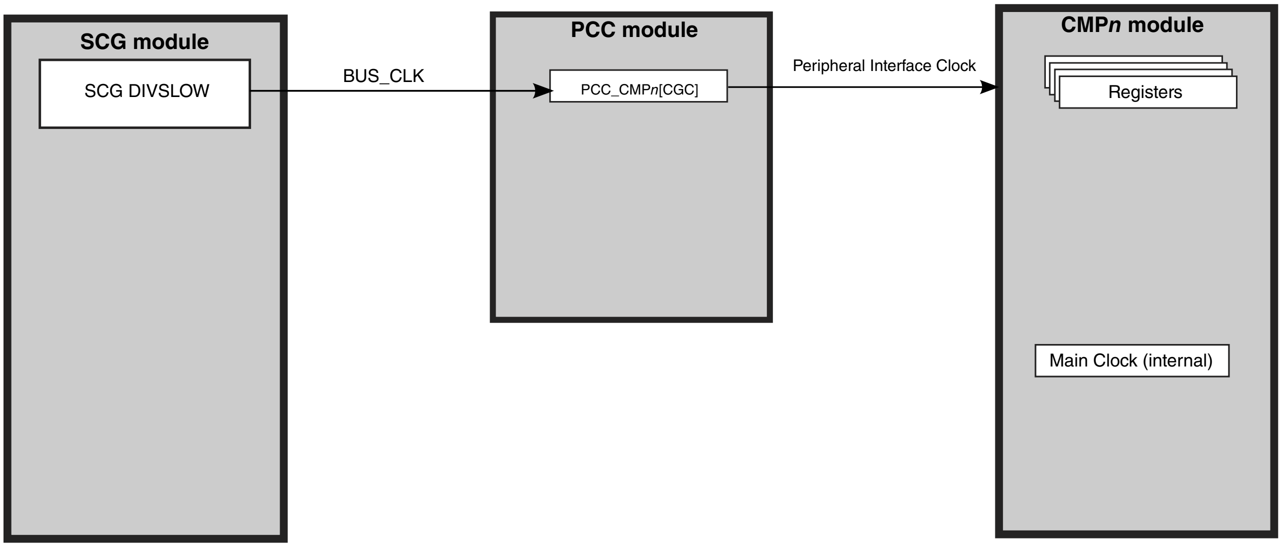
Table 33-1. CMP input connections

CMP Inputs	CMP0
IN0	ACMP0_IN0
IN1	ACMP0_IN1
IN2	ACMP0_IN2
IN3	ACMP0_IN3
IN4	ACMP0_IN4
IN5	ACMP0_IN5
IN6	Reserved
IN7	Reserved

33.1.2 CMP Clocking Information

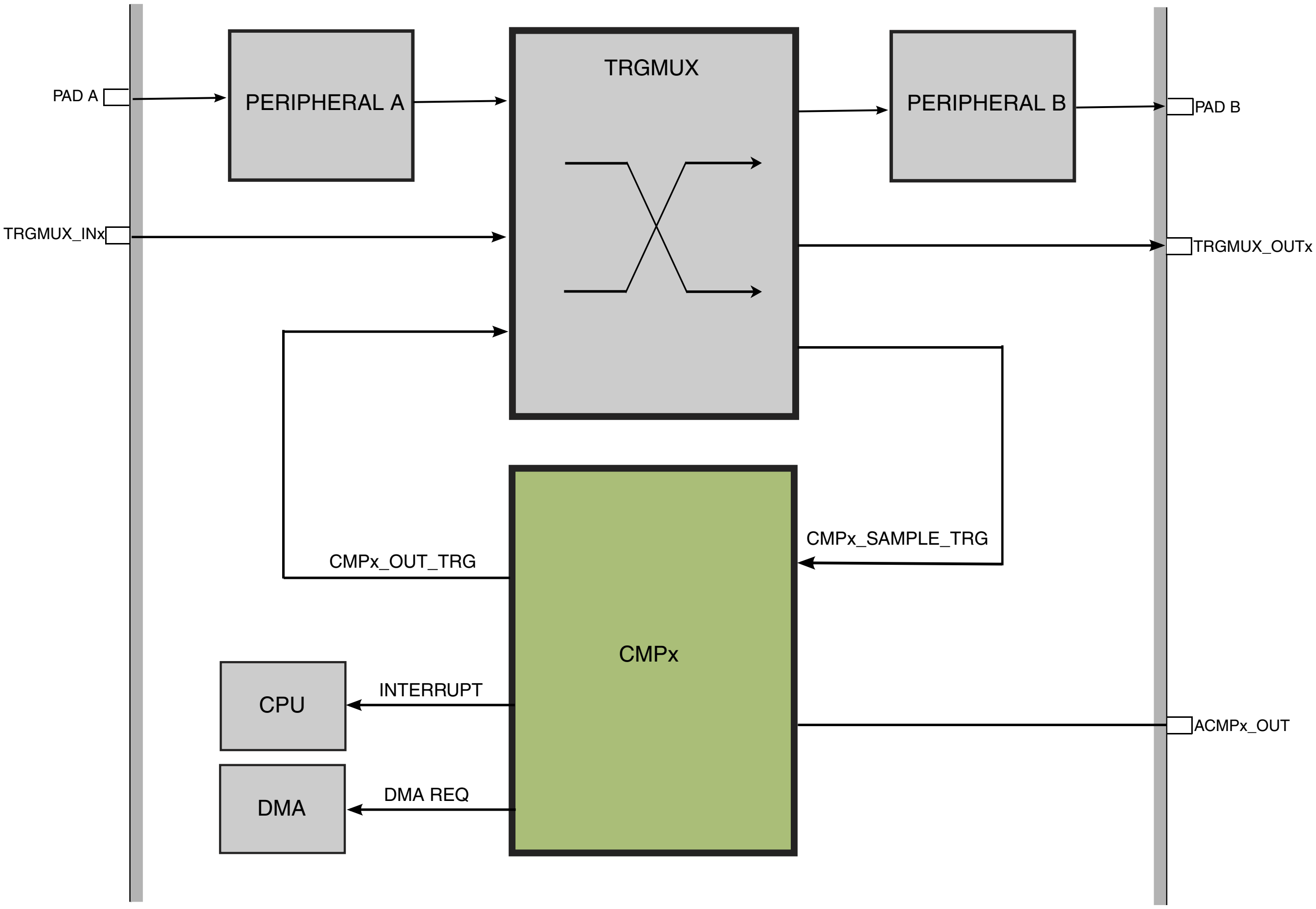
The CMP clocking input is as below.

Peripheral Clocking - CMP



33.1.3 Inter-connectivity Information

The CMP inter-connectivity is shown in following diagram.



33.1.4 Application-related Information

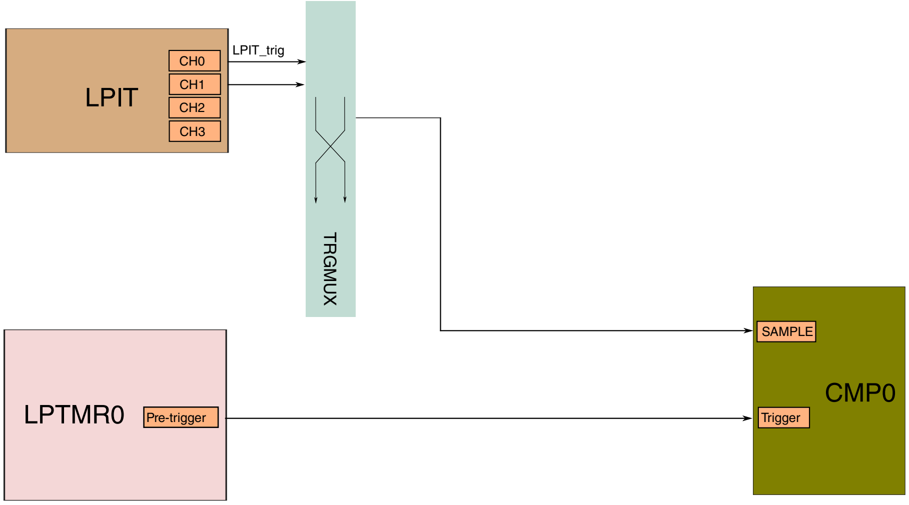
33.1.4.1 CMP external references

The CMP could get external reference through the tightly integrated 8-bit DAC sub-block. The 8-bit DAC sub-block supports selection of two references. For this device, the references are connected as follows:

- VDDA -- connected to V_{in1} of CMP
- PMC bandgap buffer out (1V reference voltage) -- connected to V_{in2} of CMP

33.1.4.2 External window/sample input

LPIT could be used to generate pulse output which can be used as sampling windows of CMP block via TRGMUX.



33.1.4.3 CMP trigger mode

The CMP and 8-bit DAC sub-block supports trigger mode operation when the chip is in STOP or VLPS mode. When trigger mode is enabled, the trigger source will provide a low power clock and the triggers to the CMP. The trigger event will initiate a compare sequence that must first enable the CMP and DAC prior to performing a CMP operation and capturing the output.

In this device, control for this two-staged sequencing is provided from, for example, LPTMR. The LPTMR provides a single trigger output to all implemented comparators. Through configuration of the CMPx_C2[RRE] bits the trigger can be used to trigger a single comparator or multiple comparators concurrently. The LPTMR only offers single wire trigger to CMP. And the configuration must be done by LPTMR itself (round robin) before entering low power mode.

33.2 Overview

The 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 8-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. The 256-tap resistor ladder network divides the supply reference V_{in} into 256 voltage levels. A 8-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/256$. V_{in} can be selected from two voltage sources, V_{in1} and V_{in2}. The DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

33.2.1 Block diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

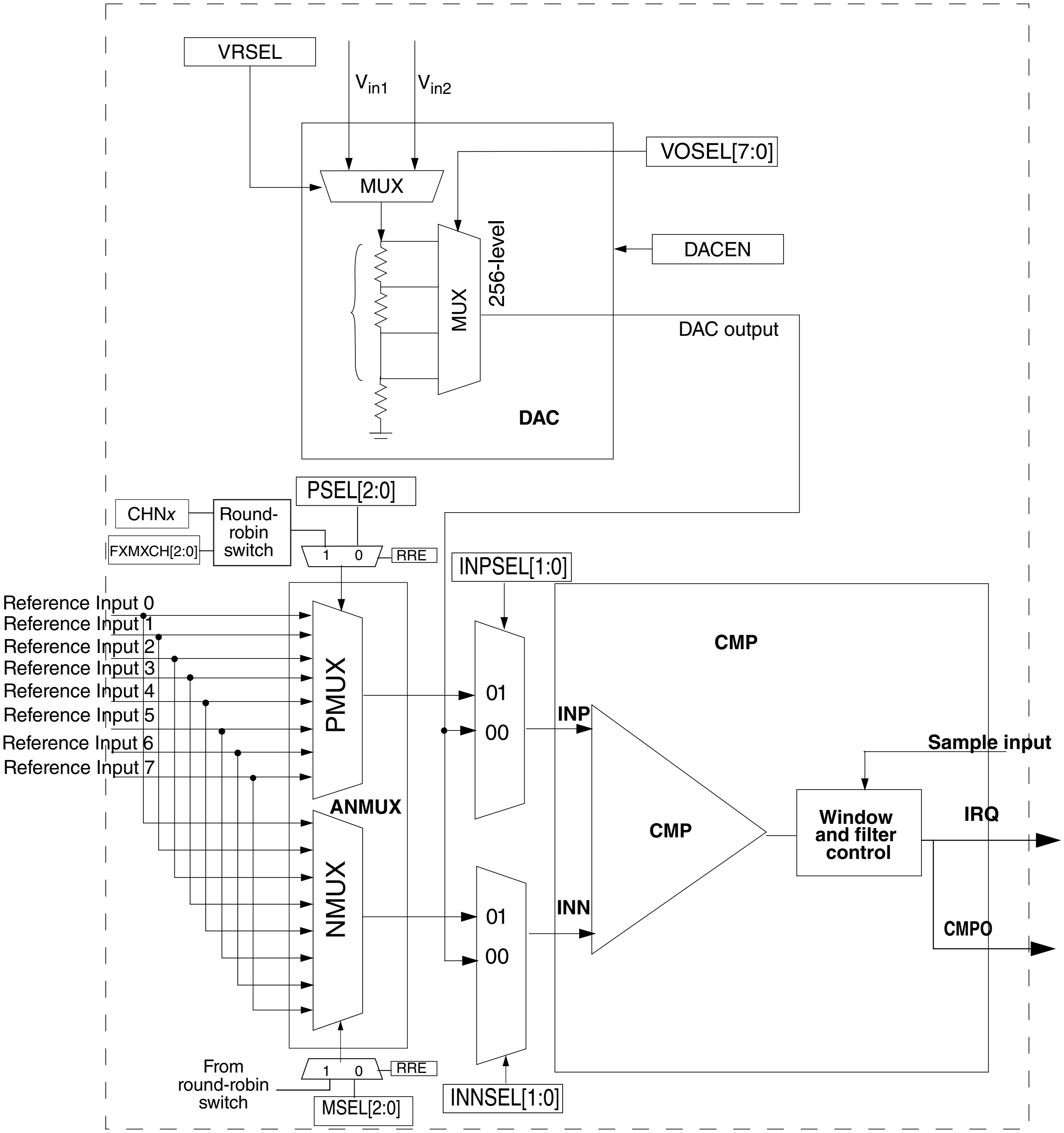


Figure 33-1. CMP high level diagram

33.2.2 Features

The following subsections list the features of the CMP, the DAC, and the ANMUX.

33.2.2.1 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 power modes available on this MCU
- The window and filter functions are not available in STOP modes
- The comparator can be triggered by other peripherals to work for only a small fraction of the time

33.2.2.2 ANMUX features

The ANMUX has the following features:

- Two 8-to-1 channel MUXes
- Operational over the entire supply range

33.2.2.3 DAC features

The DAC has the following features:

- 8-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

33.3 Functional description

33.3.1 CMP

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 C0[INVT] = 1.

C0[IER] and C0[IEF] are used to select the condition that causes the CMP module to assert an interrupt to the processor. C0[CFF] is set on a falling edge, and C0[CFR] is set on a rising edge of the comparator output. The optionally filtered CMPO can be read directly through C0[COUT].

33.3.1.1 CMP block diagram

The following figure shows the block diagram for the CMP module.

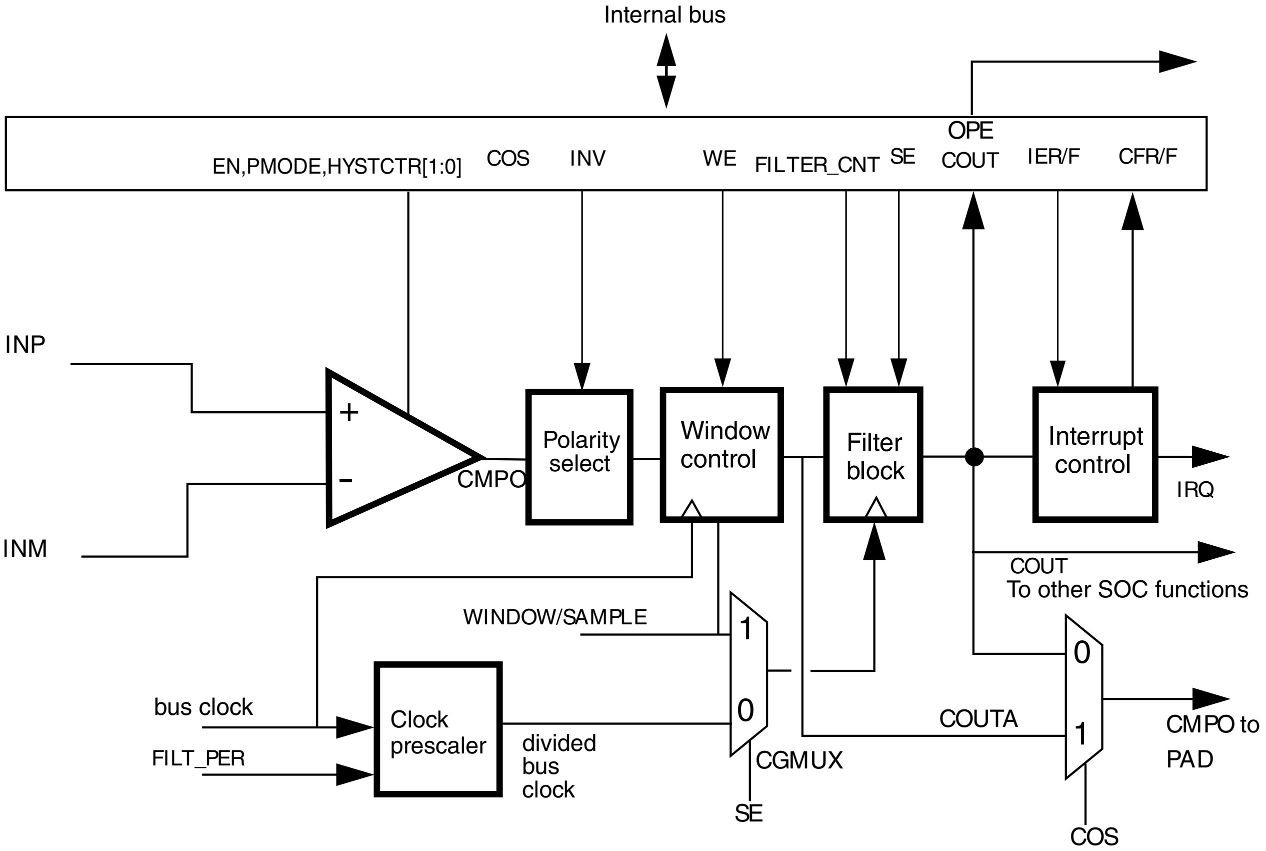


Figure 33-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when C0[WE] = 0.
- If C0[WE] = 1, the comparator output is 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.

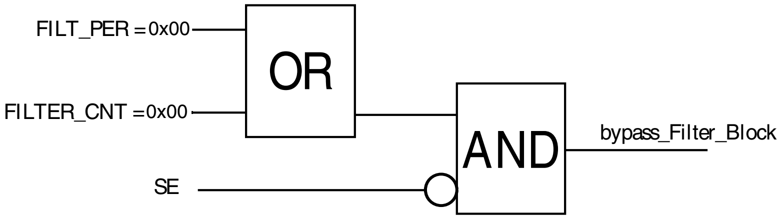


Figure 33-3. Filter block bypass logic

- The Filter block acts as a simple sampler if the filter is bypassed and C0[FILTER_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and C0[FILTER_CNT] is set greater than 0x01.

  - If C0[SE] = 1, the external SAMPLE input is used as the sampling clock.
  - IF C0[SE] = 0, the divided bus clock is used as the 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 crosses clock domain boundaries, must be resynchronized to the bus clock.
- C0[WE] and C0[SE] are mutually exclusive.
- If enabled, the filter clock and the sample period must be at least 4 times slower than the system clock to the comparator.

33.3.1.2 CMP functional modes

There are three main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, C0[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 C0[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 C0[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 33-2. Comparator sample/filter controls

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation
1	0	X	X	X	X	**Disabled** See the Disabled mode (# 1).

Table continues on the next page...

Table 33-2. Comparator sample/filter controls (continued)

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	C0[FPR]	Operation
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	> 0x04	
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 C0[FPR] to generate COUT. See the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01–0xFF	**Windowed/Filtered mode** 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 C0[EN], C0[WE], C0[SE], C0[FILTER_CNT], and C0[FPR] 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 C0[SE]=0, C0[FPR] =0 and C0[FILTER_CNT]=0x00. This resets the filter to a known state.

33.3.1.2.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

33.3.1.2.2 Continuous mode (#s 2A & 2B)

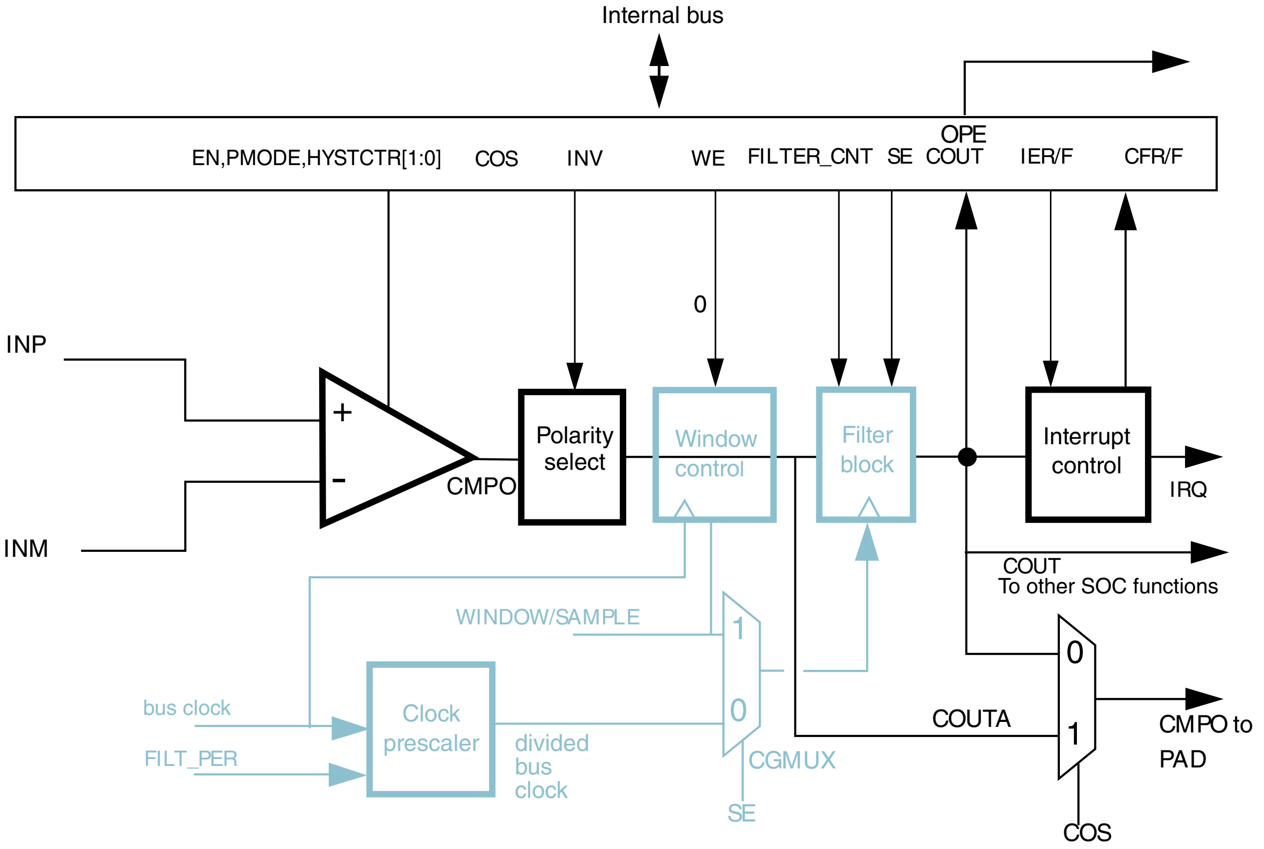


Figure 33-4. Comparator operation in Continuous mode

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 (as the grey-colored parts in the figure). C0[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 that result in disabling the filter block, see Figure 33-3.

33.3.1.2.3 Sampled, Non-Filtered mode (#s 3A & 3B)

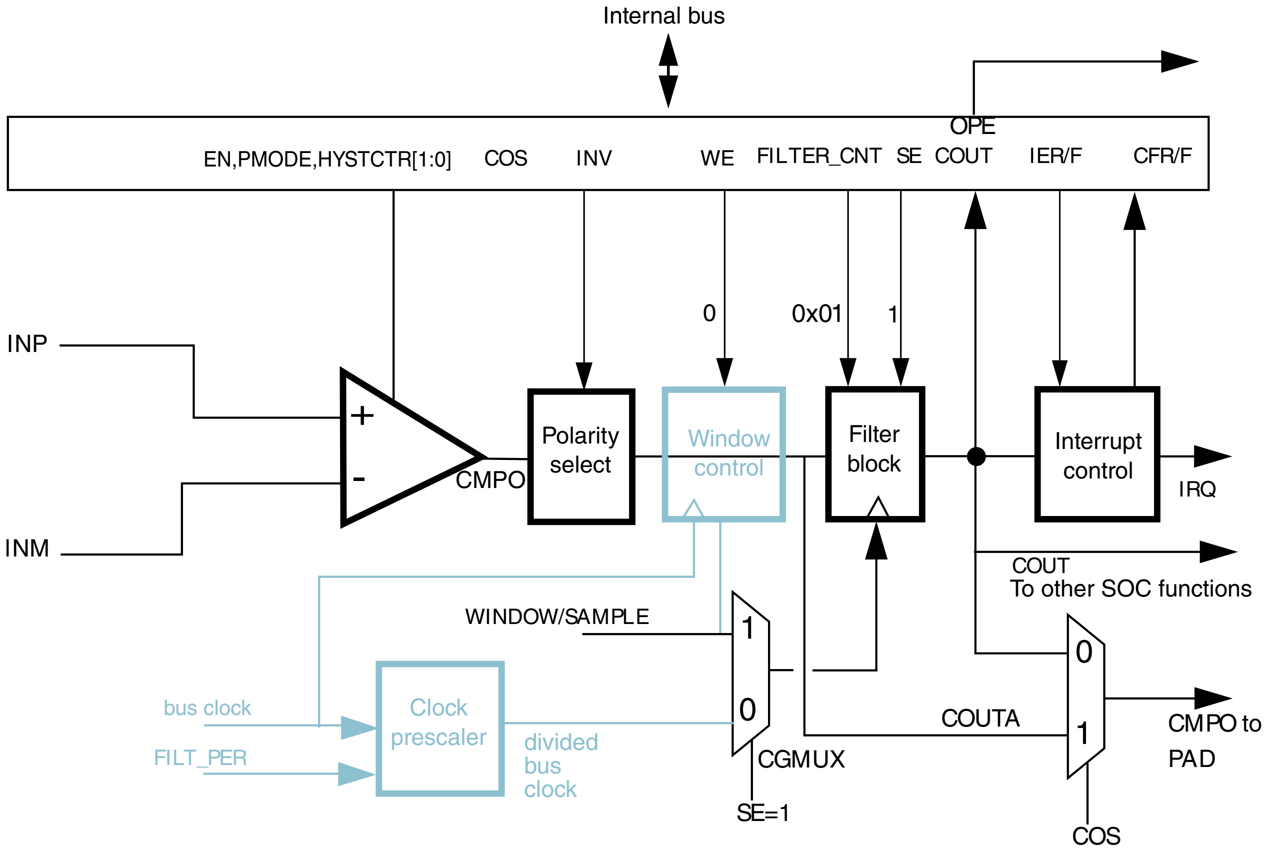


Figure 33-5. Sampled, Non-Filtered (# 3A): sampling point externally driven

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).

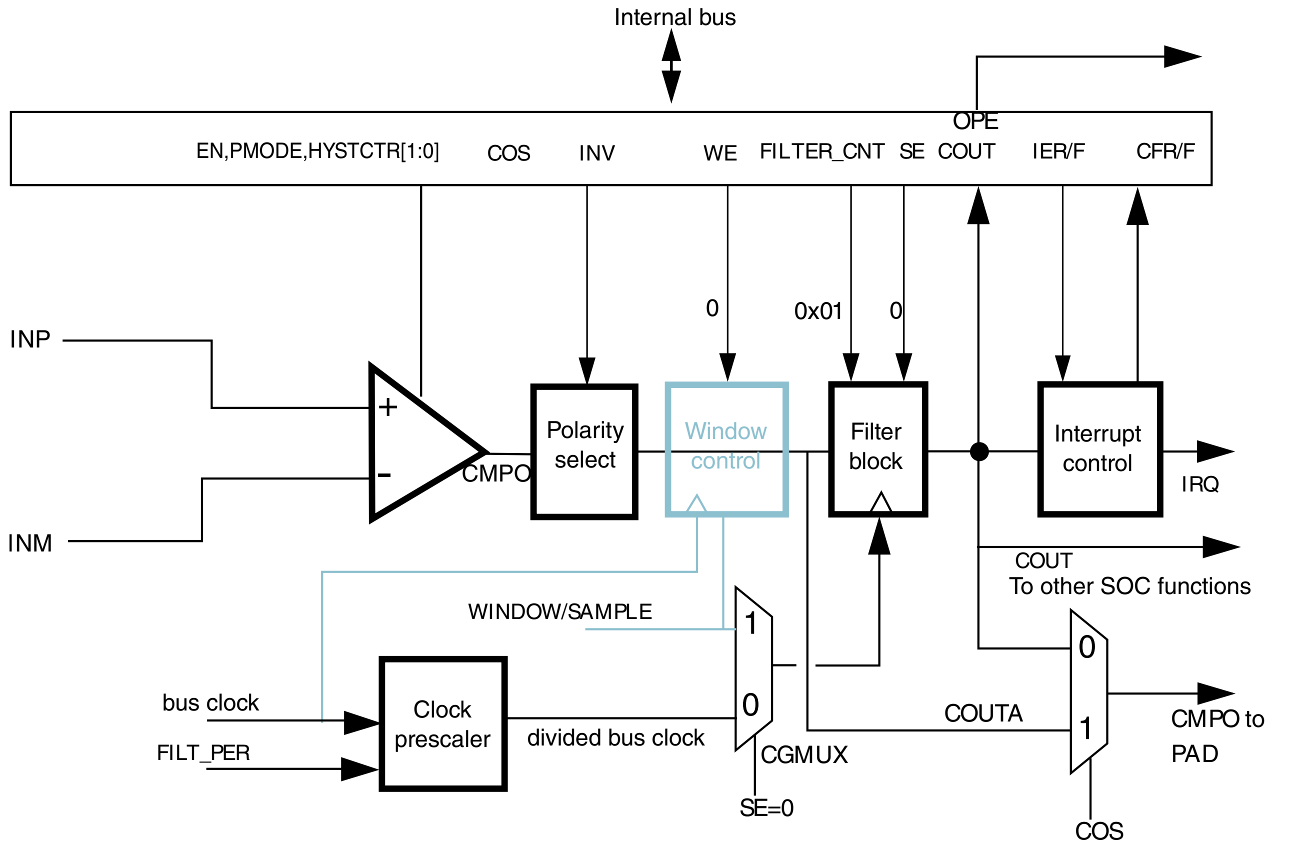


Figure 33-6. Sampled, Non-Filtered (# 3B): sampling interval internally derived

The following figure illustrates comparator operation in this mode, assuming the polarity select is set to non-inverting state.

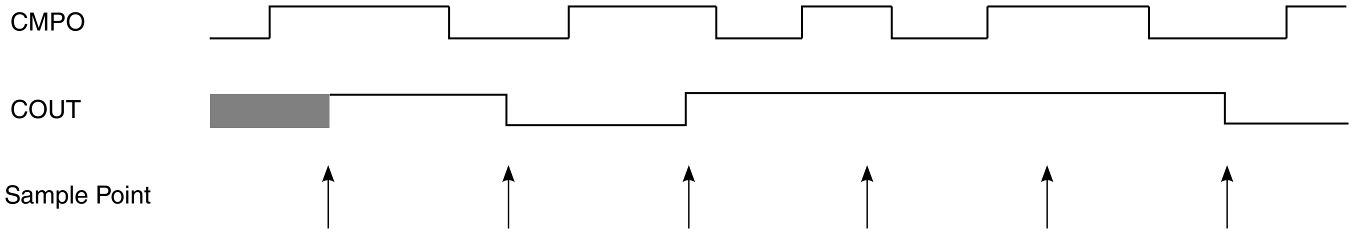


Figure 33-7. Sampled, Non-Filtered Mode Timing Diagram

33.3.1.2.4 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, C0[FILTER_CNT]>1, which activates filter operation.

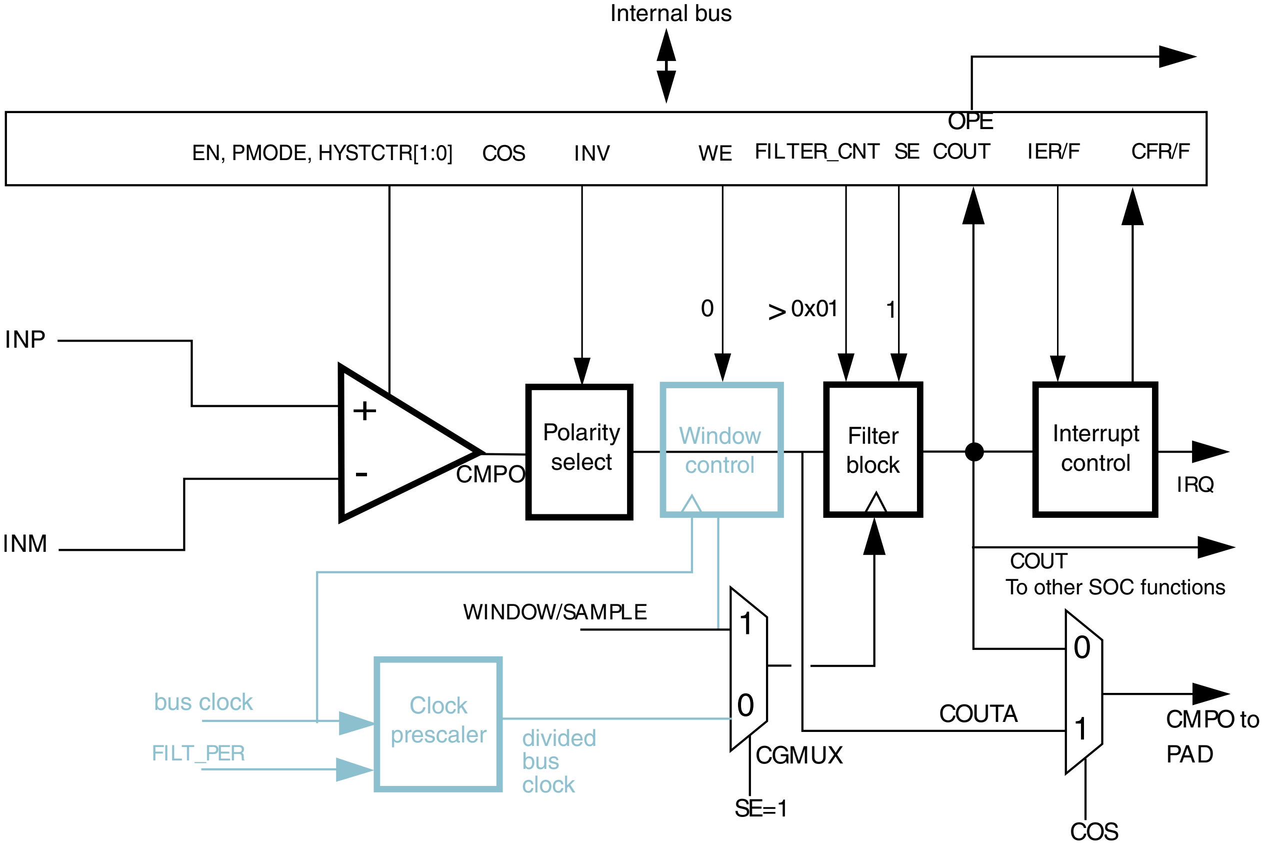


Figure 33-8. Sampled, Filtered (# 4A): sampling point externally driven

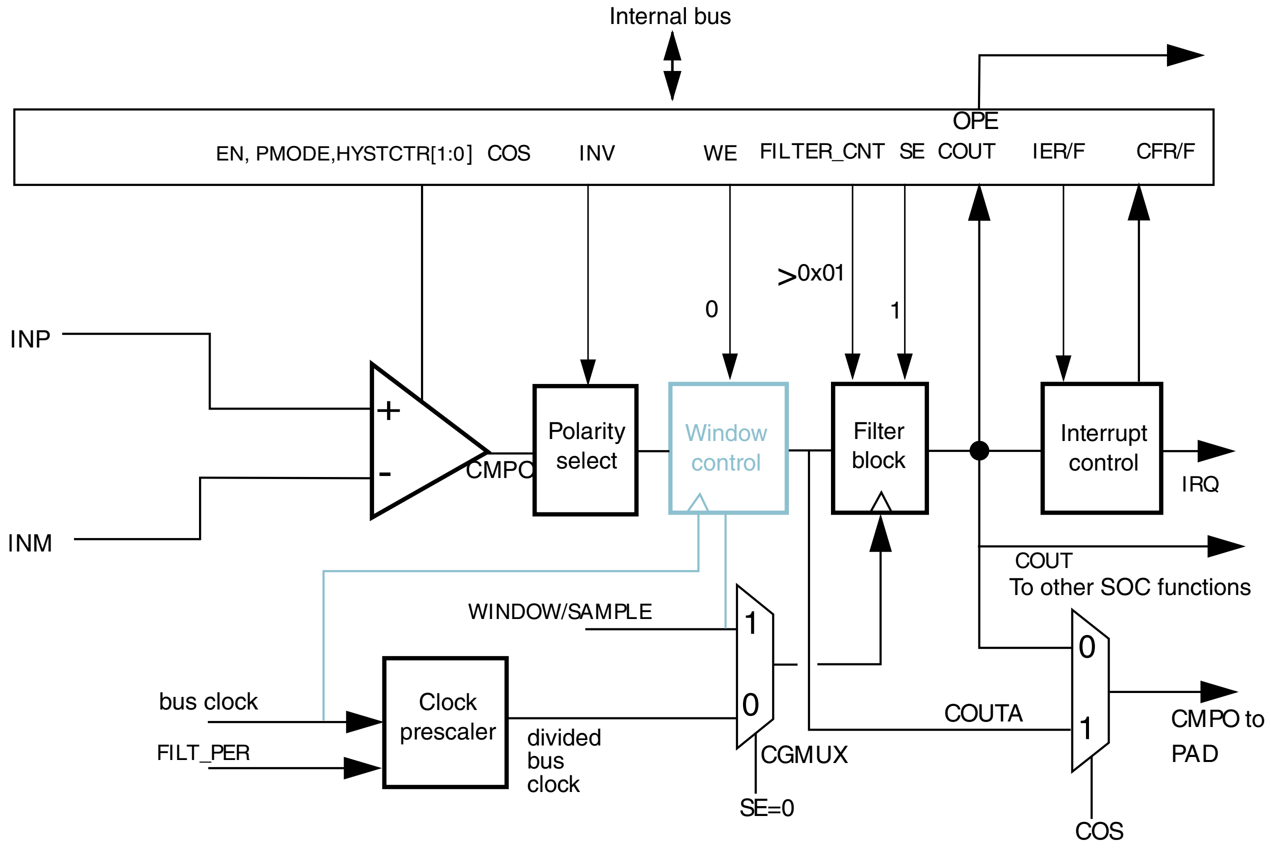


Figure 33-9. Sampled, Filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, C0[FILTER_CNT]>1, which activates filter operation.

33.3.1.2.5 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.

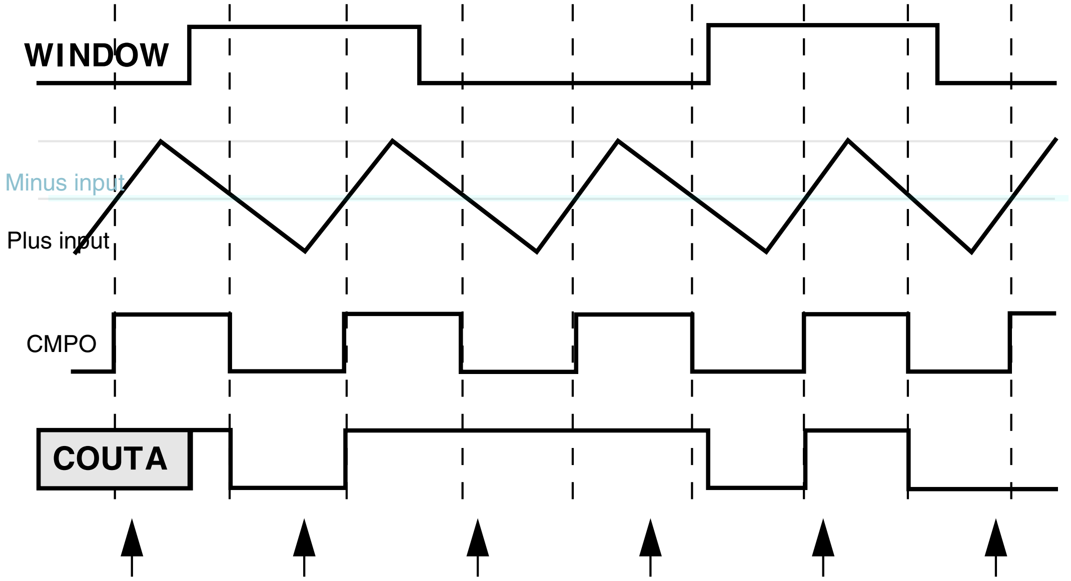


Figure 33-10. Windowed mode timing diagram

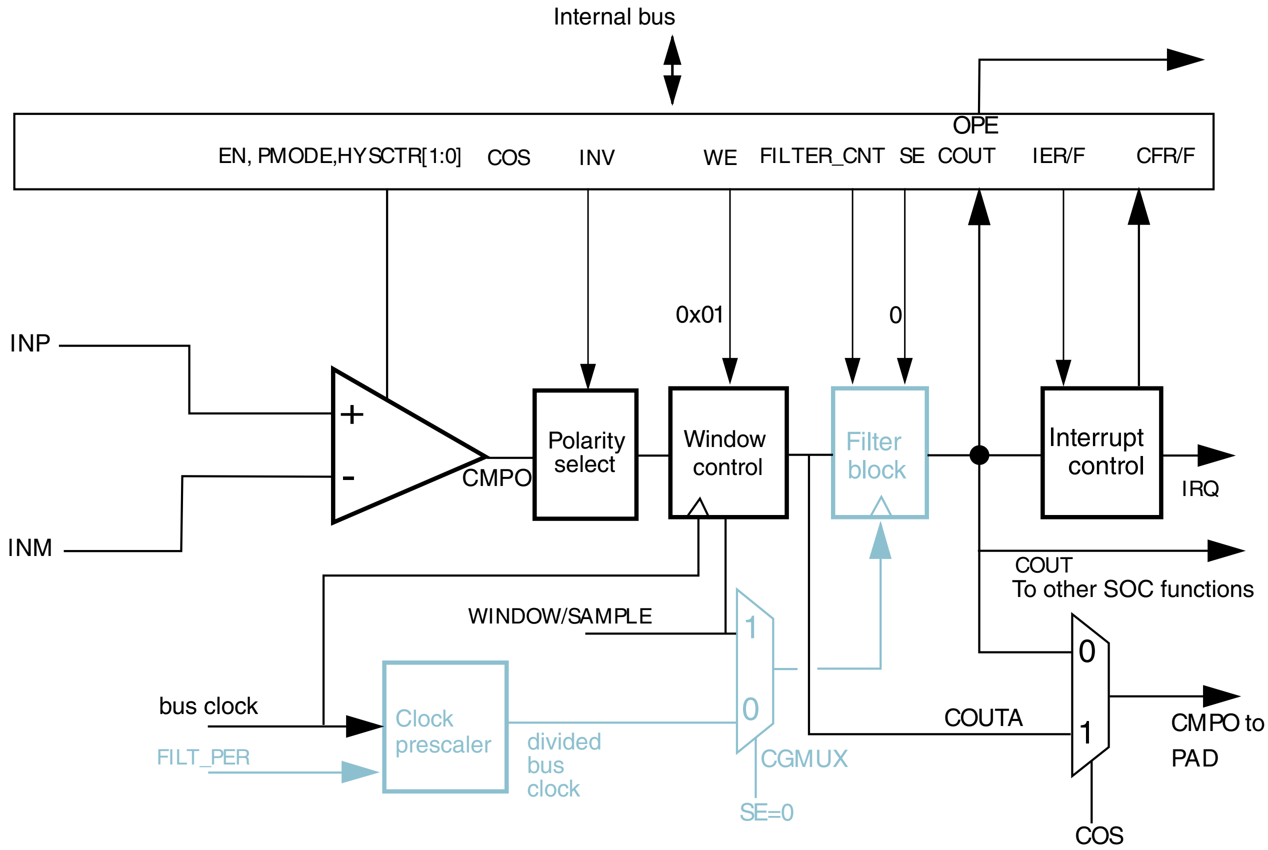


Figure 33-11. Windowed mode

For control configurations which result in disabling the filter block, see Figure 33-3.

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.

NOTE

The sample input must be high for ≥ 2.5 CMP bus clock cycles to ensure no sampling event is missed.

33.3.1.2.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 33-10, 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.

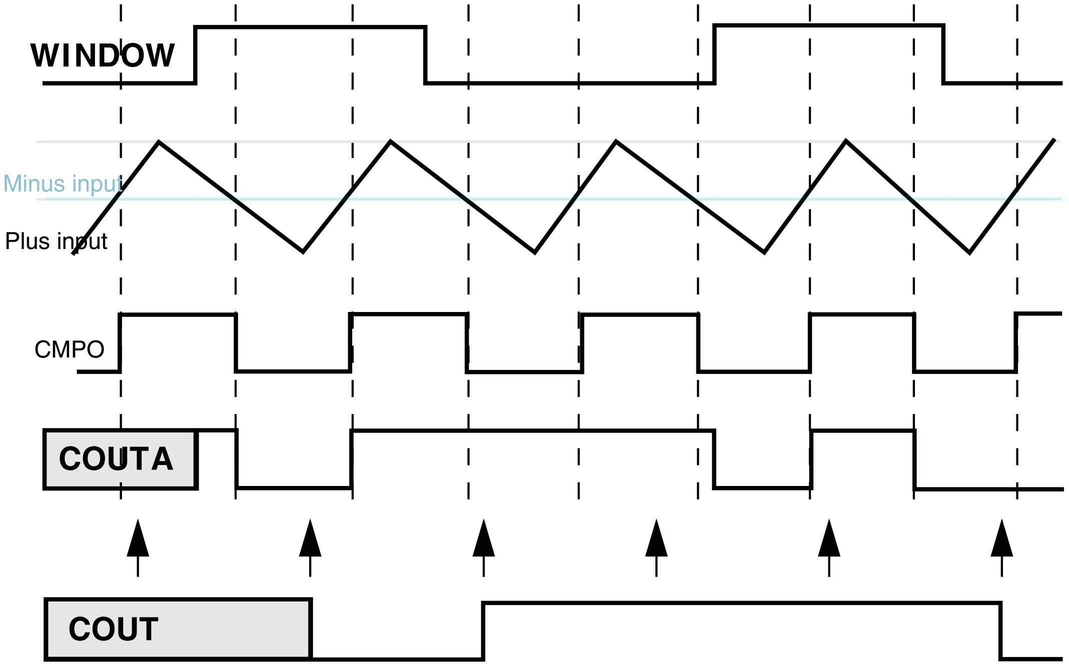


Figure 33-12. Windowed/resampled mode operation

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 C0[FILTER_CNT] must be 1.

NOTE

The sample input must be high for ≥ 2.5 CMP bus clock cycles to ensure no sampling event is missed.

33.3.1.2.7 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 + [(C0[FILTER_CNT] × C0[FPR]) + 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.

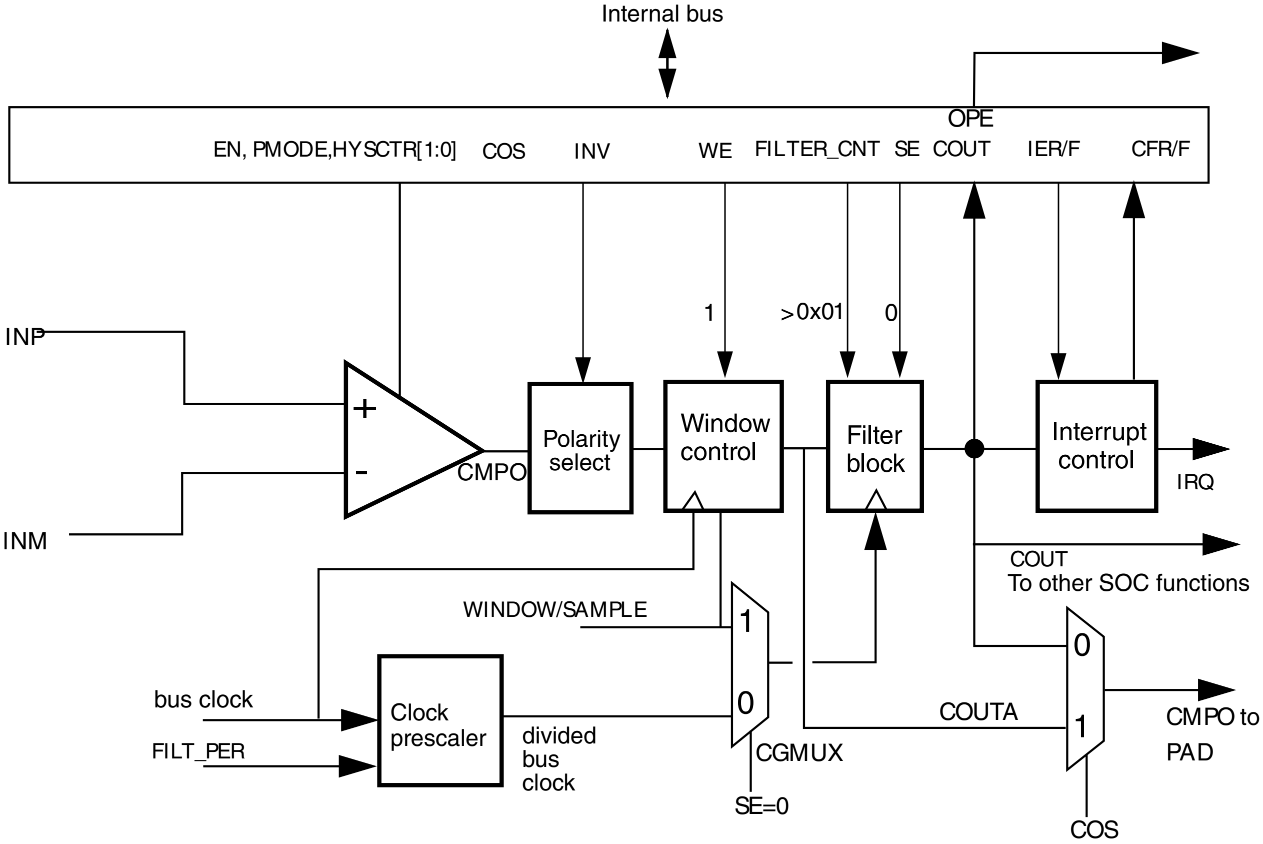


Figure 33-13. Windowed/Filtered mode

The following figure shows the operation timing for this mode, considering uncertainty is introduced by the internal synchronization for the filter block.

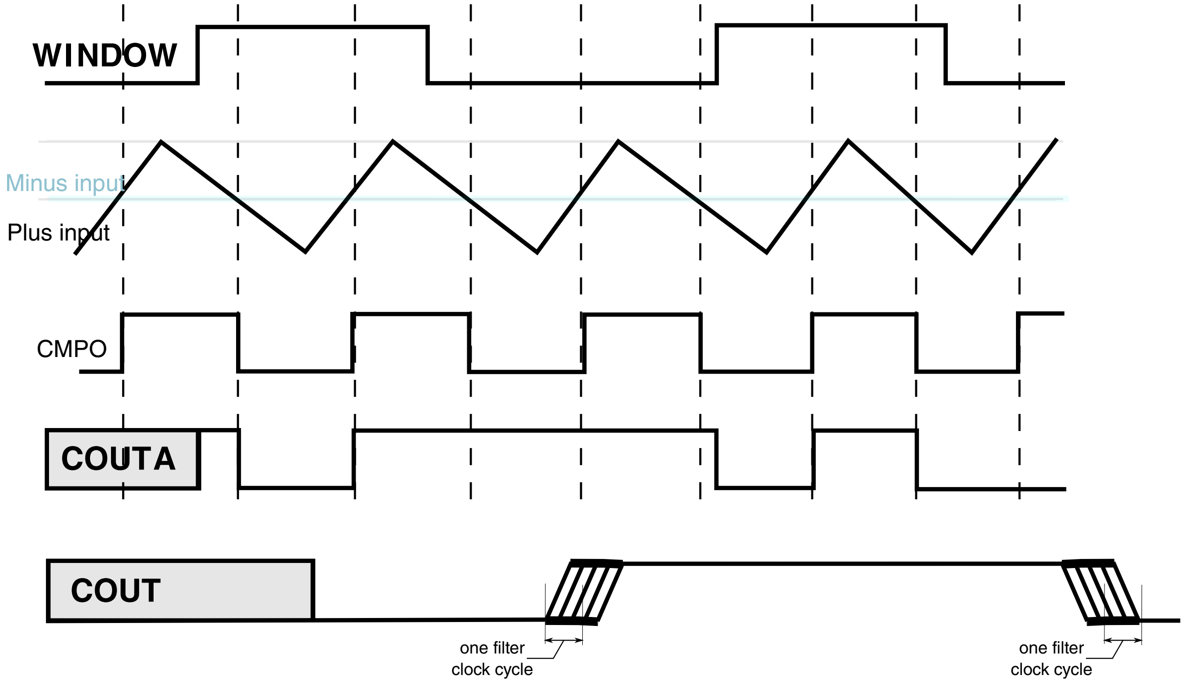


Figure 33-14. Windowed/Filtered mode operation

33.3.1.3 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 time base 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.

33.3.1.3.1 Enabling filter modes

Filter modes can be enabled by:

- Setting C0[FILTER_CNT] > 0x01 and
- Setting C0[FPR] to a nonzero value or setting C0[SE]=1

If using the divided bus clock to drive the filter, it samples COUTA every C0[FPR] bus clock cycles.

The filter output is at logic 0 when first initialized, and subsequently changes when all the consecutive C0[FILTER_CNT] samples agree that the output value has changed. In other words, C0[COUT] is 0 for some initial period, even when COUTA is at logic 1.

Setting all of C0[SE], C0[FPR] and C0[FILTER_CNT] 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 C0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If C0[SE]=1, the filter samples COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive C0[FILTER_CNT] samples agree that the output value has changed.

33.3.1.3.2 Latency issues

The value of C0[FPR] 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 C0[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 C0[FILTER_CNT].

The values of C0[FPR] or SAMPLE period and C0[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 C0[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 33-3. Comparator sample/filter maximum latencies

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	Co[FPR]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}

Table continues on the next page...

Table 33-3. Comparator sample/filter maximum latencies (continued)

Mode #	C0[EN]	C0[WE]	C0[SE]	C0[FILTER_CNT]	Co[FPR]	Operation	Maximum latency[1]
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} + (C0[FPR] * $T_{per)}$ + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (C0[FILTER_CNT] * $T_{SAMPLE)}$ + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (C0[FILTER_CNT] * C0[FPR] x $T_{per)}$ + T_{per}
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} + (C0[FPR] * $T_{per)}$ + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (C0[FILTER_CNT] * C0[FPR] 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.

33.3.2 DAC

This section provides DAC functional description.

33.3.2.1 DAC block diagram

The following figure shows the block diagram of the DAC module. It contains a 256-tap resistor ladder network and a 256-to-1 multiplexer, which selects an output voltage from one of 256 distinct levels that outputs from DACO. It is controlled through the Control register 1 (CMP_C1). 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 the Disabled mode, DACO is connected to the analog ground.

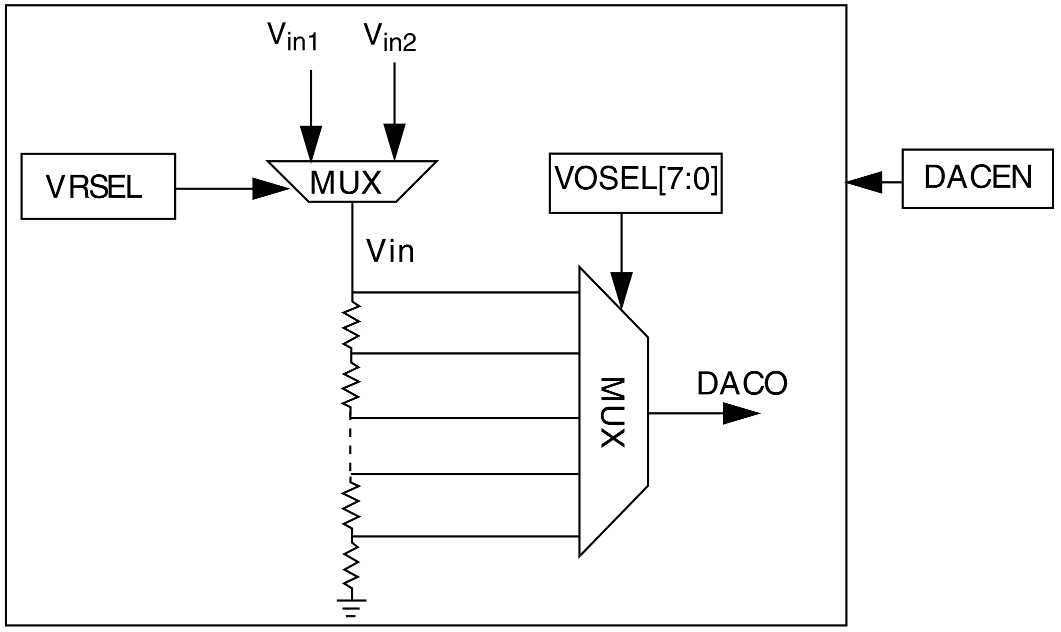


Figure 33-15. 8-bit DAC block diagram

33.3.3 Trigger mode

The CMP and the 8-bit DAC are designed to support the trigger mode operation, which is enabled when the MCU enters STOP modes with C2[RRE] and C0[EN] are set.

With this mode enabled, the trigger events that include the operation clock and a trigger start signal will initiate a compare sequence that must first enable the CMP and DAC prior to performing a CMP operation and capturing the output. A fixed channel for either the plus-side mux or the minus-side mux is selected by software via C2[FXMP] and C2[FXMXCH]. It is a mandatory request that the round-robin cycling period must be set longer than the time that all the active channels complete the specified comparison cycles set by C2[NSAM].

The active channels selected by C1[CHN*n*] are then routed to the non-fixed channel mux and compared with the reference input in a round-robin manner. In order to meet the comparator stabilization time, after the configurable number of operation clocks defined by C2[NSAM], the comparison result is sampled for the selected channel. A software pre-programmed state for each channel is configured by writing to C2[ACO*n*] field. After all the active channels are sampled, if the comparison result changes from its pre-programmed state, the corresponding flag in C2[CH*n*F] is set. If C2[RRIE] is set, an asynchronous reset is asserted to bring the MCU out of STOP mode.

NOTE

These flags do not support generating a DMA transfer event.

This mode is active when the MCU is in STOP mode, so none of the window/filter functions are available. A basic assumption of this mode is that the selected inputs are changing at a much slower rate than the operation clock. It is suggested to configure the comparator in low power comparison mode as well. In programming the C2[INITMOD] registers, the INITMOD × round-robin clock period must be longer than the initialization delay, which can be referred from the chip datasheet.

The following diagram shows the basic flow of this mode. In the diagram, C1[CHN1], C1[CHN3], and C1[CHN7] are set, so channels #1, #3, and #7 are selected for round-robin. C2[NSAM] is set to 2'b01, so one clock later the comparison result of the selected channel is sampled. When channel #7 is compared, the result is sampled, and round-robin ends. If any of the comparison results from channel #1, #3, or #7 changed from their programmed value (written to C2[ACO1], C2[ACO3], and C2[ACO7]), an interrupt is generated to wake up the MCU from the STOP mode. Software can then poll the C2[CH*n*F] to see which channel input(s) changed value during the STOP mode.

NOTE

In round-robin mode, it should be ensured that the RTC_CLK period is greater than the comparison time corresponding to the value of C0[PMODE]. It is also required to **not** select the internal reserved channels for round-robin by INPSEL and INNSEL.

NOTE

In round-robin mode, it is suggested to always configure the DAC output as the fixed port reference.

NOTE

In round-robin mode, current injection or over-voltage is not supported on the input channels.

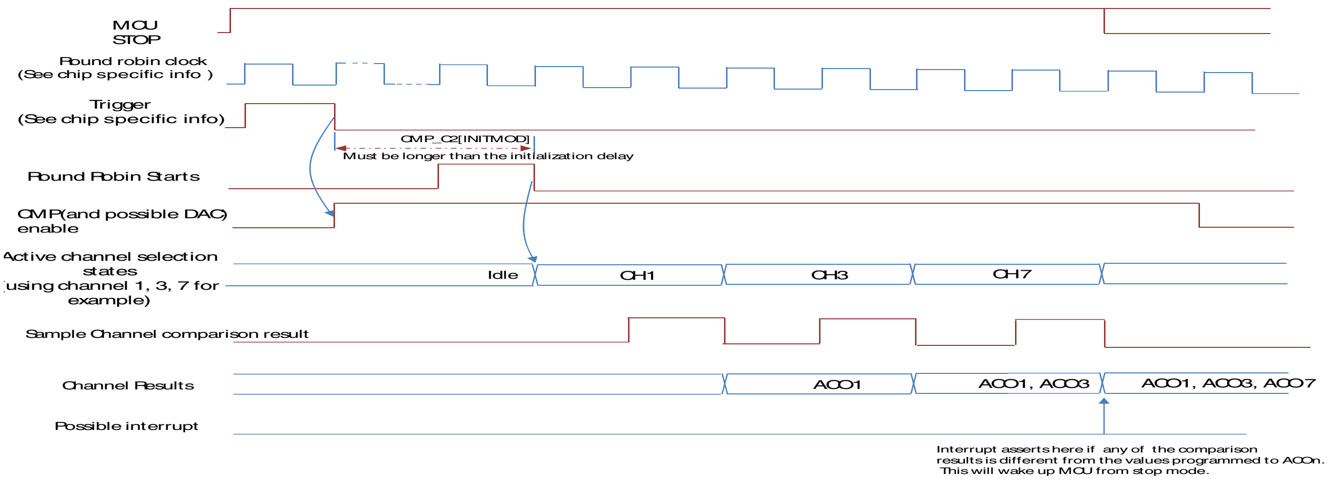

MCU STOP
Round robin clock (See chip specific info)
Trigger (See chip specific info)
CMP_C2[INITMOD]
Must be longer than the initialization delay
Round Robin Starts
CMP(and possible DAC) enable
Active channel selection states (using channel 1, 3, 7 for example)
Idle
CH1
CH3
CH7
Sample Channel comparison result
Channel Results
ACO1
ACO1, ACO3
ACO1, ACO3, ACO7
Possible interrupt
Interrupt asserts here if any of the comparison results is different from the values programmed to ACOn. This will wake up MCU from stop mode.

Figure 33-16. Trigger mode

The following table shows the channel decoding in both functional mode and trigger mode. Other cases not listed in the table are illegal.

Table 33-4. CMP channel decoding in functional mode and trigger mode

Mode	RRE	PSEL[2:0]	MSEL[2:0]	INPSEL[1:0]	INNSEL[1:0]	FXMP	FXMXCH[2:0]	CHNx	INP	INN	CMP Behavior
Functional Mode	0	x[1]	0~7	0	1	x	x	x	DAC	Channel decoded from MSEL[2:0]	Channel 0~7 can be compared with DAC
		0~7	x	1	0	x	x	x	Channel decoded from PSEL[2:0]	DAC	Channel 0~7 can be compared with DAC
		0~7	0~7	1	1	x	x	x	Channel decoded from PSEL[2:0]	Channel decoded from MSEL[2:0]	Channel 0~7 can be compared with channel 0~7[2]
Trigger Mode	1	x	x	0	1	0	x	0~7	DAC	Channel sweep (CHN*x*)	Channel 0~7 can be swept with DAC
		x	x	1	0	1	x	0~7	Channel sweep (CHN*x*)	DAC	Channel 0~7 can be swept with DAC
		x	x	1	1	0	0~7	0~7	Channel fixed by FXMXCH[2:0]	Channel sweep (CHN*x*)	Channel 0~7 can be swept with a fixed channel (0~7)[3]

Table continues on the next page...

Table 33-4. CMP channel decoding in functional mode and trigger mode (continued)

Mode	RRE	PSEL[2:0]	MSEL[2:0]	INPSEL[1:0]	INNSEL[1:0]	FXMP	FXMXCH[2:0]	CHNx	INP	INN	CMP Behavior
		x	x	1	1	1	0~7	0~7	Channel sweep (CHN*x*)	Channel fixed by FXMXCH[2:0]	Channel 0~7 can be swept with a fixed channel (0~7)[3]

1. "x" means "do not care".
2. PSEL should not be set the same as MSEL.
3. Channel in the sweep side should not be the same as the fixed side.

33.3.4 Clocking

33.3.4.1 DAC clocks

This module has a single clock input, the bus clock.

33.3.5 Reset

33.3.5.1 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

33.3.6 Interrupts

33.3.6.1 CMP interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both. Assuming the CMP DMA enable bit is not set, the following table gives the conditions in which the interrupt request is asserted and deasserted.

Table 33-5. CMP interrupt generations

When	Then
C0[IER] and C0[CFR] are set	The interrupt request is asserted

Table continues on the next page...

Table 33-5. CMP interrupt generations (continued)

When	Then
C0[IEF] and C0[CFF] are set	The interrupt request is asserted
C0[IER] and C0[CFR] are cleared for a rising-edge interrupt	The interrupt request is deasserted
C0[IEF] and C0[CFF] are cleared for a falling-edge interrupt	The interrupt request is deasserted

33.3.6.2 DAC interrupts

This module has no interrupts.

33.3.7 DMA

33.3.7.1 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting C0[DMAEN] and the interrupt is enabled by setting C0[IER], C0[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 signal 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.

33.4 External signals

33.4.1 CMP pin descriptions

This section provides the comparator pin descriptions. The external inputs IN[7:0] are muxed by CMP_C1[PSEL] and CMP_C1[MSEL] beforehand and multiplexed output will then go to the second stage of multiplex with the input of 8-bit DAC and other two internal reserved test signals, determined by CMP_C1[INPSEL] and CMP_C1[INNSEL]. The output of the second multiplex will finally go to the positive and negative ports of the comparator respectively.

Table 33-6. CMP signal descriptions

Signal	Description	I/O
IN[7:0]	Analog voltage inputs	I

NOTE

If comparing one input channel with the DAC output, and if there is injection or over-voltage in the input channels, the DAC output may be corrupted. For such case, the software workaround is to configure the DAC side SEL[2:0] same as the non-DAC side, i.e. configuration of MSEL and PSEL register bits must be the same.

33.4.1.1 External pins

The CMP has two analog inputs: INP and INM. Each of these pins can accept an input voltage that varies across the full operating range of the MCU. If the module is not enabled, each pin can be used as a digital input or output. Consult the specific MCU documentation to determine what functions are shared with these analog inputs.

The user can select either filtered or unfiltered comparator outputs for use on an external I/O pad.

33.5 Initialization

A typical startup sequence is as follows.

The time required to stabilize COUT is 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 datasheet 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 section.

During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and C0[CFR]/C0[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 initially equals 0 until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

33.6 Memory map/register definitions

CMP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_3000	CMP Control Register 0 (CMP0_C0)	32	R/W	0000_0000h	33.6.1/729
4007_3004	CMP Control Register 1 (CMP0_C1)	32	R/W	0000_0000h	33.6.2/732
4007_3008	CMP Control Register 2 (CMP0_C2)	32	R/W	0000_0000h	33.6.3/735

33.6.1 CMP Control Register 0 (CMPx_C0)

Access:

- Supervisor read/write
- User read/write

Address: 4007_3000h base + 0h offset = 4007_3000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DMAEN	0	IER	IEF	CFR	CFF	COUT	FPR							
W						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	SE	WE	0	PMODE	INVT	COS	OPE	EN	0	FILTER_CNT			0	OFFSET	HYSTCTR	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMPx_C0 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 DMAEN	DMA Enable 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.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 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.
27 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.
26 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 A rising edge has not been detected on COUT. 1 A rising edge on COUT has occurred.
25 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 A falling edge has not been detected on COUT. 1 A falling edge on COUT has occurred.
24 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 C0[INVT] when the Analog Comparator module is disabled, that is, when C0[EN] = 0. Writes to this field are ignored.
23–16 FPR	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when C0[SE] = 0. Setting FPR to 0x0 disables the filter. Filter programming and latency details are provided in the CMP functional description. This field has no effect when C0[SE]= 1. In that case, the external SAMPLE signal is used to determine the sampling period.
15 SE	Sample Enable

Table continues on the next page...

CMPx_C0 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 ones to both bit locations because this "11" case is reserved. 0 Sampling mode is not selected. 1 Sampling mode is selected.
14 WE	Windowing 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 ones to both bit locations because this "11" case is reserved. 0 Windowing mode is not selected. 1 Windowing mode is selected.
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 PMODE	Power Mode Select 0 Low Speed (LS) comparison mode is selected. 1 High Speed (HS) comparison mode is selected, in VLPx mode, or Stop mode switched to Low Speed (LS) mode.
11 INVT	Comparator invert This bit allows selecting the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as C0[COUT]) when C0[OPE]=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
10 COS	Comparator Output Select 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
9 OPE	Comparator Output Pin Enable The OPE bit enables the path from the comparator output to a selected pin. 0 When OPE is 0, the comparator output (after window/filter settings dependent on software configuration) is not available to a packaged pin. 1 When OPE is 1, and if the software has configured the comparator to own a packaged pin, the comparator is available in a packaged pin.
8 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, the analog part remains in the off state, and consumes no power. 0 Analog Comparator is disabled. 1 Analog Comparator is enabled.
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_C0 field descriptions (continued)

Field	Description
	This field specifies the number of consecutive samples that must agree prior to the comparator output filter accepting a new output state. For information regarding filter programming and latency, please see the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (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 OFFSET	Comparator hard block offset control. See chip data sheet to get the actual offset value with each level **NOTE:** • If OFFSET = 1, then there will be no hysteresis in the case of INP crossing INN in the positive direction (or INN crossing INP in the negative direction). A Half Hysteresis value still exists for INP crossing INN in the falling direction. • If OFFSET = 0, then the hysteresis selected by HYSTCTR is valid for both directions. 0 The comparator hard block output has level 0 offset internally. 1 The comparator hard block output has level 1 offset internally.
HYSTCTR	Comparator hard block hysteresis control. See chip data sheet to get the actual hysteresis value with each level 00 The hard block output has level 0 hysteresis internally. 01 The hard block output has level 1 hysteresis internally. 10 The hard block output has level 2 hysteresis internally. 11 The hard block output has level 3 hysteresis internally.

33.6.2 CMP Control Register 1 (CMPx_C1)

Access:

- Supervisor read/write
- User read/write

Address: 4007_3000h base + 4h offset = 4007_3004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0	INPSEL		0	INNSEL		CHN7	CHN6	CHN5	CHN4	CHN3	CHN2	CHN1	CHN0
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	DACEN	VRSEL	PSEL			MSEL			VOSEL							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMPx_C1 field descriptions

Field	Description
31–30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28–27 INPSEL	Selection of the input to the positive port of the comparator Determines which input is selected for the plus input of the comparator. NOTE: These selections is used to select the final positive input to the comparator. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMP*x*_C1 register must have different values. 00 IN0, from the 8-bit DAC output 01 IN1, from the analog 8-1 mux 10 Reserved 11 Reserved
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 INNSEL	Selection of the input to the negative port of the comparator Determines which input is selected for the minus input of the comparator. NOTE: These selections is used to select the final negative input to the comparator. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMP*x*_C1 register must have different values. 00 IN0, from the 8-bit DAC output 01 IN1, from the analog 8-1 mux 10 Reserved 11 Reserved
23 CHN7	Channel 7 input enable Channel 7 of the input enable for the round-robin checker. If CHN7 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
22 CHN6	Channel 6 input enable Channel 6 of the input enable for the round-robin checker. If CHN6 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
21 CHN5	Channel 5 input enable

Table continues on the next page...

CMPx_C1 field descriptions (continued)

Field	Description
	Channel 5 of the input enable for the round-robin checker. If CHN5 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
20 CHN4	Channel 4 input enable Channel 4 of the input enable for the round-robin checker. If CHN4 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
19 CHN3	Channel 3 input enable Channel 3 of the input enable for the round-robin checker. If CHN3 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
18 CHN2	Channel 2 input enable Channel 2 of the input enable for the round-robin checker. If CHN2 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
17 CHN1	Channel 1 input enable Channel 1 of the input enable for the round-robin checker. If CHN1 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
16 CHN0	Channel 0 input enable Channel 0 of the input enable for the round-robin checker. If CHN0 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
15 DACEN	DAC Enable This bit is used to enable the DAC. When the DAC is disabled, it is powered down to conserve power. 0 DAC is disabled. 1 DAC is enabled.
14 VRSEL	Supply Voltage Reference Source Select 0 Vin1 is selected as resistor ladder network supply reference Vin. 1 Vin2 is selected as resistor ladder network supply reference Vin.
13–11 PSEL	Plus Input MUX Control Determines which input is selected for the plus mux. NOTE: These bits are used to select the external 8 inputs for the plus mux, the actual input to the positive port of the comparator is selected between this mux out and other inputs finally, see the definition in INPSEL. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMP*x*_C1 register must have different values. 000 IN0 001 IN1 010 IN2 011 IN3

Table continues on the next page...

CMPx_C1 field descriptions (continued)

Field	Description
	100 IN4 101 IN5 110 IN6 111 IN7
10–8 MSEL	Minus Input MUX Control Determines which input is selected for the minus mux. NOTE: These bits are used to select the external 8 inputs for the minus mux, the actual input to the negative port of the comparator is selected between this mux out and other inputs finally, see the definition in INNSEL. Note: For the round robin mode of operation, the MSEL and PSEL bitfields in CMP*x*_C1 register must have different values. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7
VOSEL	DAC Output Voltage Select This bit selects an output voltage from one of 256 distinct levels. DACO = (Vin/256) × (VOSEL[7:0] + 1), so the DACO range is from Vin/256 to Vin.

33.6.3 CMP Control Register 2 (CMPx_C2)

Access:

- Supervisor read/write
- User read/write

Address: 4007_3000h base + 8h offset = 4007_3008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RRE	RRIE	FXMP	0	FXMXCH			0	CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NSAM		INITMOD						ACOn							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMPx_C2 field descriptions

Field	Description
31 RRE	Round-Robin Enable This bit enables the round-robin operation. 0 Round-robin operation is disabled. 1 Round-robin operation is enabled.
30 RRIE	Round-Robin interrupt enable This bit enables the interrupt/wake-up when the comparison result changes for a given channel. 0 The round-robin interrupt is disabled. 1 The round-robin interrupt is enabled when a comparison result changes from the last sample.
29 FXMP	Fixed MUX Port This bit is used to fix the analog mux port for the round-robin mode. 0 The Plus port is fixed. Only the inputs to the Minus port are swept in each round. 1 The Minus port is fixed. Only the inputs to the Plus port are swept in each round.
28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–25 FXMXCH	Fixed channel selection This field indicates which channel in the mux port is fixed in a given round-robin mode. 000 Channel 0 is selected as the fixed reference input for the fixed mux port. 001 Channel 1 is selected as the fixed reference input for the fixed mux port. 010 Channel 2 is selected as the fixed reference input for the fixed mux port. 011 Channel 3 is selected as the fixed reference input for the fixed mux port. 100 Channel 4 is selected as the fixed reference input for the fixed mux port. 101 Channel 5 is selected as the fixed reference input for the fixed mux port. 110 Channel 6 is selected as the fixed reference input for the fixed mux port. 111 Channel 7 is selected as the fixed reference input for the fixed mux port.
24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 CH7F	Channel 7 input changed flag. This bit is set if the channel 7 input changed from the last comparison with the fixed mux port.

Table continues on the next page...

CMPx_C2 field descriptions (continued)

Field	Description
22 CH6F	Channel 6 input changed flag. This bit is set if the channel 6 input changed from the last comparison with the fixed mux port.
21 CH5F	Channel 5 input changed flag. This bit is set if the channel 5 input changed from the last comparison with the fixed mux port.
20 CH4F	Channel 4 input changed flag. This bit is set if the channel 4 input changed from the last comparison with the fixed mux port.
19 CH3F	Channel 3 input changed flag. This bit is set if the channel 3 input changed from the last comparison with the fixed mux port.
18 CH2F	Channel 2 input changed flag. This bit is set if the channel 2 input changed from the last comparison with the fixed mux port.
17 CH1F	Channel 1 input changed flag. This bit is set if the channel 1 input changed from the last comparison with the fixed mux port.
16 CH0F	Channel 0 input changed flag. This bit is set if the channel 0 input changed from the last comparison with the fixed mux port.
15–14 NSAM	Number of sample clocks For a given channel, this field specifies how many round-robin clock cycles later the sample takes place. 00 The comparison result is sampled as soon as the active channel is scanned in one round-robin clock. 01 The sampling takes place 1 round-robin clock cycle after the next cycle of the round-robin clock. 10 The sampling takes place 2 round-robin clock cycles after the next cycle of the round-robin clock. 11 The sampling takes place 3 round-robin clock cycles after the next cycle of the round-robin clock.
13–8 INITMOD	Comparator and DAC initialization delay modulus. These values specify the round robin clock cycles used to determine the comparator and DAC initialization delays specified by the datasheet. For example the initialization delay is 80us and the round robin clock is 100kHz, then INITMOD should be set to 80us/10us = 8. 000000 The modulus is set to 64 (same with 111111). other values Initialization delay is set to INITMOD × round robin clock period
ACOn	The result of the input comparison for channel *n* . This field stores the latest comparison result of the input channel *n* with the fixed mux port. Reading this bit returns the latest comparison result. Writing this field defines the pre-set state of channel *n*.

33.7 Usage Guide

33.7.1 Zero Crossing Detection

A zero-crossing is a point where the sign of a signal’s mathematical function changes (e.g. from positive to negative), represented by a crossing of the axis (zero value) in the graph of the signal function. It is a commonly used in electronics application especially for systems which send digital data over AC circuits.

When in some cases, the “Zero point” could be other voltage than actual 0 V. This “Zero point” would be used to judge whether the indicated voltage level is reached. In this situation, the internal DAC could generate the reference voltage level for “Zero point” to make the comparison with the other input channel of CMP module, and then output the result of logic “0” and “1”.

To enable the internal DAC and set it as the comparator’s input of minus side, the code could be as follow:

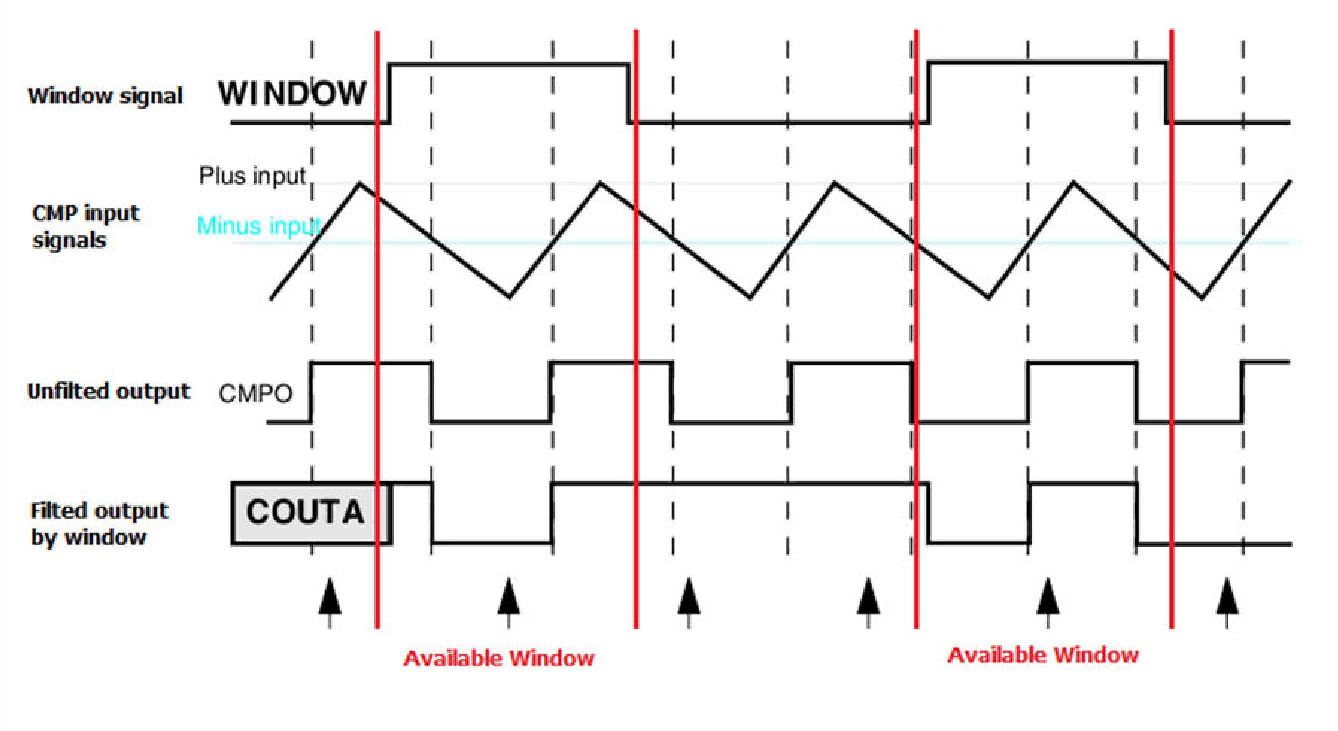
```
/* Set internal DAC as minus input. */
CMPx_C1 &= ~CMP_C1_ INNSEL_MASK;

/* Set input channel 3 as plus input. */
CMPx_C1 = (CMPx_C1 & ~(CMP_C1_ INPSEL_MASK | CMP_C1_PSEL_MASK))
          | CMP_C1_INPSEL(1) | CMP_C1_PSEL(3);
```

Then, the CMP output interrupts with their flags would be used to indicate the event of Zero Crossing Detection.

33.7.2 Window Mode

This mode could be used to create a kind of filter for input signal. When enabling the window mode, the compare would only launch the comparison in available window, which could be generated by some timer modules (e.g. PDB or LPIT). And output of CMP in unavailable window would be hold.



To enable the window mode for CMP, the code could be as follows:

```
/* Enable the window mode and disable the sample mode. */
CMPx_C0 = (CMPx_C0 & ~ CMP_C0_SE_MASK ) | CMP_C0_WE_MASK;
```

Then enable the window's generator (to produce the WINDOW signal) of related module.

For detailed information about CMP's window feature, please see to section "Windowed mode" in this chapter.

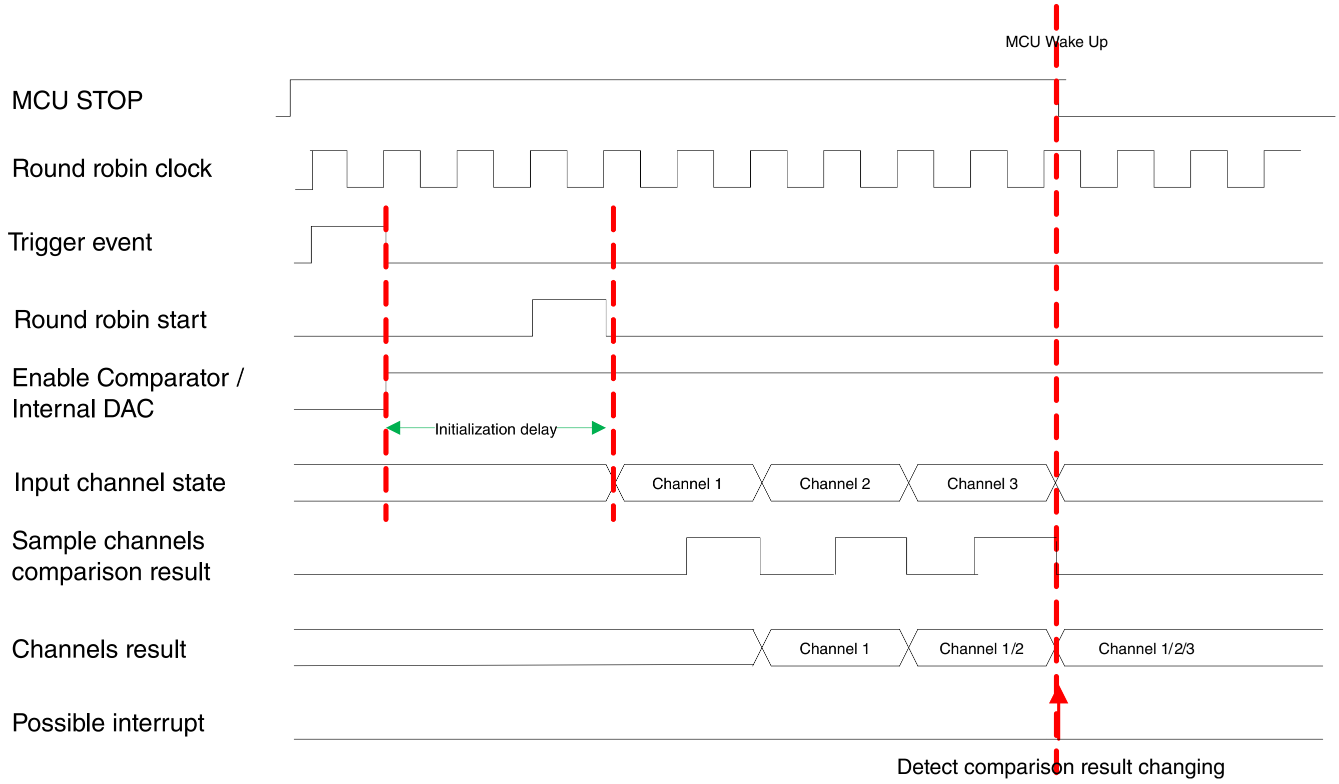
33.7.3 Round Robin Mode

This mode compares multiple input channels with the reference input channel (fixed) in a round-robin manner. It is commonly used to provide a trigger mode to wake up the MCU in STOP mode.

This mode needs some trigger events to work. The trigger events include the operation clock and a trigger start signal which can be provided by other module (e.g. LPTMR).

Round robin mode works as follows:

1. The trigger start signal will enable the comparator and internal DAC in the initialization delay period;
2. The comparator will then compare the multiple input channels with the reference input channel in turn under the operation clock until all input channels complete comparison;
3. If current comparison result is different with the pre-set state or the previous comparison result and round robin interrupt is enabled, an interrupt will generate to bring the MCU out of STOP mode.



The code snippet to enable the round robin mode is:

```
/* Set the positive port input from DAC and negative port input from minus mux input */
/* Plus mux input must be different from minus mux input even though they aren't functional
in round robin mode. */
CMPx_C1 = ((CMPx_C1 & (~(CMP_C1_INPSEL_MASK | CMP_C1_INNSEL_MASK | CMP_C1_PSEL_MASK |
CMP_C1_MSEL_MASK))
           | (CMP_C1_INPSEL(0) | CMP_C1_INNSEL(1) | CMP_C1_PSEL(0) | CMP_C1_MSEL(1)));

/* Set following round robin attribute:
positive port as fixed port.
All channel0~7 as the round robin checker channel in non-fixed port.
The comparison result is sampled as soon as the active channel is scanned in one round-robin
clock.
The initialization delay modulus is set to 64.
Enable round robin mode.
Enable round robin interrupt.
*/
CMPx_C1 = ((CMPx_C1 & (~(CMP_C1_CHN0_MASK | CMP_C1_CHN1_MASK | CMP_C1_CHN2_MASK |
CMP_C1_CHN3_MASK |
           CMP_C1_CHN4_MASK | CMP_C1_CHN5_MASK | CMP_C1_CHN6_MASK | CMP_C1_CHN7_MASK)))
           |(0xFF << CMP_C1_CHN0_SHIFT));

CMPx_C2 = ((CMPx_C2 & (~(CMP_C2_FXMP_MASK | CMP_C2_FXMXCH_MASK | CMP_C2_NSAM_MASK
           | CMP_C2_INITMOD_MASK | CMP_C2_CHnF_MASK))) | (CMP_C2_FXMP(0)
           | CMP_C2_FXMXCH(0) | CMP_C2_NSAM(0)
           | CMP_C2_INITMOD(0) | CMP_C2_RRE_MASK | CMP_C2_RRIE_MASK));

/* Set all the pre-state of round robin checker channel0~7 to 1. */
CMPx ->C2 = ((CMPx ->C2 & (~CMP_C2_ACOn_MASK | CMP_C2_CHnF_MASK)) | (0xFF <<
CMP_C2_ACOn_SHIFT));

/* Set round robin comparison trigger. See the chip configuration about the available
trigger in the SoC. */
```

```
/* Set SoC enter into STOP mode. See the power management chapter. */

/* Change the voltage of input channel to wake up the SoC. */
```

Chapter 34
FlexTimer Module (FTM)

34.1 Chip-specific information for this module

34.1.1 Instantiation Information

This device contains three FlexTimer modules.

The following table shows how these modules are configured.

Table 34-1. FTM Instantiations

FTM instance	Number of channels	Features/usage
FTM0	8	• FTM enhanced features • Global time base • Fault Control supported in FTM0
FTM1	4	• FTM basic features • Global time base
FTM2	4	• FTM basic features • Global time base

NOTE

The Global Load mechanism of FTM module is not supported on this device.

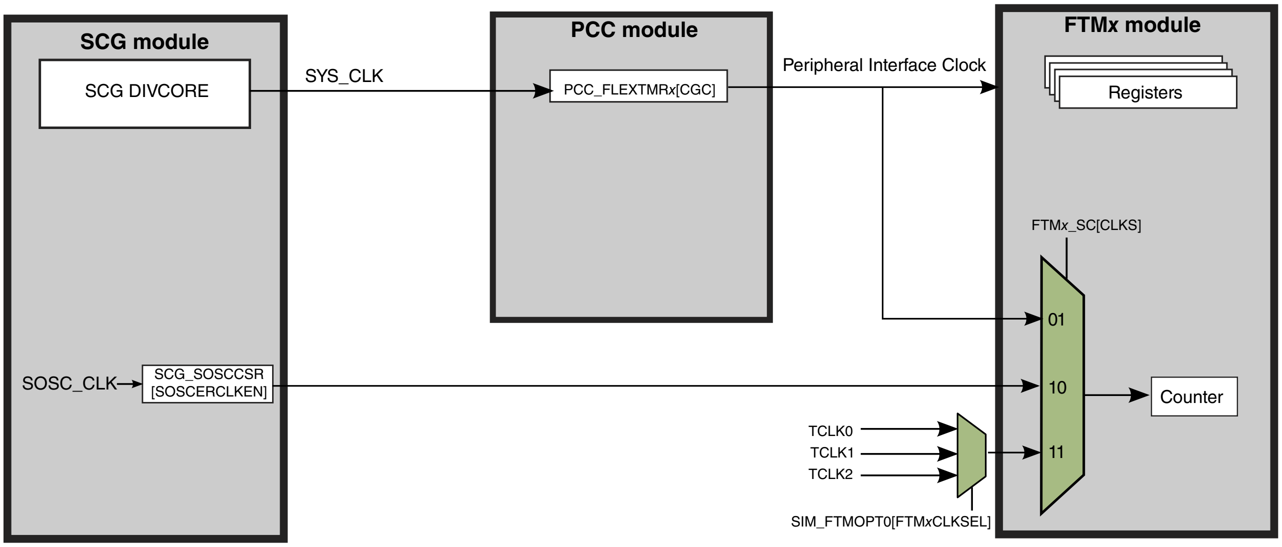
34.1.2 FTM Clocking Information

The following figure shows the input clock sources available for this module.

NOTE

It is recommended to clear the FTM channel (n) flag bits CHF right after writing a non-zero value to CLKS[1:0]. This procedure guarantees that the FTM will not capture spurious inputs edges in its input modes while CLKS[1:0] is 00b.

Peripheral Clocking - FTM



NOTE

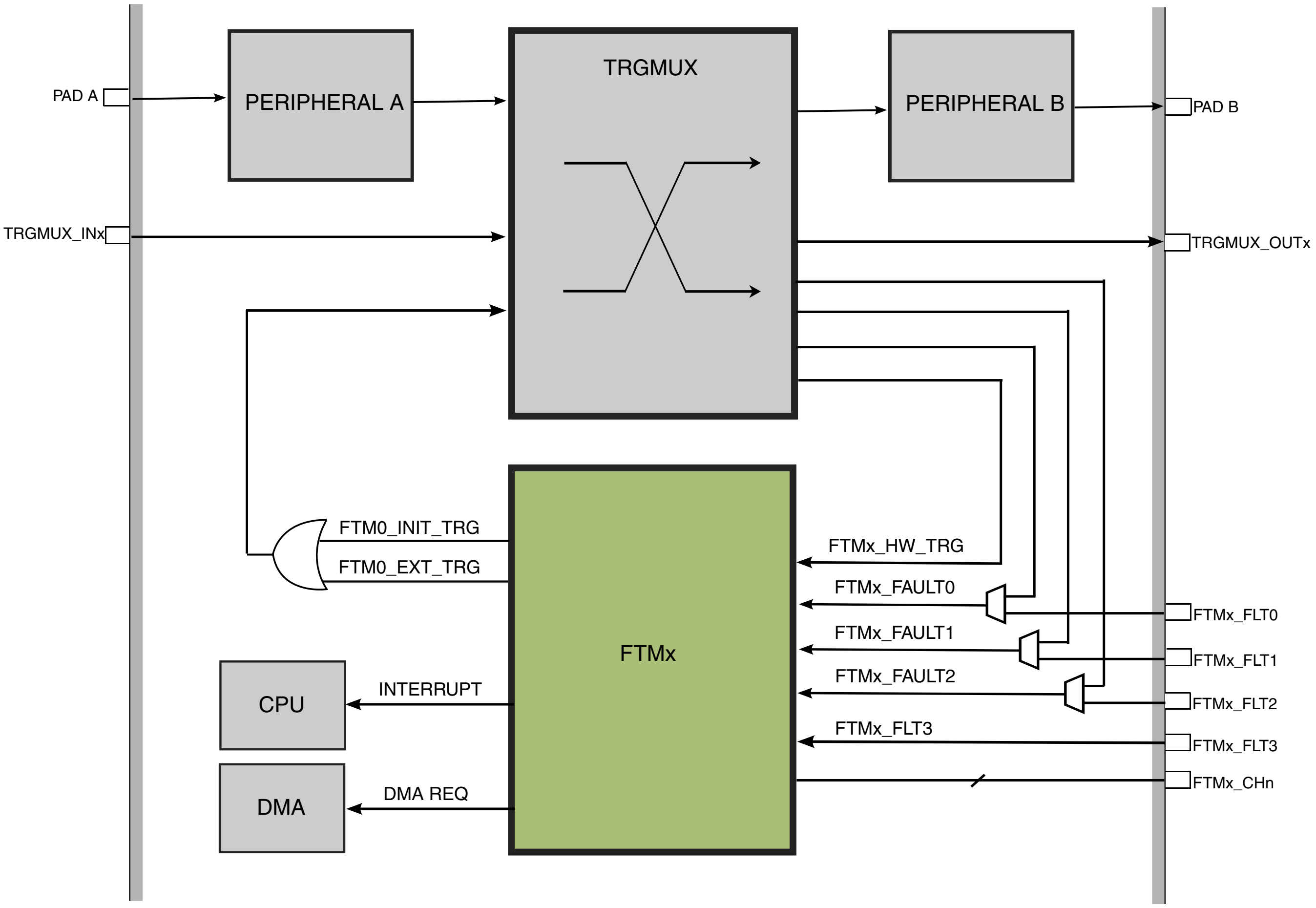
Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM system clock frequency (SYS_CLK).

NOTE

The external clock are synchronized by FTM system clock (SYS_CLK). Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

34.1.3 Inter-connectivity Information

The FTM inter-connectivity is shown in the following diagram.



NOTE

The diagram only shows some possible fault input sources. For the actual connections of each FTM, see FTM Fault Detection Inputs for details.

34.1.3.1 FTM Fault Detection Inputs

The following fault detection input options for the FTM modules are selected via the SIM_FTMOPT0 register. The external pin option is selected by default.

- FTM0 FAULT0 = FTM0_FLT0 pin or TRGMUX output
- FTM0 FAULT1 = FTM0_FLT1 pin or TRGMUX output
- FTM0 FAULT2 = FTM0_FLT2 pin or TRGMUX output
- FTM0 FAULT3 = FTM0_FLT3 pin

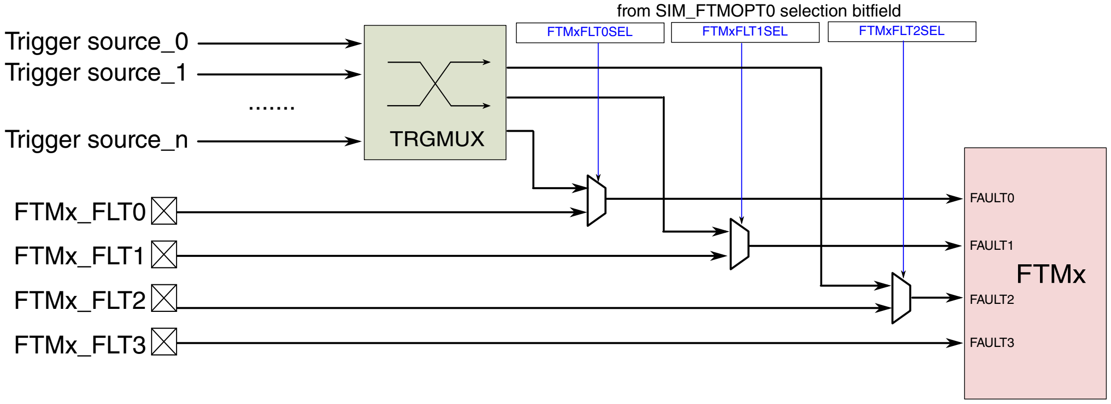


Figure 34-1. FTM0 Fault Detection Inputs

34.1.3.2 FTM Hardware Triggers and Synchronization

The FlexTimer support external hardware trigger input which can be used for timer dynamic synchronization between multiple FlexTimers or counter reset. The FlexTimer hardware trigger are implemented as following.

FTM0:

- FTM0 hardware trigger 0 = TRGMUX trigger output
- FTM0 hardware trigger 1 = SIM_FTMOPT1[FTM0SYNCBIT]
- FTM0 hardware trigger 2 = FTM0_FLT0 pin

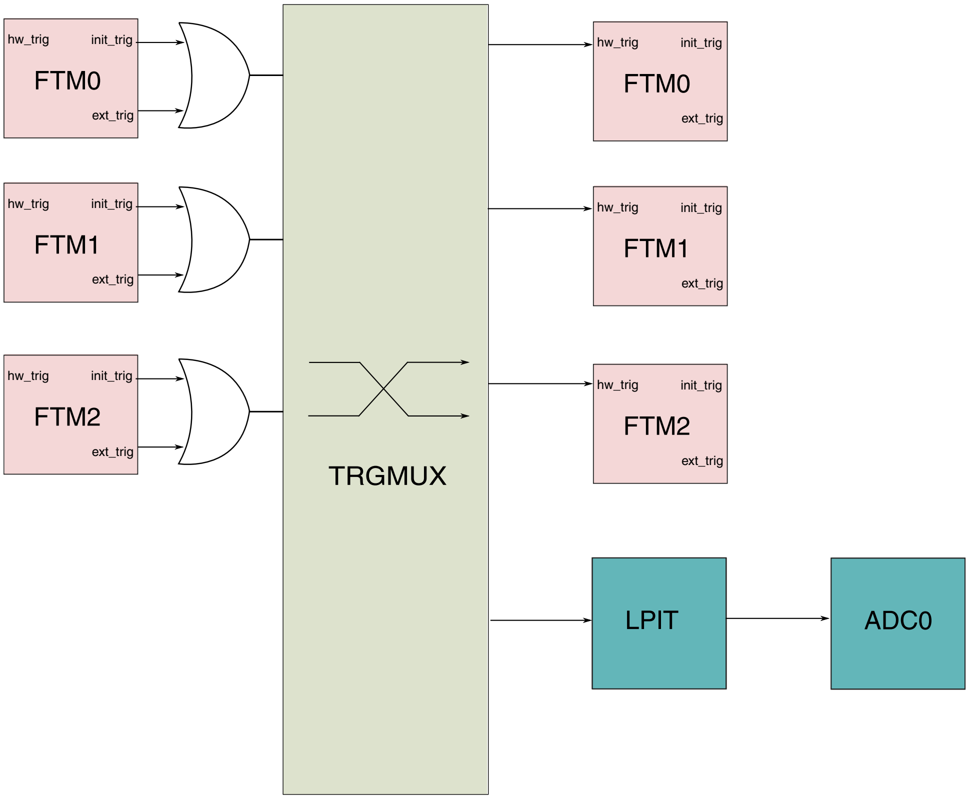
FTM1:

- FTM1 hardware trigger 0 = TRGMUX trigger output
- FTM1 hardware trigger 1 = SIM_FTMOPT1[FTM1SYNCBIT]

FTM2:

- FTM2 hardware trigger 0 = TRGMUX trigger output
- FTM2 hardware trigger 1 = SIM_FTMOPT1[FTM2SYNCBIT]

The hardware trigger source can be from many other modules via TRGMUX, like LPIT, Low Power Timer, CMP, etc. It also supports FlexTimer's self trigger outputs, e.g. counter initialization trigger (init_trig) and channel match trigger (ext_trig), through the flexible TRGMUX module.



The FlexTimer trigger outputs are also usually used as trigger source by other modules, for example, the above diagram shows a case of triggering ADC. See ADC Trigger Sources in ADC chapter for details.

34.1.3.3 FTM Input Capture Options

The following channel 0 input capture source options are selected via SIM_FTMOPT1. The external pin option is selected by default.

- FTM1 channel 0 input capture = FTM1_CH0 pin or CMP0 output
- FTM2 channel 0 input capture = FTM2_CH0 pin or CMP0 output
- FTM2 channel 1 input capture = FTM2_CH1 pin or exclusive OR of FTM2_CH0, FTM2_CH1, and FTM1_CH1. See FTM Hall sensor support.

34.2 Introduction

The FlexTimer module (FTM) is a two-to-eight channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

34.2.1 Features

The FTM features include:

- FTM source clock is selectable
  - Source clock can be the FTM input clock, the fixed frequency clock, or an external clock
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the FTM input clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit counter
  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In Input Capture mode:
  - The capture can occur on rising edges, falling edges or both edges
  - An input filter can be selected for some channels.
- In Output Compare mode the output signal can be set, cleared, or toggled on match
- All channels can be configured for center-aligned PWM mode
- Each pair of channels can be combined to generate a PWM signal with independent control of both edges of PWM signal

- The FTM channels can operate as pairs with equal outputs, pairs with complementary outputs, or independent channels with independent outputs
- The deadtime insertion is available for each complementary pair
- Generation of match triggers
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- The polarity of each channel is configurable
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when the fault condition is detected
- The generation of an interrupt when a register reload point occurs
- Synchronized loading of write buffered FTM registers
- Half cycle and Full cycle register reload capacity
- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input capture mode
- Direct access to input pin states
- Dual edge capture for pulse and period width measurement
- The FTM channels can be selected to generate a trigger pulse on channel output instead of a PWM
- Dithering capability to simulate fine edge control for both PWM period or PWM duty cycle

34.2.2 Modes of operation

When the chip is in an active Debug mode, the FTM temporarily suspends all counting until the chip returns to normal user operating mode. During Stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During Wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the chip from Wait mode, the power can then be saved by disabling FTM functions before entering Wait mode.

34.2.3 Block Diagram

The FTM uses one input/output (I/O) pin per channel, CHn (FTM channel (n)) where n is the channel number (0–7).

NOTE

> The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

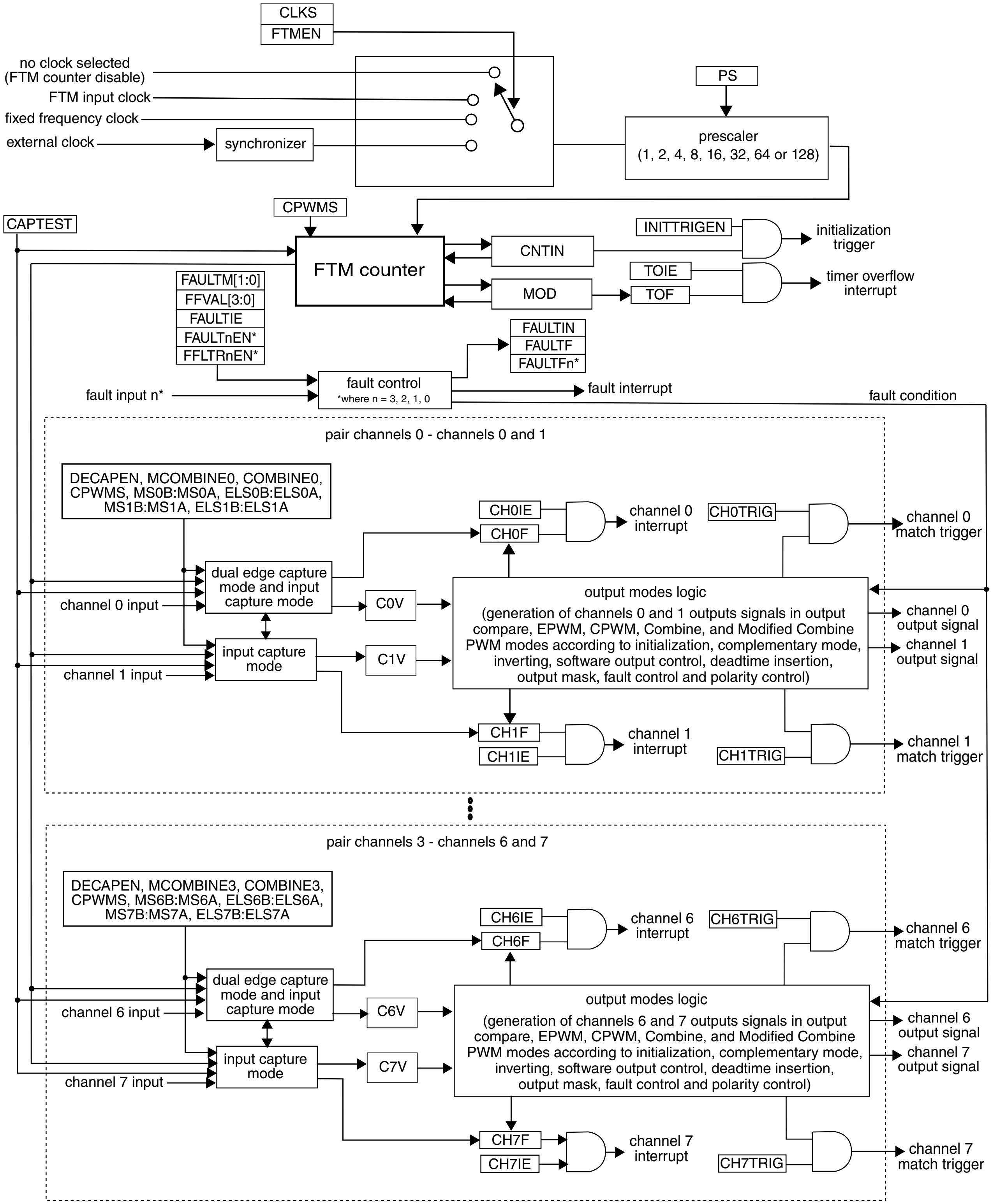

CLKS
FTMEN
no clock selected
(FTM counter disable)
FTM input clock
fixed frequency clock
external clock
synchronizer
PS
prescaler
(1, 2, 4, 8, 16, 32, 64 or 128)
CPWMS
CAPTEST
FTM counter
CNTIN
MOD
INITTRIGEN
initialization trigger
TOIE
TOF
timer overflow interrupt
FAULTM[1:0]
FFVAL[3:0]
FAULTIE
FAULTnEN*
FFLTRnEN*
FAULTIN
FAULTF
FAULTFn*
fault input n*
fault control
*where n = 3, 2, 1, 0
fault interrupt
fault condition
pair channels 0 - channels 0 and 1
DECAPEN, MCOMBINE0, COMBINE0, CPWMS, MS0B:MS0A, ELS0B:ELS0A, MS1B:MS1A, ELS1B:ELS1A
CH0IE
CH0F
channel 0 interrupt
CH0TRIG
channel 0 match trigger
dual edge capture mode and input capture mode
channel 0 input
C0V
output modes logic
(generation of channels 0 and 1 outputs signals in output compare, EPWM, CPWM, Combine, and Modified Combine PWM modes according to initialization, complementary mode, inverting, software output control, deadtime insertion, output mask, fault control and polarity control)
channel 0 output signal
channel 1 output signal
input capture mode
channel 1 input
C1V
CH1F
CH1IE
channel 1 interrupt
CH1TRIG
channel 1 match trigger
pair channels 3 - channels 6 and 7
DECAPEN, MCOMBINE3, COMBINE3, CPWMS, MS6B:MS6A, ELS6B:ELS6A, MS7B:MS7A, ELS7B:ELS7A
CH6IE
CH6F
channel 6 interrupt
CH6TRIG
channel 6 match trigger
dual edge capture mode and input capture mode
channel 6 input
C6V
output modes logic
(generation of channels 6 and 7 outputs signals in output compare, EPWM, CPWM, Combine, and Modified Combine PWM modes according to initialization, complementary mode, inverting, software output control, deadtime insertion, output mask, fault control and polarity control)
channel 6 output signal
channel 7 output signal
input capture mode
channel 7 input
C7V
CH7F
CH7IE
channel 7 interrupt
CH7TRIG
channel 7 match trigger

Figure 34-2. FTM Block Diagram

34.3 FTM signal descriptions

Table 34-2 shows the user-accessible signals for the FTM.

Table 34-2. FTM signal descriptions

Signal	Description	I/O	Function
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I	The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of FTM input clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.
CHn	FTM channel (n), where n can be 7-0	I/O	Each FTM 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.
FAULTj	Fault input (j), where j can be 3-0	I	The fault input signals are used to control the CHn channel output state. If a fault is detected, the FAULTj signal is asserted and the channel output is put in a safe state. The behavior of the fault logic is defined by the FAULTM[1:0] control bits in the MODE register and FAULTEN bit in the COMBINE register. Note that each FAULTj input may affect all channels selectively since FAULTM[1:0] and FAULTEN control bits are defined for each pair of channels. Because there are several FAULTj inputs, maximum of 4 for the FTM module, each one of these inputs is activated by the FAULTjEN bit in the FLTCTRL register.

34.4 Memory map and register definition

34.4.1 Memory map

This section presents a high-level summary of the FTM registers and how they are mapped.

The registers and bits of an unavailable function in the FTM remain in the memory map and in the reset value, but they have no active function.

NOTE

> The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance.

34.4.2 Register descriptions

Accesses to reserved addresses result in transfer errors. Registers for absent channels are considered reserved. Double buffered register writes must be done using 32-bit operations.

34.4.3 FTM register descriptions

34.4.3.1 FTM memory map

FTM0 base address: 4003_8000h

FTM1 base address: 4003_9000h

FTM2 base address: 4003_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Status And Control (SC)	32	RW	0000_0000h
4h	Counter (CNT)	32	RW	0000_0000h
8h	Modulo (MOD)	32	RW	0000_0000h
Ch	Channel (n) Status And Control (C0SC)	32	RW	0000_0000h
10h	Channel (n) Value (C0V)	32	RW	0000_0000h
14h	Channel (n) Status And Control (C1SC)	32	RW	0000_0000h
18h	Channel (n) Value (C1V)	32	RW	0000_0000h
1Ch	Channel (n) Status And Control (C2SC)	32	RW	0000_0000h
20h	Channel (n) Value (C2V)	32	RW	0000_0000h
24h	Channel (n) Status And Control (C3SC)	32	RW	0000_0000h
28h	Channel (n) Value (C3V)	32	RW	0000_0000h
2Ch	Channel (n) Status And Control (C4SC)	32	RW	0000_0000h
30h	Channel (n) Value (C4V)	32	RW	0000_0000h
34h	Channel (n) Status And Control (C5SC)	32	RW	0000_0000h
38h	Channel (n) Value (C5V)	32	RW	0000_0000h
3Ch	Channel (n) Status And Control (C6SC)	32	RW	0000_0000h
40h	Channel (n) Value (C6V)	32	RW	0000_0000h
44h	Channel (n) Status And Control (C7SC)	32	RW	0000_0000h
48h	Channel (n) Value (C7V)	32	RW	0000_0000h
4Ch	Counter Initial Value (CNTIN)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
50h	Capture And Compare Status (STATUS)	32	RW	0000_0000h
54h	Features Mode Selection (MODE)	32	RW	0000_0004h
58h	Synchronization (SYNC)	32	RW	0000_0000h
5Ch	Initial State For Channels Output (OUTINIT)	32	RW	0000_0000h
60h	Output Mask (OUTMASK)	32	RW	0000_0000h
64h	Function For Linked Channels (COMBINE)	32	RW	0000_0000h
68h	Deadtime Configuration (DEADTIME)	32	RW	0000_0000h
6Ch	FTM External Trigger (EXTTRIG)	32	RW	0000_0000h
70h	Channels Polarity (POL)	32	RW	0000_0000h
74h	Fault Mode Status (FMS)	32	RW	0000_0000h
78h	Input Capture Filter Control (FILTER)	32	RW	0000_0000h
7Ch	Fault Control (FLTCTRL)	32	RW	0000_0000h
84h	Configuration (CONF)	32	RW	0000_0000h
88h	FTM Fault Input Polarity (FLTPOL)	32	RW	0000_0000h
8Ch	Synchronization Configuration (SYNCONF)	32	RW	0000_0000h
90h	FTM Inverting Control (INVCTRL)	32	RW	0000_0000h
94h	FTM Software Output Control (SWOCTRL)	32	RW	0000_0000h
98h	FTM PWM Load (PWMLOAD)	32	RW	0000_0000h
9Ch	Half Cycle Register (HCR)	32	RW	0000_0000h
200h	Mirror of Modulo Value (MOD_MIRROR)	32	RW	0000_0000h
204h - 220h	Mirror of Channel (n) Match Value (C0V_MIRROR - C7V_MIRROR)	32	RW	See section

34.4.3.2 Status And Control (SC)

34.4.3.2.1 Offset

Register	Offset
SC	0h

34.4.3.2.2 Function

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, and prescaler factor.

This register also contains the output enable control bits and the reload opportunity flag control.

These controls relate to all channels within this module.

34.4.3.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				PWMEN7	PWMEN6	PWMEN5	PWMEN4	PWMEN3	PWMEN2	PWMEN1	PWMEN0
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						TOF	TOIE	RF	RIE	CPWMS	CLKS		PS		
W							0		0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.2.4 Fields

Field	Function
31-28 —	Reserved
27-24 —	Reserved
23 PWMEN7	Channel 7 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. **NOTE:** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>FTM0</td><td>SC</td><td>—</td></tr><tr><td>FTM1</td><td>—</td><td>SC</td></tr><tr><td>FTM2</td><td>—</td><td>SC</td></tr></table> 0b - Channel output port is disabled. 1b - Channel output port is enabled.
22 PWMEN6	Channel 6 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. **NOTE:** This field is not supported in every instance. The following table includes only supported registers.

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SC</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SC</td></tr>
<tr><td>FTM2</td><td>—</td><td>SC</td></tr>
</table>
0b - Channel output port is disabled.
1b - Channel output port is enabled.</td></tr>
<tr><td>21
PWMEN5</td><td>Channel 5 PWM enable bit
This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SC</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SC</td></tr>
<tr><td>FTM2</td><td>—</td><td>SC</td></tr>
</table>
0b - Channel output port is disabled.
1b - Channel output port is enabled.</td></tr>
<tr><td>20
PWMEN4</td><td>Channel 4 PWM enable bit
This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SC</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SC</td></tr>
<tr><td>FTM2</td><td>—</td><td>SC</td></tr>
</table>
0b - Channel output port is disabled.
1b - Channel output port is enabled.</td></tr>
<tr><td>19
PWMEN3</td><td>Channel 3 PWM enable bit
This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used.
0b - Channel output port is disabled.
1b - Channel output port is enabled.</td></tr>
<tr><td>18
PWMEN2</td><td>Channel 2 PWM enable bit
This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used.
0b - Channel output port is disabled.</td></tr>
</table>

Table continues on the next page...

Field	Function
	1b - Channel output port is enabled.
17 PWMEN1	Channel 1 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0b - Channel output port is disabled. 1b - Channel output port is enabled.
16 PWMEN0	Channel 0 PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0b - Channel output port is disabled. 1b - Channel output port is enabled.
15-10 —	Reserved
9 TOF	Timer Overflow Flag Set by hardware when the FTM counter passes the value in the MOD register. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect. If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case, a TOF interrupt request is not lost due to the clearing sequence for a previous TOF. 0b - FTM counter has not overflowed. 1b - FTM counter has overflowed.
8 TOIE	Timer Overflow Interrupt Enable Enables FTM overflow interrupts. 0b - Disable TOF interrupts. Use software polling. 1b - Enable TOF interrupts. An interrupt is generated when TOF equals one.
7 RF	Reload Flag The RF bit is set at each selected reload point. See Reload Points. The RF bit is cleared by reading the SC register while RF is set and then writing a 0 to RF bit. Writing 1 to RF has no effect. If another selected reload point happens between the read and write operations, the write operation has no effect; therefore, RF remains set. 0b - A selected reload point did not happen. 1b - A selected reload point happened.
6 RIE	Reload Point Interrupt Enable Enables the reload point interrupt. 0b - Reload point interrupt is disabled. 1b - Reload point interrupt is enabled.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the FTM to operate in Up-Down Counting mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - FTM counter operates in Up Counting mode. 1b - FTM counter operates in Up-Down Counting mode.
4-3 CLKS	Clock Source Selection Selects one of the three FTM counter clock sources. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00b - No clock selected. This in effect disables the FTM counter.

Table continues on the next page...

Field	Function
	01b - FTM input clock 10b - Fixed frequency clock 11b - External clock
2-0 PS	Prescale Factor Selection Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next FTM input clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

34.4.3.3 Counter (CNT)

34.4.3.3.1 Offset

Register	Offset
CNT	4h

34.4.3.3.2 Function

The CNT register contains the FTM counter value.

Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value, CNTIN.

34.4.3.3.3 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 W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.3.4 Fields

Field	Function
31-16 —	Reserved
15-0 COUNT	Counter Value

34.4.3.4 Modulo (MOD)

34.4.3.4.1 Offset

Register	Offset
MOD	8h

34.4.3.4.2 Function

The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock cycle, and the next value of FTM counter depends on the selected counting method; see Counter.

Writes to the MOD register are done on its write buffer. The MOD register is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to SC register resets manually this write coherency mechanism.

Initialize the FTM counter, by writing to CNT, before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

34.4.3.4.3 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 W	MOD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.4.4 Fields

Field	Function
31-16 —	Reserved
15-0 MOD	MOD Modulo Value

34.4.3.5 Channel (n) Status And Control (C0SC - C7SC)

34.4.3.5.1 Offset

For a = 0 to 7:

Register	Offset
CaSC	Ch + (a × 8h)

34.4.3.5.2 Function

CnSC contains channel (n) status bits and control bits that select the channel (n) mode and its functionality.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
FTM0	C0SC–C7SC	—
FTM1	C0SC–C3SC	C4SC–C7SC
FTM2	C0SC–C3SC	C4SC–C7SC

34.4.3.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See #d6159e2238a1310.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					CHOV	CHIS	TRIGMODE	CHF	CHIE	MSB	MSA	ELSB	ELSA	ICRST	DMA
W									0							
Reset	See #d6159e2238a1310.															

34.4.3.5.4 Register reset values

Register	Reset value
C0SC–C3SC	FTM0–FTM2: 0000_0000h
C4SC–C7SC	0000_0000h

34.4.3.5.5 Fields

Field	Function
31-11 —	Reserved
10 CHOV	Channel (n) Output Value The CHOV bit has the final value of the channel (n) output. **NOTE:** The CHOV bit should be ignored when the channel (n) is not in an output mode. 0b - The channel (n) output is zero. 1b - The channel (n) output is one.
9 CHIS	Channel (n) Input State

Table continues on the next page...

Field	Function
	The CHIS bit has the value of the channel (n) input after the double-sampling or the filtering (if the channel (n) filter is enabled) both them are inside the FTM. **NOTE:** The CHIS bit should be ignored when the channel (n) is not in an input mode. **NOTE:** When the pair channels is on dual edge mode, the channel (n+1) CHIS bit is the channel (n+1) input value and not the channel (n) input value (this signal is the input signal used by the dual edge mode). 0b - The channel (n) input is zero. 1b - The channel (n) input is one.
8 TRIGMODE	Trigger mode control This bit controls the trigger generation on FTM channel outputs. This mode is allowed only if when FTM channel is configured to EPWM or CPWM modes. If a match in the channel occurs, a trigger pulse with one FTM clock cycle width will be generated in the channel output. See Channel trigger output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Channel outputs will generate the normal PWM outputs without generating a pulse. 1b - If a match in the channel occurs, a trigger generation on channel output will happen. The trigger pulse width has one FTM clock cycle.
7 CHF	Channel (n) Flag Set by hardware when an event occurs on the channel (n). CHF is cleared by reading the CnSC register while CHF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect. If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF. 0b - No channel (n) event has occurred. 1b - A channel (n) event has occurred.
6 CHIE	Channel (n) Interrupt Enable Enables channel (n) interrupt. 0b - Disable channel (n) interrupt. Use software polling. 1b - Enable channel (n) interrupt.
5 MSB	Channel (n) Mode Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
4 MSA	Channel (n) Mode Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
3 ELSB	Channel (n) Edge or Level Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
2 ELSA	Channel (n) Edge or Level Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 ICRST	FTM counter reset by the selected input capture event. FTM counter reset is driven by the selected event of the channel (n) in the Input Capture mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - FTM counter is not reset when the selected channel (n) input event is detected.

Table continues on the next page...

Field	Function
	1b - FTM counter is reset when the selected channel (n) input event is detected.
0 DMA	DMA Enable Enables DMA transfers for the channel. 0b - Disable DMA transfers. 1b - Enable DMA transfers.

34.4.3.6 Channel (n) Value (C0V - C7V)

34.4.3.6.1 Offset

For a = 0 to 7:

Register	Offset
CaV	10h + (a × 8h)

34.4.3.6.2 Function

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.

In Input Capture , Capture Test, and Dual Edge Capture modes, any write to a CnV register is ignored.

In output modes, writes to the CnV register are done on its write buffer. The CnV register is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to CnSC register resets manually this write coherency mechanism.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
FTM0	C0V–C7V	—
FTM1	C0V–C3V	C4V–C7V
FTM2	C0V–C3V	C4V–C7V

34.4.3.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See #d6159e2721a1310.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VAL															
Reset	See #d6159e2721a1310.															

34.4.3.6.4 Register reset values

Register	Reset value
C0V–C3V	FTM0–FTM2: 0000_0000h
C4V–C7V	0000_0000h

34.4.3.6.5 Fields

Field	Function
31-16 —	Reserved
15-0 VAL	Channel Value Captured FTM counter value of the input modes or the match value for the output modes

34.4.3.7 Counter Initial Value (CNTIN)

34.4.3.7.1 Offset

Register	Offset
CNTIN	4Ch

34.4.3.7.2 Function

The Counter Initial Value register contains the initial value for the FTM counter.

Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to Registers updated from write buffers.

When the FTM clock is initially selected, by writing a non-zero value to the CLKS bits, the FTM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the FTM clock, write the new value to the CNTIN register and then initialize the FTM counter by writing any value to the CNT register.

34.4.3.7.3 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 W	INIT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.7.4 Fields

Field	Function
31-16 —	Reserved
15-0 INIT	INIT Initial Value Of The FTM Counter

34.4.3.8 Capture And Compare Status (STATUS)

34.4.3.8.1 Offset

Register	Offset
STATUS	50h

34.4.3.8.2 Function

The STATUS register contains a copy of the status flag CHF bit in CnSC for each FTM channel for software convenience.

Each CHF bit in STATUS is a mirror of CHF bit in CnSC. All CHF bits can be checked using only one read of STATUS. All CHF bits can be cleared by reading STATUS followed by writing 0x00 to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. CHF is cleared by reading STATUS while CHF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect.

If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case, a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

34.4.3.8.3 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								CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.8.4 Fields

Field	Function
31-8 —	Reserved
7 CH7F	Channel 7 Flag See the register description. **NOTE:** This field is not supported in every instance. The following table includes only supported registers.

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>STATUS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>STATUS</td></tr>
<tr><td>FTM2</td><td>—</td><td>STATUS</td></tr>
</table>
0b - No channel event has occurred.
1b - A channel event has occurred.
</td></tr>
<tr><td>6
CH6F</td><td>
Channel 6 Flag
See the register description.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>STATUS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>STATUS</td></tr>
<tr><td>FTM2</td><td>—</td><td>STATUS</td></tr>
</table>
0b - No channel event has occurred.
1b - A channel event has occurred.
</td></tr>
<tr><td>5
CH5F</td><td>
Channel 5 Flag
See the register description.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>STATUS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>STATUS</td></tr>
<tr><td>FTM2</td><td>—</td><td>STATUS</td></tr>
</table>
0b - No channel event has occurred.
1b - A channel event has occurred.
</td></tr>
<tr><td>4
CH4F</td><td>
Channel 4 Flag
See the register description.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>STATUS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>STATUS</td></tr>
<tr><td>FTM2</td><td>—</td><td>STATUS</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Field	Function
	0b - No channel event has occurred. 1b - A channel event has occurred.
3 CH3F	Channel 3 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
2 CH2F	Channel 2 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
1 CH1F	Channel 1 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.

34.4.3.9 Features Mode Selection (MODE)

34.4.3.9.1 Offset

Register	Offset
MODE	54h

34.4.3.9.2 Function

This register contains the global enable bit for FTM-specific features and the control bits used to configure:

- Fault control mode and interrupt
- Capture Test mode
- PWM synchronization
- Write protection
- Channel output initialization

These controls relate to all channels within this module.

34.4.3.9.3 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								FAULTIE	FAULTM		CAPTEST	PWMSYNC	WPDIS	0	FTMEN
W															INIT	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

34.4.3.9.4 Fields

Field	Function
31-8 —	Reserved
7 FAULTIE	Fault Interrupt Enable Enables the generation of an interrupt when a fault is detected by FTM and the FTM fault control is enabled. **NOTE:** This field is not supported in every instance. The following table includes only supported registers. Instance / Field supported in / Field not supported in: FTM0 / MODE / — FTM1 / — / MODE FTM2 / — / MODE 0b - Fault control interrupt is disabled. 1b - Fault control interrupt is enabled.
6-5 FAULTM	Fault Control Mode Defines the FTM fault control mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. **NOTE:** This field is not supported in every instance. The following table includes only supported registers. Instance / Field supported in / Field not supported in: FTM0 / MODE / — FTM1 / — / MODE

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td><table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>FTM2</td><td>—</td><td>MODE</td></tr></table>
00b - Fault control is disabled for all channels.
01b - Fault control is enabled for even channels only (channels 0, 2, 4, and 6), and the selected mode is the manual fault clearing.
10b - Fault control is enabled for all channels, and the selected mode is the manual fault clearing.
11b - Fault control is enabled for all channels, and the selected mode is the automatic fault clearing.</td></tr>
<tr><td>4
CAPTEST</td><td>Capture Test Mode Enable
Enables the capture test mode.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
0b - Capture test mode is disabled.
1b - Capture test mode is enabled.</td></tr>
<tr><td>3
PWMSYNC</td><td>PWM Synchronization Mode
Selects which triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. See PWM synchronization. The PWMSYNC bit configures the synchronization when SYNCMODE is 0.
0b - No restrictions. Software and hardware triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization.
1b - Software trigger can only be used by MOD and CnV synchronization, and hardware triggers can only be used by OUTMASK and FTM counter synchronization.</td></tr>
<tr><td>2
WPDIS</td><td>Write Protection Disable
When write protection is enabled (WPDIS = 0), write protected bits cannot be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN. WPDIS is set when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPDIS has no effect.
0b - Write protection is enabled.
1b - Write protection is disabled.</td></tr>
<tr><td>1
INIT</td><td>Initialize The Channels Output
When a 1 is written to INIT bit the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect.
The INIT bit is always read as 0.</td></tr>
<tr><td>0
FTMEN</td><td>FTM Enable
This field is write protected. It can be written only when MODE[WPDIS] = 1.
0b - TPM compatibility. Free running counter and synchronization compatible with TPM.
1b - Free running counter and synchronization are different from TPM behavior.</td></tr>
</table>

34.4.3.10 Synchronization (SYNC)

34.4.3.10.1 Offset

Register	Offset
SYNC	58h

34.4.3.10.2 Function

This register configures the PWM synchronization.

A synchronization event can perform the synchronized update of MOD, CnV, and OUTMASK registers with the value of their write buffer and the FTM counter initialization.

NOTE

> The software trigger, SWSYNC bit, and hardware triggers TRIG0, TRIG1, and TRIG2 bits have a potential conflict if used together when SYNCMODE = 0. Use only hardware or software triggers but not both at the same time, otherwise unpredictable behavior is likely to happen.
>
> The selection of the loading point, CNTMAX and CNTMIN bits, is intended to provide the update of MOD, CNTIN, and CnV registers across all enabled channels simultaneously. The use of the loading point selection together with SYNCMODE = 0 and hardware trigger selection, TRIG0, TRIG1, or TRIG2 bits, is likely to result in unpredictable behavior.
>
> The synchronization event selection also depends on the PWMSYNC (MODE register) and SYNCMODE (SYNCONF register) bits. See PWM synchronization.

34.4.3.10.3 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								SWSYNC	TRIG2	TRIG1	TRIG0	SYNCHOM	REINIT	CNTMAX	CNTMIN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.10.4 Fields

Field	Function
31-8 —	Reserved
7 SWSYNC	PWM Synchronization Software Trigger Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit. 0b - Software trigger is not selected. 1b - Software trigger is selected.
6 TRIG2	PWM Synchronization Hardware Trigger 2 Enables hardware trigger 2 to the PWM synchronization. Hardware trigger 2 happens when a rising edge is detected at the trigger 2 input signal. 0b - Trigger is disabled. 1b - Trigger is enabled.
5 TRIG1	PWM Synchronization Hardware Trigger 1 Enables hardware trigger 1 to the PWM synchronization. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal. 0b - Trigger is disabled. 1b - Trigger is enabled.
4 TRIG0	PWM Synchronization Hardware Trigger 0 Enables hardware trigger 0 to the PWM synchronization. Hardware trigger 0 occurs when a rising edge is detected at the trigger 0 input signal. 0b - Trigger is disabled. 1b - Trigger is enabled.
3 SYNCHOM	Output Mask Synchronization Selects when the OUTMASK register is updated with the value of its buffer. 0b - OUTMASK register is updated with the value of its buffer in all rising edges of the FTM input clock. 1b - OUTMASK register is updated with the value of its buffer only by the PWM synchronization.
2	FTM Counter Reinitialization by Synchronization

Table continues on the next page...

Field	Function
REINIT	Determines if the FTM counter is reinitialized when the selected trigger for the synchronization is detected (FTM counter synchronization). The REINIT bit configures the synchronization when SYNCMODE is zero. 0b - FTM counter continues to count normally. 1b - FTM counter is updated with its initial value when the selected trigger is detected.
1 CNTMAX	Maximum Loading Point Enable Selects the maximum loading point to PWM synchronization (Synchronization Points). If CNTMAX is 1, the selected loading point is when the FTM counter reaches its maximum value (MOD register). 0b - The maximum loading point is disabled. 1b - The maximum loading point is enabled.
0 CNTMIN	Minimum Loading Point Enable Selects the minimum loading point to PWM synchronization (Synchronization Points). If CNTMIN is 1, the selected loading point is when the FTM counter reaches its minimum value (CNTIN register). 0b - The minimum loading point is disabled. 1b - The minimum loading point is enabled.

34.4.3.11 Initial State For Channels Output (OUTINIT)

34.4.3.11.1 Offset

Register	Offset
OUTINIT	5Ch

34.4.3.11.2 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								CH7OI	CH6OI	CH5OI	CH4OI	CH3OI	CH2OI	CH1OI	CH0OI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.11.3 Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-8
—</td><td>Reserved</td></tr>
<tr><td>7
CH7OI</td><td>Channel 7 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTINIT</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTINIT</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTINIT</td></tr>
</table>
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>6
CH6OI</td><td>Channel 6 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTINIT</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTINIT</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTINIT</td></tr>
</table>
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>5
CH5OI</td><td>Channel 5 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTINIT</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTINIT</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTINIT</td></tr>
</table>
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>4</td><td>Channel 4 Output Initialization Value</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>CH4OI</td><td>Selects the value that is forced into the channel output when the initialization occurs.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTINIT</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTINIT</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTINIT</td></tr>
</table>
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>3
CH3OI</td><td>Channel 3 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>2
CH2OI</td><td>Channel 2 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>1
CH1OI</td><td>Channel 1 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
<tr><td>0
CH0OI</td><td>Channel 0 Output Initialization Value
Selects the value that is forced into the channel output when the initialization occurs.
0b - The initialization value is 0.
1b - The initialization value is 1.</td></tr>
</table>

34.4.3.12 Output Mask (OUTMASK)

34.4.3.12.1 Offset

Register	Offset
OUTMASK	60h

34.4.3.12.2 Function

This register provides a mask for each FTM channel. The mask of a channel determines if its output responds, that is, it is masked or not, when a match occurs. This feature is used for BLDC control where the PWM signal is presented to an electric motor at specific times to provide electronic commutation.

Any write to the OUTMASK register, stores the value in its write buffer. The register is updated with the value of its write buffer according to PWM synchronization.

Output Mask bits must not be set for trigger mode.

34.4.3.12.3 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								CH7OM	CH6OM	CH5OM	CH4OM	CH3OM	CH2OM	CH1OM	CH0OM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.12.4 Fields

Field	Function
31-8 —	Reserved
7 CH7OM	Channel 7 Output Mask Defines if the channel output is masked or unmasked. **NOTE:** This field is not supported in every instance. The following table includes only supported registers. 0b - Channel output is not masked. It continues to operate normally. 1b - Channel output is masked. It is forced to its inactive state.

Instance	Field supported in	Field not supported in
FTM0	OUTMASK	—
FTM1	—	OUTMASK
FTM2	—	OUTMASK

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>6
CH6OM</td><td>Channel 6 Output Mask
Defines if the channel output is masked or unmasked.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTMASK</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTMASK</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTMASK</td></tr>
</table>
0b - Channel output is not masked. It continues to operate normally.
1b - Channel output is masked. It is forced to its inactive state.</td></tr>
<tr><td>5
CH5OM</td><td>Channel 5 Output Mask
Defines if the channel output is masked or unmasked.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTMASK</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTMASK</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTMASK</td></tr>
</table>
0b - Channel output is not masked. It continues to operate normally.
1b - Channel output is masked. It is forced to its inactive state.</td></tr>
<tr><td>4
CH4OM</td><td>Channel 4 Output Mask
Defines if the channel output is masked or unmasked.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>OUTMASK</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>OUTMASK</td></tr>
<tr><td>FTM2</td><td>—</td><td>OUTMASK</td></tr>
</table>
0b - Channel output is not masked. It continues to operate normally.
1b - Channel output is masked. It is forced to its inactive state.</td></tr>
<tr><td>3
CH3OM</td><td>Channel 3 Output Mask
Defines if the channel output is masked or unmasked.
0b - Channel output is not masked. It continues to operate normally.
1b - Channel output is masked. It is forced to its inactive state.</td></tr>
<tr><td>2</td><td>Channel 2 Output Mask</td></tr>
</table>

Table continues on the next page...

Field	Function
CH2OM	Defines if the channel output is masked or unmasked. 0b - Channel output is not masked. It continues to operate normally. 1b - Channel output is masked. It is forced to its inactive state.
1 CH1OM	Channel 1 Output Mask Defines if the channel output is masked or unmasked. 0b - Channel output is not masked. It continues to operate normally. 1b - Channel output is masked. It is forced to its inactive state.
0 CH0OM	Channel 0 Output Mask Defines if the channel output is masked or unmasked. 0b - Channel output is not masked. It continues to operate normally. 1b - Channel output is masked. It is forced to its inactive state.

34.4.3.13 Function For Linked Channels (COMBINE)

34.4.3.13.1 Offset

Register	Offset
COMBINE	64h

34.4.3.13.2 Function

This register contains the configuration bits for each pair of channels.

34.4.3.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCOMBINE3	FAULTEN3	SYNCEN3	DTEN3	DECAP3	DECAPEN3	COMP3	COMBINE3	MCOMBINE2	FAULTEN2	SYNCEN2	DTEN2	DECAP2	DECAPEN2	COMP2	COMBINE2
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	MCOMBINE1	FAULTEN1	SYNCEN1	DTEN1	DECAP1	DECAPEN1	COMP1	COMBINE1	MCOMBINE0	FAULTEN0	SYNCEN0	DTEN0	DECAP0	DECAPEN0	COMP0	COMBINE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.13.4 Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31
MCOMBINE3</td><td>Modified Combine Mode For n = 6
Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table></td></tr>
<tr><td>30
FAULTEN3</td><td>Fault Control Enable For n = 6
Enables the fault control in channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The fault control in this pair of channels is disabled.
1b - The fault control in this pair of channels is enabled.</td></tr>
<tr><td>29
SYNCEN3</td><td>Synchronization Enable For n = 6
Enables PWM synchronization of registers C(n)V and C(n+1)V.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The PWM synchronization in this pair of channels is disabled.
1b - The PWM synchronization in this pair of channels is enabled.</td></tr>
<tr><td>28</td><td>Deadtime Enable For n = 6</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>DTEN3</td><td>Enables the deadtime insertion in the channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The deadtime insertion in this pair of channels is disabled.
1b - The deadtime insertion in this pair of channels is enabled.</td></tr>
<tr><td>27
DECAP3</td><td>Dual Edge Capture Mode Captures For n = 6
Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits.
This field applies only when DECAPEN = 1.
DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The dual edge captures are inactive.
1b - The dual edge captures are active.</td></tr>
<tr><td>26
DECAPEN3</td><td>Dual Edge Capture Mode Enable For n = 6
Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table></td></tr>
<tr><td>25
COMP3</td><td>Complement Of Channel (n) for n = 6
In Complementary mode the channel (n+1) output is the inverse of the channel (n) output.
This field is write protected. It can be written only when MODE[WPDIS] = 1.</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td><b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>

0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output.
1b - The channel (n+1) output is the complement of the channel (n) output.</td></tr>
<tr><td>24
COMBINE3</td><td>Combine Channels For n = 6
Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table></td></tr>
<tr><td>23
MCOMBINE2</td><td>Modified Combine Mode For n = 4
Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table></td></tr>
<tr><td>22
FAULTEN2</td><td>Fault Control Enable For n = 4
Enables the fault control in channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The fault control in this pair of channels is disabled.
1b - The fault control in this pair of channels is enabled.
</td></tr>
<tr><td>21
SYNCEN2</td><td>
Synchronization Enable For n = 4
Enables PWM synchronization of registers C(n)V and C(n+1)V.
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The PWM synchronization in this pair of channels is disabled.
1b - The PWM synchronization in this pair of channels is enabled.
</td></tr>
<tr><td>20
DTEN2</td><td>
Deadtime Enable For n = 4
Enables the deadtime insertion in the channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The deadtime insertion in this pair of channels is disabled.
1b - The deadtime insertion in this pair of channels is enabled.
</td></tr>
<tr><td>19
DECAP2</td><td>
Dual Edge Capture Mode Captures For n = 4
Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits.
This field applies only when DECAPEN = 1.
DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.
NOTE: This field is not supported in every instance. The following table includes only supported registers.
</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The dual edge captures are inactive.
1b - The dual edge captures are active.
</td></tr>
<tr><td>18
DECAPEN2</td><td>
Dual Edge Capture Mode Enable For n = 4
Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
</td></tr>
<tr><td>17
COMP2</td><td>
Complement Of Channel (n) For n = 4
In Complementary mode the channel (n+1) output is the inverse of the channel (n) output.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output.
1b - The channel (n+1) output is the complement of the channel (n) output.
</td></tr>
<tr><td>16
COMBINE2</td><td>
Combine Channels For n = 4
Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td><table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr><tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr><tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr></table></td></tr>
<tr><td>15
MCOMBINE1</td><td>Modified Combine Mode For n = 2
Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.</td></tr>
<tr><td>14
FAULTEN1</td><td>Fault Control Enable For n = 2
Enables the fault control in channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.<table><tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr><tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr><tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr></table>0b - The fault control in this pair of channels is disabled.
1b - The fault control in this pair of channels is enabled.</td></tr>
<tr><td>13
SYNCEN1</td><td>Synchronization Enable For n = 2
Enables PWM synchronization of registers C(n)V and C(n+1)V.
0b - The PWM synchronization in this pair of channels is disabled.
1b - The PWM synchronization in this pair of channels is enabled.</td></tr>
<tr><td>12
DTEN1</td><td>Deadtime Enable For n = 2
Enables the deadtime insertion in the channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
0b - The deadtime insertion in this pair of channels is disabled.
1b - The deadtime insertion in this pair of channels is enabled.</td></tr>
<tr><td>11
DECAP1</td><td>Dual Edge Capture Mode Captures For n = 2
Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits.
This field applies only when DECAPEN = 1.
DECAP bit is cleared automatically by hardware if Dual Edge Capture – One-Shot mode is selected and when the capture of channel (n+1) event is made.
0b - The dual edge captures are inactive.
1b - The dual edge captures are active.</td></tr>
<tr><td>10
DECAPEN1</td><td>Dual Edge Capture Mode Enable For n = 2
Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>9
COMP1</td><td>Complement Of Channel (n) For n = 2
In Complementary mode the channel (n+1) output is the inverse of the channel (n) output.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output.
1b - The channel (n+1) output is the complement of the channel (n) output.</td></tr>
<tr><td>8
COMBINE1</td><td>Combine Channels For n = 2
Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.</td></tr>
<tr><td>7
MCOMBINE0</td><td>Modified Combine Mode For n = 0
Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes.
This field is write protected. It can be written only when MODE[WPDIS] = 1.</td></tr>
<tr><td>6
FAULTEN0</td><td>Fault Control Enable For n = 0
Enables the fault control in channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>COMBINE</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>COMBINE</td></tr>
<tr><td>FTM2</td><td>—</td><td>COMBINE</td></tr>
</table>
0b - The fault control in this pair of channels is disabled.
1b - The fault control in this pair of channels is enabled.</td></tr>
<tr><td>5
SYNCEN0</td><td>Synchronization Enable For n = 0
Enables PWM synchronization of registers C(n)V and C(n+1)V.
0b - The PWM synchronization in this pair of channels is disabled.
1b - The PWM synchronization in this pair of channels is enabled.</td></tr>
<tr><td>4
DTEN0</td><td>Deadtime Enable For n = 0
Enables the deadtime insertion in the channels (n) and (n+1).
This field is write protected. It can be written only when MODE[WPDIS] = 1.
0b - The deadtime insertion in this pair of channels is disabled.
1b - The deadtime insertion in this pair of channels is enabled.</td></tr>
<tr><td>3
DECAP0</td><td>Dual Edge Capture Mode Captures For n = 0
Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits.
This field applies only when DECAPEN = 1.
DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.</td></tr>
</table>

Table continues on the next page...

Field	Function
	0b - The dual edge captures are inactive. 1b - The dual edge captures are active.
2 DECAPEN0	Dual Edge Capture Mode Enable For n = 0 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 COMP0	Complement Of Channel (n) For n = 0 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output. 1b - The channel (n+1) output is the complement of the channel (n) output.
0 COMBINE0	Combine Channels For n = 0 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.

34.4.3.14 Deadtime Configuration (DEADTIME)

34.4.3.14.1 Offset

Register	Offset
DEADTIME	68h

34.4.3.14.2 Function

This register selects the deadtime prescaler and value for all pair of channels.

34.4.3.14.3 Diagram

Bits	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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DTPS		DTVAL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.14.4 Fields

Field	Function
31-20 —	Reserved
19-16 —	Reserved
15-8 —	Reserved
7-6 DTPS	Deadtime Prescaler Value Selects the division factor of the FTM input clock. This prescaled clock is used by the deadtime counter. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0xb - Divide the FTM input clock by 1. 10b - Divide the FTM input clock by 4. 11b - Divide the FTM input clock by 16.
5-0 DTVAL	Deadtime Value Selects the deadtime value. Deadtime insert value = (DTPS × DTVAL). This field is write protected. It can be written only when MODE[WPDIS] = 1.

34.4.3.15 FTM External Trigger (EXTTRIG)

34.4.3.15.1 Offset

Register	Offset
EXTTRIG	6Ch

34.4.3.15.2 Function

This register:

- Indicates when the external trigger was generated
- Enables the generation of a trigger when the FTM counter is equal to its initial value
- Selects which channels are used in the generation of the external trigger

34.4.3.15.3 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						CH7TRIG	CH6TRIG	TRIGF	INITTRIGEN	CH1TRIG	CH0TRIG	CH5TRIG	CH4TRIG	CH3TRIG	CH2TRIG
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.15.4 Fields

Field	Function
31-10 —	Reserved
9 CH7TRIG	Channel 7 External Trigger Enable Enables the generation of the external trigger when FTM counter = C7V. **NOTE:** This field is not supported in every instance. The following table includes only supported registers. | Instance | Field supported in | Field not supported in | | FTM0 | EXTTRIG | — | | FTM1 | — | EXTTRIG | | FTM2 | — | EXTTRIG | 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
8 CH6TRIG	Channel 6 External Trigger Enable Enables the generation of the external trigger when FTM counter = C6V. **NOTE:** This field is not supported in every instance. The following table includes only supported registers. | Instance | Field supported in | Field not supported in | | FTM0 | EXTTRIG | — | | FTM1 | — | EXTTRIG | | FTM2 | — | EXTTRIG |

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - The generation of this external trigger is disabled.
1b - The generation of this external trigger is enabled.</td></tr>
<tr><td>7
TRIGF</td><td>Channel Trigger Flag
Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. Writing a 1 to TRIGF has no effect.
If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF.
0b - No channel trigger was generated.
1b - A channel trigger was generated.</td></tr>
<tr><td>6
INITTRIGEN</td><td>Initialization Trigger Enable
Enables the generation of the trigger when the FTM counter is equal to the CNTIN register.
0b - The generation of initialization trigger is disabled.
1b - The generation of initialization trigger is enabled.</td></tr>
<tr><td>5
CH1TRIG</td><td>Channel 1 External Trigger Enable
Enables the generation of the external trigger when FTM counter = C1V.
0b - The generation of this external trigger is disabled.
1b - The generation of this external trigger is enabled.</td></tr>
<tr><td>4
CH0TRIG</td><td>Channel 0 External Trigger Enable
Enables the generation of the external trigger when FTM counter = C0V.
0b - The generation of this external trigger is disabled.
1b - The generation of this external trigger is enabled.</td></tr>
<tr><td>3
CH5TRIG</td><td>Channel 5 External Trigger Enable
Enables the generation of the external trigger when FTM counter = C5V.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>EXTTRIG</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>EXTTRIG</td></tr>
<tr><td>FTM2</td><td>—</td><td>EXTTRIG</td></tr>
</table>
0b - The generation of this external trigger is disabled.
1b - The generation of this external trigger is enabled.</td></tr>
<tr><td>2
CH4TRIG</td><td>Channel 4 External Trigger Enable
Enables the generation of the external trigger when FTM counter = C4V.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>EXTTRIG</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>EXTTRIG</td></tr>
<tr><td>FTM2</td><td>—</td><td>EXTTRIG</td></tr>
</table></td></tr>
</table>

Table continues on the next page...

Field	Function
	0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
1 CH3TRIG	Channel 3 External Trigger Enable Enables the generation of the external trigger when FTM counter = C3V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
0 CH2TRIG	Channel 2 External Trigger Enable Enables the generation of the external trigger when FTM counter = C2V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.

34.4.3.16 Channels Polarity (POL)

34.4.3.16.1 Offset

Register	Offset
POL	70h

34.4.3.16.2 Function

This register defines the output polarity of the FTM channels.

NOTE

The channel safe value is the value of its POL bit.The channel safe value is driven on the channel output when the fault control is enabled and a fault condition is detected.

34.4.3.16.3 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								POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.16.4 Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-8
—</td><td>Reserved</td></tr>
<tr><td>7
POL7</td><td>Channel 7 Polarity
Defines the polarity of the channel output.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>POL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>POL</td></tr>
<tr><td>FTM2</td><td>—</td><td>POL</td></tr>
</table>
0b - The channel polarity is active high.
1b - The channel polarity is active low.</td></tr>
<tr><td>6
POL6</td><td>Channel 6 Polarity
Defines the polarity of the channel output.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>POL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>POL</td></tr>
<tr><td>FTM2</td><td>—</td><td>POL</td></tr>
</table>
0b - The channel polarity is active high.
1b - The channel polarity is active low.</td></tr>
<tr><td>5
POL5</td><td>Channel 5 Polarity
Defines the polarity of the channel output.
This field is write protected. It can be written only when MODE[WPDIS] = 1.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>POL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>POL</td></tr>
<tr><td>FTM2</td><td>—</td><td>POL</td></tr>
</table></td></tr>
</table>

Table continues on the next page...

Field	Function
	0b - The channel polarity is active high. 1b - The channel polarity is active low.
4 POL4	Channel 4 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. **NOTE:** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
FTM0	POL	—
FTM1	—	POL
FTM2	—	POL

Field	Function
	0b - The channel polarity is active high. 1b - The channel polarity is active low.
3 POL3	Channel 3 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The channel polarity is active high. 1b - The channel polarity is active low.
2 POL2	Channel 2 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The channel polarity is active high. 1b - The channel polarity is active low.
1 POL1	Channel 1 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The channel polarity is active high. 1b - The channel polarity is active low.
0 POL0	Channel 0 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The channel polarity is active high. 1b - The channel polarity is active low.

34.4.3.17 Fault Mode Status (FMS)

34.4.3.17.1 Offset

Register	Offset
FMS	74h

34.4.3.17.2 Function

This register contains:

- the write protection enable bit
- the fault detection flags
- the logic OR of the enabled fault inputs

34.4.3.17.3 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								FAULTF	WPEN	FAULTIN	0	FAULTF3	FAULTF2	FAULTF1	FAULTF0
W									0				0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.17.4 Fields

Field	Function
31-8 —	Reserved
7 FAULTF	Fault Detection Flag Represents the logic OR of the FAULTF bit of each enabled fault input. Clear FAULTF by reading the FMS register while FAULTF is set and then writing a 0 to FAULTF while there is no existing fault condition at the enabled fault inputs. Writing a 1 to FAULTF has no effect. If another fault condition is detected in an enabled fault input before the clearing sequence is completed, the sequence is reset so FAULTF remains set after the clearing sequence is completed for the earlier fault condition. FAULTF is also cleared when FAULTF bit of each enabled fault input is cleared. **NOTE:** This field is not supported in every instance. The following table includes only supported registers.

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>FMS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>FMS</td></tr>
<tr><td>FTM2</td><td>—</td><td>FMS</td></tr>
</table>
0b - No fault condition was detected.
1b - A fault condition was detected.
</td></tr>
<tr><td>6
WPEN</td><td>Write Protection Enable
The WPEN bit is the negation of the WPDIS bit. WPEN is set when 1 is written to it. WPEN is cleared when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPEN has no effect.
0b - Write protection is disabled. Write protected bits can be written.
1b - Write protection is enabled. Write protected bits cannot be written.</td></tr>
<tr><td>5
FAULTIN</td><td>Fault Inputs
Represents the logic OR of the enabled fault inputs after their filter (if their filter is enabled) when fault control is enabled.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>FMS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>FMS</td></tr>
<tr><td>FTM2</td><td>—</td><td>FMS</td></tr>
</table>
0b - The logic OR of the enabled fault inputs is 0.
1b - The logic OR of the enabled fault inputs is 1.
</td></tr>
<tr><td>4
—</td><td>Reserved</td></tr>
<tr><td>3
FAULTF3</td><td>Fault Detection Flag 3
Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.
Clear FAULTF3 by reading the FMS register while FAULTF3 is set and then writing a 0 to FAULTF3 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF3 has no effect. FAULTF3 bit is also cleared when FAULTF bit is cleared.
If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF3 remains set after the clearing sequence is completed for the earlier fault condition.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>FMS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>FMS</td></tr>
<tr><td>FTM2</td><td>—</td><td>FMS</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - No fault condition was detected at the fault input.
1b - A fault condition was detected at the fault input.</td></tr>
<tr><td>2
FAULTF2</td><td>Fault Detection Flag 2
Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.
Clear FAULTF2 by reading the FMS register while FAULTF2 is set and then writing a 0 to FAULTF2 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF2 has no effect. FAULTF2 bit is also cleared when FAULTF bit is cleared.
If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF2 remains set after the clearing sequence is completed for the earlier fault condition.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>FMS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>FMS</td></tr>
<tr><td>FTM2</td><td>—</td><td>FMS</td></tr>
</table>
0b - No fault condition was detected at the fault input.
1b - A fault condition was detected at the fault input.</td></tr>
<tr><td>1
FAULTF1</td><td>Fault Detection Flag 1
Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.
Clear FAULTF1 by reading the FMS register while FAULTF1 is set and then writing a 0 to FAULTF1 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF1 has no effect. FAULTF1 bit is also cleared when FAULTF bit is cleared.
If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF1 remains set after the clearing sequence is completed for the earlier fault condition.
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>FMS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>FMS</td></tr>
<tr><td>FTM2</td><td>—</td><td>FMS</td></tr>
</table>
0b - No fault condition was detected at the fault input.
1b - A fault condition was detected at the fault input.</td></tr>
<tr><td>0
FAULTF0</td><td>Fault Detection Flag 0
Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.</td></tr>
</table>

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<p>Clear FAULTF0 by reading the FMS register while FAULTF0 is set and then writing a 0 to FAULTF0 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF0 has no effect. FAULTF0 bit is also cleared when FAULTF bit is cleared.</p>
<p>If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF0 remains set after the clearing sequence is completed for the earlier fault condition.</p>
<p><b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.</p>
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>FMS</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>FMS</td></tr>
<tr><td>FTM2</td><td>—</td><td>FMS</td></tr>
</table>
<p>0b - No fault condition was detected at the fault input.
1b - A fault condition was detected at the fault input.</p>
</td></tr>
</table>

34.4.3.18 Input Capture Filter Control (FILTER)

34.4.3.18.1 Offset

Register	Offset
FILTER	78h

34.4.3.18.2 Function

This register selects the filter value for the inputs of channels.

Channels 4, 5, 6 and 7 do not have an input filter.

NOTE

Writing to the FILTER register has immediate effect and must be done only when the channels 0, 1, 2, and 3 are not in input modes. Failure to do this could result in a missing valid signal.

34.4.3.18.3 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 W	CH3FVAL				CH2FVAL				CH1FVAL				CH0FVAL			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.18.4 Fields

Field	Function
31-16 —	Reserved
15-12 CH3FVAL	Channel 3 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
11-8 CH2FVAL	Channel 2 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
7-4 CH1FVAL	Channel 1 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
3-0 CH0FVAL	Channel 0 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

34.4.3.19 Fault Control (FLTCTRL)

34.4.3.19.1 Offset

Register	Offset
FLTCTRL	7Ch

34.4.3.19.2 Function

This register contains:

- the state of channels output when a fault event happens
- the enable for each fault input
- the filter enable for each fault input
- the filter value for enabled fault inputs and with filter

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
FTM0	FLTCTRL	—
FTM1	—	FLTCTRL
FTM2	—	FLTCTRL

34.4.3.19.3 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	FSTATE	0			FFVAL				FFLTR3EN	FFLTR2EN	FFLTR1EN	FFLTR0EN	FAULT3EN	FAULT2EN	FAULT1EN	FAULT0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.19.4 Fields

Field	Function
31-16 —	Reserved
15 FSTATE	Fault output state This configuration allows to put the FTM outputs tri-stated when a fault event is ongoing. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

Field	Function
	0b - FTM outputs will be placed into safe values when fault events in ongoing (defined by POL bits). 1b - FTM outputs will be tri-stated when fault event is ongoing
14-12 —	Reserved
11-8 FFVAL	Fault Input Filter Selects the filter value for the fault inputs. The fault filter is disabled when the value is zero. **NOTE:** Writing to this field has immediate effect and must be done only when the fault control or all fault inputs are disabled. Failure to do this could result in a missing fault detection.
7 FFLTR3EN	Fault Input 3 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input filter is disabled. 1b - Fault input filter is enabled.
6 FFLTR2EN	Fault Input 2 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input filter is disabled. 1b - Fault input filter is enabled.
5 FFLTR1EN	Fault Input 1 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input filter is disabled. 1b - Fault input filter is enabled.
4 FFLTR0EN	Fault Input 0 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input filter is disabled. 1b - Fault input filter is enabled.
3 FAULT3EN	Fault Input 3 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input is disabled. 1b - Fault input is enabled.
2 FAULT2EN	Fault Input 2 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input is disabled. 1b - Fault input is enabled.
1 FAULT1EN	Fault Input 1 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

Field	Function
	0b - Fault input is disabled. 1b - Fault input is enabled.
0 FAULT0EN	Fault Input 0 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Fault input is disabled. 1b - Fault input is enabled.

34.4.3.20 Configuration (CONF)

34.4.3.20.1 Offset

Register	Offset
CONF	84h

34.4.3.20.2 Function

This register selects the frequency of the reload opportunities, the FTM behavior in Debug mode, the use of an external global time base, and the global time base signal generation.

This register also controls if initialization trigger should be generated when a reload point is reached.

34.4.3.20.3 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				ITRIGR	GTBEOUT	GTBEEN	0	BDMMODE		0	LDFQ				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.20.4 Fields

Field	Function
31-12 —	Reserved
11 ITRIGR	Initialization trigger on Reload Point This bit controls whether an initialization trigger is generated when a reload point configured by PWMLOAD register is reached considering the FTM_CONF[LDFQ] settings. 0b - Initialization trigger is generated on counter wrap events. 1b - Initialization trigger is generated when a reload point is reached.
10 GTBEOUT	Global Time Base Output Enables the global time base signal generation to other FTMs. 0b - A global time base signal generation is disabled. 1b - A global time base signal generation is enabled.
9 GTBEEN	Global Time Base Enable Configures the FTM to use an external global time base signal that is generated by another FTM. 0b - Use of an external global time base is disabled. 1b - Use of an external global time base is enabled.
8 —	Reserved
7-6 BDMMODE	Debug Mode Selects the FTM behavior in Debug mode. See Debug mode.
5 —	Reserved
4-0 LDFQ	Frequency of the Reload Opportunities The LDFQ[4:0] bits define the number of enabled reload opportunities should happen until an enabled reload opportunity becomes a reload point. See Reload Points LDFQ = 0: All reload opportunities are reload points. LDFQ = 1: There is a reload point each 2 reload opportunities. LDFQ = 2: There is a reload point each 3 reload opportunities. LDFQ = 3: There is a reload point each 4 reload opportunities. This pattern continues up to a maximum of 32.

34.4.3.21 FTM Fault Input Polarity (FLTPOL)

34.4.3.21.1 Offset

Register	Offset
FLTPOL	88h

34.4.3.21.2 Function

This register defines the fault inputs polarity.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
FTM0	FLTPOL	—
FTM1	—	FLTPOL
FTM2	—	FLTPOL

34.4.3.21.3 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												FLT3POL	FLT2POL	FLT1POL	FLT0POL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.21.4 Fields

Field	Function
31-4 —	Reserved
3 FLT3POL	Fault Input 3 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The fault input polarity is active high. A 1 at the fault input indicates a fault. 1b - The fault input polarity is active low. A 0 at the fault input indicates a fault.
2 FLT2POL	Fault Input 2 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

Field	Function
	0b - The fault input polarity is active high. A 1 at the fault input indicates a fault. 1b - The fault input polarity is active low. A 0 at the fault input indicates a fault.
1 FLT1POL	Fault Input 1 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The fault input polarity is active high. A 1 at the fault input indicates a fault. 1b - The fault input polarity is active low. A 0 at the fault input indicates a fault.
0 FLT0POL	Fault Input 0 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The fault input polarity is active high. A 1 at the fault input indicates a fault. 1b - The fault input polarity is active low. A 0 at the fault input indicates a fault.

34.4.3.22 Synchronization Configuration (SYNCONF)

34.4.3.22.1 Offset

Register	Offset
SYNCONF	8Ch

34.4.3.22.2 Function

This register selects the PWM synchronization configuration, SWOCTRL, INVCTRL and CNTIN registers synchronization, if FTM clears the TRIGj bit, where j = 0, 1, 2, when the hardware trigger j is detected.

34.4.3.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											HWSOC	HWINVC	HWOM	HWWRBUF	HWRSTCNT
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			SWSOC	SWINVC	SWOM	SWWRBUF	SWRSTCNT	SYNCMODE	0	SWOC	INVC	0	CNTINC	0	HWTRIGMODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.22.4 Fields

Field	Function
31-21 —	Reserved
20 HWSOC	Software output control synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the SWOCTRL register synchronization. 1b - A hardware trigger activates the SWOCTRL register synchronization.
19 HWINVC	Inverting control synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the INVCTRL register synchronization. 1b - A hardware trigger activates the INVCTRL register synchronization.
18 HWOM	Output mask synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the OUTMASK register synchronization. 1b - A hardware trigger activates the OUTMASK register synchronization.
17 HWWRBUF	MOD, HCR, CNTIN, and CV registers synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate MOD, HCR, CNTIN, and CV registers synchronization. 1b - A hardware trigger activates MOD, HCR, CNTIN, and CV registers synchronization.
16 HWRSTCNT	FTM counter synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the FTM counter synchronization. 1b - A hardware trigger activates the FTM counter synchronization.
15-13 —	Reserved
12 SWSOC	Software output control synchronization is activated by the software trigger 0b - The software trigger does not activate the SWOCTRL register synchronization. 1b - The software trigger activates the SWOCTRL register synchronization.
11 SWINVC	Inverting control synchronization is activated by the software trigger 0b - The software trigger does not activate the INVCTRL register synchronization. 1b - The software trigger activates the INVCTRL register synchronization.

Table continues on the next page...

Field	Function
10 SWOM	Output mask synchronization is activated by the software trigger 0b - The software trigger does not activate the OUTMASK register synchronization. 1b - The software trigger activates the OUTMASK register synchronization.
9 SWWRBUF	MOD, HCR, CNTIN, and CV registers synchronization is activated by the software trigger 0b - The software trigger does not activate MOD, HCR, CNTIN, and CV registers synchronization. 1b - The software trigger activates MOD, HCR, CNTIN, and CV registers synchronization.
8 SWRSTCNT	FTM counter synchronization is activated by the software trigger 0b - The software trigger does not activate the FTM counter synchronization. 1b - The software trigger activates the FTM counter synchronization.
7 SYNCMODE	Synchronization Mode Selects the PWM Synchronization mode. 0b - Legacy PWM synchronization is selected. 1b - Enhanced PWM synchronization is selected.
6 —	Reserved
5 SWOC	SWOCTRL Register Synchronization 0b - SWOCTRL register is updated with its buffer value at all rising edges of FTM input clock. 1b - SWOCTRL register is updated with its buffer value by the PWM synchronization.
4 INVC	INVCTRL Register Synchronization 0b - INVCTRL register is updated with its buffer value at all rising edges of FTM input clock. 1b - INVCTRL register is updated with its buffer value by the PWM synchronization.
3 —	Reserved
2 CNTINC	CNTIN Register Synchronization 0b - CNTIN register is updated with its buffer value at all rising edges of FTM input clock. 1b - CNTIN register is updated with its buffer value by the PWM synchronization.
1 —	Reserved
0 HWTRIGMODE	Hardware Trigger Mode 0b - FTM clears the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2. 1b - FTM does not clear the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2.

34.4.3.23 FTM Inverting Control (INVCTRL)

34.4.3.23.1 Offset

Register	Offset
INVCTRL	90h

34.4.3.23.2 Function

This register controls when the channel (n) output becomes the channel (n+1) output, and channel (n+1) output becomes the channel (n) output. Each INVmEN bit enables the inverting operation for the corresponding pair channels m.

This register has a write buffer. The INVmEN bit is updated by the INVCTRL register synchronization.

34.4.3.23.3 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												INV3EN	INV2EN	INV1EN	INV0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.23.4 Fields

Field	Function
31-4 —	Reserved
3 INV3EN	Pair Channels 3 Inverting Enable **NOTE:** This field is not supported in every instance. The following table includes only supported registers. 0b - Inverting is disabled. 1b - Inverting is enabled.
2 INV2EN	Pair Channels 2 Inverting Enable **NOTE:** This field is not supported in every instance. The following table includes only supported registers.

Instance	Field supported in	Field not supported in
FTM0	INVCTRL	—
FTM1	—	INVCTRL
FTM2	—	INVCTRL

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td><table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>INVCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>INVCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>INVCTRL</td></tr>
</table>
0b - Inverting is disabled.
1b - Inverting is enabled.</td></tr>
<tr><td>1
INV1EN</td><td>Pair Channels 1 Inverting Enable
0b - Inverting is disabled.
1b - Inverting is enabled.</td></tr>
<tr><td>0
INV0EN</td><td>Pair Channels 0 Inverting Enable
0b - Inverting is disabled.
1b - Inverting is enabled.</td></tr>
</table>

34.4.3.24 FTM Software Output Control (SWOCTRL)

34.4.3.24.1 Offset

Register	Offset
SWOCTRL	94h

34.4.3.24.2 Function

This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CH(n)OC bits enable the control of the corresponding channel (n) output by software.
- The CH(n)OCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

34.4.3.24.3 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	CH7OCV	CH6OCV	CH5OCV	CH4OCV	CH3OCV	CH2OCV	CH1OCV	CH0OCV	CH7OC	CH6OC	CH5OC	CH4OC	CH3OC	CH2OC	CH1OC	CH0OC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.24.4 Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-16
—</td><td>Reserved</td></tr>
<tr><td>15
CH7OCV</td><td>Channel 7 Software Output Control Value
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>14
CH6OCV</td><td>Channel 6 Software Output Control Value
NOTE: This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>13</td><td>Channel 5 Software Output Control Value</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>CH5OCV</td><td><b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>12
CH4OCV</td><td>Channel 4 Software Output Control Value
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>11
CH3OCV</td><td>Channel 3 Software Output Control Value
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>10
CH2OCV</td><td>Channel 2 Software Output Control Value
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>9
CH1OCV</td><td>Channel 1 Software Output Control Value
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>8
CH0OCV</td><td>Channel 0 Software Output Control Value
0b - The software output control forces 0 to the channel output.
1b - The software output control forces 1 to the channel output.</td></tr>
<tr><td>7
CH7OC</td><td>Channel 7 Software Output Control Enable
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The channel output is not affected by software output control.</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>1b - The channel output is affected by software output control.</td></tr>
<tr><td>6
CH6OC</td><td>Channel 6 Software Output Control Enable
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The channel output is not affected by software output control.
1b - The channel output is affected by software output control.</td></tr>
<tr><td>5
CH5OC</td><td>Channel 5 Software Output Control Enable
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The channel output is not affected by software output control.
1b - The channel output is affected by software output control.</td></tr>
<tr><td>4
CH4OC</td><td>Channel 4 Software Output Control Enable
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>SWOCTRL</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>SWOCTRL</td></tr>
<tr><td>FTM2</td><td>—</td><td>SWOCTRL</td></tr>
</table>
0b - The channel output is not affected by software output control.
1b - The channel output is affected by software output control.</td></tr>
<tr><td>3
CH3OC</td><td>Channel 3 Software Output Control Enable
0b - The channel output is not affected by software output control.
1b - The channel output is affected by software output control.</td></tr>
<tr><td>2
CH2OC</td><td>Channel 2 Software Output Control Enable
0b - The channel output is not affected by software output control.
1b - The channel output is affected by software output control.</td></tr>
<tr><td>1
CH1OC</td><td>Channel 1 Software Output Control Enable
0b - The channel output is not affected by software output control.
1b - The channel output is affected by software output control.</td></tr>
</table>

Table continues on the next page...

Field	Function
0 CH0OC	Channel 0 Software Output Control Enable 0b - The channel output is not affected by software output control. 1b - The channel output is affected by software output control.

34.4.3.25 FTM PWM Load (PWMLOAD)

34.4.3.25.1 Offset

Register	Offset
PWMLOAD	98h

34.4.3.25.2 Function

Enables the reload of the MOD, HCR, CNTIN, C(n)V, and C(n+1)V registers with the values of their write buffers when the FTM counter changes from the MOD register value to its next value or when a channel (j) match occurs. A match occurs for channel (j) when FTM counter = C(j)V. A reload can also occurs when FTM counter = HCR register at a half cycle match. This register also controls the local and global load mechanisms.

34.4.3.25.3 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				0	GLEN	LDOK	HCSEL	CH7SEL	CH6SEL	CH5SEL	CH4SEL	CH3SEL	CH2SEL	CH1SEL	CH0SEL
W					GLDOK											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.25.4 Fields

Field	Function
31-12	Reserved

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>—</td><td></td></tr>
<tr><td>11
GLDOK</td><td>Global Load OK
This bit controls the global load mechanism. It generates a pulse at FTM module global load output with one FTM clock cycle width, which is used to set LDOK bits of FTM and other modules (including other FTMs). This bit is self-cleared and read value is always zero.
The global load mechanism depends on SoC specific information. Refer to FTM SoC specific information to more details.
0b - No action.
1b - LDOK bit is set.</td></tr>
<tr><td>10
GLEN</td><td>Global Load Enable
This bit enables the global load mechanism implemented by GLDOK. If GLEN bit is set, then an external event on the FTM global load input sets the LDOK bit. The clear of the LDOK bit is done by CPU writes '0' to the bit.
0b - Global Load Ok disabled.
1b - Global Load OK enabled. A pulse event on the module global load input sets the LDOK bit.</td></tr>
<tr><td>9
LDOK</td><td>Load Enable
Enables the loading of the MOD, CNTIN, HCR and CV registers with the values of their buffers.
The LDOK bit can also be set by the Global Load mechanism if GLEN bit is enabled.
0b - Loading updated values is disabled.
1b - Loading updated values is enabled.</td></tr>
<tr><td>8
HCSEL</td><td>Half Cycle Select
This bit enables the half cycle match as a reload opportunity. A half cycle is defined by when the FTM counter matches the HCR register.
0b - Half cycle reload is disabled and it is not considered as a reload opportunity.
1b - Half cycle reload is enabled and it is considered as a reload opportunity.</td></tr>
<tr><td>7
CH7SEL</td><td>Channel 7 Select
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>PWMLOAD</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>PWMLOAD</td></tr>
<tr><td>FTM2</td><td>—</td><td>PWMLOAD</td></tr>
</table>
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>6
CH6SEL</td><td>Channel 6 Select
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>PWMLOAD</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>PWMLOAD</td></tr>
<tr><td>FTM2</td><td>—</td><td>PWMLOAD</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>5
CH5SEL</td><td>Channel 5 Select
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>PWMLOAD</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>PWMLOAD</td></tr>
<tr><td>FTM2</td><td>—</td><td>PWMLOAD</td></tr>
</table>
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>4
CH4SEL</td><td>Channel 4 Select
<b>NOTE:</b> This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Instance</th><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>FTM0</td><td>PWMLOAD</td><td>—</td></tr>
<tr><td>FTM1</td><td>—</td><td>PWMLOAD</td></tr>
<tr><td>FTM2</td><td>—</td><td>PWMLOAD</td></tr>
</table>
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>3
CH3SEL</td><td>Channel 3 Select
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>2
CH2SEL</td><td>Channel 2 Select
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>1
CH1SEL</td><td>Channel 1 Select
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
<tr><td>0
CH0SEL</td><td>Channel 0 Select
0b - Channel match is not included as a reload opportunity.
1b - Channel match is included as a reload opportunity.</td></tr>
</table>

34.4.3.26 Half Cycle Register (HCR)

34.4.3.26.1 Offset

Register	Offset
HCR	9Ch

34.4.3.26.2 Function

The Half Cycle Register contains the match value for FTM half cycle reload feature. After FTM counter reaches this value, a reload opportunity is generated if FTM_PWMLOAD[HCSEL] is enabled.

Writing to the HCR register latches the value into a buffer. The HCR register is updated with the value of its write buffer according to Registers updated from write buffers.

34.4.3.26.3 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 W	HCVAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.26.4 Fields

Field	Function
31-16 —	Reserved
15-0 HCVAL	Half Cycle Value

34.4.3.27 Mirror of Modulo Value (MOD_MIRROR)

34.4.3.27.1 Offset

Register	Offset
MOD_MIRROR	200h

34.4.3.27.2 Function

This register contains the integer and fractional modulo value for the FTM counter.

34.4.3.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MOD															
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	FRACMOD					0										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.4.3.27.4 Fields

Field	Function
31-16 MOD	Mirror of the Modulo Integer Value See the field MOD of the register MOD.
15-11 FRACMOD	Modulo Fractional Value The modulo fractional value is used in the PWM period dithering. This value is added to an internal accumulator at the end of each PWM period. Writes to the field FRACMOD are done on its write buffer. The FRACMOD is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to SC register resets manually this write coherency mechanism.
10-0 —	Reserved

34.4.3.28 Mirror of Channel (n) Match Value (C0V_MIRROR - C7V_MIRROR)

34.4.3.28.1 Offset

For a = 0 to 7:

Register	Offset
CaV_MIRROR	204h + (a × 4h)

34.4.3.28.2 Function

This register contains the integer and fractional value of the channel (n) match.

NOTE

Each module instance supports a different number of registers.

Instance	Register supported	Register not supported
FTM0	C0V_MIRROR–C7V_MIRROR	—
FTM1	C0V_MIRROR–C3V_MIRROR	C4V_MIRROR–C7V_MIRROR
FTM2	C0V_MIRROR–C3V_MIRROR	C4V_MIRROR–C7V_MIRROR

34.4.3.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VAL															
Reset	See #d6159e14933a1310.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRACVAL					0										
W																
Reset	See #d6159e14933a1310.															

34.4.3.28.4 Register reset values

Register	Reset value
C0V_MIRROR–C3V_MIRROR	FTM0–FTM2: 0000_0000h
C4V_MIRROR–C7V_MIRROR	0000_0000h

34.4.3.28.5 Fields

Field	Function
31-16 VAL	Mirror of the Channel (n) Match Integer Value See the field VAL of the register CnV.
15-11 FRACVAL	Channel (n) Match Fractional Value The channel (n) match fractional value is used in the PWM edge dithering. This value is added to the channel (n) internal accumulator at the end of each PWM period. Writes to the field FRACVAL are done on its write buffer. The FRACVAL is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to CnSC register resets manually this write coherency mechanism.
10-0 —	Reserved

34.5 Functional Description

34.5.1 Clock source

The FTM has only one clock domain: the FTM input clock.

34.5.1.1 Counter clock source

The CLKS[1:0] bits select one of three possible clock sources for the FTM counter or disable the FTM counter. After any chip reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The fixed frequency clock is an alternative clock source for the FTM counter that allows the selection of a clock other than the FTM input clock or an external clock. This clock input is defined by chip integration. Refer to the chip specific documentation for further information. Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM input clock frequency.

The external clock passes through a synchronizer clocked by the FTM input clock to assure that counter transitions are properly aligned to FTM input clock transitions. Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the FTM input clock frequency.

34.5.2 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 FTM counter.

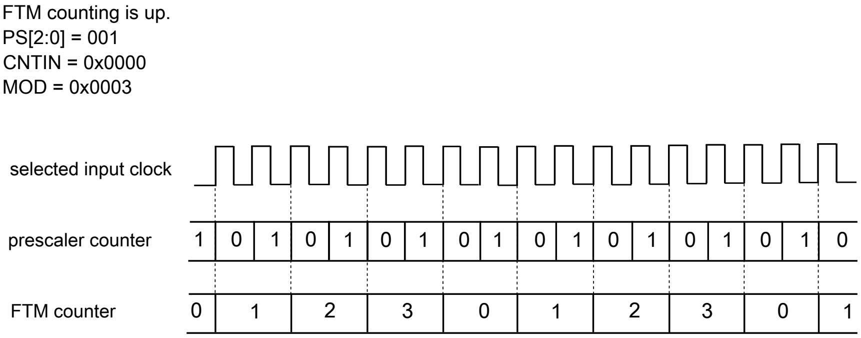


Figure 34-3. Example of the prescaler counter

34.5.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler.

The FTM counter has these modes of operation:

- Up counting
- Up-down counting

34.5.3.1 Up counting

Up counting is selected when:

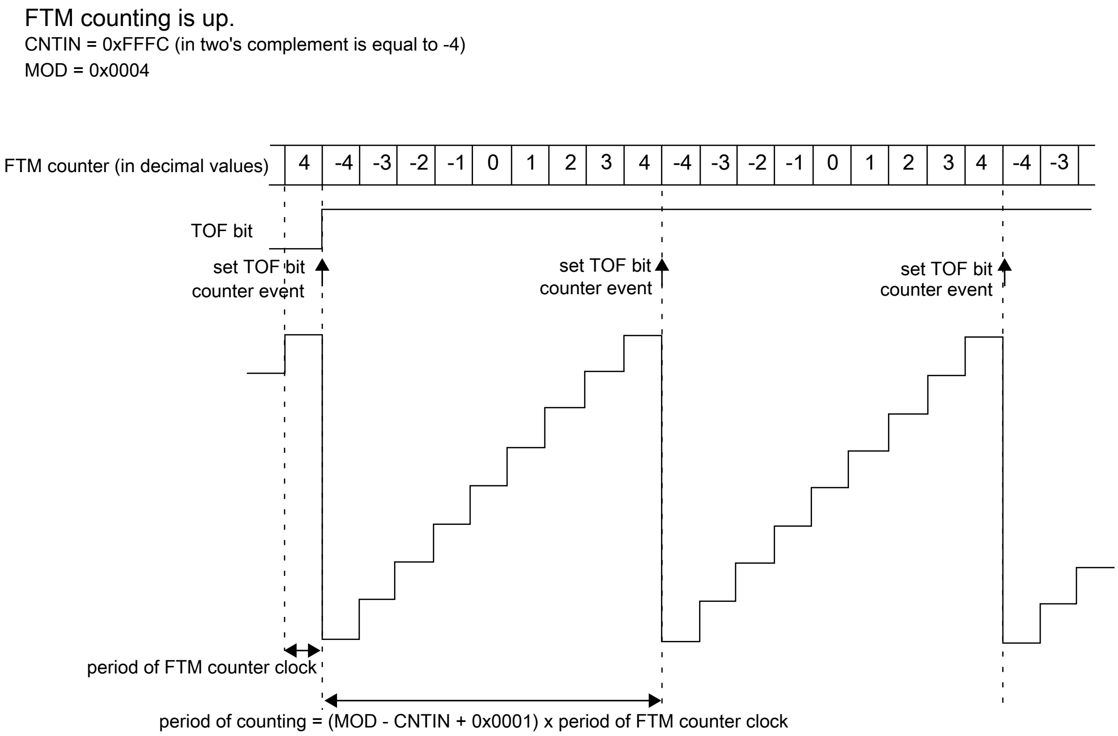
- CPWMS = 0

CNTIN defines the starting value of the count and MOD defines the final value of the count, see the following figure. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.

The FTM period when using up counting is (MOD – CNTIN + 0x0001) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to CNTIN.

A counter event happens at the same time of TOF bit set when the FTM counter changes from MOD to CNTIN. See Counter events for more details.



FTM counting is up.
CNTIN = 0xFFFC (in two's complement is equal to -4)
MOD = 0x0004
FTM counter (in decimal values)
4 -4 -3 -2 -1 0 1 2 3 4 -4 -3 -2 -1 0 1 2 3 4 -4 -3
TOF bit
set TOF bit
counter event
set TOF bit
counter event
set TOF bit
counter event
period of FTM counter clock
period of counting = (MOD - CNTIN + 0x0001) x period of FTM counter clock

Figure 34-4. Example of FTM up and signed counting

Table 34-3. FTM counting based on CNTIN value

When	Then
CNTIN = 0x0000	The FTM counting is equivalent to TPM up counting, that is, up and unsigned counting. See the following figure.
CNTIN[15] = 1	The initial value of the FTM counter is a negative number in two's complement, so the FTM counting is up and signed.
CNTIN[15] = 0 and CNTIN ≠ 0x0000	The initial value of the FTM counter is a positive number, so the FTM counting is up and unsigned.

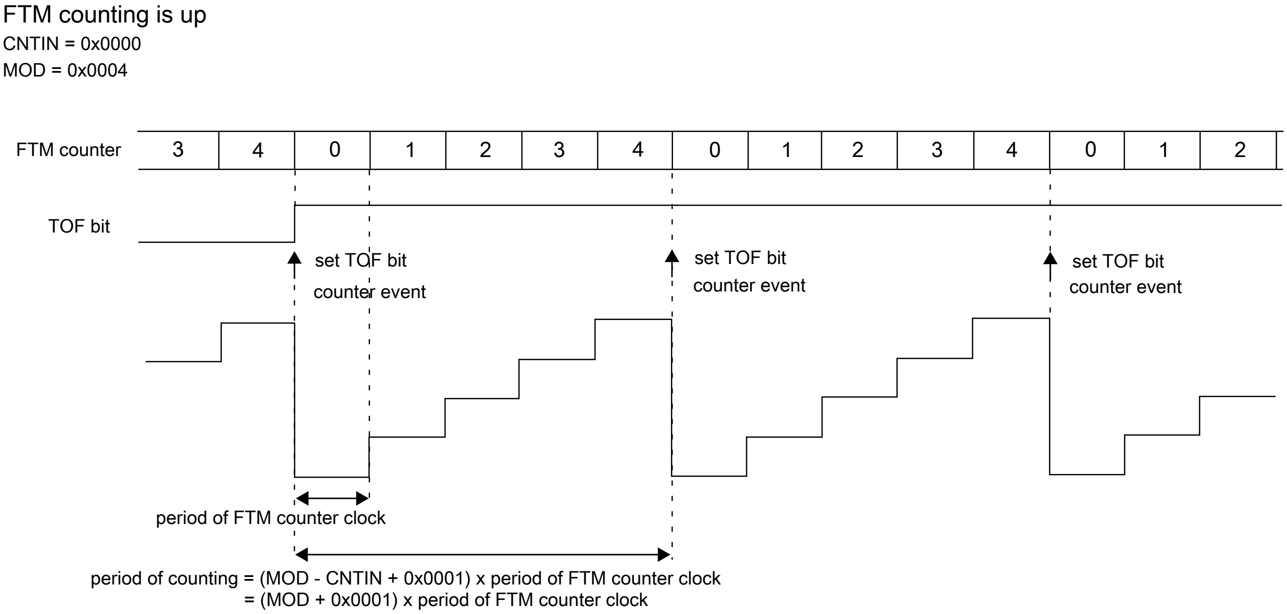


Figure 34-5. Example of FTM up counting with CNTIN = 0x0000

Note

- FTM operation is only valid when the value of the CNTIN register is less than the value of the MOD register, either in the unsigned counting or signed counting. It is the responsibility of the software to ensure that the values in the CNTIN and MOD registers meet this requirement. Any values of CNTIN and MOD that do not satisfy this criteria can result in unpredictable behavior.
- MOD = CNTIN is a redundant condition. In this case, the FTM counter is always equal to MOD and the TOF bit is set in each rising edge of the FTM counter clock.
- When MOD = 0x0000, CNTIN = 0x0000, for example after reset, and FTMEN = 1, the FTM counter remains stopped at 0x0000 until a non-zero value is written into the MOD or CNTIN registers.
- Setting CNTIN to be greater than the value of MOD is not recommended as this unusual setting may make the FTM operation difficult to comprehend. However, there is no restriction on this configuration, and an example is shown in the following figure.

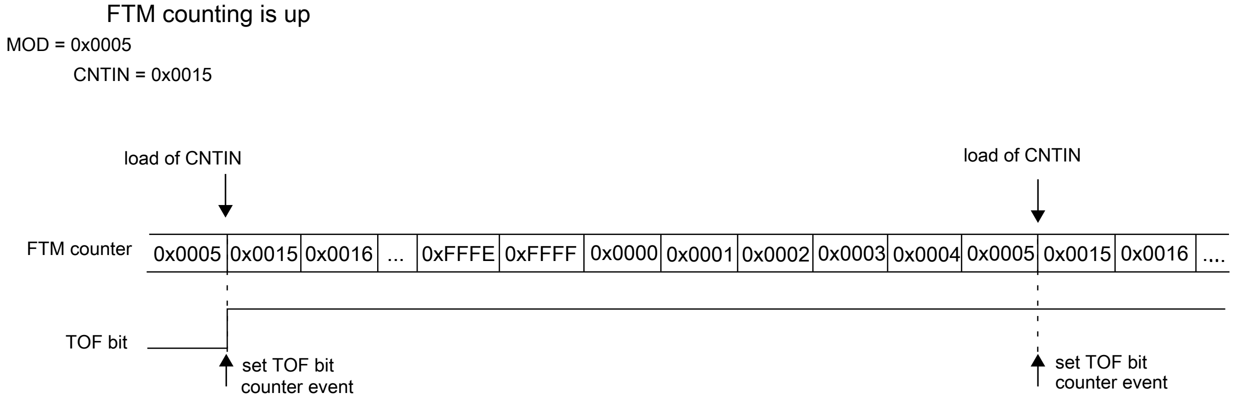


Figure 34-6. Example of up counting when the value of CNTIN is greater than the value of MOD

34.5.3.2 Up-down counting

Up-down counting is selected when:

- CPWMS = 1

CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The FTM period when using up-down counting is 2 × (MOD – CNTIN) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to (MOD – 1).

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up-down counting, that is, up-down and unsigned counting. See the following figure.

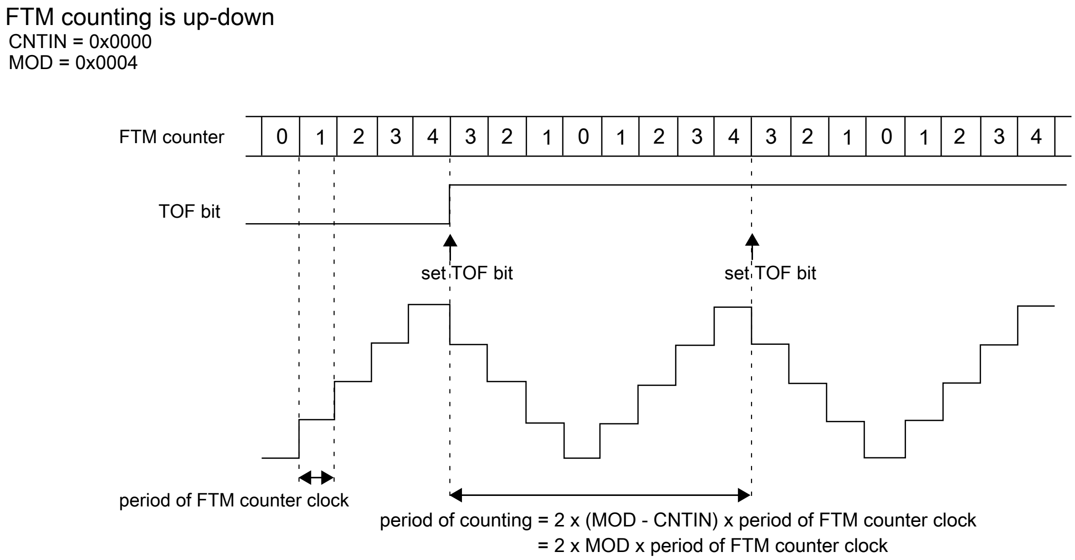

FTM counting is up-down
CNTIN = 0x0000
MOD = 0x0004
FTM counter
0 1 2 3 4 3 2 1 0 1 2 3 4 3 2 1 0 1 2 3 4
TOF bit
set TOF bit
set TOF bit
period of FTM counter clock
period of counting = 2 x (MOD - CNTIN) x period of FTM counter clock
= 2 x MOD x period of FTM counter clock

Figure 34-7. Example of up-down counting when CNTIN = 0x0000

Note

> When CNTIN is different from zero in the up-down counting, a valid CPWM signal is generated:
>
> - if CnV > CNTIN, or
> - if CnV = 0 or if CnV[15] = 1. In this case, 0% CPWM is generated.

The figure below shows the possible counter events when in up-down counting mode. See Counter events for more details.

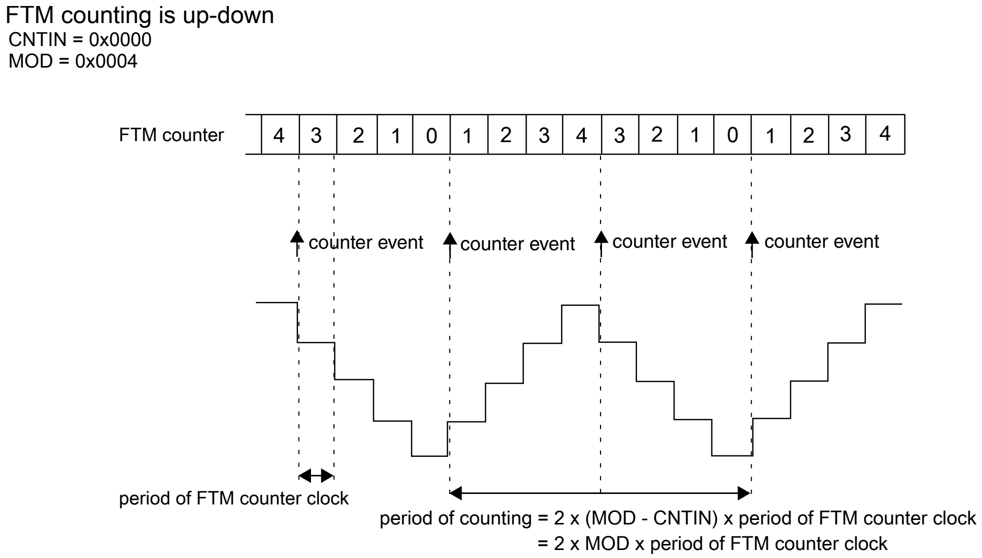


Figure 34-8. Example of counter events in up-down counting mode when CNTIN = 0x0000

34.5.3.3 Free running counter

If (FTMEN = 0) and (MOD = 0x0000 or MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000. See the following figure.

A counter event occurs at the same time of TOF bit set when the FTM counter changes from 0xFFFF to 0x0000. See Counter events for more details.

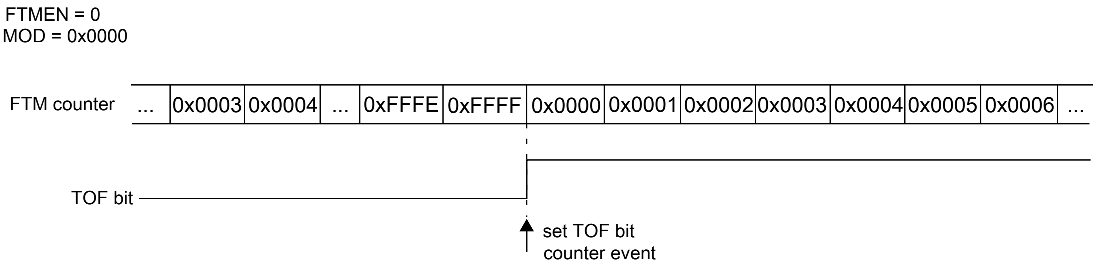


Figure 34-9. Example when the FTM counter is free running

The FTM counter is also a free running counter when:

- FTMEN = 1
- CPWMS = 0
- CNTIN = 0x0000, and
- MOD = 0xFFFF

34.5.3.4 Counter reset

Any one of the following cases resets the FTM counter to the value in the CNTIN register and the channels output to its initial value, except for channels in Output Compare mode.

- Any write to CNT.
- FTM counter synchronization.
- A channel in Input Capture mode with ICRST = 1 (FTM Counter Reset in Input Capture Mode).

Note that resetting the counter also generates a counter event. See Counter events for more details.

34.5.3.5 Counter events

Counter events can be used as reload opportunities to FTM register synchronization mechanism. See Reload Points for more details. There are some possible counter events depending on the counter mode. Please see the table below for more details.

Table 34-4. FTM counter events

When	Then
FTM counter is in up counting mode or freerunning	• A counter event happens at the same time of TOF bit set when the FTM counter changes from MOD to CNTIN (counter wrap). Figure at Up counting shows the counter event generation. • When in freerunning, there is a counter event when FTM counter changes from 0xFFFF to 0x0000. Figure at Free running counter shows the counter event generation.
FTM counter is in up-down counting mode	• In up-down counting mode, there are two possible counter events when FTM counter turns from down to up counting and when counter turns from up to down counting. User can select which point will be used to generate the counter event. Figure at Up-down counting shows the possible counter events.
FTM counter is reset (see Counter reset) or a value different from zero is written at CLKS field	• In up-counting mode, all counter reset events or a write in the CLKS with a value different from zero generates a counter event. • In up-down counting mode, counter reset events only generates a counter event if the minimum load point when FTM counter turns from down to up counting is configured. A write in the CLKS with a value different from zero always generates a counter event in up-down counting mode.

34.5.4 Channel Modes

The following table shows the channel modes selection.

Table 34-5. Channel Modes Selection

DECAPEN	MCOMBINE	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
X	X	X	X	XX	00	Pin not used for FTM—revert the channel pin to general purpose I/O or other peripheral control	
0	0	0	0	00	01	Input Capture	Capture on Rising Edge Only

Table continues on the next page...

Table 34-5. Channel Modes Selection (continued)

DECAPEN	MCOMBINE	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
					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
				1X	10	Edge-Aligned PWM	High-true pulses (clear Output on match)
					X1		Low-true pulses (set Output on match)
			1	XX	10	Center-Aligned PWM	High-true pulses (clear Output on match-up)
					X1		Low-true pulses (set Output on match-up)
		1	0	XX	10	Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
					X1		Low-true pulses (clear on channel (n) match, and set on channel (n+1) match)
	1	0	X	XX	XX	Reserved for future use	
	1	1	0	XX	10	Modified Combine PWM	High-true pulses (set on channel (n) match, and

Table continues on the next page...

Table 34-5. Channel Modes Selection (continued)

DECAPEN	MCOMBINE	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
							clear on channel (n+1) match)
					X1		Low-true pulses (clear on channel (n) match, and set on channel (n+1) match)
1	0	0	0	X0	See Table 34-6.	Dual Edge Capture	One-Shot Capture mode
				X1			Continuous Capture mode

Table 34-6. Dual Edge Capture Mode — Edge Polarity Selection

ELSB	ELSA	Channel Port Enable	Detected Edges
0	0	Disabled	No edge
0	1	Enabled	Rising edge
1	0	Enabled	Falling edge
1	1	Enabled	Rising and falling edges

34.5.5 Input Capture Mode

The Input Capture mode is selected when:

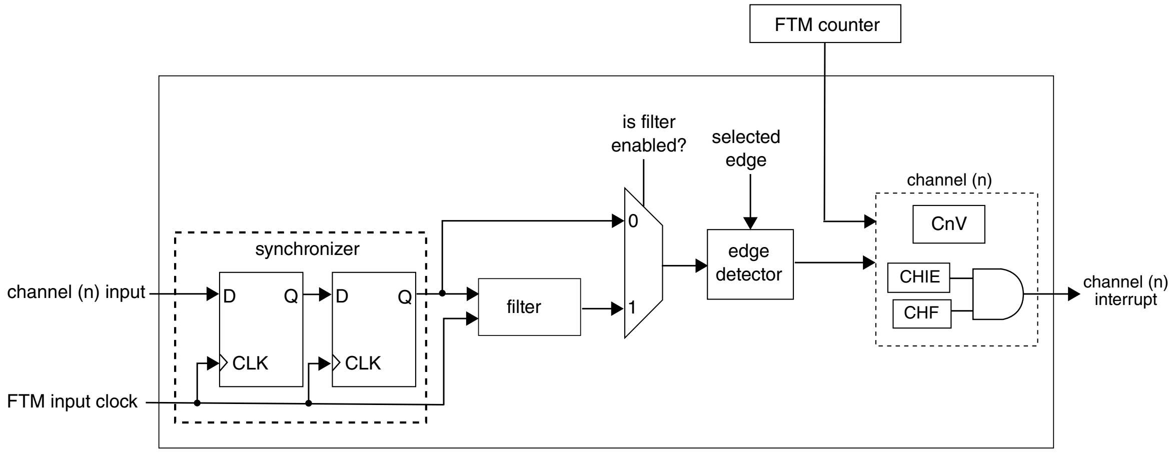
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0
- MSB:MSA = 0:0, and
- ELSB:ELSA ≠ 0:0

When a selected edge occurs on the channel input, the current value of the FTM counter is captured into the CnV register, at the same time the CHF bit is set and the channel interrupt is generated if enabled by CHIE = 1. See the following figure.

When a channel is configured for input capture, the FTMxCHn pin is an edge-sensitive input. ELSB:ELSA bits determine which edge, falling or rising, triggers input-capture event.

Writes to the CnV register are ignored in input capture mode.

While in Debug mode, the input capture function works as configured. When a selected edge event occurs, the FTM counter value, which is frozen because of Debug, is captured into the CnV register and the CHF bit is set.



Note: The filter is only available for the channels 0, 1, 2, and 3 inputs.

Figure 34-10. Diagram for Input Capture Mode

34.5.5.1 Filter for Input Capture Mode

The filter is only available on channels 0, 1, 2, and 3.

The channel input after being synchronized by FTM input clock (Figure 34-10) is the filter input.

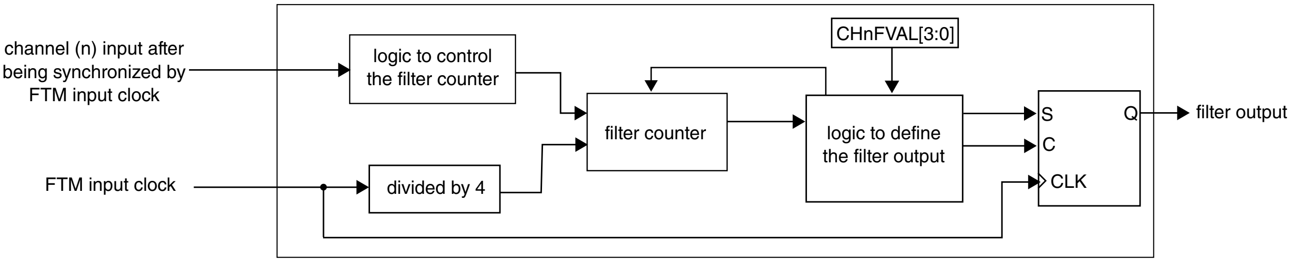

channel (n) input after being synchronized by FTM input clock
logic to control the filter counter
CHnFVAL[3:0]
filter counter
logic to define the filter output
S
Q
C
CLK
filter output
FTM input clock
divided by 4

Figure 34-11. Channel Input Filter

NOTE

The maximum frequency for the channel input to be detected correctly is FTM input clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

When there is a state change in the channel input, the counter is reset and starts counting up. As long as the new state is stable on the channel input, the counter continues to increment. When the counter is equal to CHnFVAL[3:0], the new channel input signal value is validated. It is then transmitted as a pulse to the edge detector.

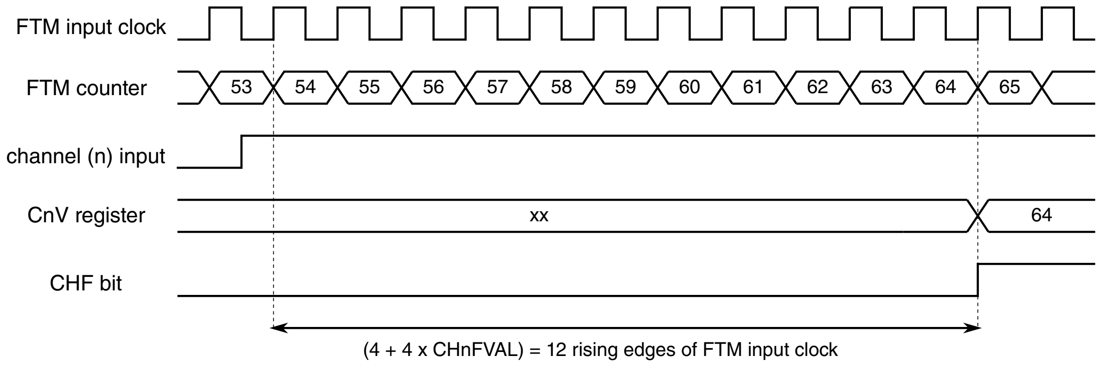
If the opposite edge appears on the channel input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. If a pulse is sampled as a value less than (CHnFVAL[3:0] x 4) consecutive rising edges of FTM input clock, it is regarded as a glitch and is not passed on to the edge detector.

The table below shows the delay that is added by the FTM channel input filter according to its configuration.

Table 34-7. FTM Channel Input Filter Delay

FTM channel input filter	Number of rising edges between the selected edge on channel input and setting CHF bit
• channel does not have the input filter, or • channel input filter is disabled (CHnFVAL[3:0] = 0)	• 3 rising edges of FTM input clock
• channel has the input filter, and • channel input filter is enabled (CHnFVAL[3:0] ≠ 0)	• (4 + 4 × CHnFVAL[3:0]) rising edges of FTM input clock

The following figure illustrates an example of channel input filter.



Note:
PS[2:0] = 3'b000
channel (n) in input capture mode with capture only on rising edges
CHnFVAL[3:0] = 4'h2 (channel (n) input filter is enabled)

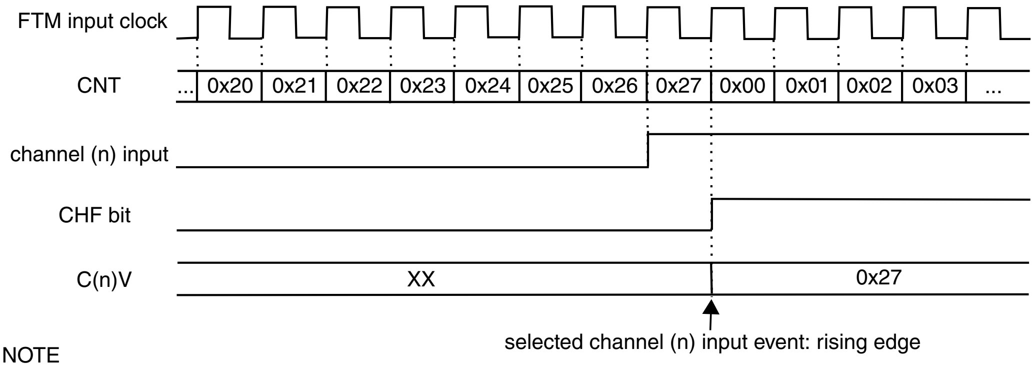
Figure 34-12. Example of Channel Input Filter

34.5.5.2 FTM Counter Reset in Input Capture Mode

If the channel (n) is in input capture mode and CnSC[ICRST = 1], then when the selected input capture event occurs in the channel (n) input signal, the current value of the FTM counter is captured into the CnV register, the CHF bit is set, the channel (n) interrupt is generated (if CHIE = 1) and the FTM counter is reset to the CNTIN register value.

This allows the FTM to measure a period/pulse being applied to the channel (n) input (number of the FTM input clocks) without having to implement a subtraction calculation in software subsequent to the event occurring.

The figure below shows the FTM counter reset when the selected input capture event is detected in a channel in input capture mode with ICRST = 1.



NOTE
Channel (n) input after its synchronizer and filter
MOD = 0xFFFF
CNTIN = 0x0000
PS[2:0] = 3'b000
ICRST = 1'b1

Figure 34-13. Example of the Input Capture mode with ICRST = 1

NOTE

- It is expected that the ICRST bit be set only when the channel is in input capture mode.
- If the FTM counter is reset because the channel is in input capture mode with ICRST = 1, then the prescaler counter (Prescaler) is also reset.

34.5.6 Output Compare mode

The Output Compare mode is selected when:

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0

- CPWMS = 0, and
- MSB:MSA = 0:1

In Output Compare mode, the FTM 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.

When a channel is initially configured to Toggle mode, the previous value of the channel output is held until the first output compare event occurs.

The CHF bit is set and the channel (n) interrupt is generated if CHIE = 1 at the channel (n) match (FTM counter = CnV).

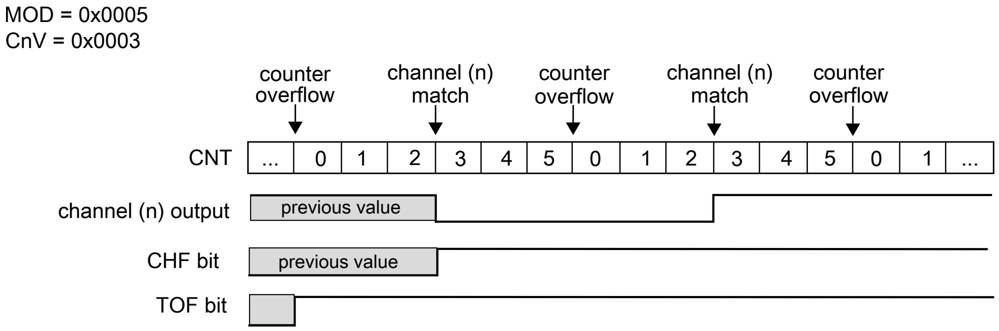


Figure 34-14. Example of the Output Compare mode when the match toggles the channel output

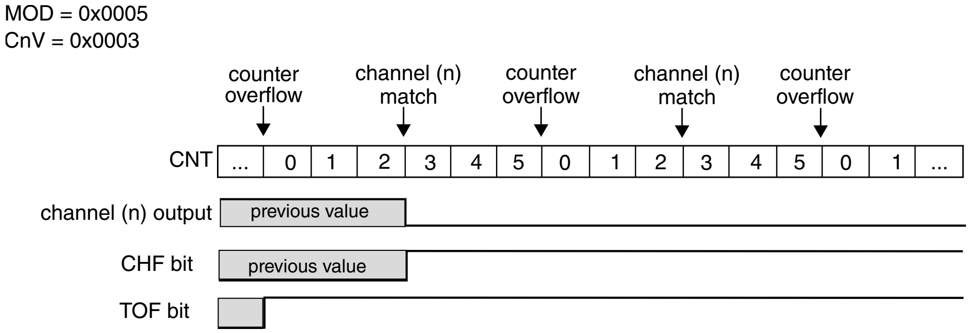


Figure 34-15. Example of the Output Compare mode when the match clears the channel output

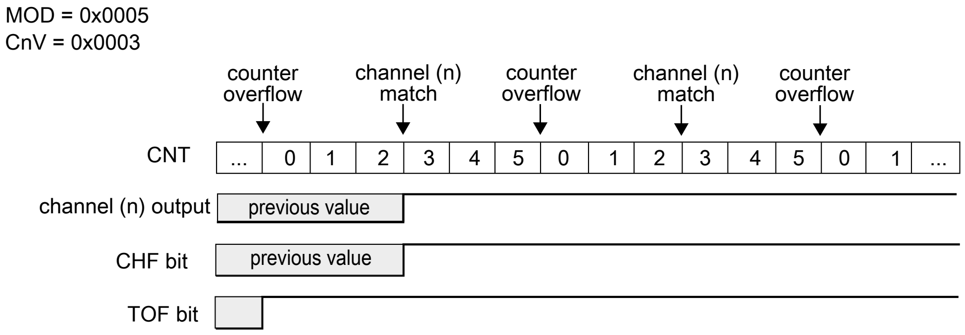


Figure 34-16. Example of the Output Compare mode when the match sets the channel output

If (ELSB:ELSA = 0:0) when the counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated if CHIE = 1, however the channel (n) output is not modified and controlled by FTM.

34.5.7 Edge-Aligned PWM (EPWM) mode

The Edge-Aligned mode is selected when:

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0, and
- MSB = 1

The EPWM period is determined by (MOD − CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV − CNTIN).

The CHF bit is set and the channel (n) interrupt is generated if CHIE = 1 at the channel (n) match (FTM 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 FTM.

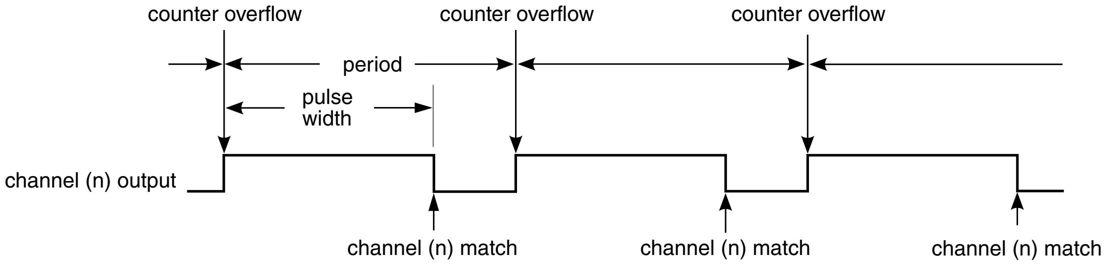


Figure 34-17. EPWM period and pulse width with ELSB:ELSA = 1:0

If (ELSB:ELSA = 0:0) when the counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated if CHIE = 1, however the channel (n) output is not controlled by FTM.

If (ELSB:ELSA = 1:0), then the channel (n) output is forced high at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced low at the channel (n) match (FTM counter = CnV). See the following figure.

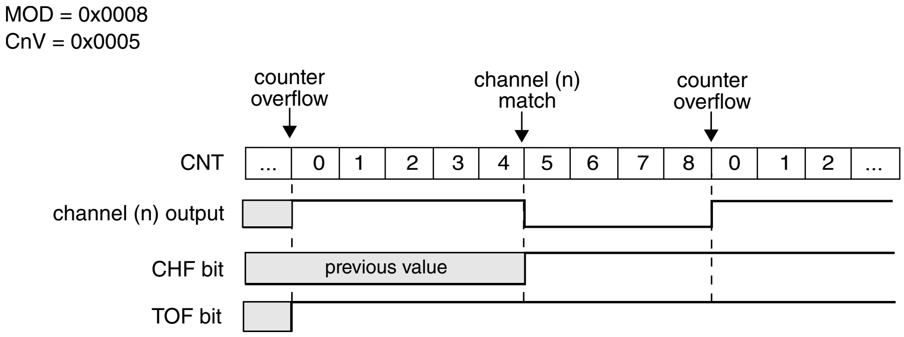


Figure 34-18. EPWM signal with ELSB:ELSA = 1:0

If (ELSB:ELSA = X:1), then the channel (n) output is forced low at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced high at the channel (n) match (FTM counter = CnV). See the following figure.

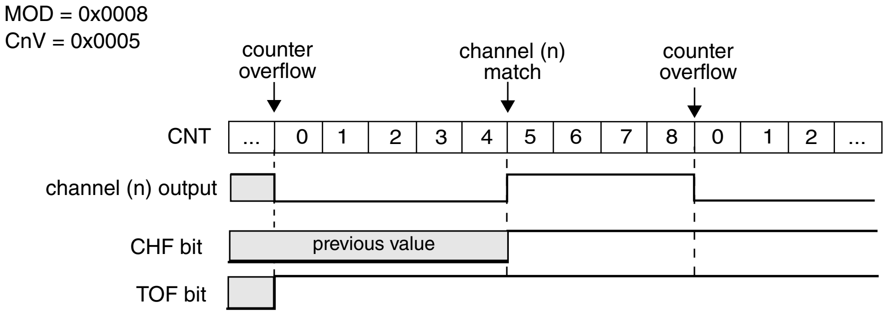


Figure 34-19. EPWM signal with ELSB:ELSA = X:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHF bit is not set even when there is the channel (n) match.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHF bit is not set. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

Note

When CNTIN is different from zero the following EPWM signals can be generated:

- 0% EPWM signal if CnV = CNTIN,

- EPWM signal between 0% and 100% if CNTIN < CnV <= MOD,
- 100% EPWM signal when CNTIN > CnV or CnV > MOD.

34.5.8 Center-Aligned PWM (CPWM) mode

The Center-Aligned mode is selected when:

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0, and
- CPWMS = 1

The CPWM pulse width (duty cycle) is determined by 2 × (CnV − CNTIN) and the period is determined by 2 × (MOD − CNTIN). 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 FTM counter counts up until it reaches MOD and then counts down until it reaches CNTIN.

The CHF bit is set and channel (n) interrupt is generated (if CHIE = 1) at the channel (n) match (FTM counter = CnV) when the FTM counting is down (at the begin of the pulse width) and when the FTM 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 aligned with the value of CNTIN.

The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).

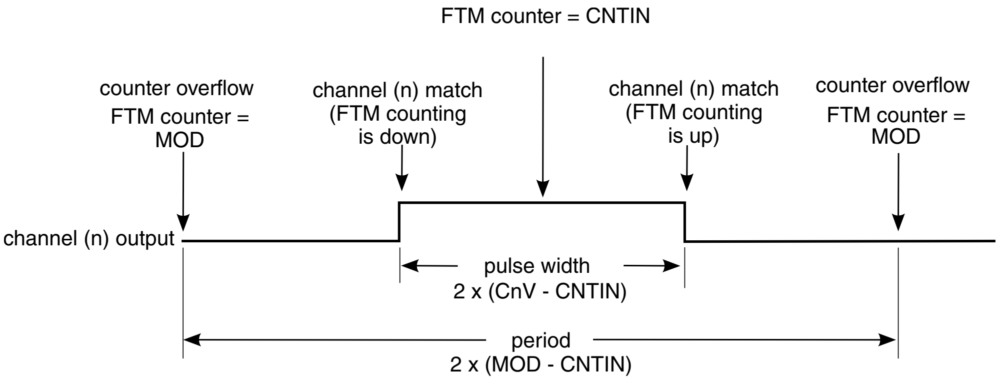


Figure 34-20. CPWM period and pulse width with ELSB:ELSA = 1:0

If (ELSB:ELSA = 0:0) when the FTM counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated (if CHIE = 1), however the channel (n) output is not controlled by FTM.

If (ELSB:ELSA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. See the following figure.

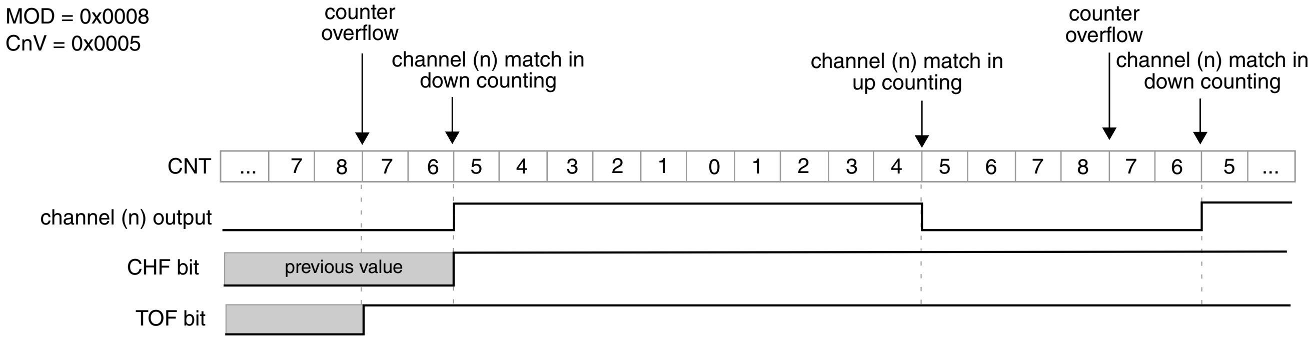


Figure 34-21. CPWM signal with ELSB:ELSA = 1:0

If (ELSB:ELSA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. See the following figure.

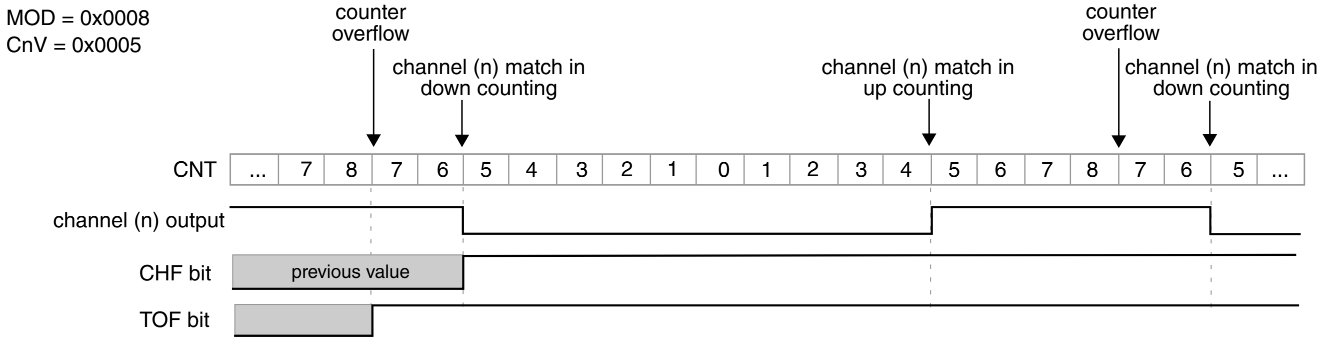


Figure 34-22. CPWM signal with ELSB:ELSA = X:1

If (CnV = 0x0000) or CnV is a negative value, that is (CnV[15] = 1), then the channel (n) output is a 0% duty cycle CPWM signal and CHF bit is not set even when there is the channel (n) match.

If CnV is a positive value, that is (CnV[15] = 0), (CnV ≥ MOD), and (MOD ≠ 0x0000), then the channel (n) output is a 100% duty cycle CPWM signal and CHF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MOD is 0x0001 through 0x7FFE, 0x7FFF if you do not need to generate a 100% duty cycle CPWM signal. This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

34.5.9 Combine mode

The Combine mode is selected when:

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 1, and
- CPWMS = 0

In Combine 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 − CNTIN + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|).

The channel (n) CHF bit is set and its interrupt is generated, if channel (n) CHIE = 1, at the channel (n) match (FTM counter = C(n)V). The channel (n+1) CHF bit is set and its interrupt is generated, if channel (n+1) CHIE = 1, at the channel (n+1) match (FTM counter = C(n+1)V).

If channel (n) ELSB:ELSA = 1:0, then the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n+1)V). It is forced high at the channel (n) match (FTM counter = C(n)V). See the following figure.

If channel (n) ELSB:ELSA = X:1, then the channel (n) output is forced high at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n+1)V). It is forced low at the channel (n) match (FTM counter = C(n)V). See the following figure.

In Combine mode, the channel (n+1) ELSB:ELSA bits are not used in the generation of the channels (n) and (n+1) output. However, if channel (n) ELSB:ELSA = 0:0, then the channel (n) output is not controlled by FTM, and if channel (n+1) ELSB:ELSA = 0:0, then the channel (n+1) output is not controlled by FTM.

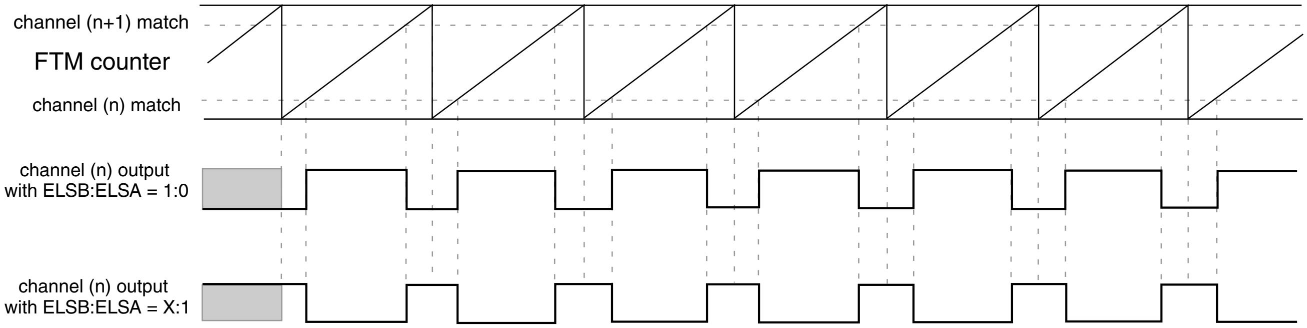

channel (n+1) match
FTM counter
channel (n) match
channel (n) output with ELSB:ELSA = 1:0
channel (n) output with ELSB:ELSA = X:1

Figure 34-23. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

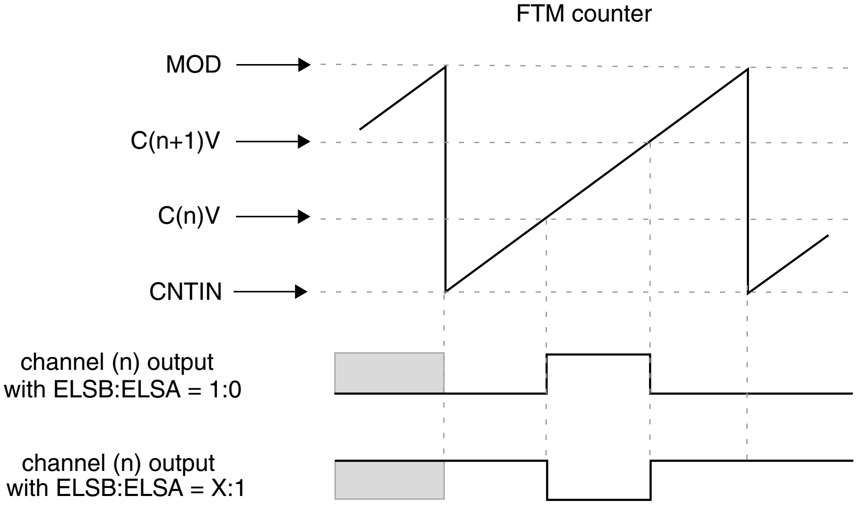

FTM counter
MOD
C(n+1)V
C(n)V
CNTIN
channel (n) output with ELSB:ELSA = 1:0
channel (n) output with ELSB:ELSA = X:1

Figure 34-24. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V < C(n+1)V)

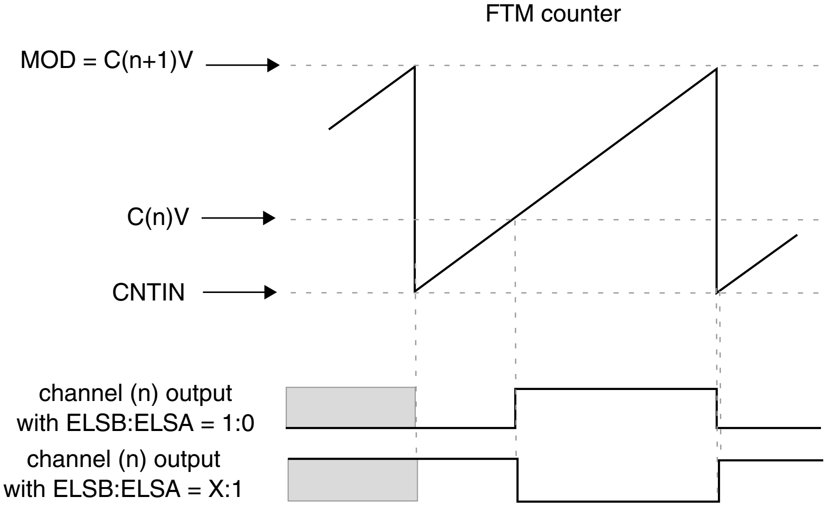

FTM counter
MOD = C(n+1)V
C(n)V
CNTIN
channel (n) output with ELSB:ELSA = 1:0
channel (n) output with ELSB:ELSA = X:1

Figure 34-25. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n+1)V = MOD)

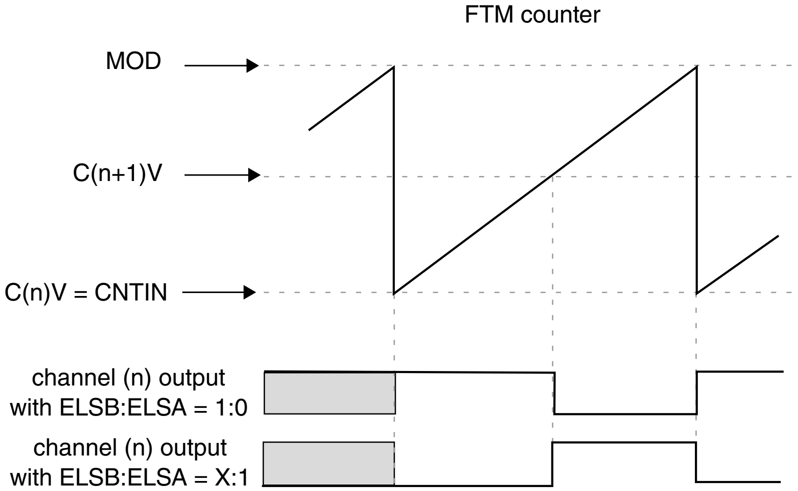


Figure 34-26. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD)

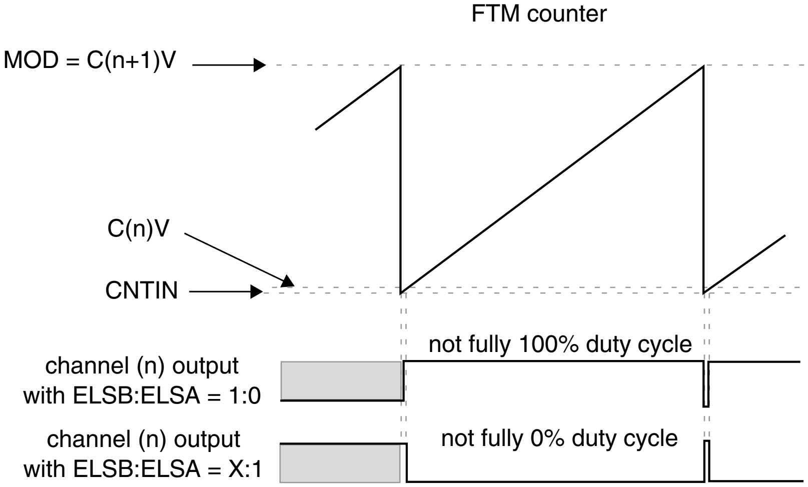


Figure 34-27. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n)V is Almost Equal to CNTIN) and (C(n+1)V = MOD)

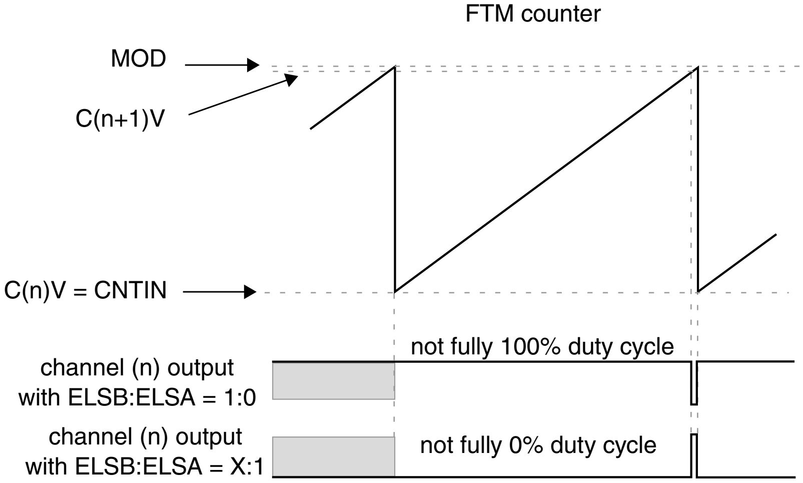


Figure 34-28. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD) and (C(n+1)V is Almost Equal to MOD)

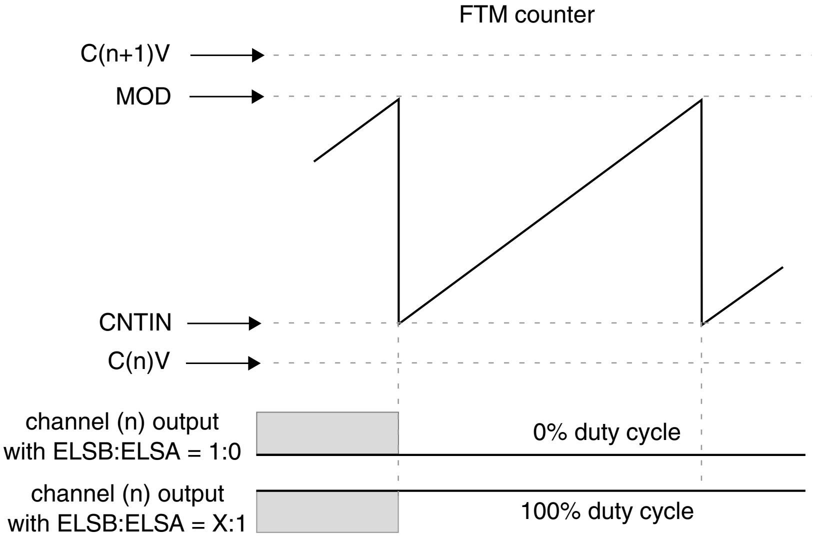


Figure 34-29. Channel (n) output if C(n)V and C(n+1)V are not between CNTIN and MOD

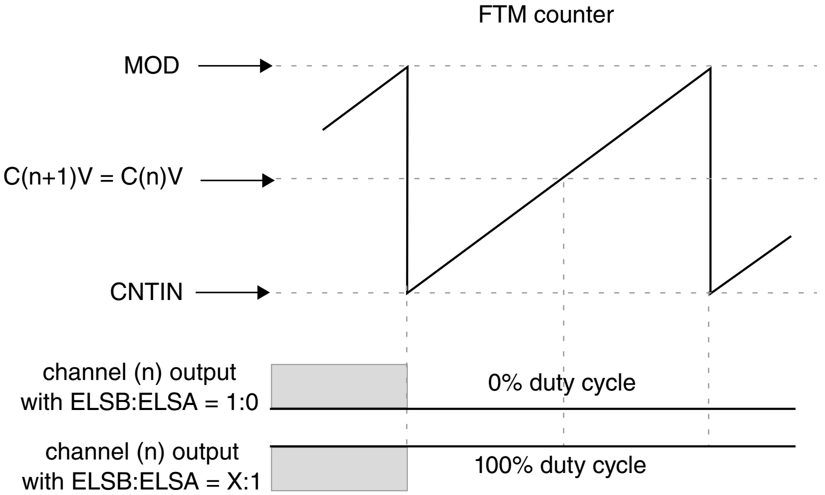


Figure 34-30. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V = C(n+1)V)

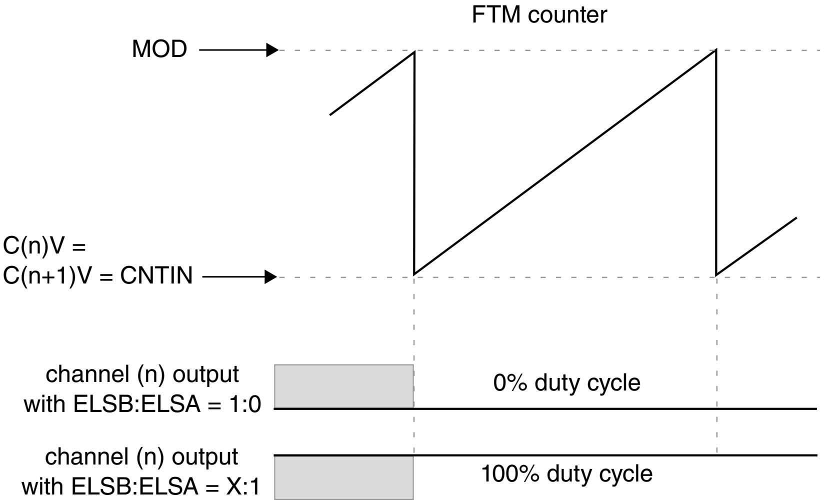


Figure 34-31. Channel (n) output if (C(n)V = C(n+1)V = CNTIN)

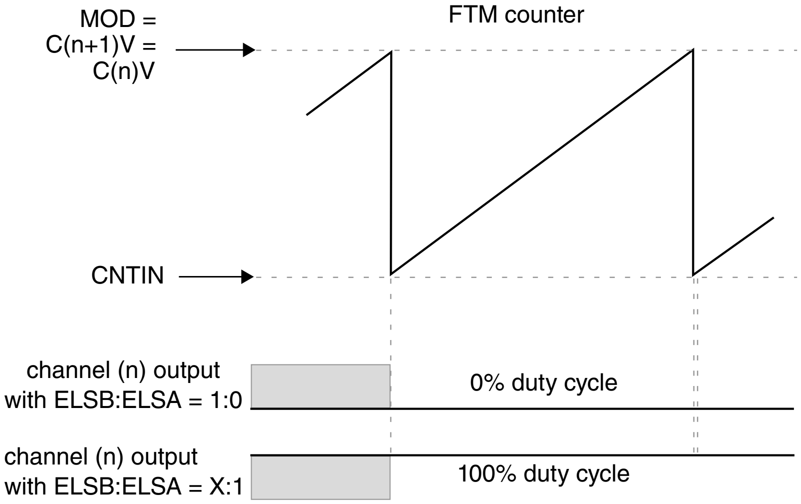


Figure 34-32. Channel (n) output if (C(n)V = C(n+1)V = MOD)

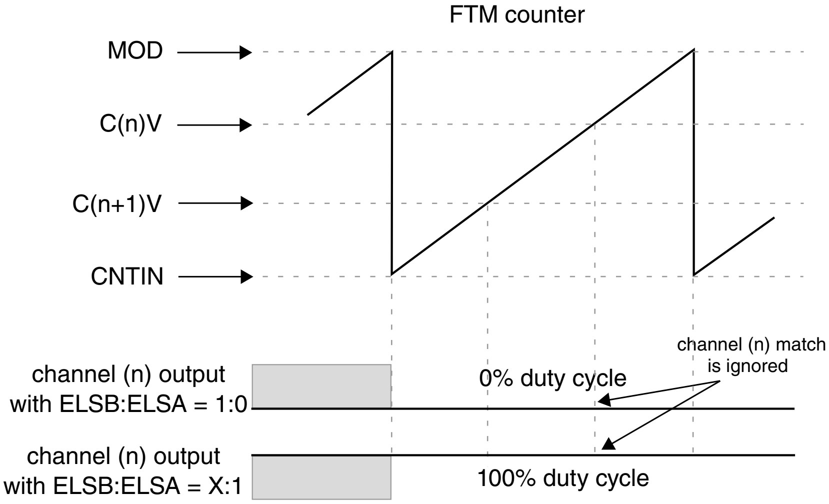


Figure 34-33. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V > C(n+1)V)

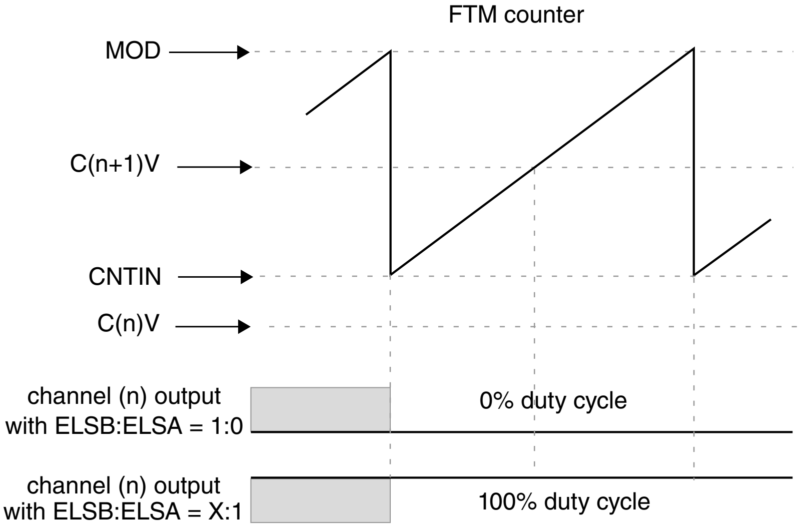


Figure 34-34. Channel (n) output if (C(n)V < CNTIN) and (CNTIN < C(n+1)V < MOD)

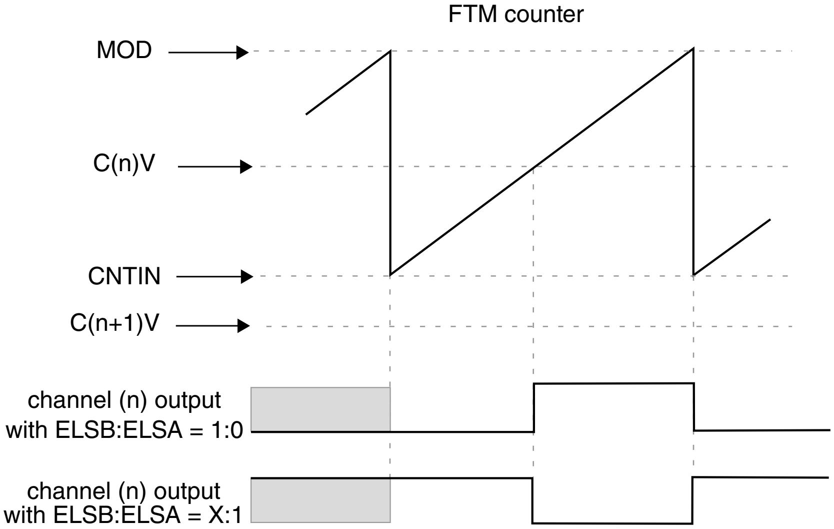


Figure 34-35. Channel (n) output if (C(n+1)V < CNTIN) and (CNTIN < C(n)V < MOD)

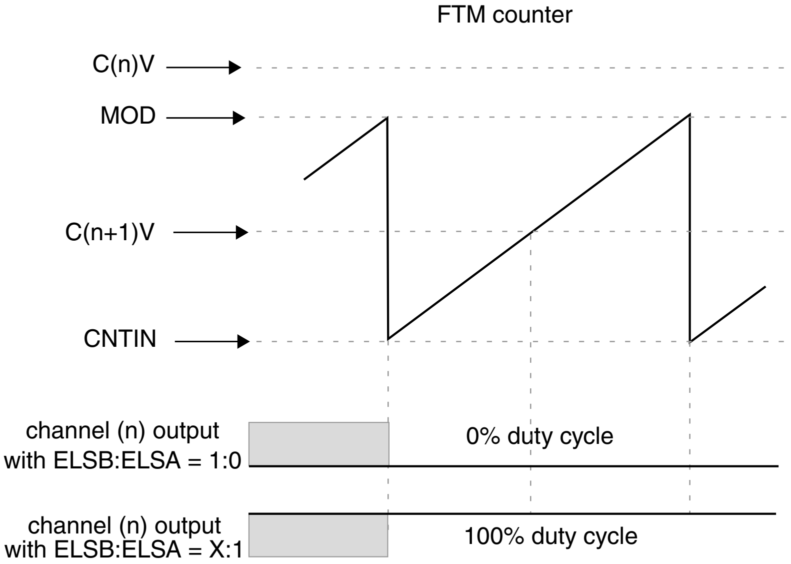


Figure 34-36. Channel (n) output if (C(n)V > MOD) and (CNTIN < C(n+1)V < MOD)

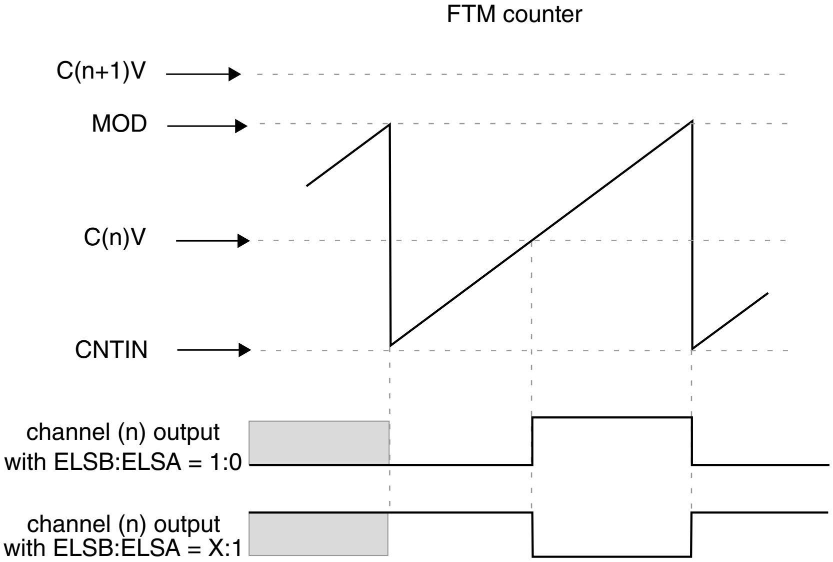


Figure 34-37. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V < MOD)

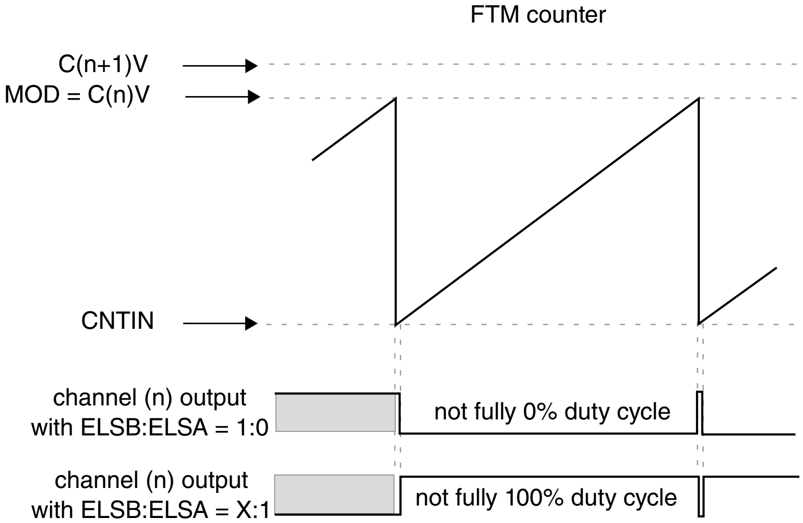


Figure 34-38. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V = MOD)

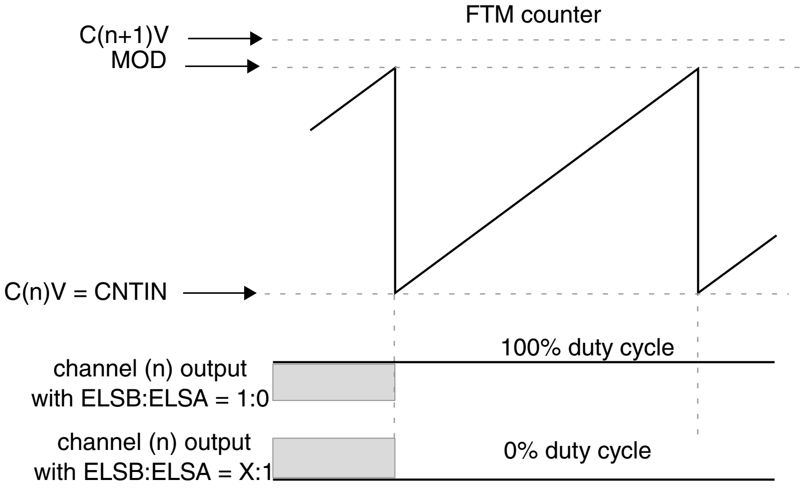


Figure 34-39. Channel (n) output if (C(n)V = CNTIN) and (C(n+1)V > MOD)

34.5.9.1 Asymmetrical PWM

In Combine mode and Modified Combine PWM Mode, the PWM first edge (channel (n) match: FTM counter = C(n)V) is independent of the PWM second edge (channel (n+1) match: FTM counter = C(n+1)V).

34.5.10 Modified Combine PWM Mode

The Modified Combine PWM mode is selected when:

- DECAPEN = 0
- MCOMBINE = 1

- COMBINE = 1, and
- CPWMS = 0

The Modified Combine PWM mode is intended to support the generation of PWM signals where the period is not modified while the signal is being generated, but the duty cycle will be varied. In this mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output. Thus, the channel (n) match edge is fixed and the channel (n+1) match edge can be varied.

When a pair of channels is in Modified Combine PWM mode, it is recommend that the other pairs also be in Modified Combine PWM mode.

In the Modified Combine PWM mode, assuming that CNTIN ≥ 0, MOD > 0, and CNTIN < MOD:

- The PWM period is determined by (MOD - CNTIN + 0x0001);
- The channel (n) PWM duty cycle is calculated according to the following table.

Table 34-8. Modified Combine PWM Mode - Duty Cycles

Channel (n) PWM Duty Cycle	Condition
0% duty cycle	For CNTIN ≤ (C(n)V and C(n+1)V) ≤ MOD: C(n)V = C(n+1)V
duty cycle between 0% and 100%	For CNTIN ≤ (C(n)V and C(n+1)V) ≤ MOD: • if (C(n)V < C(n+1)V), then the duty cycle is (C(n+1)V - C(n)V) • if (C(n)V > C(n+1)V), then the duty cycle is [(MOD - C(n)V) + (C(n+1)V - CNTIN) + 1]
100% duty cycle	CNTIN ≤ C(n)V ≤ MOD and C(n+1)V > MOD

The channel (n) CHF bit is set and its interrupt is generated, if channel (n) CHIE = 1, at the channel (n) match (FTM counter = C(n)V). The channel (n+1) CHF bit is set and its interrupt is generated, if channel (n+1) CHIE = 1, at the channel (n+1) match (FTM counter = C(n+1)V).

If channel (n) ELSB:ELSA = 1:0, then the channel (n) output is forced high at the channel (n) match (FTM counter = C(n)V) and it is forced low at the channel (n+1) match (FTM counter = C(n+1)V).

If channel (n) ELSB:ELSA = X:1, then the channel (n) output is forced low at the channel (n) match (FTM counter = C(n)V) and it is forced high at the channel (n+1) match (FTM counter = C(n+1)V).

In Modified Combine PWM mode, the channel (n+1) ELSB:ELSA bits are not used in the generation of the channels (n) and (n+1) output. However, if channel (n) ELSB:ELSA = 0:0, then the channel (n) output is not controlled by FTM, and if channel (n+1) ELSB:ELSA = 0:0, then the channel (n+1) output is not controlled by FTM.

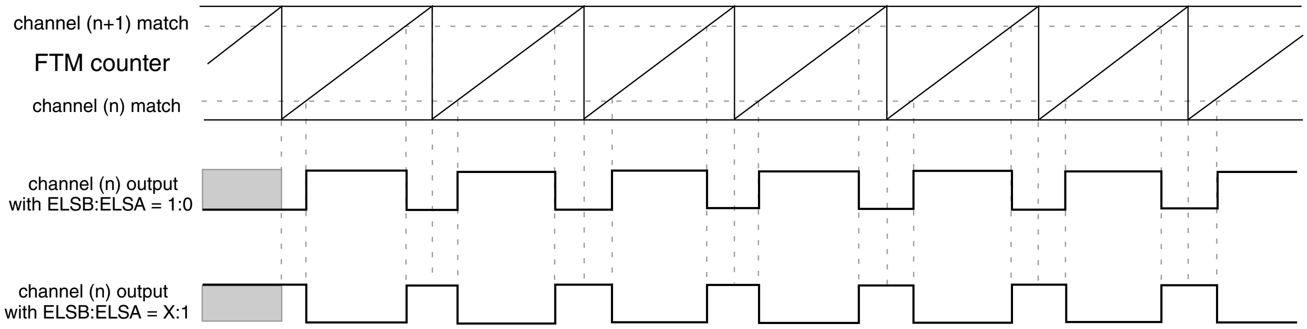


Figure 34-40. Modified Combine PWM Mode

The Modified Combine PWM mode allows the offset addition of the duty cycle, thus, in some cases, the C(n+1)V match can happen on the next FTM counter period. For CNTIN ≥ 0, MOD > 0, and CNTIN < MOD, this situation happens when C(n)V > C(n+1)V.

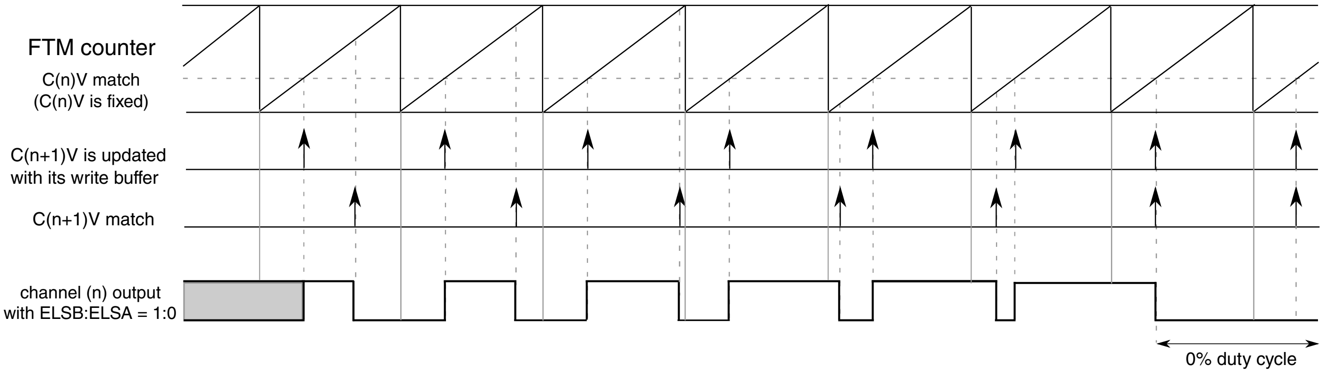


Figure 34-41. Modified Combine PWM Mode Examples

If more than one pair of channels are configured in Modified Combine PWM Mode, it is possible to fix an offset for the channel (n) match edge of each pair with respect to other pairs. This behavior is useful in the generation of lighting PWM control signals where it is desirable that edges are not coincident with each other pair to help eliminate noise generation. The C(n)V register value is the shift of the PWM pulse with respect to the beginning of FTM counter period (FTM counter = CNTIN).

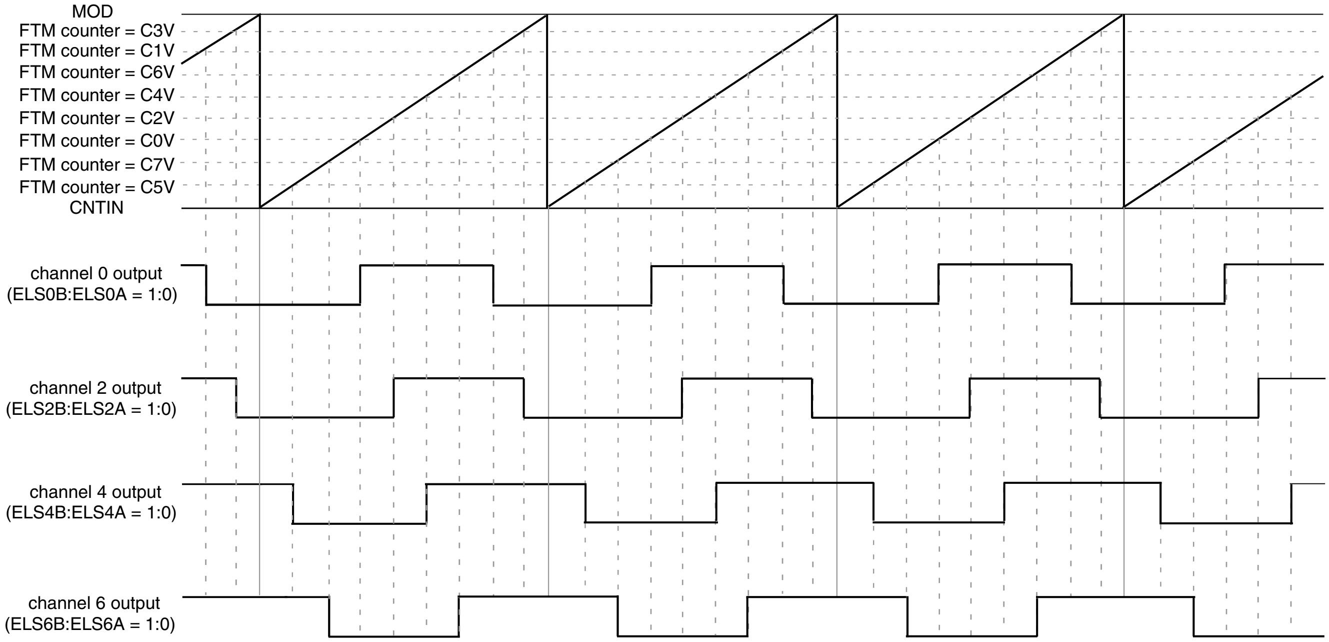


Figure 34-42. Example of Four Pairs of Channels in Modified Combine PWM Mode

34.5.10.1 Synchronization

In the Modified Combine Mode, the following registers should be updated when the FTM counter clock is disabled (CLKS[1:0] = 0:0).

- CNTIN (CNTIN register update)
- MOD (MOD and HCR registers update)
- C(n)V and C(n+1)V (CnV register update)

In the Modified Combine Mode, if (FTMEN = 1), (CLKS[1:0] ≠ 0:0), and there was a write to the register C(n+1)V, then the register C(n+1)V is updated with its write buffer value on the next channel (n) match (FTM counter = C(n)V). This feature allows to vary the PWM duty cycle value in this mode.

NOTE

In the Modified Combine Mode, the bit SYNCEN(n) should be zero bit for the channels (n) and (n+1). So, the following features are not available for this mode.

- C(n)V and C(n+1)V register synchronization
- Reload Points
- Global Load

34.5.11 Complementary Mode

The Complementary mode is selected when:

- DECAPEN = 0
- COMP = 1

In Complementary mode, the channel (n+1) output is the inverse of the channel (n) output.

NOTE

The Complementary Mode is not available on Output Compare mode.

The channel (n+1) output is the same as the channel (n) output when:

- DECAPEN = 0
- COMP = 0
- channels (n) and (n+1) are on Combine Mode or Modified Combine PWM Mode

The channel (n+1) output is independent from channel (n) output when:

- DECAPEN = 0
- COMP = 0
- channel (n) is on Output Compare Mode, EPWM or CPWM

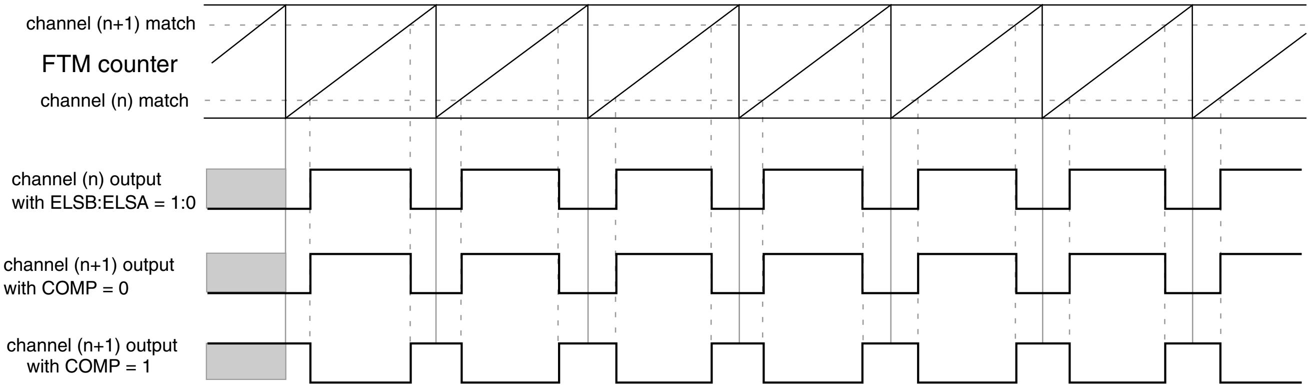


Figure 34-43. Channel (n+1) output in Complementary mode with (ELSB:ELSA = 1:0)

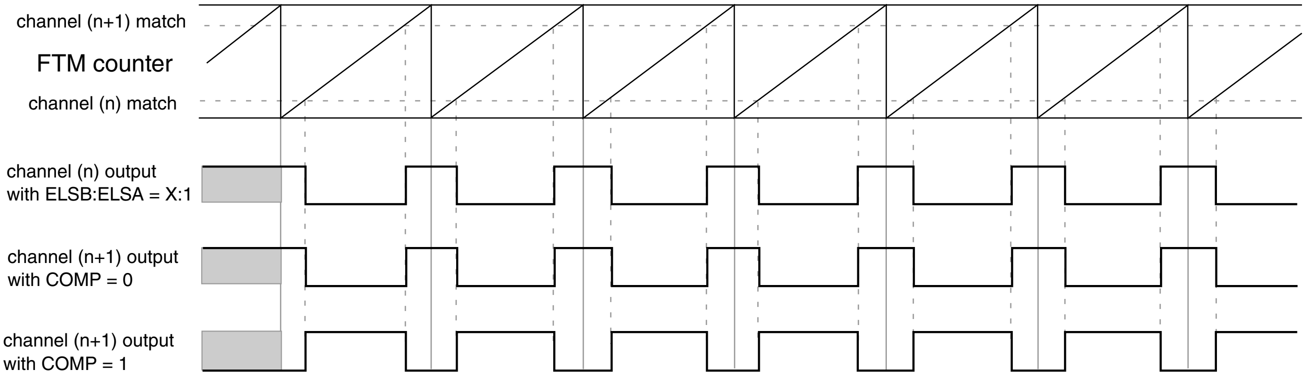

channel (n+1) match
FTM counter
channel (n) match
channel (n) output with ELSB:ELSA = X:1
channel (n+1) output with COMP = 0
channel (n+1) output with COMP = 1

Figure 34-44. Channel (n+1) output in Complementary mode with (ELSB:ELSA = X:1)

34.5.12 Registers updated from write buffers

34.5.12.1 CNTIN register update

The following table describes when CNTIN register is updated:

Table 34-9. CNTIN register update

When	Then CNTIN register is updated
CLKS[1:0] = 0:0	When CNTIN register is written, independent of FTMEN bit.
• FTMEN = 0, or • CNTINC = 0	At the next FTM input clock after CNTIN was written.
• FTMEN = 1, • SYNCMODE = 1, and • CNTINC = 1	By the CNTIN register synchronization.
• CNTINC = 1, and • LDOK = 1	By the Reload Points.

34.5.12.2 MOD and HCR registers update

The following table describes when MOD or HCR registers are updated:

Table 34-10. MOD and HCR updates

When	Then MOD or HCR is updated
CLKS[1:0] = 0:0	When MOD (or HCR) is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the CPWMS bit, that is:

Table continues on the next page...

Table 34-10. MOD and HCR updates (continued)

When	Then MOD or HCR is updated
	• If the selected mode is not CPWM then MOD (or HCR) is updated after MOD (or HCR) register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM then MOD (or HCR) register is updated after MOD (or HCR) register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	By the MOD register synchronization. HCR follows the same procedure of MOD register in this case.
• LDOK = 1	By the Reload Points.

34.5.12.3 CnV register update

The following table describes when CnV register is updated:

Table 34-11. CnV register update

When	Then CnV register is updated
CLKS[1:0] = 0:0	When CnV register is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the selected mode, that is: • If the selected mode is Output Compare, then CnV register is updated on the next FTM counter change, 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 FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	According to the selected mode, that is: • If the selected mode is output compare then CnV register is updated according to the SYNCEN bit. If (SYNCEN = 0) then CnV register is updated after CnV register was written at the next change of the FTM counter, the end of the prescaler counting. If (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization. • If the selected mode is not output compare and (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization.
• SYNCEN = 1, and • LDOK = 1	By the Reload Points.

34.5.13 PWM synchronization

The PWM synchronization provides an opportunity to update the MOD, HCR, CNTIN, CnV, OUTMASK, INVCTRL and SWOCTRL registers with their buffered value and force the FTM counter to the CNTIN register value.

Note

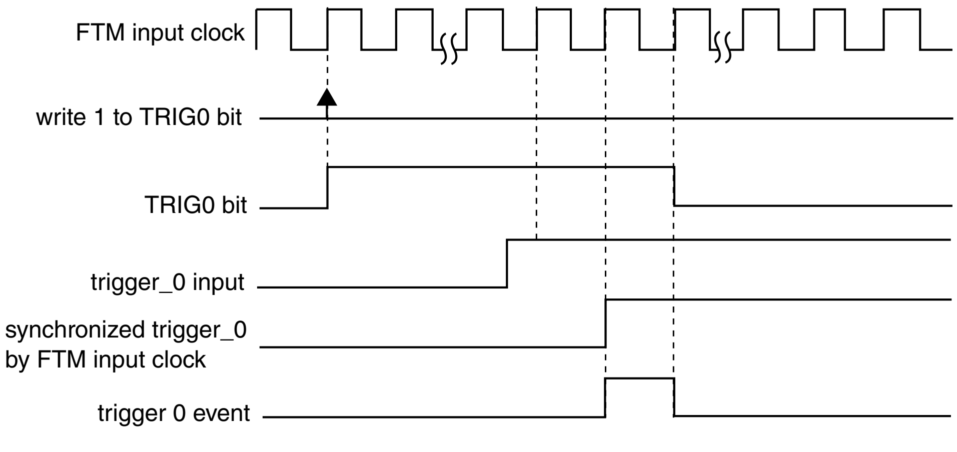
> The legacy PWM synchronization (SYNCMODE = 0) is a subset of the enhanced PWM synchronization (SYNCMODE = 1). Thus, only the enhanced PWM synchronization must be used.

34.5.13.1 Hardware trigger

Three hardware trigger signal inputs of the FTM module are enabled when TRIGn = 1, where n = 0, 1 or 2 corresponding to each one of the input signals, respectively. The hardware trigger input n is synchronized by the FTM input clock. The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs.

If (HWTRIGMODE = 0) then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case, if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs, then only TRIG1 bit is cleared. If a trigger n event occurs together with a write setting TRIGn bit, then the synchronization is initiated, but TRIGn bit remains set due to the write operation.



Note
All hardware trigger inputs have the same behavior.

Figure 34-45. Hardware trigger event with HWTRIGMODE = 0

If HWTRIGMODE = 1, then the TRIGn bit is only cleared when 0 is written to it.

NOTE

The HWTRIGMODE bit must be 1 only with enhanced PWM synchronization (SYNCMODE = 1).

34.5.13.2 Software trigger

A software trigger event occurs when 1 is written to the SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization, initiated by the software event, is completed.

If another software trigger event occurs (by writing another 1 to the SWSYNC bit) at the same time the PWM synchronization initiated by the previous software trigger event is ending, a new PWM synchronization is started and the SWSYNC bit remains equal to 1.

If SYNCMODE = 0 then the SWSYNC bit is also cleared by FTM according to PWMSYNC and REINIT bits. In this case if (PWMSYNC = 1) or (PWMSYNC = 0 and REINIT = 0) then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see Synchronization Points and the following figure. If (PWMSYNC = 0) and (REINIT = 1) then SWSYNC bit is cleared when the software trigger event occurs.

If SYNCMODE = 1 then the SWSYNC bit is also cleared by FTM according to the SWRSTCNT bit. If SWRSTCNT = 0 then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see the following figure. If SWRSTCNT = 1 then SWSYNC bit is cleared when the software trigger event occurs.

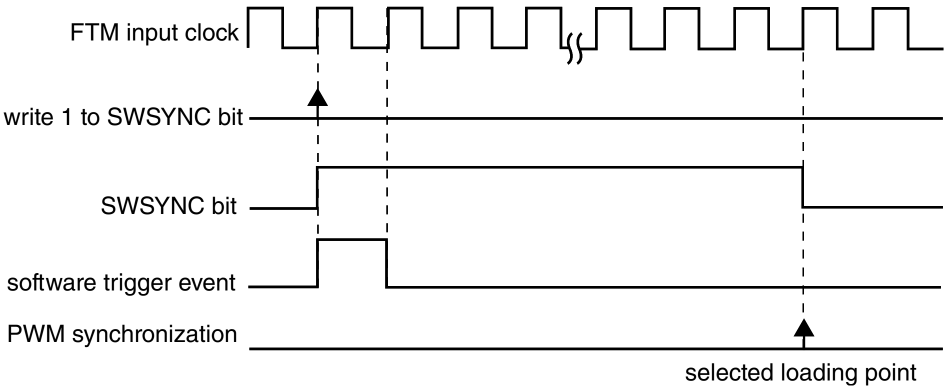


Figure 34-46. Software trigger event

34.5.13.3 Synchronization Points

The synchronization points are points where the registers can be updated with their write buffer by PWM synchronization. These synchronization points are safe points because guarantee smooth transitions in the generated PWM signals.

In Up counting, the synchronization points are when the FTM counter changes from MOD to CNTIN. In this case, the synchronization points are enabled if (CNTMIN = 1) or (CNTMAX = 1).

In Up-down counting, the synchronization points are:

- if (CNTMAX = 1), when the FTM counter changes from (MOD) to (MOD - 1);
- if (CNTMIN = 1), when the FTM counter changes from (CNTIN) to (CNTIN + 1).

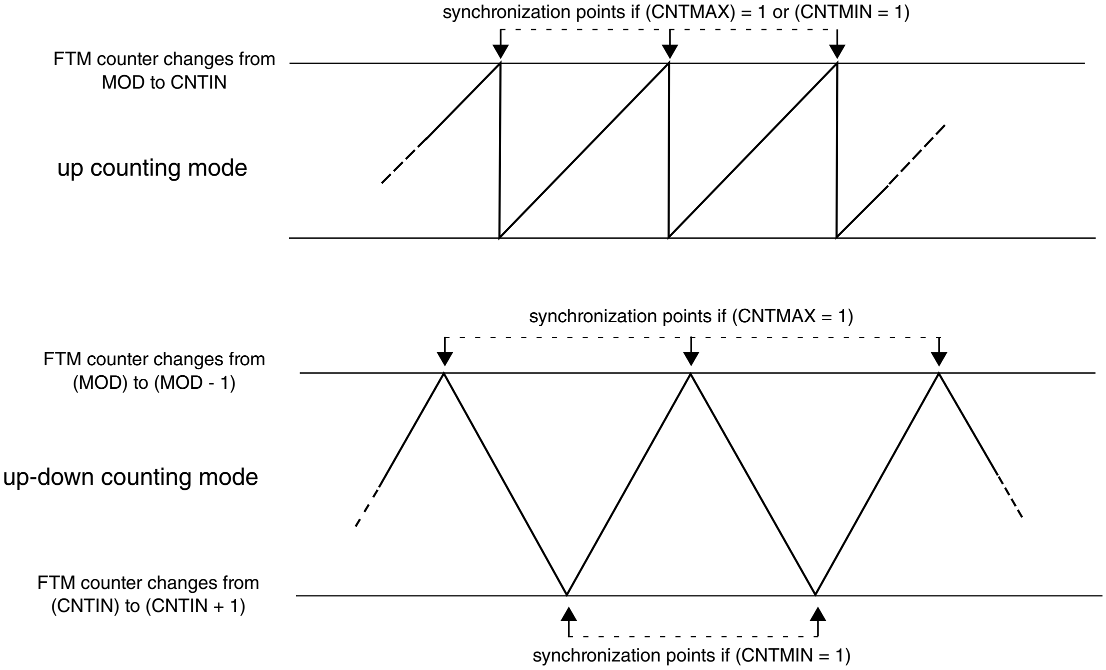

synchronization points if (CNTMAX) = 1 or (CNTMIN = 1)
FTM counter changes from MOD to CNTIN
up counting mode
synchronization points if (CNTMAX = 1)
FTM counter changes from (MOD) to (MOD - 1)
up-down counting mode
FTM counter changes from (CNTIN) to (CNTIN + 1)
synchronization points if (CNTMIN = 1)

Figure 34-47. Synchronization Points

34.5.13.4 MOD register synchronization

The MOD register synchronization updates the MOD register with its buffer value. This synchronization is enabled if (FTMEN = 1).

The MOD register synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the MOD register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the MOD register synchronization depends on SWWRBUF, SWRSTCNT, HWWRBUF, and HWRSTCNT bits according to this flowchart:

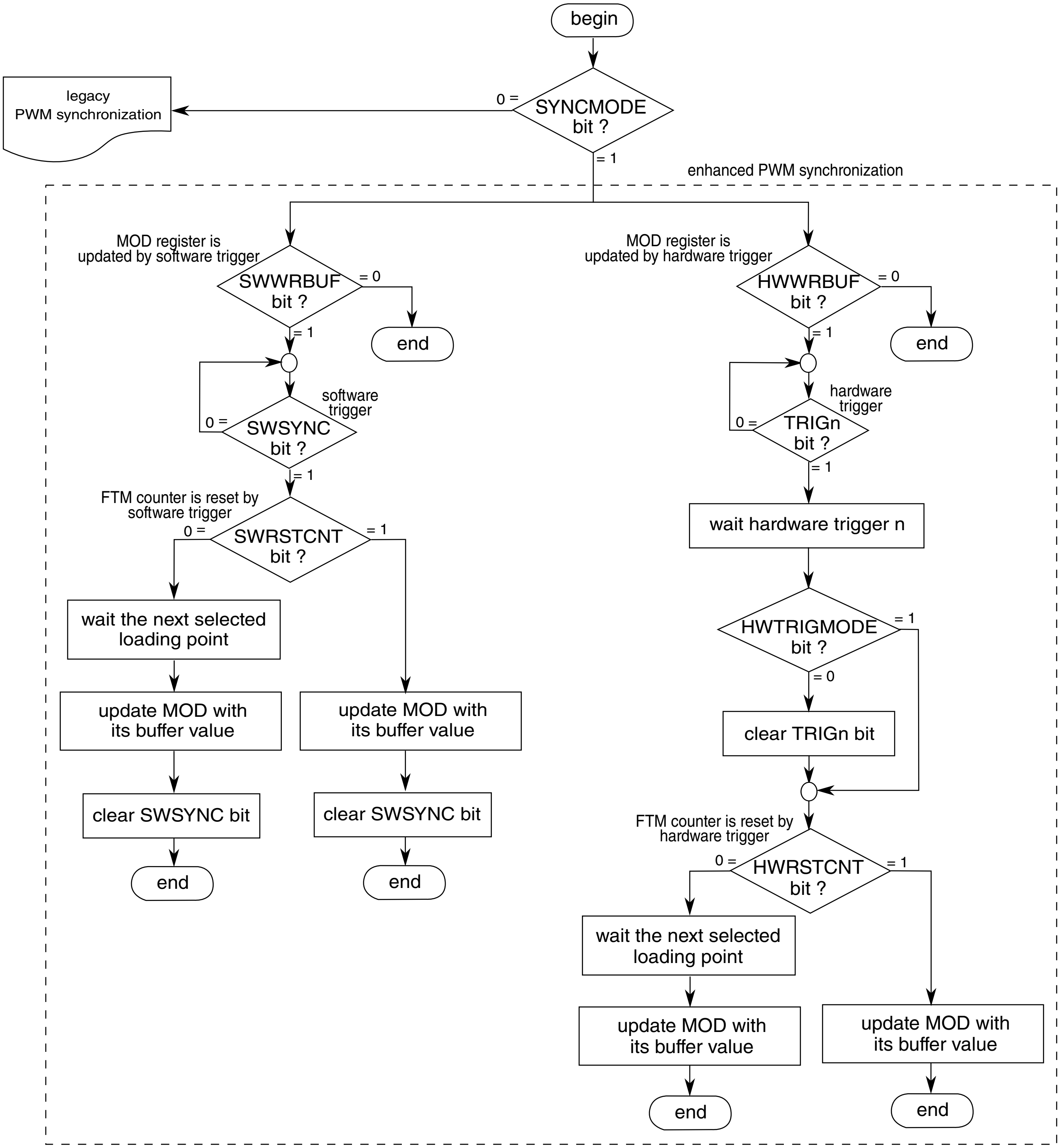


Figure 34-48. MOD register synchronization flowchart

In the case of legacy PWM synchronization, the MOD register synchronization depends on PWMSYNC and REINIT bits according to the following description.

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 0), then this synchronization is made on the next selected loading point after an enabled trigger event takes place. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the trigger enable bit (TRIGn) is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

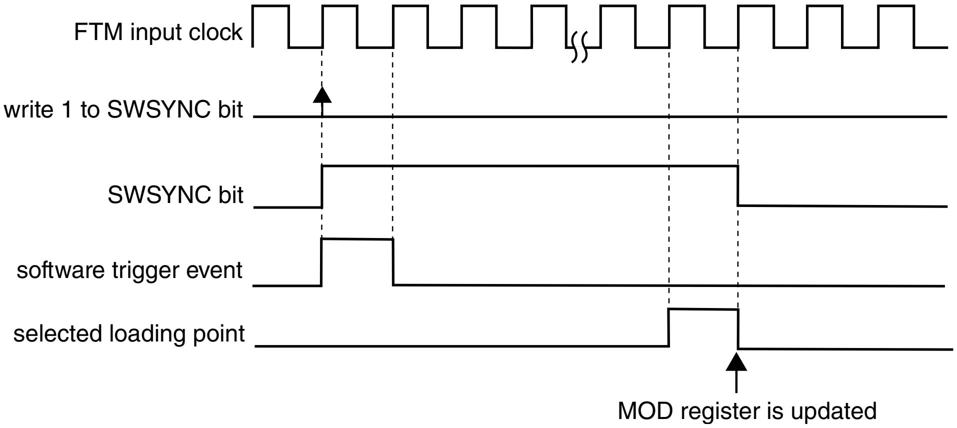


Figure 34-49. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 0), and software trigger was used

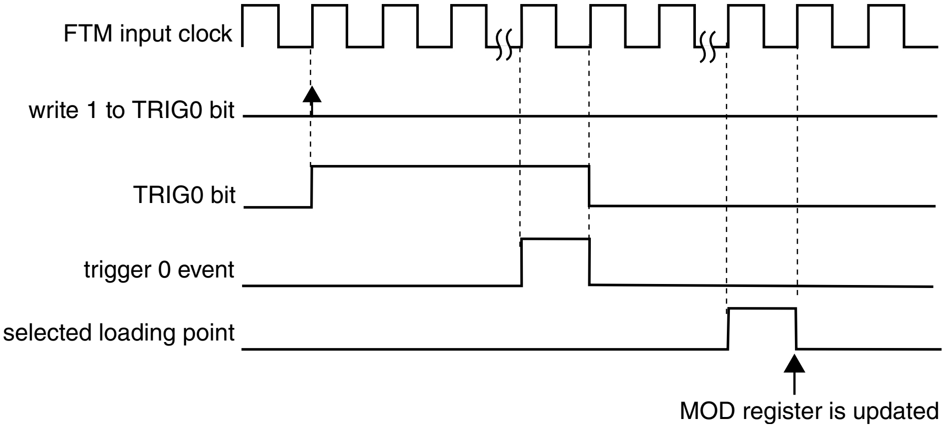


Figure 34-50. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 0), and a hardware trigger was used

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 1), then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

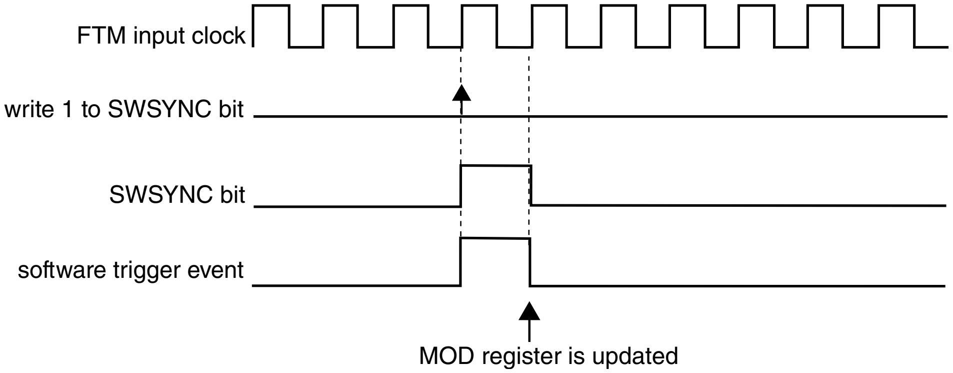


Figure 34-51. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 1), and software trigger was used

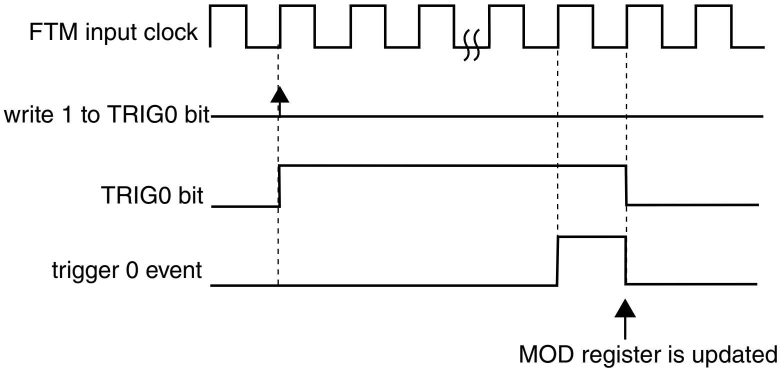


Figure 34-52. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 1), and a hardware trigger was used

If (SYNCMODE = 0) and (PWMSYNC = 1), then this synchronization is made on the next selected loading point after the software trigger event takes place. The SWSYNC bit is cleared on the next selected loading point:

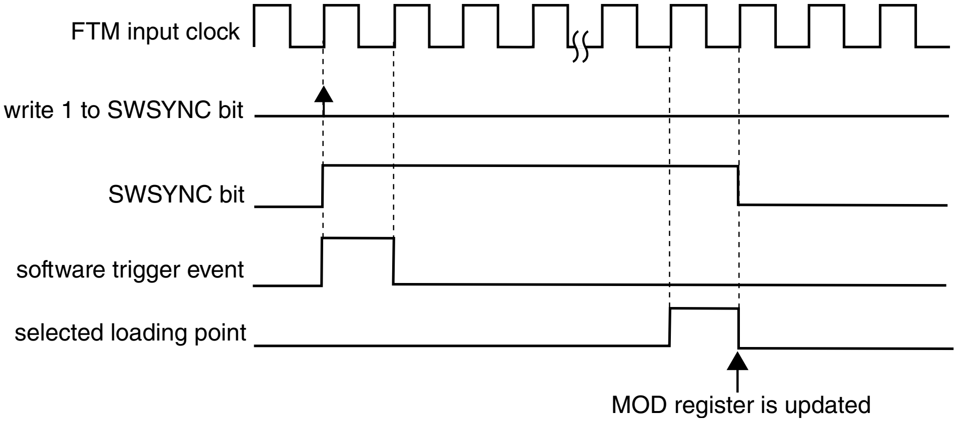


Figure 34-53. MOD synchronization with (SYNCMODE = 0) and (PWMSYNC = 1)

34.5.13.5 CNTIN register synchronization

The CNTIN register synchronization updates the CNTIN register with its buffer value.

This synchronization is enabled if (FTMEN = 1), (SYNCMODE = 1), and (CNTINC = 1). The CNTIN register synchronization can be done only by the enhanced PWM synchronization (SYNCMODE = 1). The synchronization mechanism is the same as the MOD register synchronization done by the enhanced PWM synchronization; see MOD register synchronization.

34.5.13.6 C(n)V and C(n+1)V register synchronization

The C(n)V and C(n+1)V registers synchronization updates the C(n)V and C(n+1)V registers with their buffer values.

This synchronization is enabled if (FTMEN = 1) and (SYNCEN = 1). The synchronization mechanism is the same as the MOD register synchronization. However, it is expected that the C(n)V and C(n+1)V registers be synchronized only by the enhanced PWM synchronization (SYNCMODE = 1).

34.5.13.7 OUTMASK register synchronization

The OUTMASK register synchronization updates the OUTMASK register with its buffer value.

The OUTMASK register can be updated at each rising edge of FTM input clock (SYNCHOM = 0), by the enhanced PWM synchronization (SYNCHOM = 1 and SYNCMODE = 1) or by the legacy PWM synchronization (SYNCHOM = 1 and SYNCMODE = 0). However, it is expected that the OUTMASK register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the OUTMASK register synchronization depends on SWOM and HWOM bits. See the following flowchart:

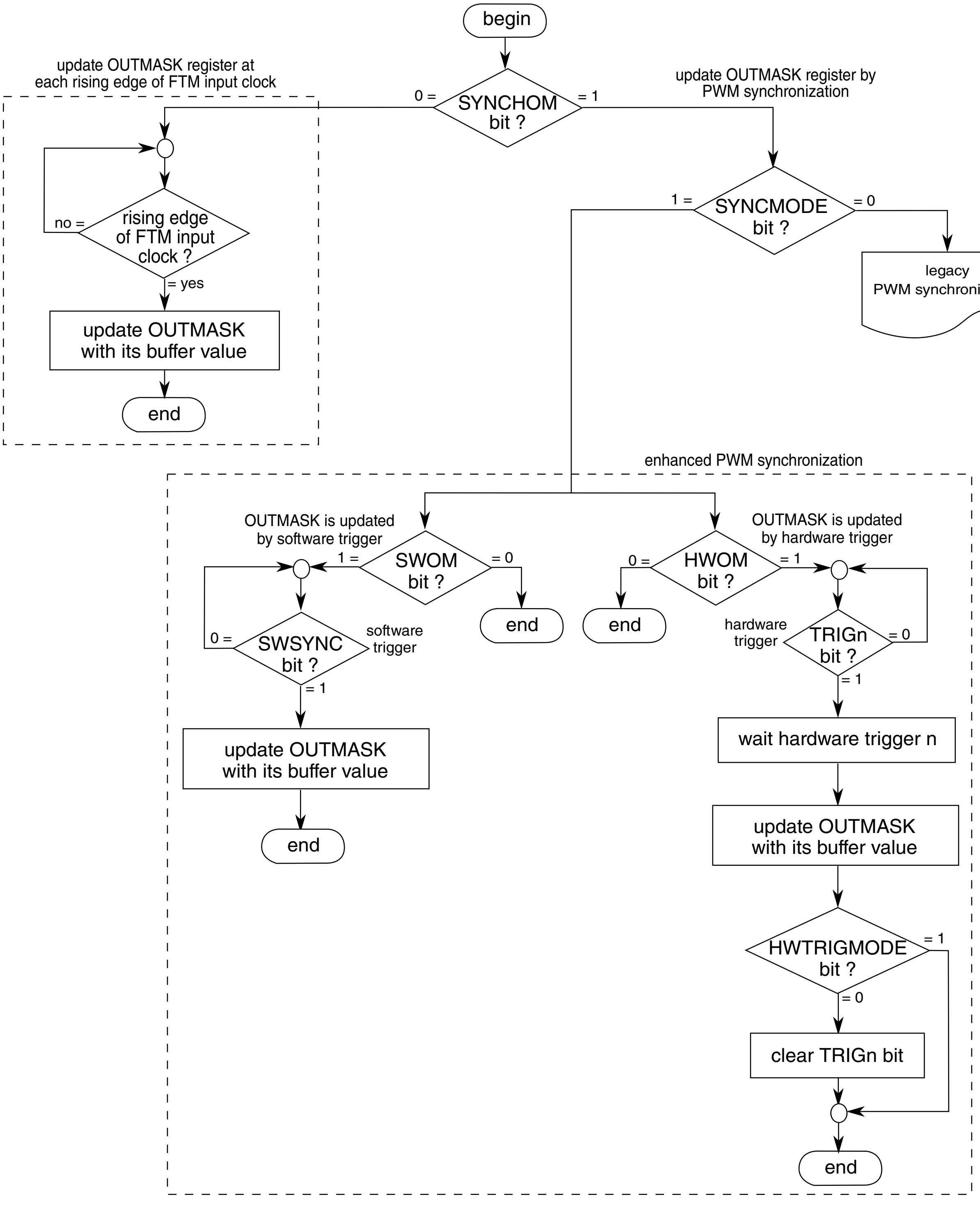

begin
SYNCHOM bit ?
0 =
= 1
update OUTMASK register at each rising edge of FTM input clock
update OUTMASK register by PWM synchronization
rising edge of FTM input clock ?
no =
= yes
update OUTMASK with its buffer value
end
SYNCMODE bit ?
1 =
= 0
legacy PWM synchronization
enhanced PWM synchronization
OUTMASK is updated by software trigger
SWOM bit ?
1 =
= 0
end
SWSYNC bit ?
0 =
software trigger
= 1
update OUTMASK with its buffer value
end
OUTMASK is updated by hardware trigger
HWOM bit ?
0 =
= 1
end
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update OUTMASK with its buffer value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 34-54. OUTMASK register synchronization flowchart

In the case of legacy PWM synchronization, the OUTMASK register synchronization depends on PWMSYNC bit according to the following description.

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 0), then this synchronization is done on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

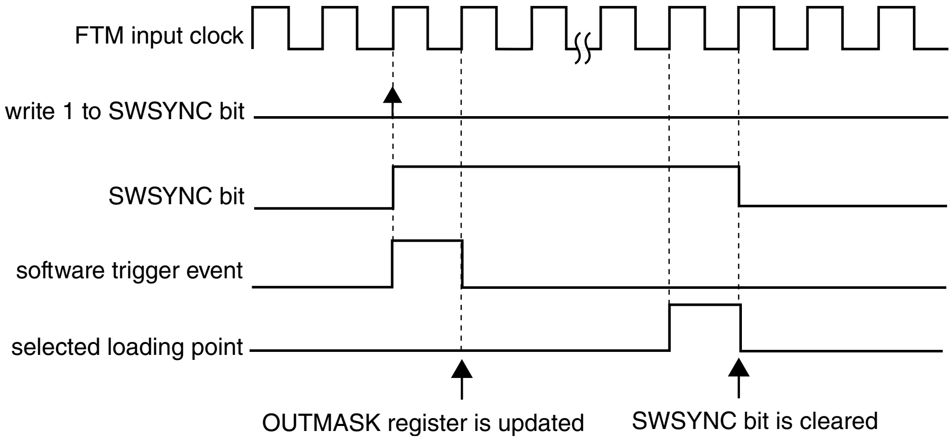


Figure 34-55. OUTMASK synchronization with (SYNCMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0) and software trigger was used

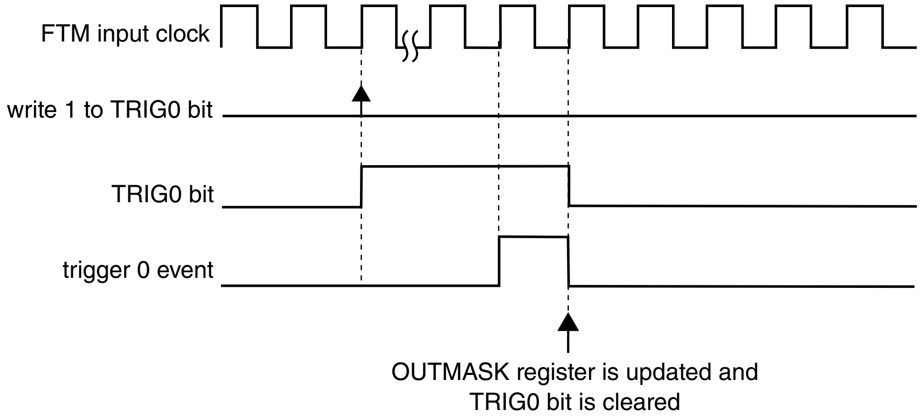


Figure 34-56. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 1), then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger. An example with a hardware trigger follows.

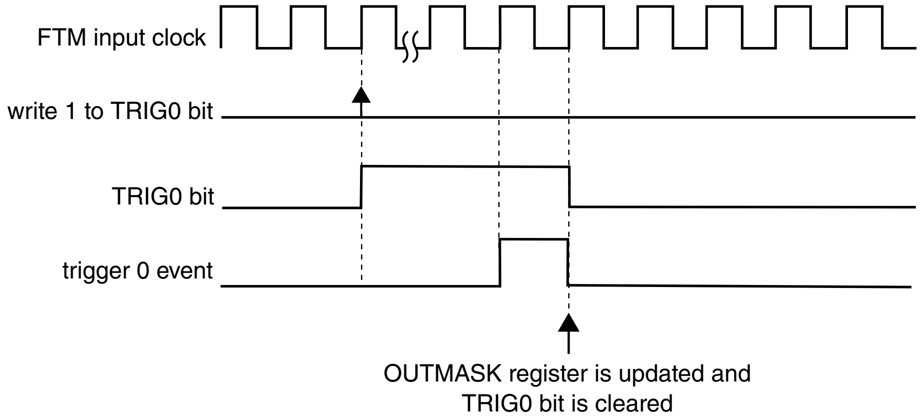


Figure 34-57. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 1), and a hardware trigger was used

34.5.13.8 INVCTRL register synchronization

The INVCTRL register synchronization updates the INVCTRL register with its buffer value.

The INVCTRL register can be updated at each rising edge of FTM input clock (INVC = 0) or by the enhanced PWM synchronization (INVC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the INVCTRL register synchronization depends on SWINVC and HWINVC bits.

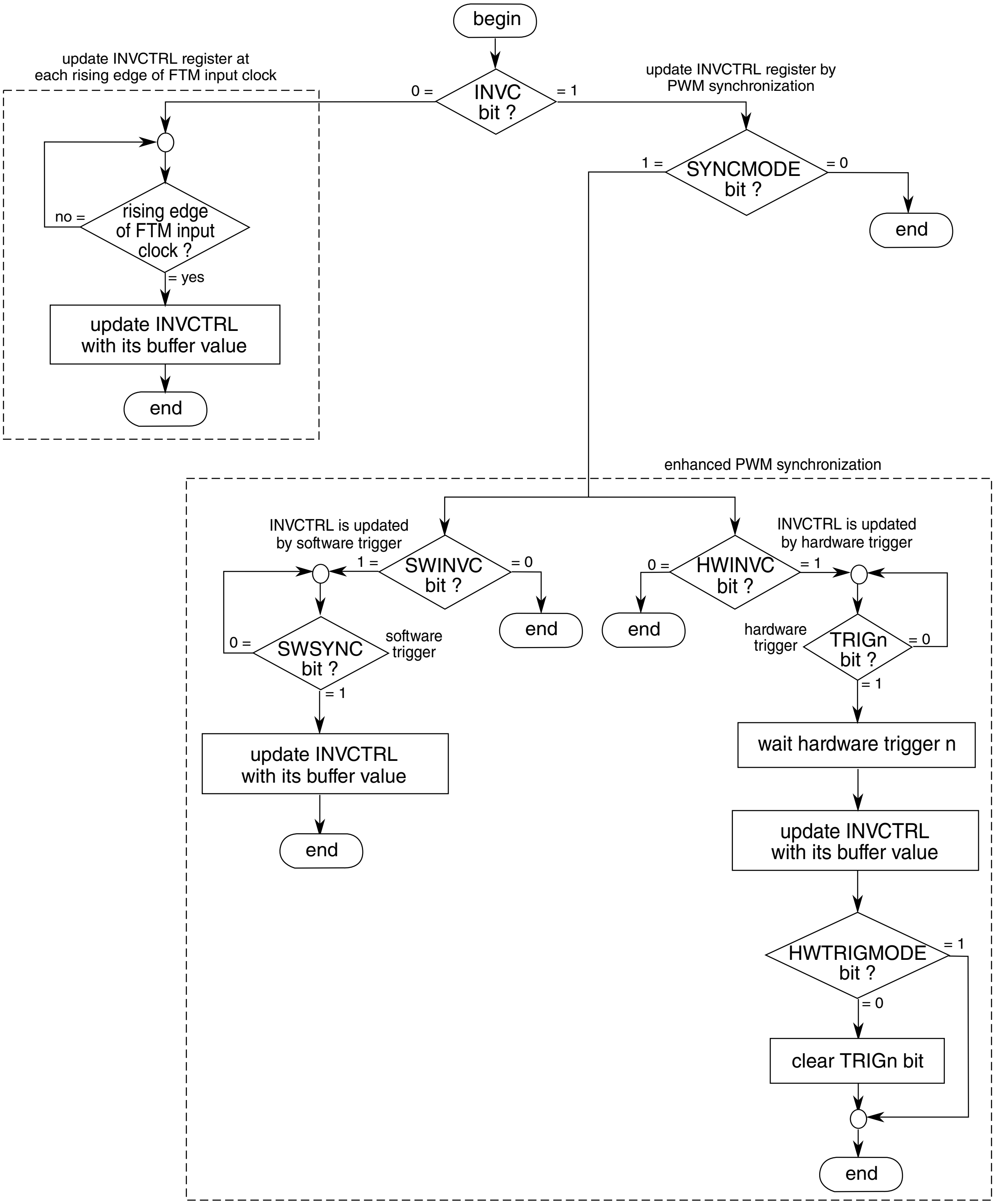

begin
INVC bit ?
0 =
= 1
update INVCTRL register at each rising edge of FTM input clock
rising edge of FTM input clock ?
no =
= yes
update INVCTRL with its buffer value
end
update INVCTRL register by PWM synchronization
SYNCMODE bit ?
1 =
= 0
end
enhanced PWM synchronization
INVCTRL is updated by software trigger
SWINVC bit ?
1 =
= 0
end
SWSYNC bit ?
0 =
= 1
software trigger
update INVCTRL with its buffer value
end
INVCTRL is updated by hardware trigger
HWINVC bit ?
0 =
= 1
end
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update INVCTRL with its buffer value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 34-58. INVCTRL register synchronization flowchart

34.5.13.9 SWOCTRL register synchronization

The SWOCTRL register synchronization updates the SWOCTRL register with its buffer value.

The SWOCTRL register can be updated at each rising edge of FTM input clock (SWOC = 0) or by the enhanced PWM synchronization (SWOC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the SWOCTRL register synchronization depends on SWSOC and HWSOC bits.

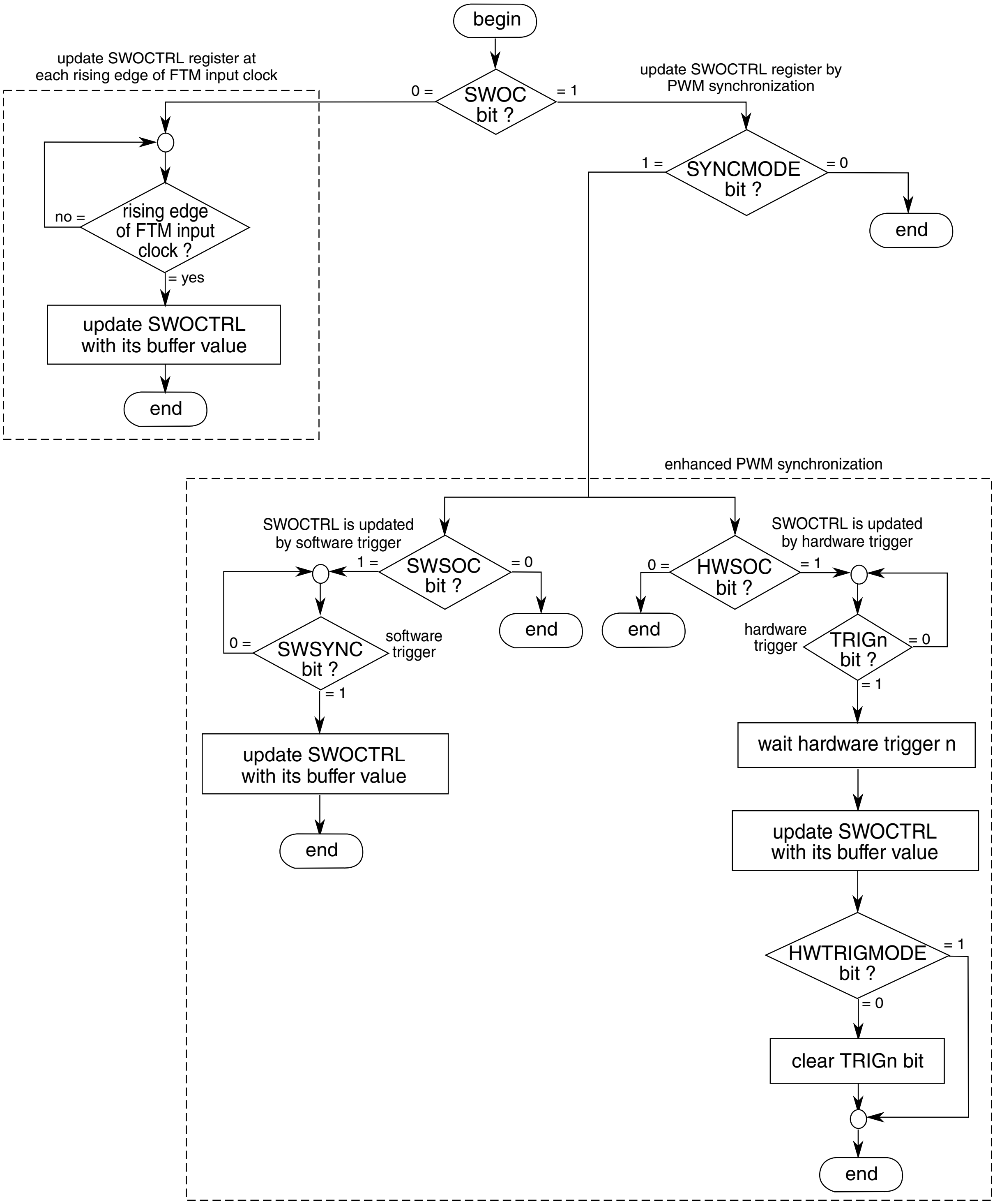

begin
SWOC bit ?
0 =
= 1
update SWOCTRL register at each rising edge of FTM input clock
no =
rising edge of FTM input clock ?
= yes
update SWOCTRL with its buffer value
end
update SWOCTRL register by PWM synchronization
SYNCMODE bit ?
1 =
= 0
end
enhanced PWM synchronization
SWOCTRL is updated by software trigger
SWSOC bit ?
1 =
= 0
end
0 =
SWSYNC bit ?
software trigger
= 1
update SWOCTRL with its buffer value
end
SWOCTRL is updated by hardware trigger
HWSOC bit ?
0 =
= 1
end
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update SWOCTRL with its buffer value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 34-59. SWOCTRL register synchronization flowchart

34.5.13.10 FTM counter synchronization

The FTM counter synchronization is a mechanism that allows the FTM to restart the PWM generation at a certain point in the PWM period. The channels outputs are forced to their initial value, except for channels in Output Compare mode, and the FTM counter is forced to its initial counting value defined by CNTIN register.

The following figure shows the FTM counter synchronization. Note that after the synchronization event occurs, the channel (n) is set to its initial value and the channel (n +1) is not set to its initial value due to a specific timing of this figure in which the deadtime insertion prevents this channel output from transitioning to 1. If no deadtime insertion is selected, then the channel (n+1) transitions to logical value 1 immediately after the synchronization event occurs.

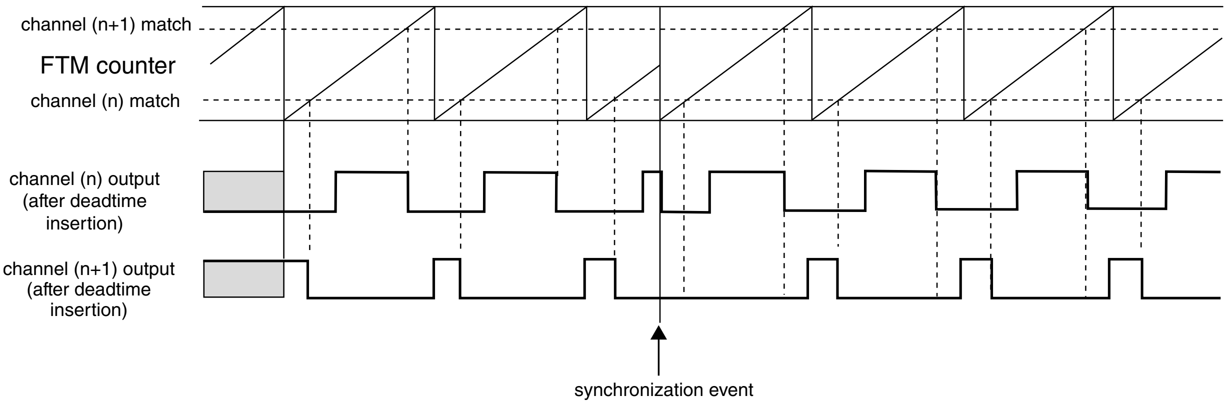


Figure 34-60. FTM counter synchronization

The FTM counter synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, the FTM counter must be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the FTM counter synchronization depends on SWRSTCNT and HWRSTCNT bits according to the following flowchart.

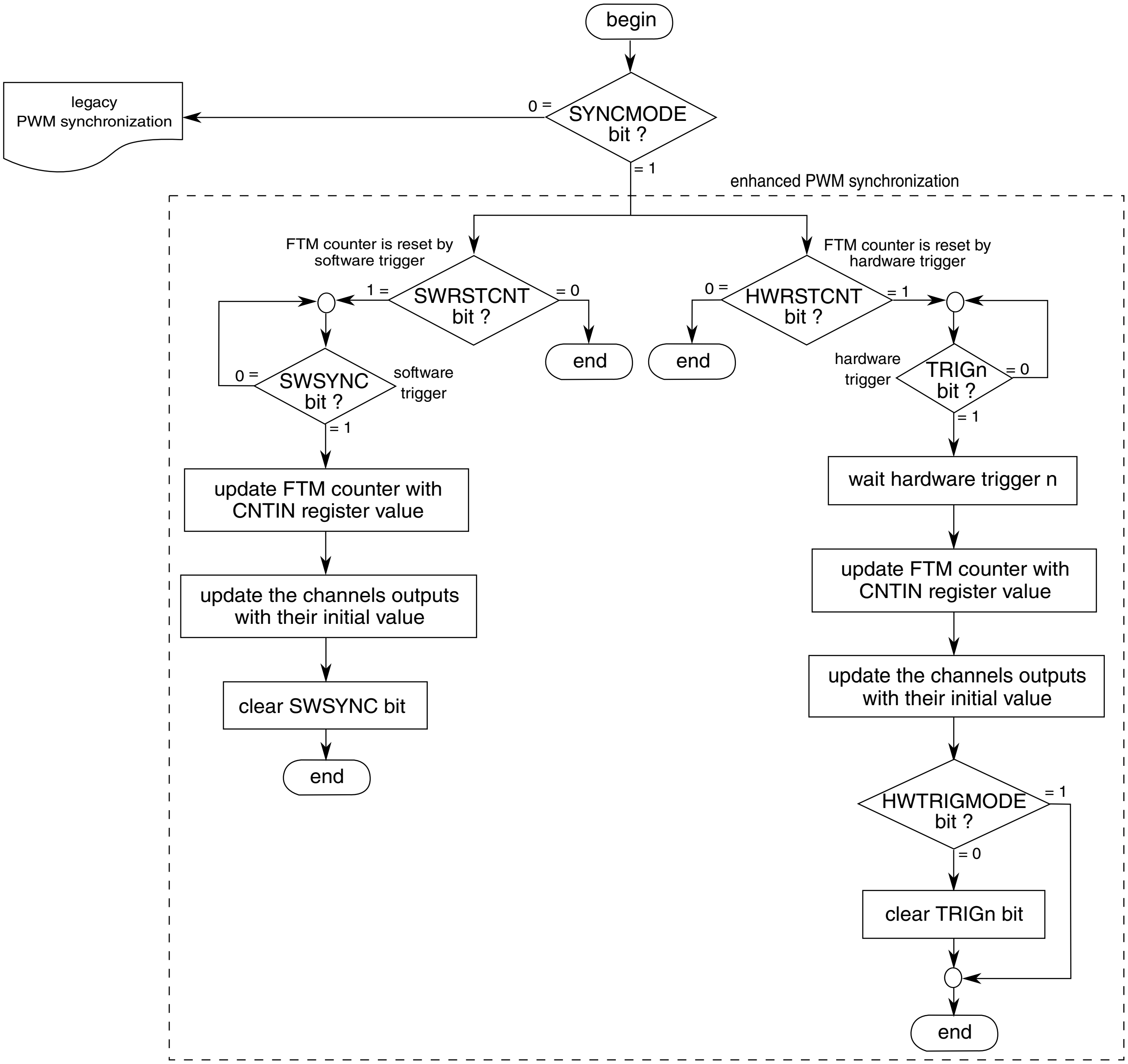

begin
SYNCMODE bit ?
0 =
legacy PWM synchronization
= 1
enhanced PWM synchronization
FTM counter is reset by software trigger
SWRSTCNT bit ?
1 =
= 0
end
SWSYNC bit ?
0 =
software trigger
= 1
update FTM counter with CNTIN register value
update the channels outputs with their initial value
clear SWSYNC bit
end
FTM counter is reset by hardware trigger
HWRSTCNT bit ?
0 =
= 1
end
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update FTM counter with CNTIN register value
update the channels outputs with their initial value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 34-61. FTM counter synchronization flowchart

In the case of legacy PWM synchronization, the FTM counter synchronization depends on REINIT and PWMSYNC bits according to the following description.

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 0) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

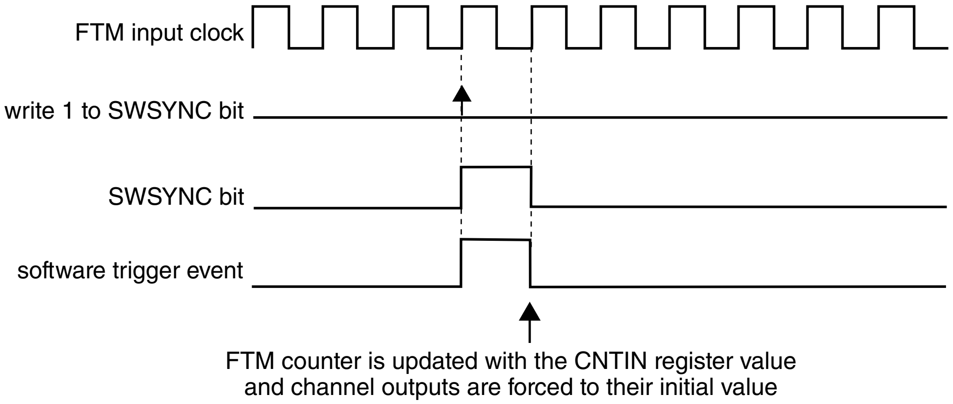

FTM input clock
write 1 to SWSYNC bit
SWSYNC bit
software trigger event
FTM counter is updated with the CNTIN register value
and channel outputs are forced to their initial value

Figure 34-62. FTM counter synchronization with (SYNCMODE = 0), (REINIT = 1), (PWMSYNC = 0), and software trigger was used

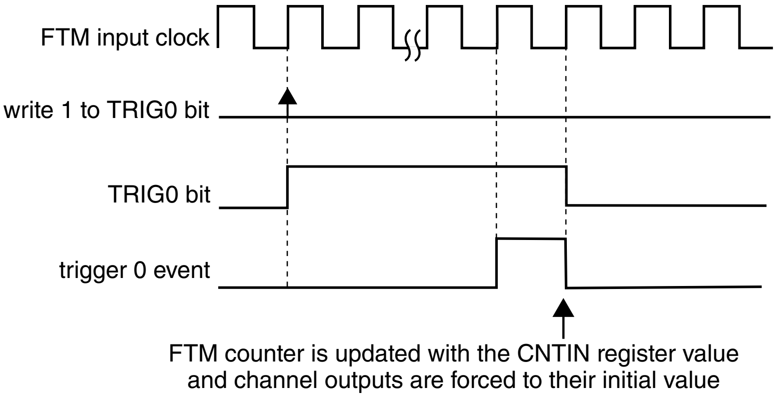

FTM input clock
write 1 to TRIG0 bit
TRIG0 bit
trigger 0 event
FTM counter is updated with the CNTIN register value
and channel outputs are forced to their initial value

Figure 34-63. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger.

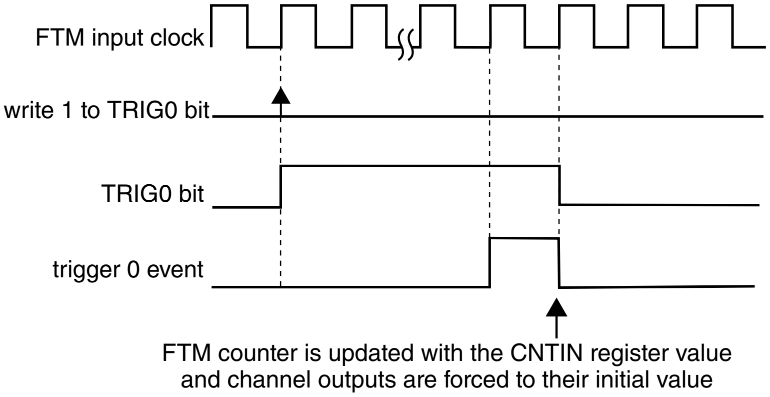

FTM input clock
write 1 to TRIG0 bit
TRIG0 bit
trigger 0 event
FTM counter is updated with the CNTIN register value
and channel outputs are forced to their initial value

Figure 34-64. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 1), and a hardware trigger was used

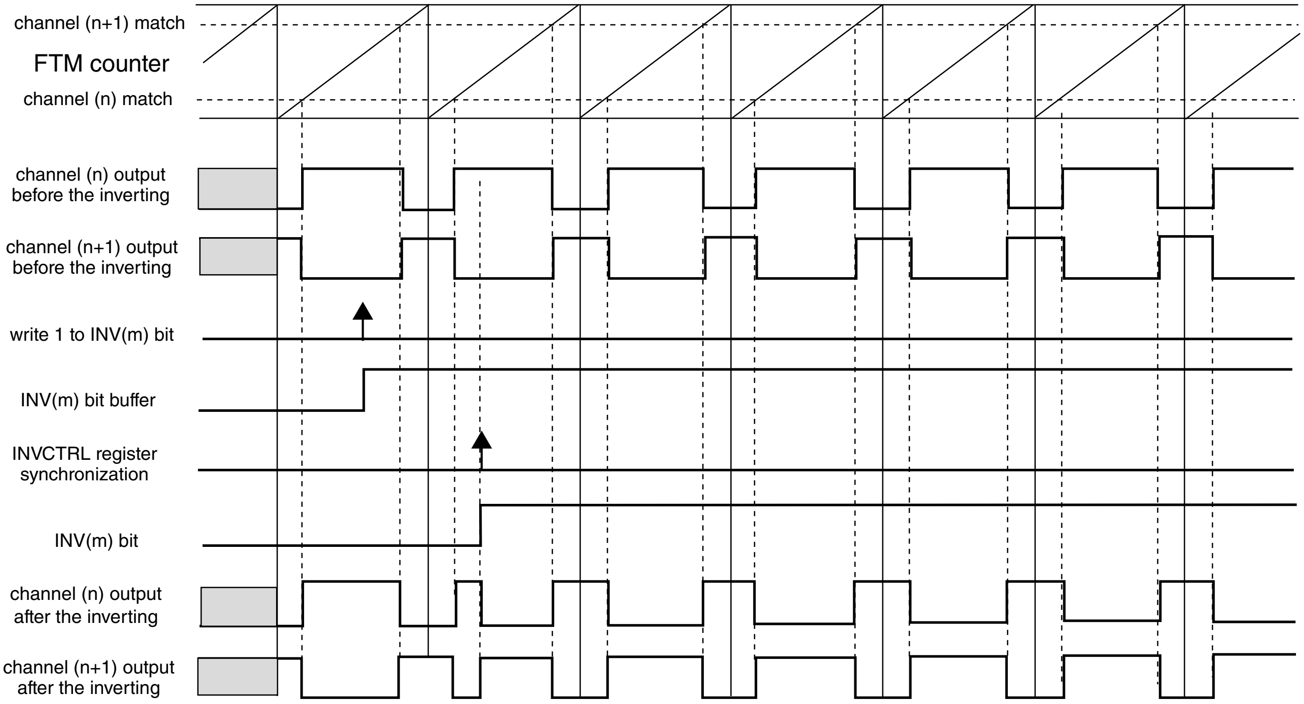
34.5.14 Inverting

The invert functionality swaps the signals between channel (n) and channel (n+1) outputs. The inverting operation is selected when:

- DECAPEN = 0
- COMP = 1, and
- INVm = 1 (where m represents a channel pair)

The INVm bit in INVCTRL register is updated with its buffer value according to INVCTRL register synchronization.

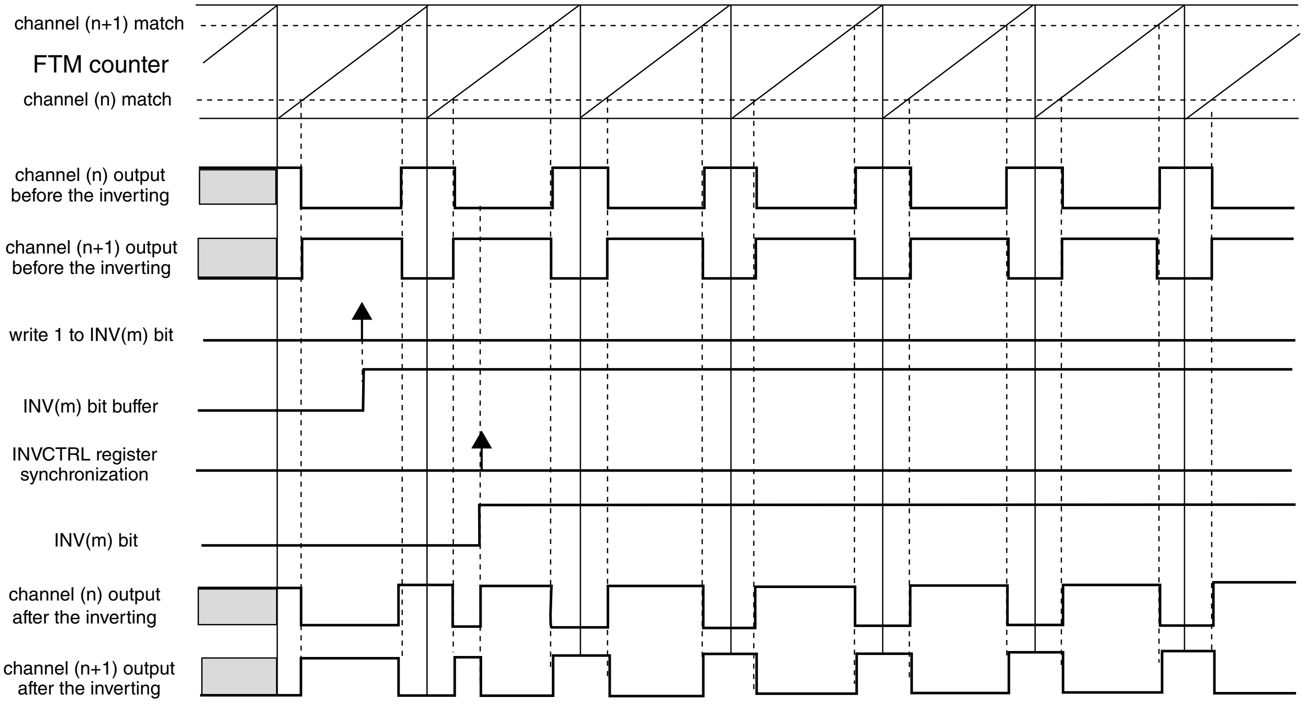
In combine mode with channel (n) ELSB:ELSA = 1:0, the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN), forced high at the channel (n) match and forced low at the channel (n+1) match. If the inverting is selected, the channel (n) output behavior is changed to force high at the beginning of the PWM period, force low at the channel (n) match and force high at the channel (n+1) match. See the following figure.


channel (n+1) match
FTM counter
channel (n) match
channel (n) output before the inverting
channel (n+1) output before the inverting
write 1 to INV(m) bit
INV(m) bit buffer
INVCTRL register synchronization
INV(m) bit
channel (n) output after the inverting
channel (n+1) output after the inverting

NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 34-65. Channels (n) and (n+1) outputs after the inverting in combine mode with channel (n) ELSB:ELSA = 1:0

Note that the channel (n) ELSB:ELSA bits should be considered because they define the active state of the channels outputs. In combine mode with channel (n) ELSB:ELSA = X: 1, the channel (n) output is forced high at the beginning of the period, forced low at the channel (n) match and forced high at the channel (n+1) match. When inverting is selected, the channels (n) and (n+1) present waveforms as shown in the following figure.


channel (n+1) match
FTM counter
channel (n) match
channel (n) output before the inverting
channel (n+1) output before the inverting
write 1 to INV(m) bit
INV(m) bit buffer
INVCTRL register synchronization
INV(m) bit
channel (n) output after the inverting
channel (n+1) output after the inverting

NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 34-66. Channels (n) and (n+1) outputs after the inverting in combine mode with channel (n) ELSB:ELSA = X:1

NOTE

The Inverting is not available in Output Compare mode and Modified Combine PWM Mode.

34.5.15 Software Output Control Mode

The software output control forces the channel output according to software defined values at a specific time in the PWM generation.

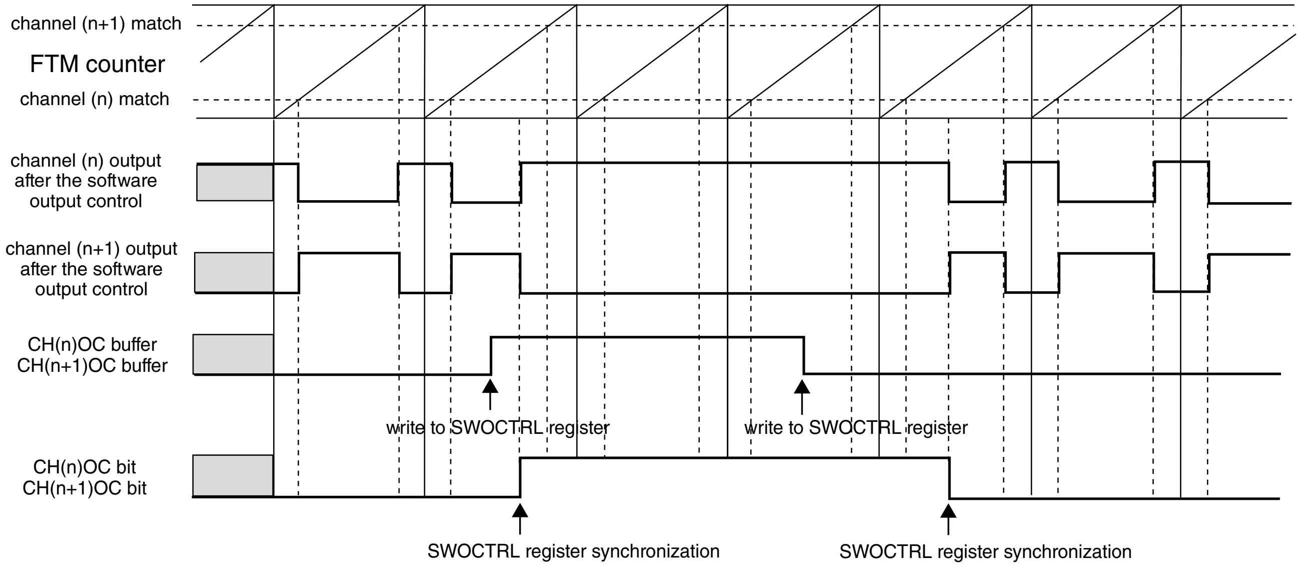
The software output control is selected when:

- DECAPEN = 0, and
- CH(n)OC = 1

The CH(n)OC bit enables the software output control for a specific channel output and the CH(n)OCV selects the value that is forced to this channel output.

Both CH(n)OC and CH(n)OCV bits in SWOCTRL register are buffered and updated with their buffer value according to SWOCTRL register synchronization.

The following figure shows the channels (n) and (n+1) outputs signals when the software output control is used. In this case the channels (n) and (n+1) are set to Combine and Complementary mode.


channel (n+1) match
FTM counter
channel (n) match
channel (n) output after the software output control
channel (n+1) output after the software output control
CH(n)OC buffer
CH(n+1)OC buffer
write to SWOCTRL register
write to SWOCTRL register
CH(n)OC bit
CH(n+1)OC bit
SWOCTRL register synchronization
SWOCTRL register synchronization

NOTE
Channel (n) ELSB:ELSA = X:1, CH(n)OCV = 1 and CH(n+1)OCV = 0.

Figure 34-67. Example of software output control in Combine and Complementary mode

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is zero.

Table 34-12. Software output control behavior when (COMP = 0)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to one

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is one.

Table 34-13. Software output control behavior when (COMP = 1)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to zero

Note

- The CH(n)OC and CH(n+1)OC bits should be equal.
- The COMP bit must not be modified when software output control is enabled, that is, CH(n)OC = 1 and/or CH(n +1)OC = 1.
- Software output control has the same behavior with disabled or enabled FTM counter (see the CLKS field description in the Status and Control register).

34.5.16 Deadtime insertion

The deadtime insertion is enabled when DTEN is set and DTVAL[5:0] is non-zero.

DEADTIME register defines the deadtime delay that can be used for all FTM channels. The clock for the DEADTIME delay is the FTM input clock divided by DTPS bits, and the DTVAL[5:0] bits define the deadtime modulo, that is, the number of the deadtime prescaler clocks.

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) = 0, and the deadtime is enabled, then when the channel (n) match (FTM 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 (FTM 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) = 1, and the deadtime is enabled, then when the channel (n) match (FTM 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 (FTM 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.

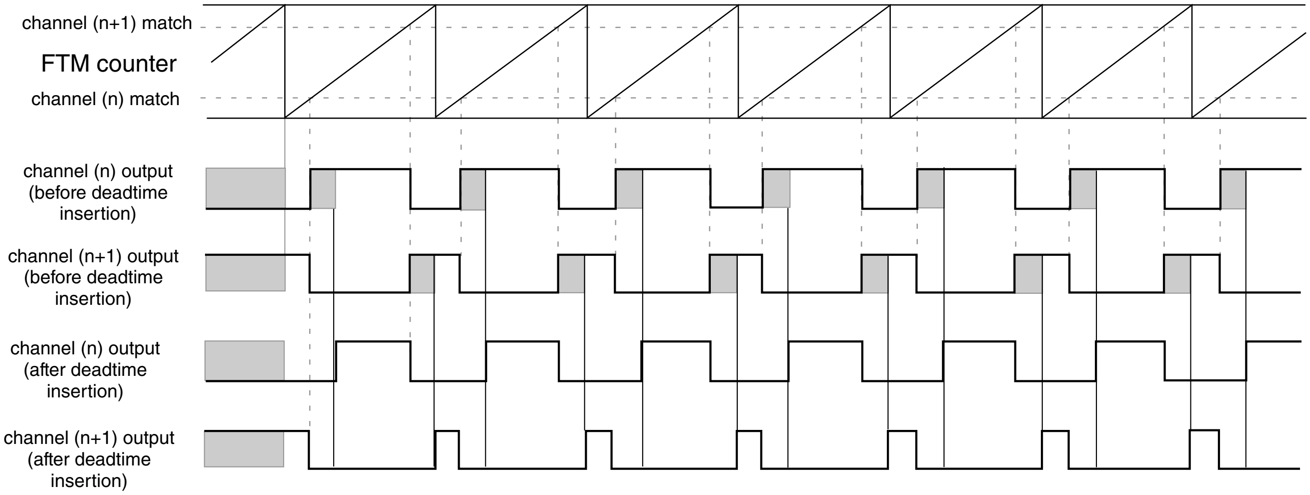

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 34-68. Deadtime insertion with ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0

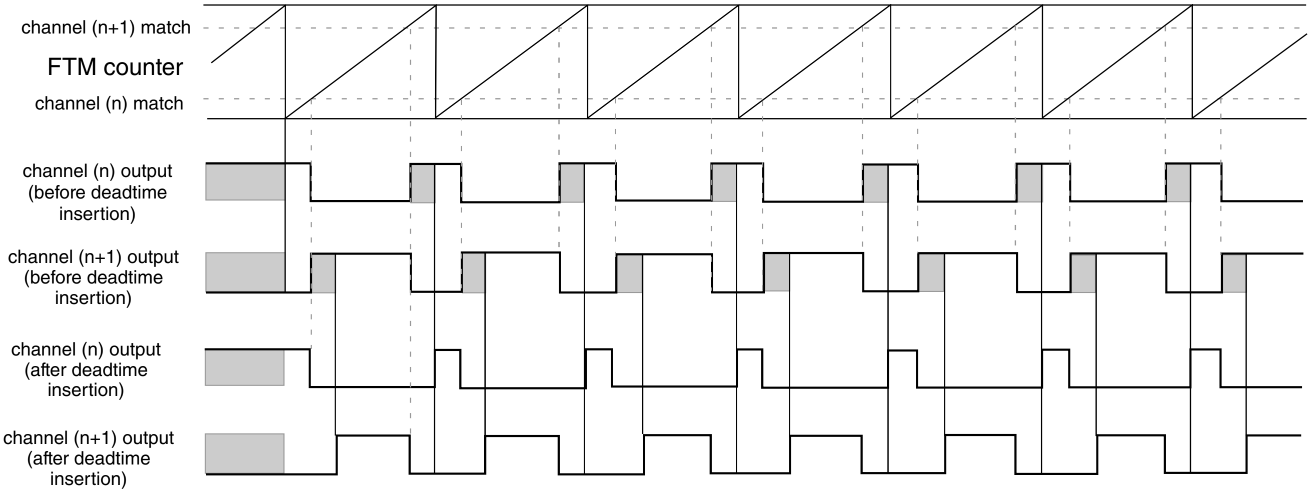

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 34-69. Deadtime insertion with ELSB:ELSA = X:1, POL(n) = 0, and POL(n+1) = 0

NOTE

- The deadtime feature must be used only in Complementary mode.
- The deadtime feature is not available in Output Compare mode.

34.5.16.1 Deadtime insertion corner cases

If (PS[2:0] is cleared), (DTPS[1:0] = 0:0 or DTPS[1:0] = 0:1):

- and the deadtime delay is greater than or equal to the channel (n) duty cycle ((C(n+1)V – C(n)V) × FTM input clock), then the channel (n) output is always the inactive value (POL(n) bit value).
- and the deadtime delay is greater than or equal to the channel (n+1) duty cycle ((MOD – CNTIN + 1 – (C(n+1)V – C(n)V)) × FTM input clock), then the channel (n+1) output is always the inactive value (POL(n+1) bit value).

Although, in most cases the deadtime delay is not comparable to channels (n) and (n+1) duty cycle, the following figures show examples where the deadtime delay is comparable to the duty cycle.

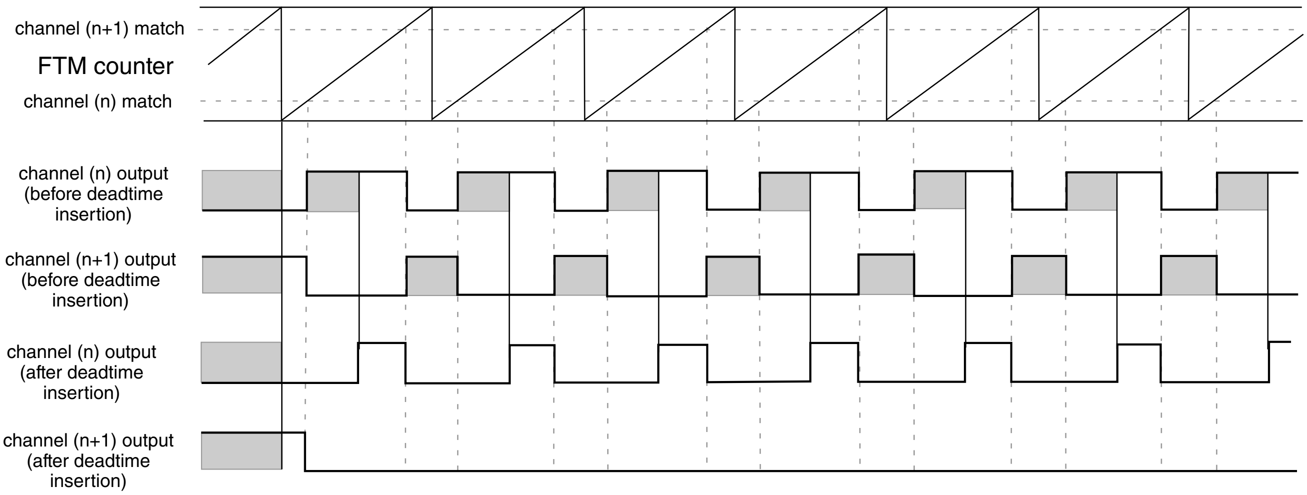

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 34-70. Example of the deadtime insertion (channel (n) ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channel (n+1) duty cycle

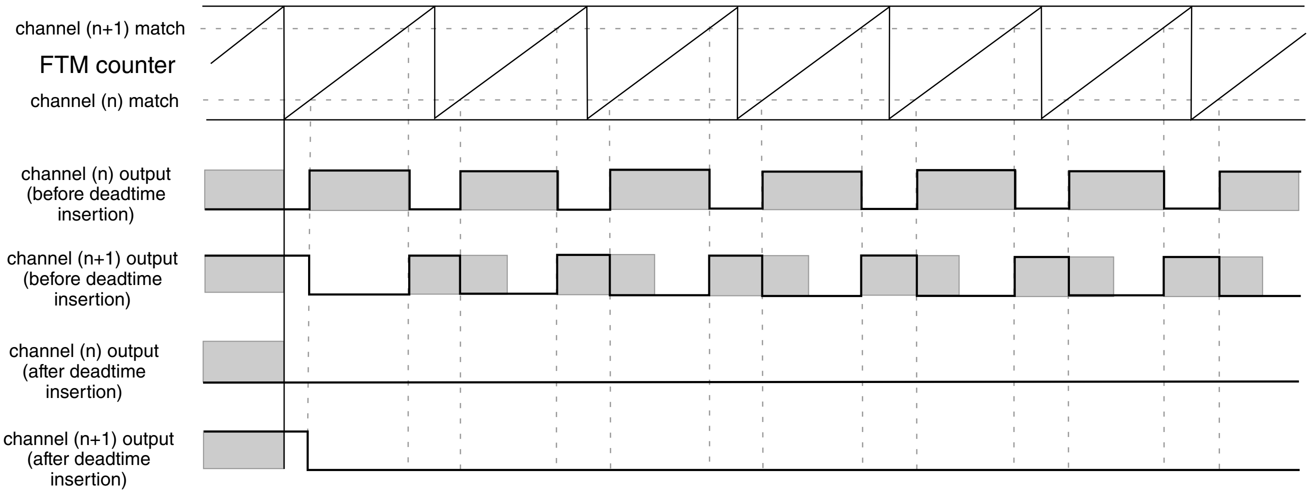

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 34-71. Example of the deadtime insertion (channel (n) ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channels (n) and (n+1) duty cycle

34.5.17 Output mask

The output mask can be used to force channels output to their inactive state through software. For example: to control a BLDC motor.

Any write to the OUTMASK register updates its write buffer. The OUTMASK register is updated with its buffer value by PWM synchronization; see OUTMASK register synchronization.

If CH(n)OM = 1, then the channel (n) output is forced to its inactive state (POLn bit value). If CH(n)OM = 0, then the channel (n) output is unaffected by the output mask. See the following figure.

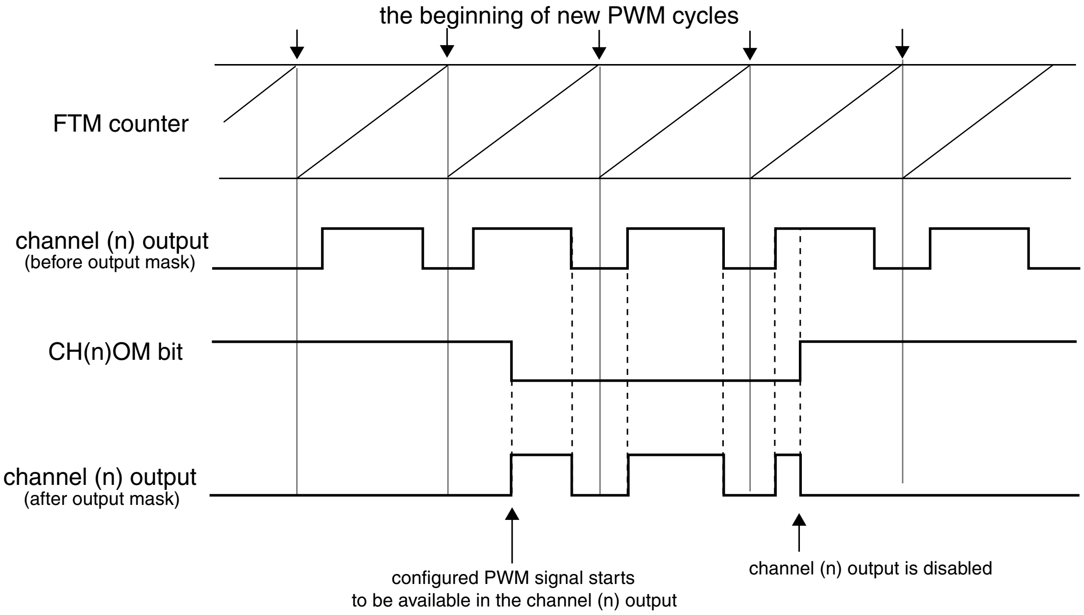

the beginning of new PWM cycles
FTM counter
channel (n) output
(before output mask)
CH(n)OM bit
channel (n) output
(after output mask)
configured PWM signal starts
to be available in the channel (n) output
channel (n) output is disabled

Figure 34-72. Output mask with POLn = 0

The following table shows the output mask result before the polarity control.

Table 34-14. Output mask result for channel (n) before the polarity control

CH(n)OM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

34.5.18 Fault Control

The fault control is enabled if FAULTM[1:0] ≠ 0:0.

FTM can have up to four fault inputs. FAULTnEN bit (where n = 0, 1, 2, 3) enables the fault input n and FFLTRnEN bit enables the fault input n filter. FFVAL[3:0] bits select the value of the enabled filter in each enabled fault input.

The fault input after being synchronized by FTM input clock is the filter input.

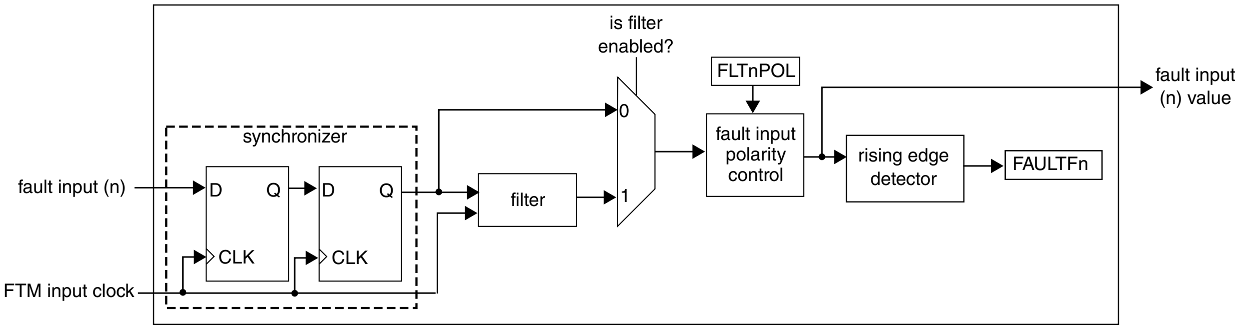


Figure 34-73. Diagram for Fault Control

When there is a state change in the fault input (n), the counter is reset and starts counting up. As long as the new state is stable on the fault input (n), the counter continues to increment. When the counter is equal to FFVAL[3:0] bits, the new fault input (n) value is validated. It is then transmitted as a pulse to the edge detector.

If the opposite edge appears on the fault input (n) before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. If a pulse is sampled as a value less than FFVAL[3:0] consecutive rising edges of FTM input clock, it is regarded as a glitch and is not passed on to the edge detector.

The table below shows the delay that is added by the FTM fault input filter according to its configuration.

Table 34-15. FTM Fault Input Filter Delay

FTM fault input filter	Number of rising edges between the selected edge on fault input and forcing the channels outputs to their safe values
• fault input does not have the input filter, or • fault input filter is disabled (FFLTRnEN = 0 or FFVAL[3:0] = 0)	• 3 rising edges of FTM input clock
• fault input has the input filter, and • fault input filter is enabled (FFLTRnEN = 1 and FFVAL[3:0] ≠ 0)	• (4 + FFVAL[3:0]) rising edges of FTM input clock

If the fault control and fault input (n) are enabled, and the selected edge at the fault input (n) is detected, then a fault condition has occurred and the FAULTFn bit is set. The FAULTF bit is the logic OR of FAULTFn[3:0] bits. See the following figure.

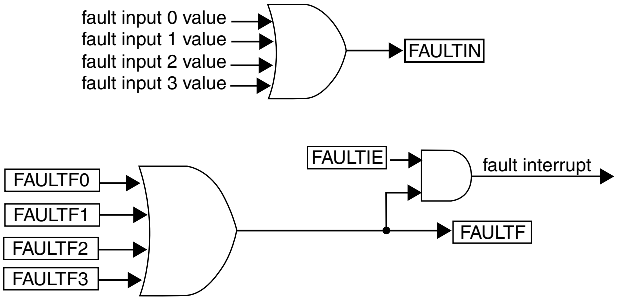


Figure 34-74. FAULTF and FAULTIN bits and fault interrupt

If the fault control is enabled (FAULTM[1:0] ≠ 0:0), a fault condition has occurred, and (FAULTENj = 1, where j is the pair j of the channels), then the channels (n) and (n+1) outputs are forced to their safe values:

- channel (n) output takes the POL(n) bit value
- channel (n+1) output takes the POL(n+1) bit value

The fault interrupt is generated when (FAULTF = 1) and (FAULTIE = 1). This interrupt request remains set until:

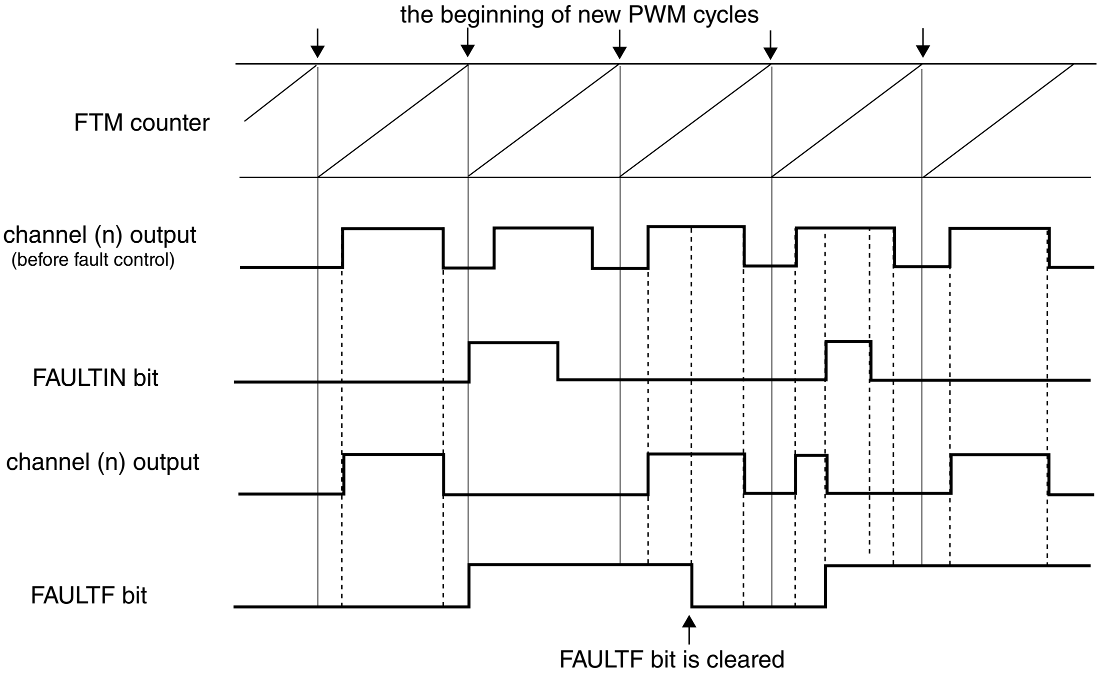
- Software clears the FAULTF bit by reading FAULTF bit as 1 and writing 0 to it
- Software clears the FAULTIE bit
- A reset occurs

NOTE

Additional Programming is required in case CPWM mode to handle fault properly. Either of CNTMIN or CNTMAX bit need to be programmed in SYNC register.

34.5.18.1 Automatic fault clearing

If the automatic fault clearing is selected (FAULTM[1:0] = 1:1), then the channels output disabled by fault control is again enabled when the fault input signal (FAULTIN) returns to zero and a new PWM cycle begins. See the following figure.

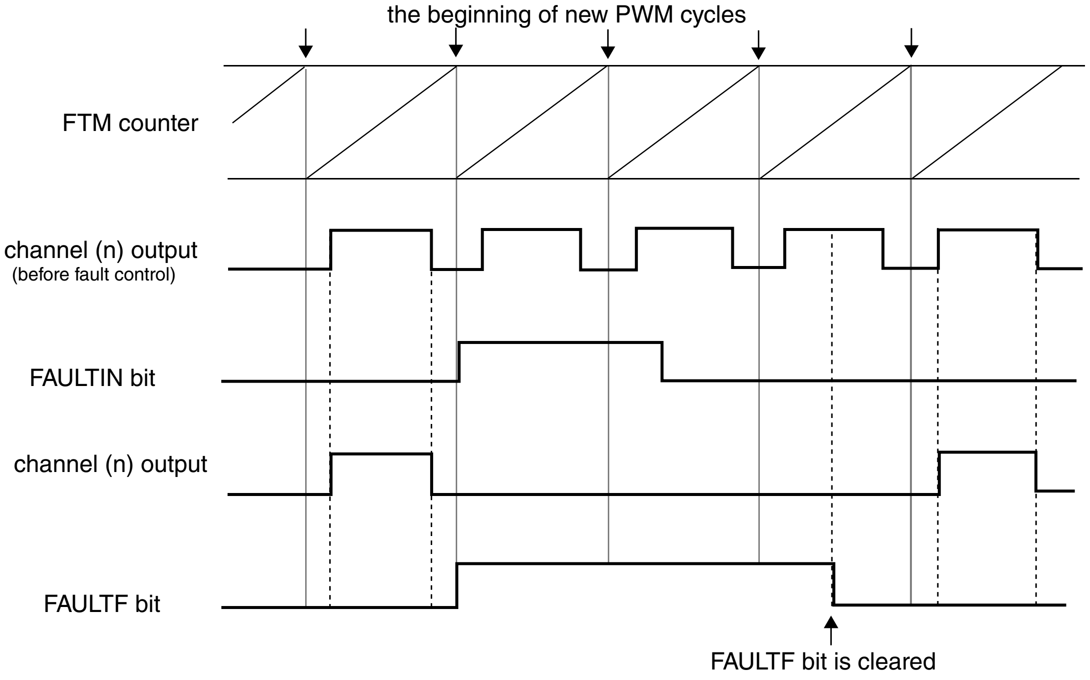


NOTE
The channel (n) output is after the fault control with automatic fault clearing and POLn = 0.

Figure 34-75. Fault control with automatic fault clearing

34.5.18.2 Manual fault clearing

If the manual fault clearing is selected (FAULTM[1:0] = 0:1 or 1:0), then the channels output disabled by fault control is again enabled when the FAULTF bit is cleared and a new PWM cycle begins. See the following figure.



NOTE
The channel (n) output is after the fault control with manual fault clearing and POLn = 0.

Figure 34-76. Fault control with manual fault clearing

34.5.18.3 Fault inputs polarity control

The FLTjPOL bit selects the fault input j polarity, where j = 0, 1, 2, 3:

- If FLTjPOL = 0, the fault j input polarity is high, so the logical one at the fault input j indicates a fault.
- If FLTjPOL = 1, the fault j input polarity is low, so the logical zero at the fault input j indicates a fault.

34.5.19 Polarity Control

The POLn bit selects the channel (n) output polarity:

- If POLn = 0, the channel (n) output polarity is high, so the logical one is the active state and the logical zero is the inactive state.
- If POLn = 1, the channel (n) output polarity is low, so the logical zero is the active state and the logical one is the inactive state.

34.5.20 Initialization

The initialization forces the CH(n)OI bit value to the channel (n) output when 1 is written to the INIT bit.

The initialization depends on COMP and DTEN bits. The following table shows the values that channels (n) and (n+1) are forced by initialization when the COMP and DTEN bits are zero.

Table 34-16. Initialization behavior when (COMP = 0 and DTEN = 0)

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	0	is forced to zero	is forced to zero
0	1	is forced to zero	is forced to one
1	0	is forced to one	is forced to zero
1	1	is forced to one	is forced to one

The following table shows the values that channels (n) and (n+1) are forced by initialization when (COMP = 1) or (DTEN = 1).

Table 34-17. Initialization behavior when (COMP = 1 or DTEN = 1)

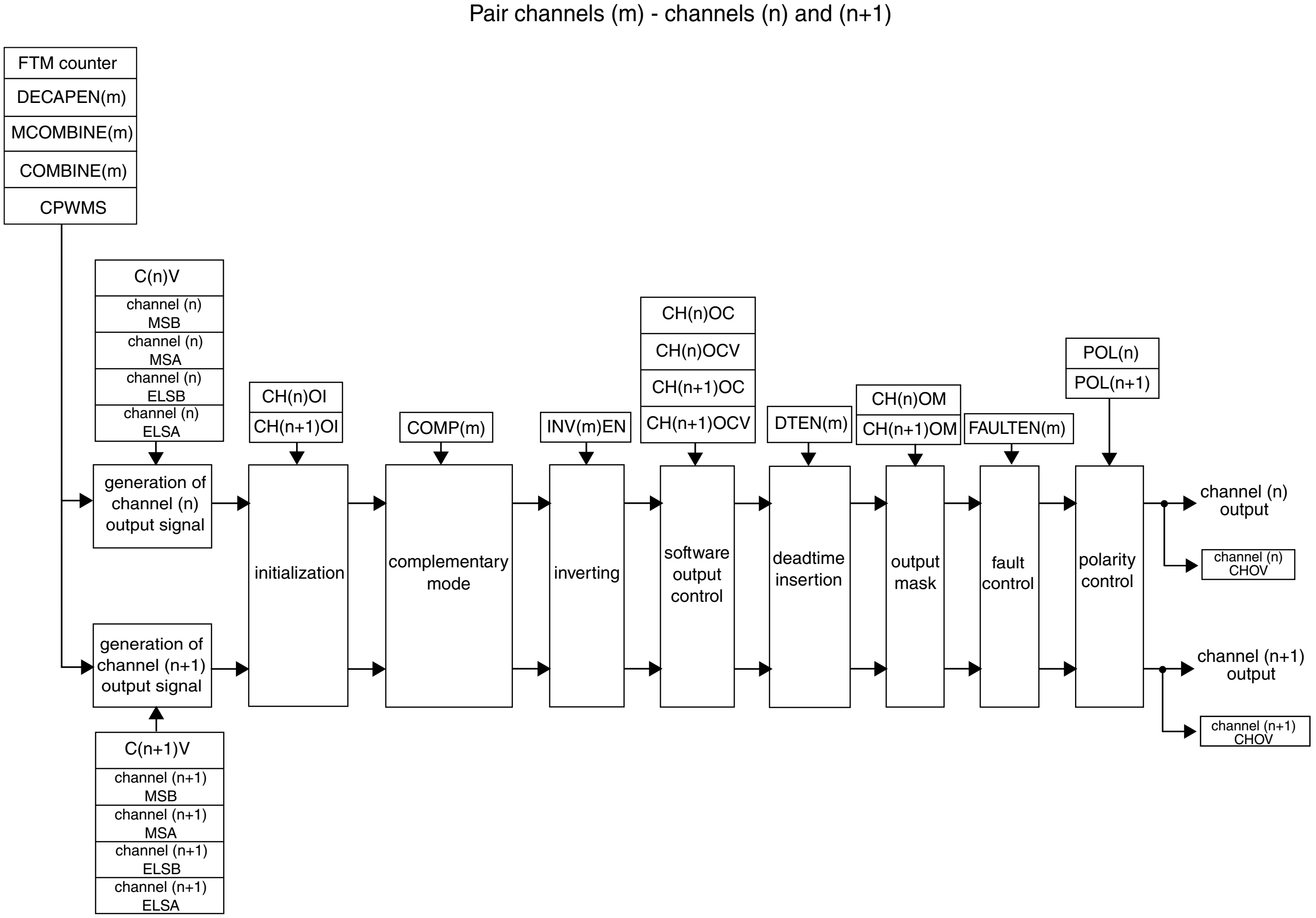
CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	X	is forced to zero	is forced to one
1	X	is forced to one	is forced to zero

Note

The initialization feature must be used only with disabled FTM counter. See the description of the CLKS field in the Status and Control register.

34.5.21 Features Priority

The following figure shows the priority of the features used at the generation of channels (n) and (n+1) outputs signals.



Note:
The channels (n) and (n+1) are in Output Compare, EPWM, CPWM, Combine or Modified Combine PWM modes.

Figure 34-77. Priority of the features used at the generation of channels (n) and (n+1) output

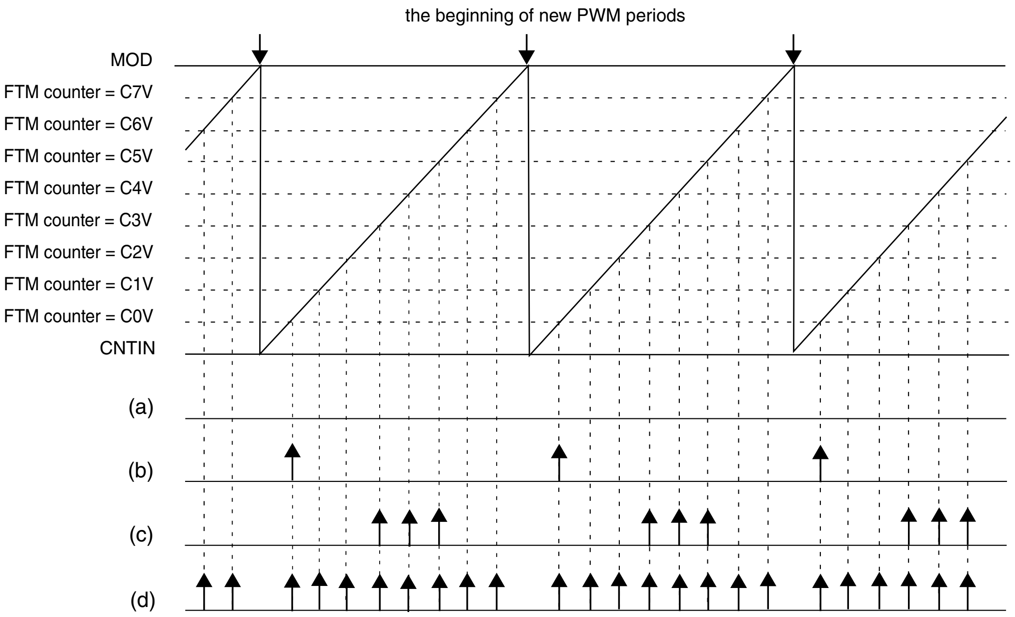
NOTE

The Initialization must not be used with Inverting and Software Output Control Mode.

34.5.22 External Trigger

If the CH(n)TRIG bit (register EXTTRIG) is set, where n = 0, 1, 2, 3, 4, 5, 6 or 7, then the FTM generates a trigger when the channel (n) match occurs (FTM counter = C(n)V) at the FTM external trigger output.

The width of a channel (n) trigger is one FTM input clock and the FTM is able to generate multiple triggers in one PWM period. See the figure below.



Note

(a) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0, CH6TRIG = 0, CH7TRIG = 0

(b) CH0TRIG = 1, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0, CH6TRIG = 0, CH7TRIG = 0

(c) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1, CH6TRIG = 0, CH7TRIG = 0

(d) CH0TRIG = 1, CH1TRIG = 1, CH2TRIG = 1, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1, CH6TRIG = 1, CH7TRIG = 1

Figure 34-78. External Trigger

34.5.23 Initialization Trigger

Initialization trigger allows FTM to generate a trigger in some specific points of FTM counter cycle. This feature is controlled by the bits INITTRIGEN and ITRIGR. The INITTRIGEN bit enables the initialization trigger generation and the ITRIGR bit selects when the initialization trigger is generated.

If INITTRIGEN = 1 and ITRIGR = 1, then the initialization trigger is generated when FTM counter reaches a reload point according to the frequency of the reload opportunities (Reload Points).

NOTE

For this configuration of initialization trigger and in CPWM mode, the bits CNTMAX and CNTMIN select where the initialization trigger is generated.

If INITTRIGEN = 1 and ITRIGR = 0, then the initialization trigger is generated when FTM counter is updated with the CNTIN register value. See the cases below.

1. When FTM counter is updated with CNTIN register value automatically.

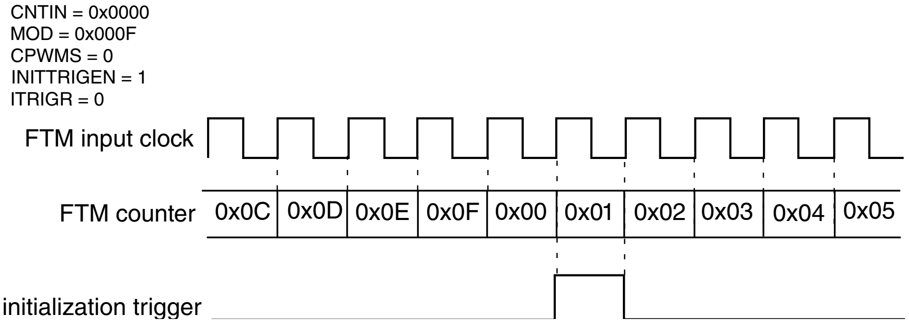


Figure 34-79. Example of the generation of the initialization trigger in the case 1.

2. When there is a write to CNT register.

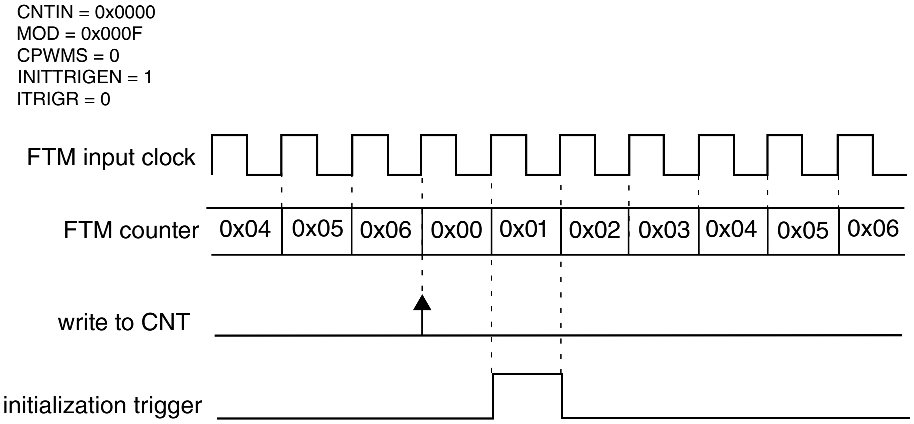


Figure 34-80. Example of the generation of the initialization trigger in the case 2.

NOTE

This behavior is not available in CPWM mode.

3. When there is the FTM counter synchronization.

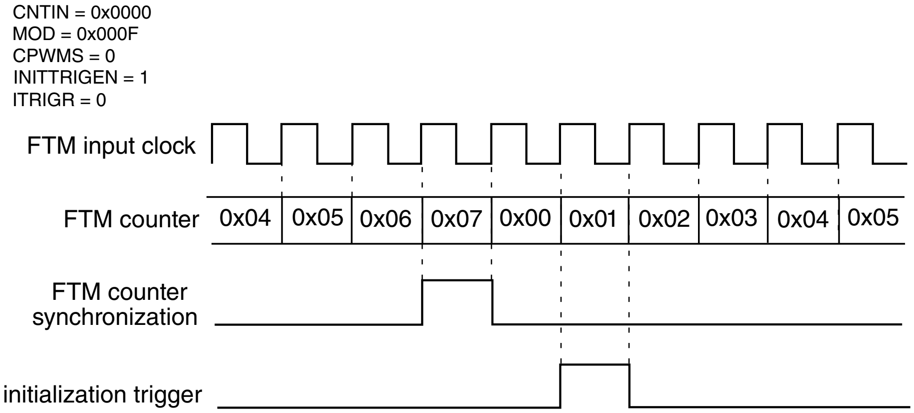


Figure 34-81. Example of the generation of the initialization trigger in the case 3.

NOTE

This behavior is not available in CPWM mode.

4. If (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits.

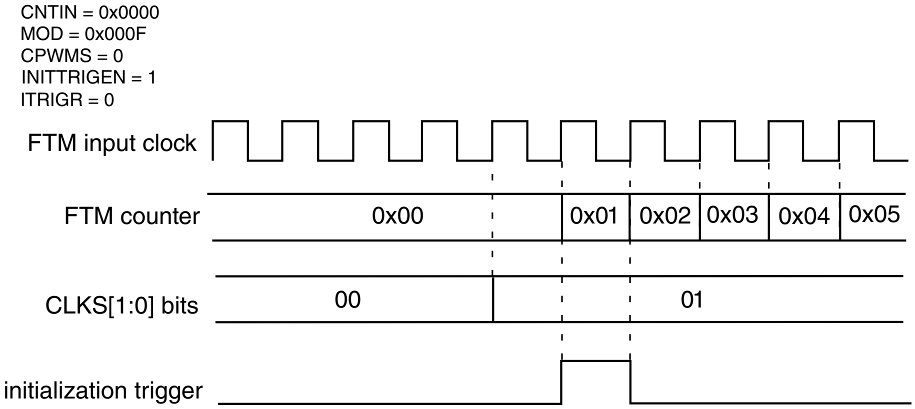


Figure 34-82. Example of the generation of the initialization trigger in the case 4.

NOTE

This behavior is not available in CPWM mode.

5. If the channel (n) is in Input Capture mode, (ICRST = 1) and the selected input capture event occurs in the channel (n) input.

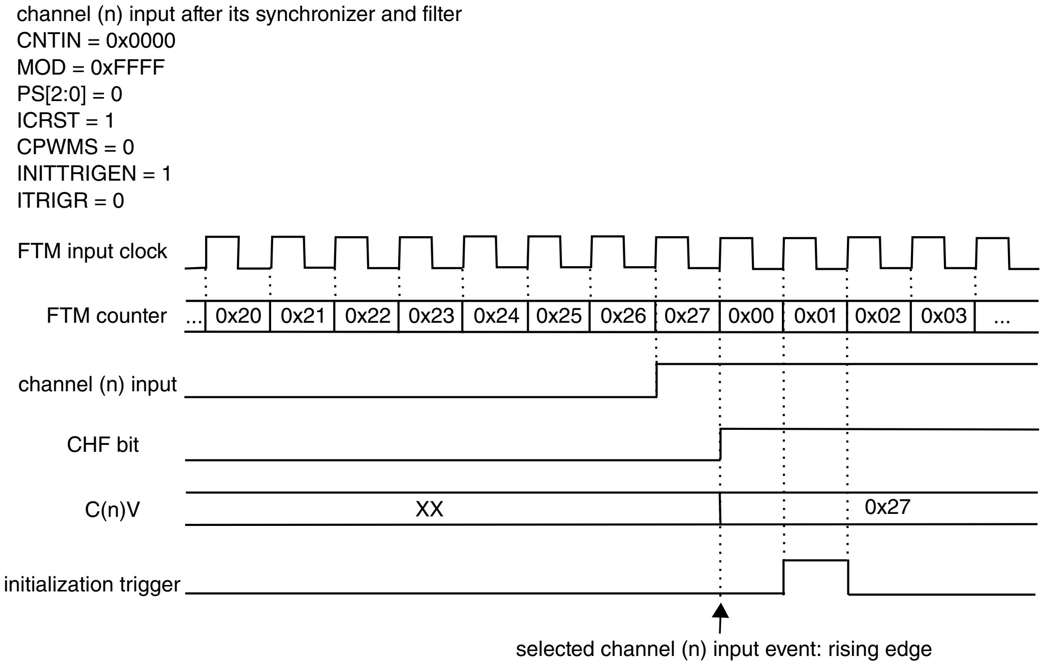


Figure 34-83. Example of the generation of the initialization trigger in the case 5.

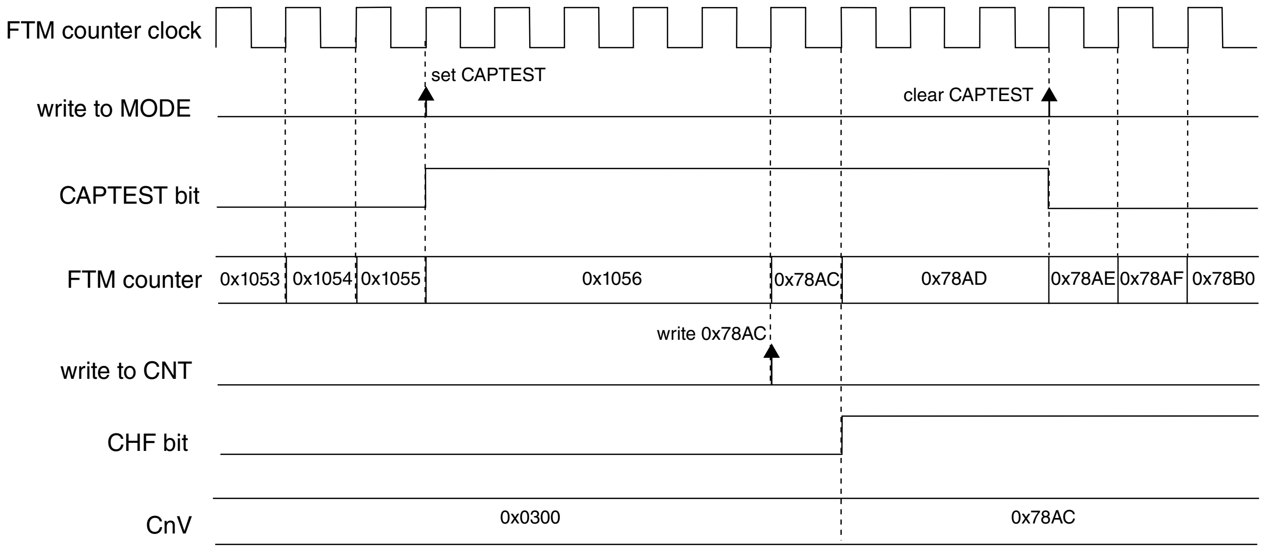
34.5.24 Capture Test Mode

The Capture Test mode allows to test the CnV registers, the FTM counter and the interconnection logic between the FTM counter and CnV registers.

In this test mode, all channels must be configured for Input Capture Mode and FTM counter must be configured to the Up counting.

When the Capture Test mode is enabled (CAPTEST = 1), the FTM counter is frozen and any write to CNT register updates directly the FTM counter; see the following figure. After it was written, all CnV registers are updated with the written value to CNT register and CHF bits are set. Therefore, the FTM counter is updated with its next value according to its configuration. Its next value depends on CNTIN, MOD, and the written value to FTM counter.

The next reads of CnV registers return the written value to the FTM counter and the next reads of CNT register return FTM counter next value.



Note:
- FTM counter is in free running
- FTMEN = 1
- FTM channel (n) is in Input Capture Mode

Figure 34-84. Capture Test Mode

34.5.25 DMA

The channel generates a DMA transfer request according to DMA and CHIE bits. See the following table.

Table 34-18. Channel DMA transfer request

DMA	CHIE	Channel DMA Transfer Request	Channel Interrupt
0	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
0	1	The channel DMA transfer request is not generated.	The channel interrupt is generated if (CHF = 1).
1	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
1	1	The channel DMA transfer request is generated if (CHF = 1).	The channel interrupt is not generated.

If DMA = 1, the CHF bit is cleared either by channel DMA transfer done or reading CnSC while CHF is set and then writing a zero to CHF bit according to CHIE bit. See the following table.

Table 34-19. Clear CHF bit when DMA = 1

CHIE	How CHF Bit Can Be Cleared
0	CHF bit is cleared either when the channel DMA transfer is done or by reading CnSC while CHF is set and then writing a 0 to CHF bit.
1	CHF bit is cleared when the channel DMA transfer is done.

34.5.26 Dual Edge Capture Mode

The dual edge capture mode is enabled if DECAPEN = 1. This mode allows to measure a pulse width or period of the channel (n) input where n = 0, 2, 4 or 6. The channel (n) filter can be enabled for n = 0 or 2.

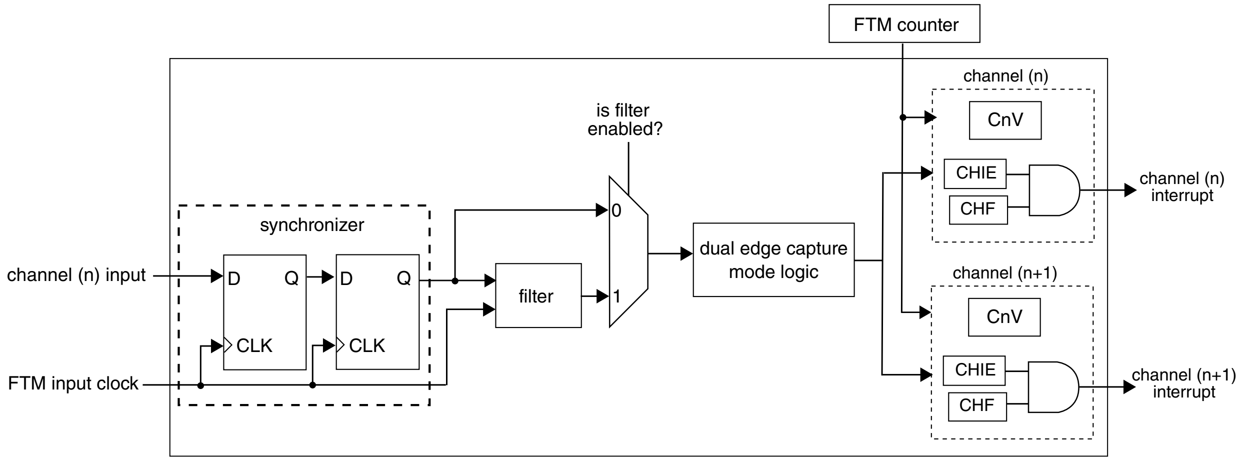


Figure 34-85. Diagram for Dual Edge Capture Mode

The channel (n) MSA bit defines if the dual edge capture mode is one-shot or continuous.

The channel (n) ELSB:ELSA bits select the edge that is captured by channel (n), and channel (n+1) ELSB:ELSA bits select the edge that is captured by channel (n+1). If both channel (n) ELSB:ELSA and channel (n+1) ELSB:ELSA bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

In the dual edge capture mode, only channel (n) input is used and channel (n+1) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then channel (n) CHF bit is set and the channel (n) interrupt is generated (if channel (n) CHIE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input and (channel (n) CHF = 1), then channel (n+1) CHF bit is set and the channel (n+1) interrupt is generated (if channel (n+1) CHIE = 1).

The C(n)V register stores the FTM counter value when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the FTM counter value when the selected edge by channel (n+1) is detected at channel (n) input.

In this mode, a coherency mechanism (for channels (n) and (n+1)) ensures coherent data when the C(n)V and C(n+1)V registers are read. The only requirement is that C(n)V must be read before C(n+1)V.

Note

- The dual edge capture mode must be used with channel (n) ELSB:ELSA = 0:1 or 1:0, channel (n+1) ELSB:ELSA = 0:1 or 1:0 and the FTM counter in Free running counter.

34.5.26.1 One-Shot Capture mode

The One-Shot Capture mode is selected when (DECAPEN = 1), and (channel (n) MSA = 0). In this capture mode, only one pair of edges at the channel (n) input is captured. The channel (n) ELSB:ELSA bits select the first edge to be captured, and channel (n+1) ELSB:ELSA bits select the second edge to be captured.

The edge captures are enabled while DECAP bit is set. For each new measurement in One-Shot Capture mode, first the channel (n) CHF and channel (n+1) CHF bits must be cleared, and then the DECAP bit must be set.

In this mode, the DECAP bit is automatically cleared by FTM when the edge selected by channel (n+1) is captured. Therefore, while DECAP bit is set, the one-shot capture is in process. When this bit is cleared, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

Similarly, when the channel (n+1) CHF bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

34.5.26.2 Continuous Capture mode

The Continuous Capture mode is selected when (DECAPEN = 1), and (channel (n) MSA = 1). In this capture mode, the edges at the channel (n) input are captured continuously. The channel (n) ELSB:ELSA bits select the initial edge to be captured, and channel (n+1) ELSB:ELSA bits select the final edge to be captured.

The edge captures are enabled while DECAP bit is set. For the initial use, first the channel (n) CHF and channel (n+1) CHF bits must be cleared, and then DECAP bit must be set to start the continuous measurements.

When the channel (n+1) CHF bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers. The latest captured values are always available in these registers even after the DECAP bit is cleared.

In this mode, it is possible to clear only the channel (n+1) CHF bit. Therefore, when the channel (n+1) CHF bit is set again, the latest captured values are available in C(n)V and C(n+1)V registers.

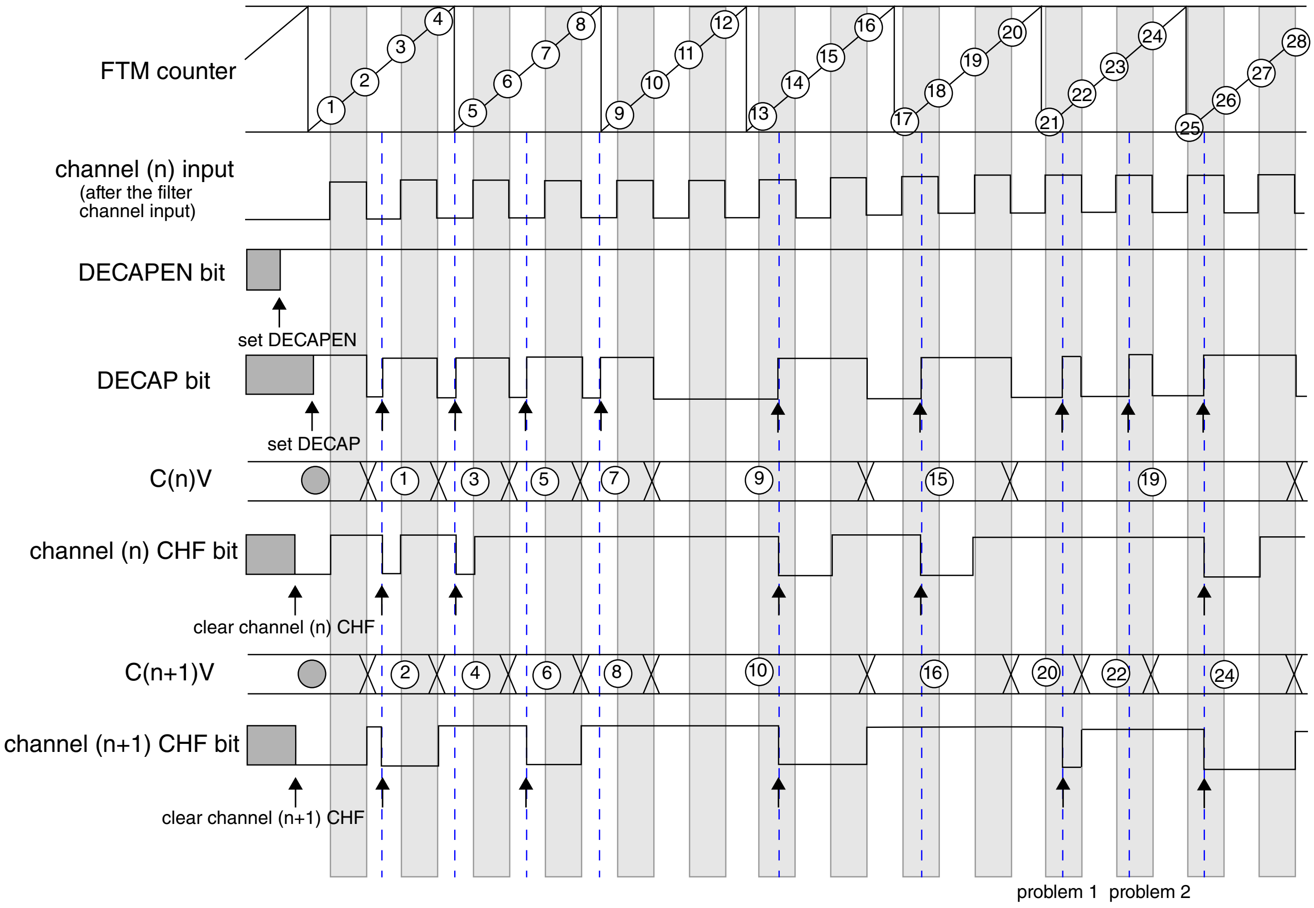
For a new sequence of the measurements in the Dual Edge Capture – Continuous mode, clear the channel (n) CHF and channel (n+1) CHF bits to start new measurements.

34.5.26.3 Pulse width measurement

If the channel (n) is configured to capture rising edges (channel (n) ELSB:ELSA = 0:1) and the channel (n+1) to capture falling edges (channel (n+1) ELSB:ELSA = 1:0), then the positive polarity pulse width is measured. If the channel (n) is configured to capture falling edges (channel (n) ELSB:ELSA = 1:0) and the channel (n+1) to capture rising edges (channel (n+1) ELSB:ELSA = 0:1), then the negative polarity pulse width is measured.

The pulse width measurement can be made in One-Shot Capture mode or Continuous Capture mode.

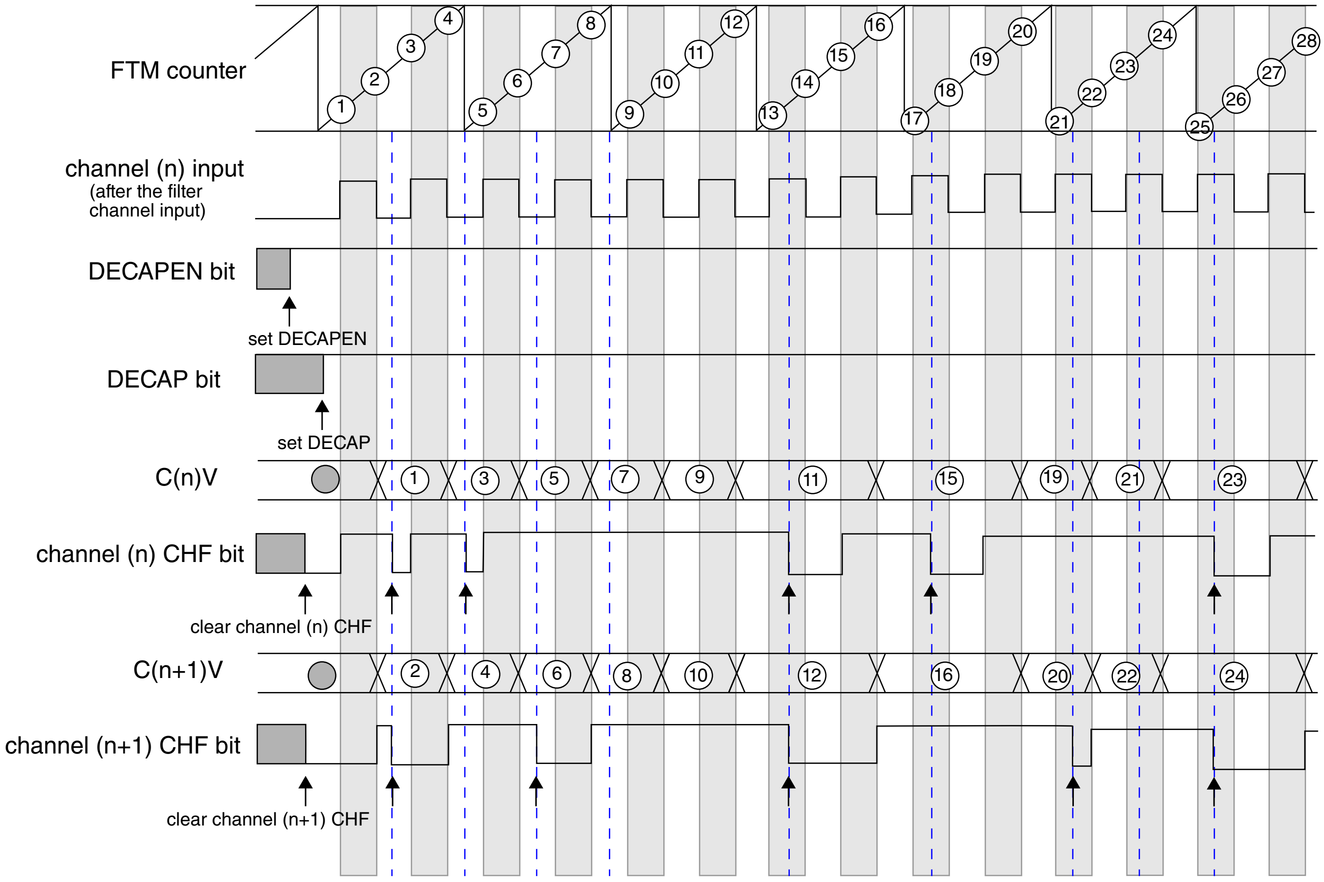
The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next positive polarity pulse width. The channel (n) CHF bit is set when the first edge of this pulse is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set and DECAP bit is cleared when the second edge of this pulse is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. Both DECAP and channel (n+1) CHF bits indicate when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.
- Problem 1: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and clear channel (n+1) CHF.
- Problem 2: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.

Figure 34-86. Dual Edge Capture – One-Shot mode for positive polarity pulse width measurement

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The channel (n) CHF bit is set when the first edge of the positive polarity pulse is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set when the second edge of this pulse is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. The channel (n+1) CHF bit indicates when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.

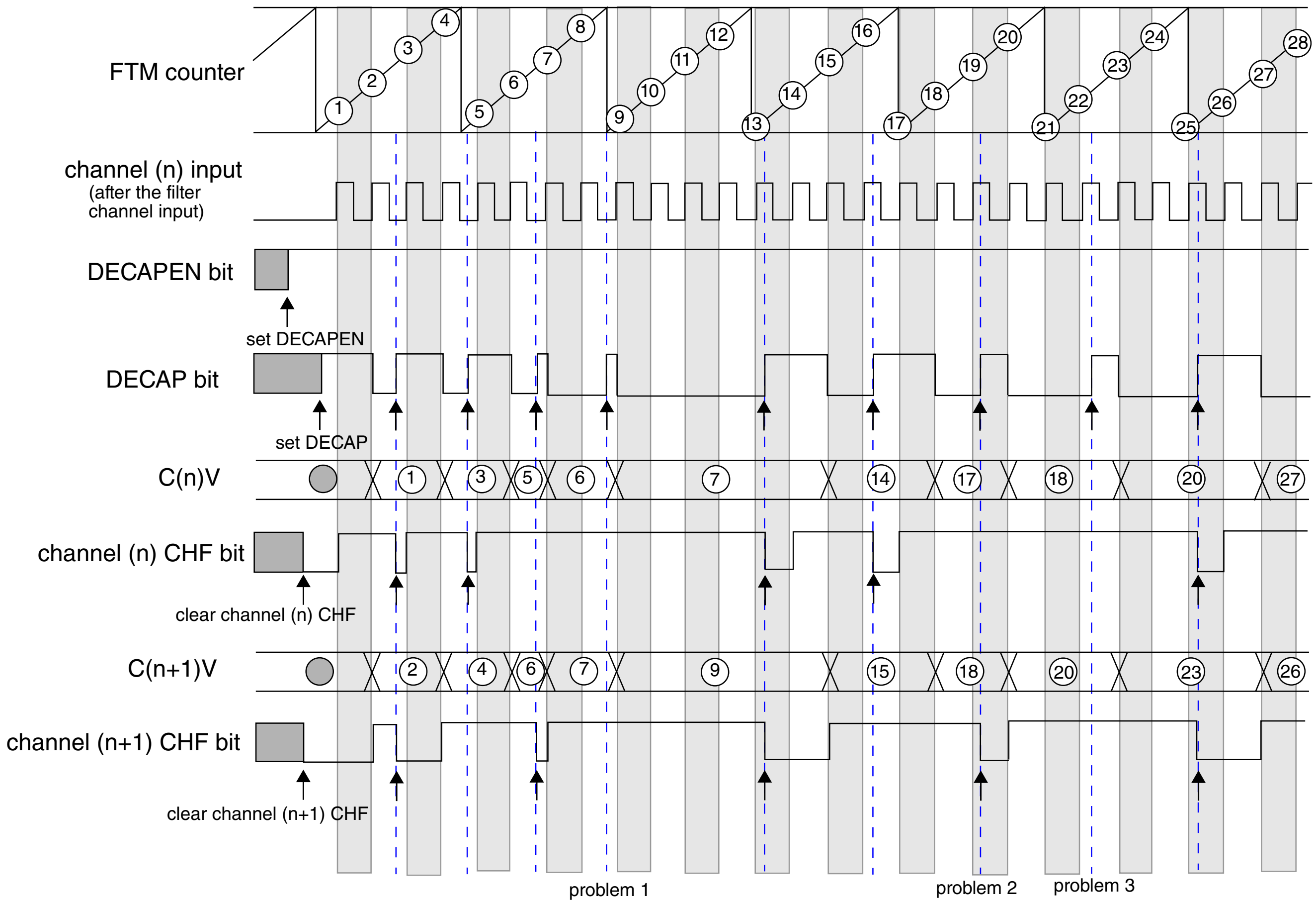
Figure 34-87. Dual Edge Capture – Continuous mode for positive polarity pulse width measurement

34.5.26.4 Period measurement

If the channels (n) and (n+1) are configured to capture consecutive edges of the same polarity, then the period of the channel (n) input signal is measured. If both channels (n) and (n+1) are configured to capture rising edges (channel (n) ELSB:ELSA = 0:1 and channel (n+1) ELSB:ELSA = 0:1), then the period between two consecutive rising edges is measured. If both channels (n) and (n+1) are configured to capture falling edges (channel (n) ELSB:ELSA = 1:0 and channel (n+1) ELSB:ELSA = 1:0), then the period between two consecutive falling edges is measured.

The period measurement can be made in One-Shot Capture mode or Continuous Capture mode.

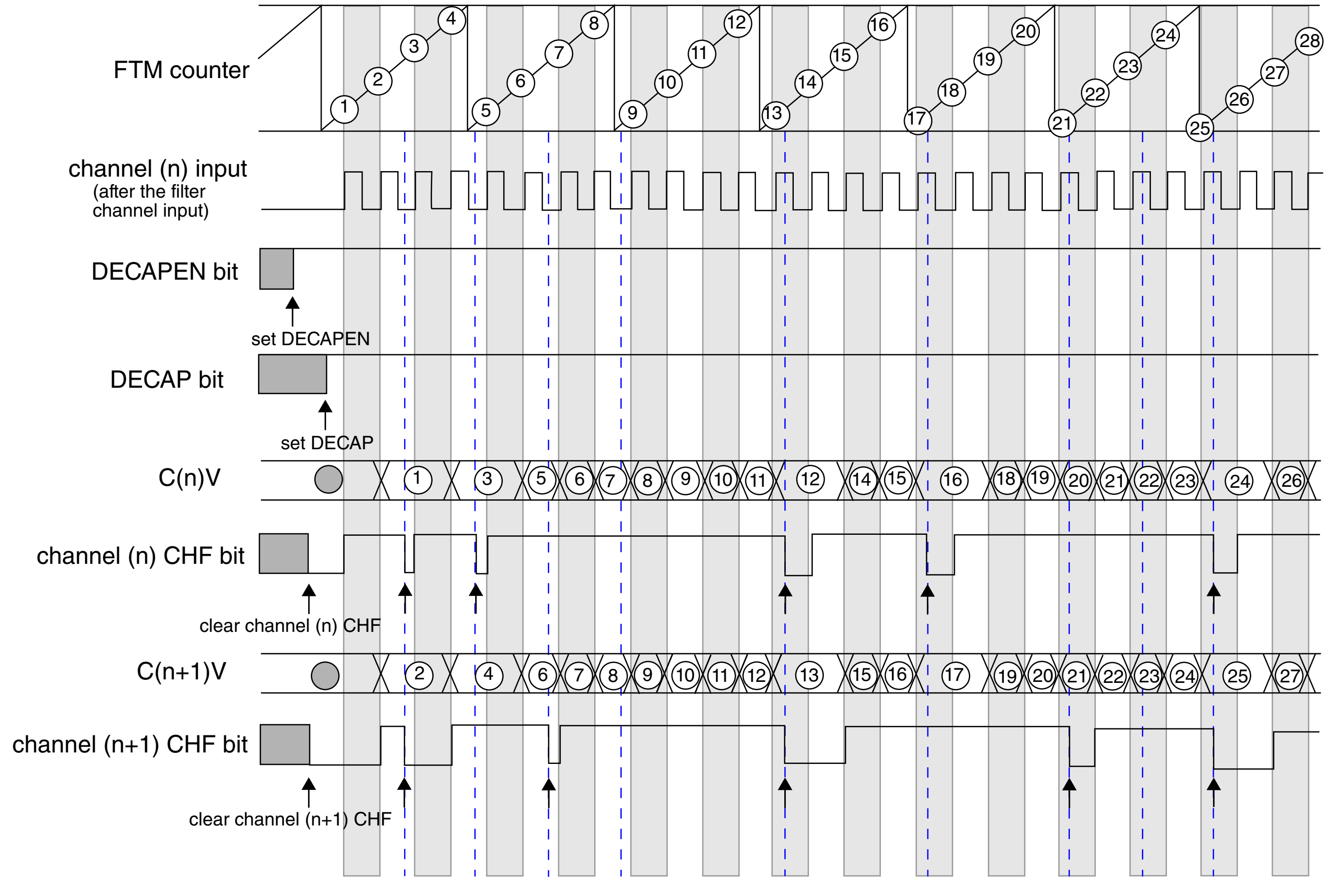
The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next period. The channel (n) CHF bit is set when the first rising edge is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set and DECAP bit is cleared when the second rising edge is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. Both DECAP and channel (n+1) CHF bits indicate when two selected edges were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.
- Problem 1: channel (n) input = 0, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.
- Problem 2: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and clear channel (n+1) CHF.
- Problem 3: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.

Figure 34-88. Dual Edge Capture – One-Shot mode to measure of the period between two consecutive rising edges

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The channel (n) CHF bit is set when the first rising edge is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set when the second rising edge is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. The channel (n+1) CHF bit indicates when two edges of the period were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.

Figure 34-89. Dual Edge Capture – Continuous mode to measure of the period between two consecutive rising edges

34.5.26.5 Read coherency mechanism

The Dual Edge Capture mode implements a read coherency mechanism between the FTM counter value captured in C(n)V and C(n+1)V registers. The read coherency mechanism is illustrated in the following figure. In this example, the channels (n) and (n +1) are in Dual Edge Capture – Continuous mode for positive polarity pulse width measurement. Thus, the channel (n) is configured to capture the FTM counter value when there is a rising edge at channel (n) input signal, and channel (n+1) to capture the FTM counter value when there is a falling edge at channel (n) input signal.

When a rising edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n) capture buffer. The channel (n) capture buffer value is transferred to C(n)V register when a falling edge occurs in the channel (n) input signal.

C(n)V register has the FTM counter value when the previous rising edge occurred, and the channel (n) capture buffer has the FTM counter value when the last rising edge occurred.

When a falling edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n+1) capture buffer. The channel (n+1) capture buffer value is transferred to C(n+1)V register when the C(n)V register is read.

In the following figure, the read of C(n)V returns the FTM counter value when the event 1 occurred and the read of C(n+1)V returns the FTM counter value when the event 2 occurred.

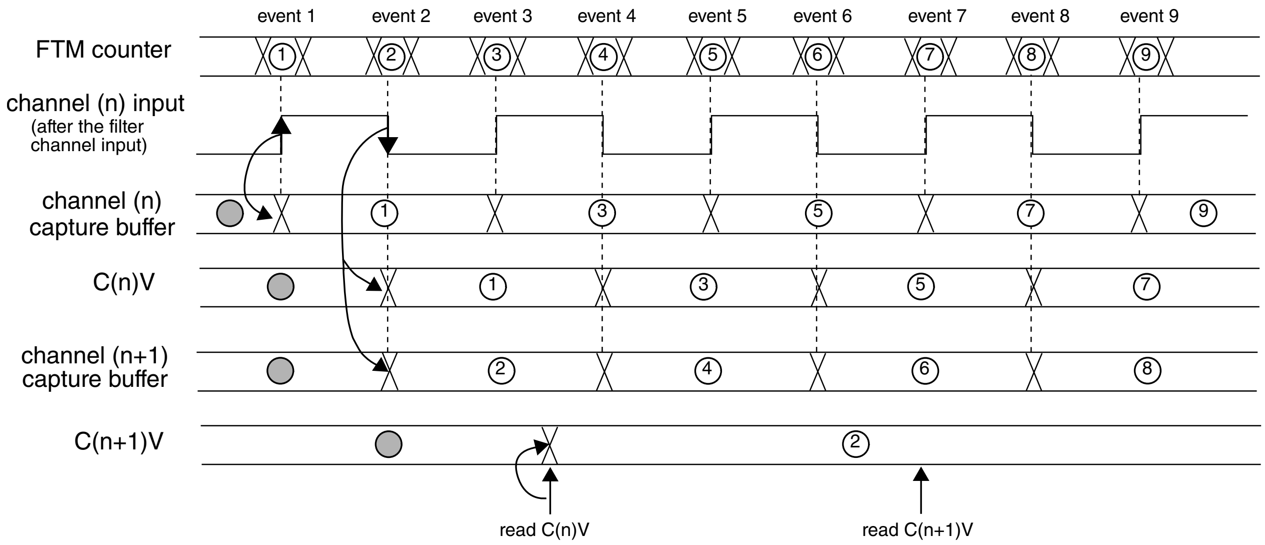


Figure 34-90. Dual Edge Capture mode read coherency mechanism

C(n)V register must be read prior to C(n+1)V register in dual edge capture one-shot and continuous modes for the read coherency mechanism works properly.

34.5.27 Debug mode

When the chip is in Debug mode, the BDMMODE[1:0] bits select the behavior of the FTM counter, the channel (n) CHF bit, the channels output, and the writes to the MOD, CNTIN, and C(n)V registers according to the following table.

Table 34-20. FTM behavior when the chip Is in Debug mode

BDMMODE	FTM Counter	channel (n) CHF bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
00	Stopped	can be set	Functional mode	Writes to these registers bypass the registers buffers

Table continues on the next page...

Table 34-20. FTM behavior when the chip Is in Debug mode (continued)

BDMMODE	FTM Counter	channel (n) CHF bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
01	Stopped	is not set	The channels outputs are forced to their safe value according to POLn bit	Writes to these registers bypass the registers buffers
10	Stopped	is not set	The channels outputs are frozen when the chip enters in Debug mode	Writes to these registers bypass the registers buffers
11	Functional mode	can be set	Functional mode	Functional mode

Note that if BDMMODE[1:0] = 2’b00 then the channels outputs remain at the value when the chip enters in Debug mode, because the FTM counter is stopped. However, the following situations modify the channels outputs in this Debug mode.

- Write any value to CNT register; see Counter reset. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for those channels set to Output Compare mode.
- FTM counter is reset by PWM Synchronization mode; see FTM counter synchronization. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for channels in Output Compare mode.
- In the channels outputs initialization, the channel (n) output is forced to the CH(n)OI bit value when the value 1 is written to INIT bit. See Initialization.

Note

The BDMMODE[1:0] = 2’b00 must not be used with the Fault Control. Even if the fault control is enabled and a fault condition exists, the channels outputs are updated as above.

Note

If CLKS[1:0] = 2'b00 in Debug, a non-zero value is written to CLKS in Debug, and CnV = CNTIN when the Debug is disabled, then the CHF bit is set (since if the channel is a 0% EPWM signal) when the Debug is disabled.

34.5.28 Reload Points

This feature allows to update the registers CNTIN, HCR, MOD and C(n)V with the value of their write buffer at the selected reload point.

NOTE

- This feature is independent of the PWM synchronization.
- At these reload points neither the channels outputs nor the FTM counter are changed. Software must select these reload points at the safe points in time.

34.5.28.1 Reload Opportunities

The reload opportunities are:

1. At the half cycle

   This reload opportunity is enabled if (HCSEL = 1) and it happens at the half cycle (FTM counter = HCR register). The software should calculate the half cycle value according to the FTM counter configuration, then writes this value to the register HCR.

2. At the channel (n) match

   This reload opportunity is enabled if (CH(n)SEL = 1) and it happens at the channel (n) match (FTM counter = C(n)V).

3. When the FTM counter is an up counter

   This reload opportunity is when the FTM counter changes from (MOD) to (CNTIN - 1) and it is always enabled.

   The following figure shows an example of the reload opportunities at the half cycle, at the channels match, and when the FTM counter is an up counter.

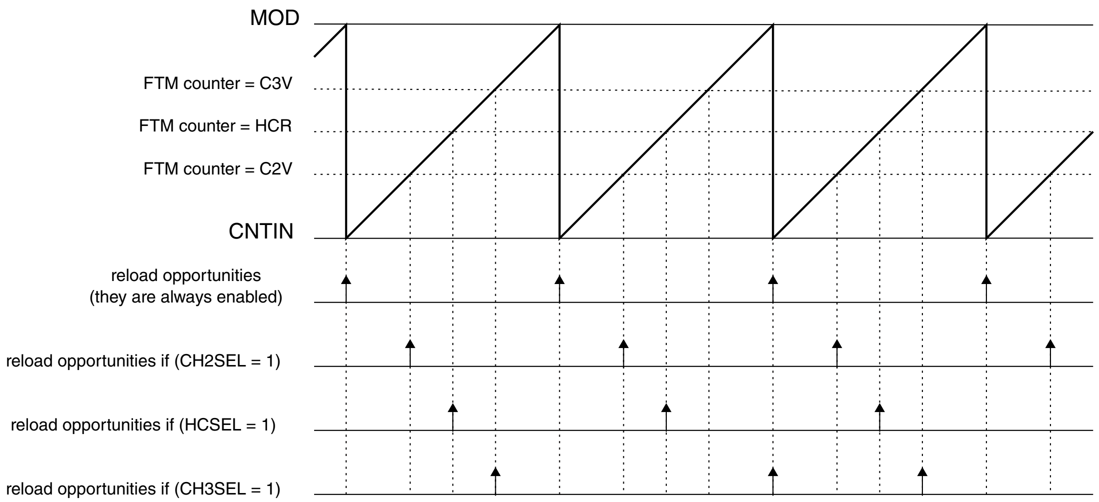

MOD
FTM counter = C3V
FTM counter = HCR
FTM counter = C2V
CNTIN
reload opportunities
(they are always enabled)
reload opportunities if (CH2SEL = 1)
reload opportunities if (HCSEL = 1)
reload opportunities if (CH3SEL = 1)

Figure 34-91. Reload opportunities when the FTM counter is an up counter

4. When the FTM counter is an up-down counter

   In this case, the reload opportunities are enabled by the bits CNTMAX and CNTMIN according to Table 34-21.

Table 34-21. Reload opportunities enabled by the bits CNTMAX and CNTMIN when the FTM counter is up-down counter

CNTMAX	CNTMIN	Reload Opportunities
0	0	when the FTM counter changes from (MOD) to (MOD - 1)
0	1	when the FTM counter changes from (CNTMIN) to (CNTMIN + 1)
1	0	when the FTM counter changes from (MOD) to (MOD - 1)
1	1	• when the FTM counter changes from (MOD) to (MOD - 1), and • when the FTM counter changes from (CNTMIN) to (CNTMIN + 1)

The following figure shows an example of the reload opportunities at the half cycle, at the channels match, and when the FTM counter is an up-down counter.

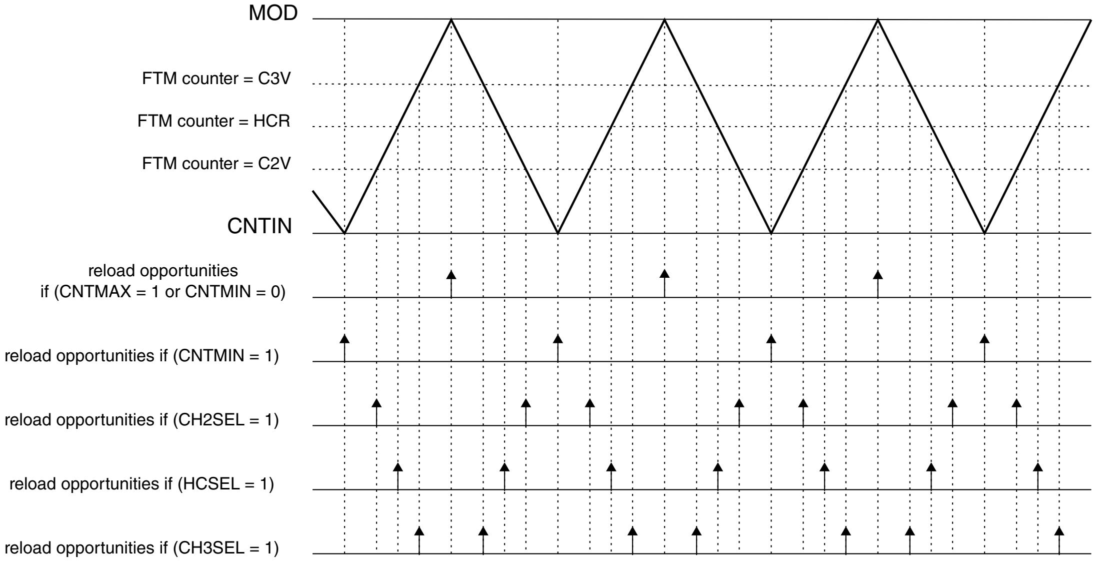

MOD
FTM counter = C3V
FTM counter = HCR
FTM counter = C2V
CNTIN
reload opportunities if (CNTMAX = 1 or CNTMIN = 0)
reload opportunities if (CNTMIN = 1)
reload opportunities if (CH2SEL = 1)
reload opportunities if (HCSEL = 1)
reload opportunities if (CH3SEL = 1)

Figure 34-92. Reload opportunities when the FTM counter is an up-down counter

34.5.28.2 Frequency of Reload Opportunities

The LDFQ[4:0] bits define the number of enabled reload opportunities should happen until an enabled reload opportunity becomes a reload point. The following figure shows an example when the LDFQ[4:0] = 4.

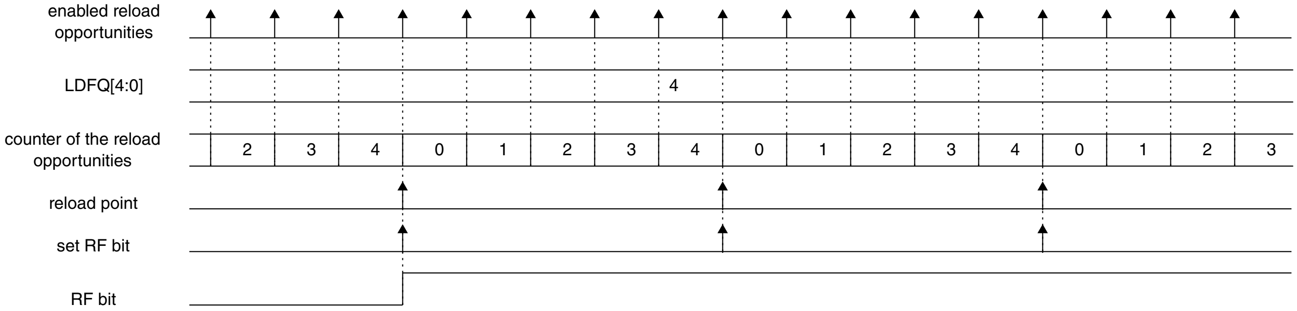

enabled reload opportunities
LDFQ[4:0]
4
counter of the reload opportunities
2
3
4
0
1
2
3
4
0
1
2
3
4
0
1
2
3
reload point
set RF bit
RF bit

Figure 34-93. Frequency of Reload Opportunities with LDFQ[4:0] = 4

If LDFQ[4:0] = 0, then all reload opportunities are reload points.

The counter of the reload opportunities is reset when there is a write to the register CNT.

The RF bit is set at each reload point (see the figure above) independent of LDOK bit value. The reload point interrupt is generated when (RF = 1) and (RIE = 1).

34.5.28.3 Update of the Registers

After writing new value to the registers with write buffer, selecting which of them will be updated (according to Table 34-22), selecting the reload opportunities, selecting the frequency of the reload opportunities, thus the LDOK bit should be set to enable the update of these registers at the next reload point.

Table 34-22. Additional conditions to update the registers

Register	Additional Condition
CNTIN	CNTINC = 1
HCR	-
MOD	-
C(n)V and C(n+1)V	SYNCENm = 1, where m is the pair of the channels (n) and (n +1)

34.5.29 Global Load

The global load mechanism allows several modules to have their double buffered registers synchronously reloaded after a synchronization event if a write to one operation is performed in the global load OK (GLDOK) bit in the PWMLOAD register. Global load may be enabled or disabled configuring the global load enable (GLEN) bit in the PWMLOAD register. Writing one in the GLDOK bit with GLEN enabled has the same effect of writing one in the LDOK bit. Refer to SoC specific information about global load connections.

Global load mechanism allows MOD, HCR, CNTIN, and C(n)V registers to be updated with the content of the register buffer at configurable reload point. The figure below shows an example of connection between FTM global load inputs and outputs considering that GLDOK bit is implemented outside from FTM module.

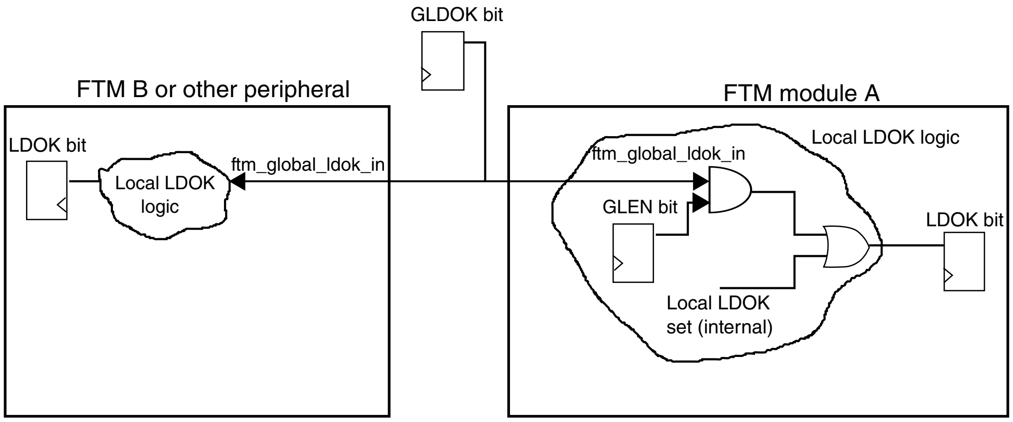


Figure 34-94. Global load logic

34.5.30 Global time base (GTB)

The global time base (GTB) is a FTM function that allows the synchronization of multiple FTM modules on a chip. The following figure shows an example of the GTB feature used to synchronize two FTM modules. In this case, the FTM A and B channels can behave as if just one FTM module was used, that is, a global time base.

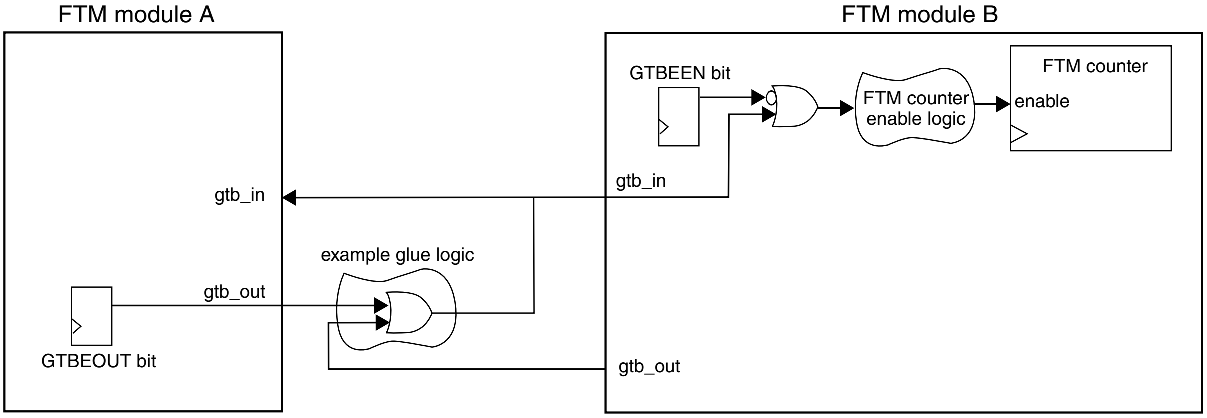


Figure 34-95. Global time base (GTB) block diagram

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the CONF register, the input signal *gtb_in*, and the output signal *gtb_out*. The GTBEEN bit enables gtb_in to control the FTM counter enable signal:

- If GTBEEN = 0, each one of FTM modules works independently according to their configured mode.
- If GTBEEN = 1, the FTM counter update is enabled only when gtb_in is 1.

In the configuration described in the preceding figure, FTM modules A and B have their FTM counters enabled if at least one of the gtb_out signals from one of the FTM modules is 1. There are several possible configurations for the interconnection of the gtb_in and gtb_out signals, represented by the example glue logic shown in the figure. Note that these configurations are chip-dependent and implemented outside of the FTM modules. See the chip-specific FTM information for the chip's specific implementation.

NOTE

- In order to use the GTB signals to synchronize the FTM counter of different FTM modules, the configuration of each FTM module should guarantee that its FTM counter starts counting as soon as the gtb_in signal is 1.
- The GTB feature does not provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during FTM operation.

The GTB feature only allows the FTM counters to *start* their operation synchronously.

34.5.30.1 Enabling the global time base (GTB)

To enable the GTB feature, follow these steps for each participating FTM module:

1. Stop the FTM counter: Write 00b to SC[CLKS].
2. Program the FTM to the intended configuration. The FTM counter mode needs to be consistent across all participating modules.
3. Write 1 to CONF[GTBEEN] and write 0 to CONF[GTBEOUT] at the same time.
4. Select the intended FTM counter clock source in SC[CLKS]. The clock source needs to be consistent across all participating modules.
5. Reset the FTM counter: Write any value to the CNT register.

To initiate the GTB feature in the configuration described in the preceding figure, write 1 to CONF[GTBEOUT] in the FTM module used as the time base.

34.5.31 Channel trigger output

The channel trigger output provides a trigger signal which has one FTM input clock period width in the channel (n) output.

If the TRIGMODE bit of the CnSC register is set (TRIGMODE = 1), a trigger pulse with one FTM input clock width is generated in the channel (n) output when a match occurs. It is only allowed to use trigger mode when channel (n) is in EPWM or CPWM modes.

The figures below show some cases of channel (n) trigger generation in the channel (n) output.

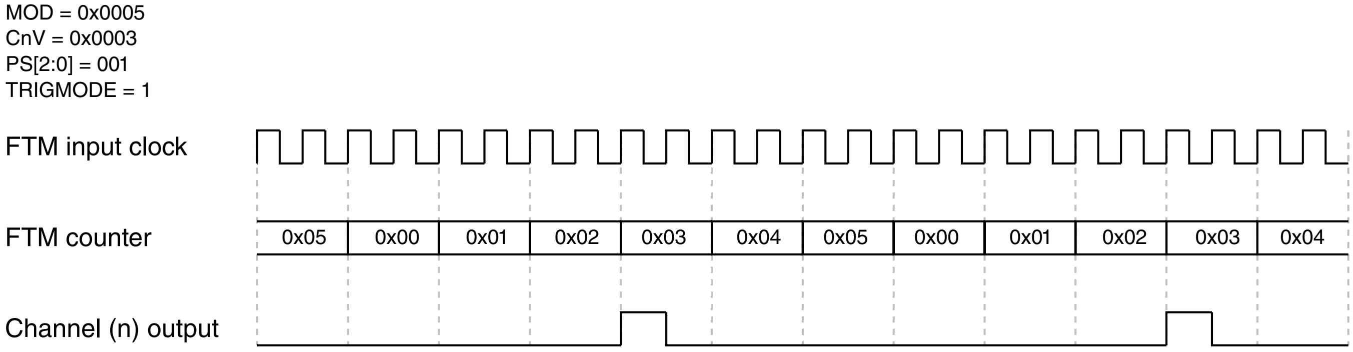


Figure 34-96. Example of channel (n) trigger at the channel (n) output in EPWM mode

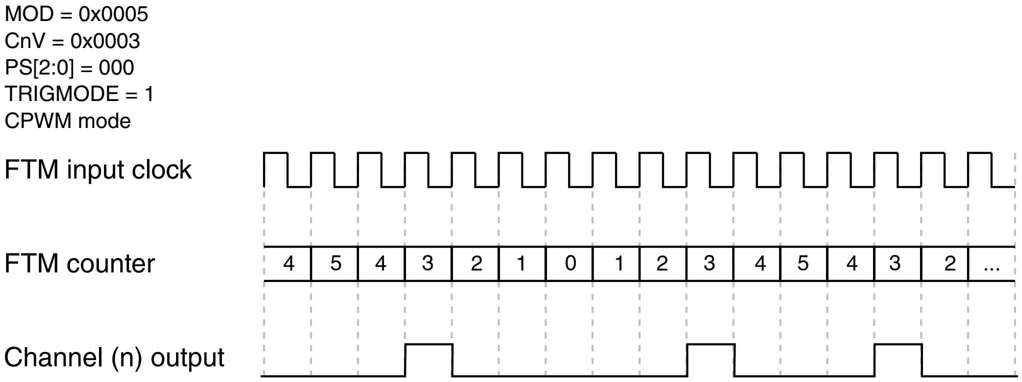


Figure 34-97. Example of channel (n) trigger at the channel (n) output in CPWM mode

34.5.32 External Control of Channels Output

The channel (n) PWMEN bit can be used in an FTM external logic to control the final value of the channel (n) output. This same logic can also control the channel (n) output when FSTATE = 1 and the channel (n) output is disabled by the Fault Control. The following figure shows an example of this external logic.

The term "channel (n) output" means the channel (n) output value after the Polarity Control. See Features Priority and Polarity Control for more details.

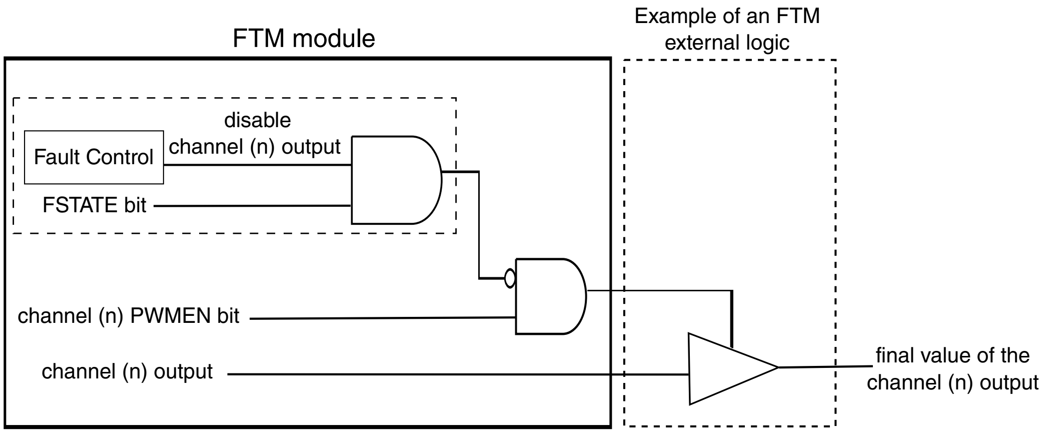


Figure 34-98. Example of the External Control of the Channel (n) Output

34.5.33 Dithering

FTM implements a fractional delay to achieve fine resolution on the generated PWM signals using dithering. The dithering can be used by applications where more resolution than one unit of the FTM counter is needed.

Two kinds of dithering are available: PWM period dithering and edge dithering.

34.5.33.1 PWM Period Dithering

The PWM period dithering is enabled when a non-zero value is written to FRACMOD.

The internal accumulator used in the PWM period dithering is reset when:

- the field MOD of the register MOD_MIRROR is updated with the value of its write buffer,
- the FRACMOD is updated with the value of its write buffer, or
- the FTM counter is stopped.

NOTE

> For the PWM period dithering, the register MOD_MIRROR should be used instead of the register MOD.
>
> To avoid inconsistencies, the field FRACMOD is cleared when the field MOD of the register MOD is updated with the value of its write buffer.
>
> The PWM period dithering is not available:
>
> - when the FTM counter is a free running counter

34.5.33.1.1 Up Counting

When the FTM counter is an up counter and the PWM period dithering is enabled, at the end of each PWM period, the FRACMOD value is added to an internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), then one unit of FTM counter is added to the end of the current PWM period, and the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

Due to one unit of FTM counter that can be added to the PWM period, the largest valid value for MOD is 0xFFFE for PWM period dithering with unsigned counting and 0x7FFE for PWM period dithering with signed counting.

The following figures show some examples of PWM period dithering when the FTM counter is an up counter.

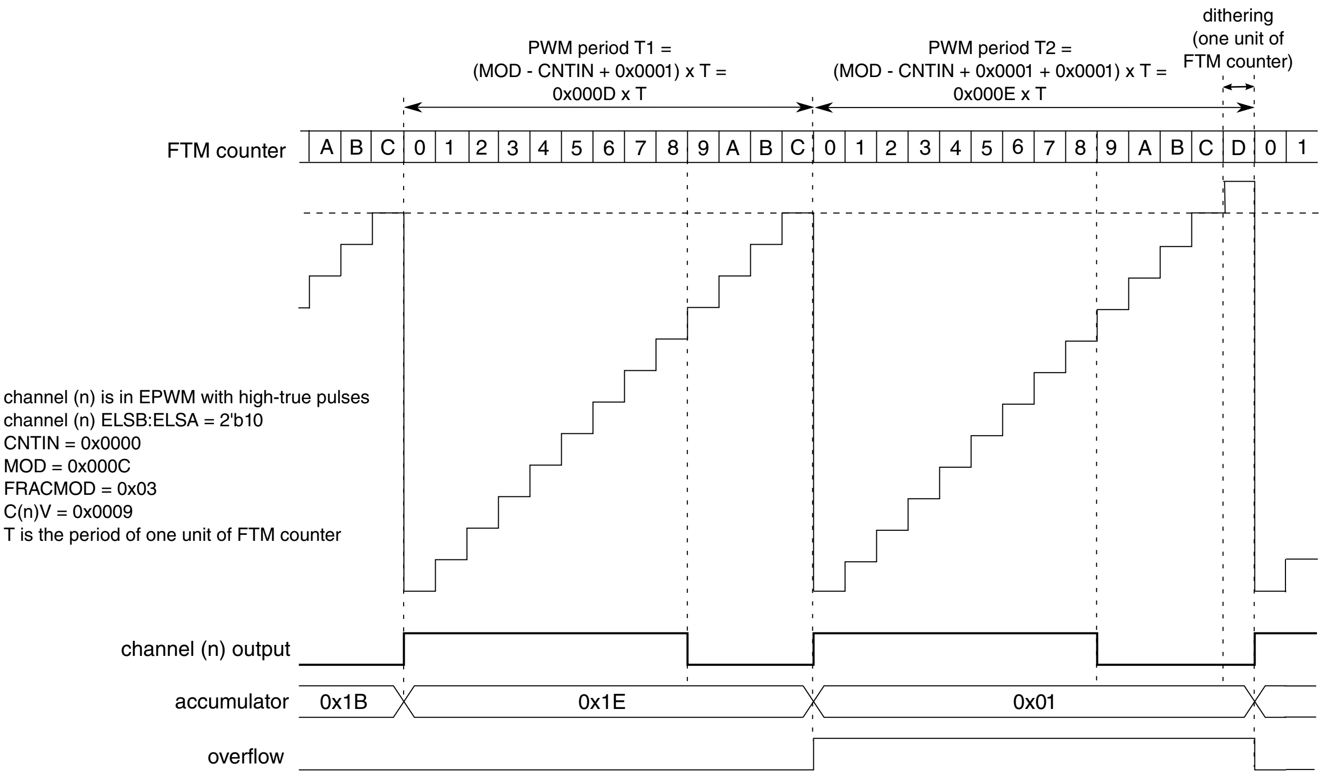

PWM period T1 =
(MOD - CNTIN + 0x0001) x T =
0x000D x T
PWM period T2 =
(MOD - CNTIN + 0x0001 + 0x0001) x T =
0x000E x T
dithering
(one unit of
FTM counter)
FTM counter
A B C 0 1 2 3 4 5 6 7 8 9 A B C 0 1 2 3 4 5 6 7 8 9 A B C D 0 1
channel (n) is in EPWM with high-true pulses
channel (n) ELSB:ELSA = 2'b10
CNTIN = 0x0000
MOD = 0x000C
FRACMOD = 0x03
C(n)V = 0x0009
T is the period of one unit of FTM counter
channel (n) output
accumulator
0x1B
0x1E
0x01
overflow

Figure 34-99. PWM Period Dithering with Up Counting

Assuming:

- the FTM counter is an up counter,
- T is one unit of FTM counter,
- the PWM period without period dithering is [(MOD - CNTIN + 1) x T],
- the number of PWM periods with period dithering is FRACMOD,
- the PWM period with period dithering is [(MOD - CNTIN + 1 + 1) x T],

thus, the average period (in decimal) is [(MOD - CNTIN + 1) + (FRACMOD/32)] x T, where the integer value is (MOD - CNTIN + 1) and the fractional value is (FRACMOD/ 32). See the example below.

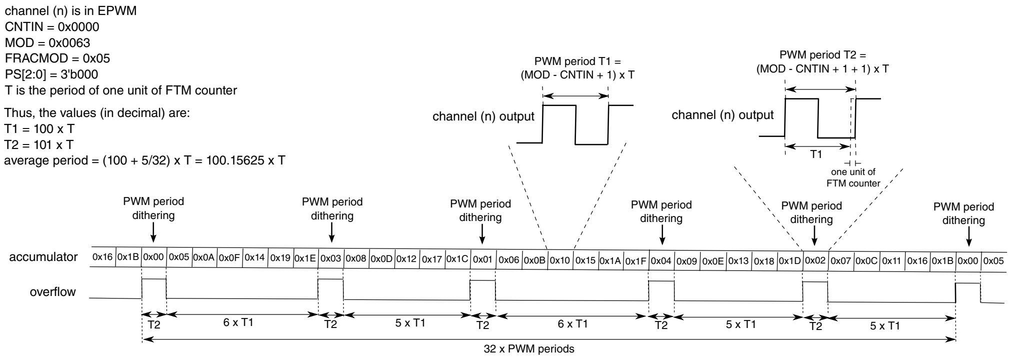

channel (n) is in EPWM
CNTIN = 0x0000
MOD = 0x0063
FRACMOD = 0x05
PS[2:0] = 3'b000
T is the period of one unit of FTM counter
Thus, the values (in decimal) are:
T1 = 100 x T
T2 = 101 x T
average period = (100 + 5/32) x T = 100.15625 x T
PWM period T1 =
(MOD - CNTIN + 1) x T
channel (n) output
PWM period T2 =
(MOD - CNTIN + 1 + 1) x T
channel (n) output
T1
one unit of
FTM counter
PWM period
dithering
accumulator
0x16 0x1B 0x00 0x05 0x0A 0x0F 0x14 0x19 0x1E 0x03 0x08 0x0D 0x12 0x17 0x1C 0x01 0x06 0x0B 0x10 0x15 0x1A 0x1F 0x04 0x09 0x0E 0x13 0x18 0x1D 0x02 0x07 0x0C 0x11 0x16 0x1B 0x00 0x05
overflow
T2
6 x T1
T2
5 x T1
T2
6 x T1
T2
5 x T1
T2
5 x T1
32 x PWM periods

Figure 34-100. Example of Average Period when the PWM Period Dithering is used with the Up Counting

NOTE

For the generation of 100% PWM signal in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using EPWM mode and PWM Period Dithering, it is recommended to use (C(n) > MOD + 1).

For the generation of PWM signals in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Period Dithering, it is recommended to use:

- For 0% PWM signal: (C(n)V > MOD + 1) and (C(n+1)V > MOD + 1);
- For 100% PWM signal: (C(n)V = CNTIN) and (C(n+1)V > MOD + 1).

For the generation of PWM signals in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Modified Combine PWM mode and PWM Period Dithering, it is recommended to use:

- For 0% PWM signal: (C(n)V > MOD + 1) and (CNTIN ≤ C(n+1)V ≤ MOD);
- For 100% PWM signal: (CNTIN ≤ C(n)V ≤ MOD) and (C(n+1)V > MOD + 1).

34.5.33.1.2 Up-Down Counting

When the FTM counter is an up-down counter and the PWM period dithering is enabled, at the end of each PWM period, the FRACMOD value is added to an internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), then one unit of FTM counter is added to the end of the current PWM period and other unit is added to the begin of the next PWM period (see the figure below). After the accumulator overflows, the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

Due to one unit of FTM counter that can be added to the PWM period, the largest valid value for MOD is 0x7FFE for PWM period dithering in up-down counting (CPWM mode).

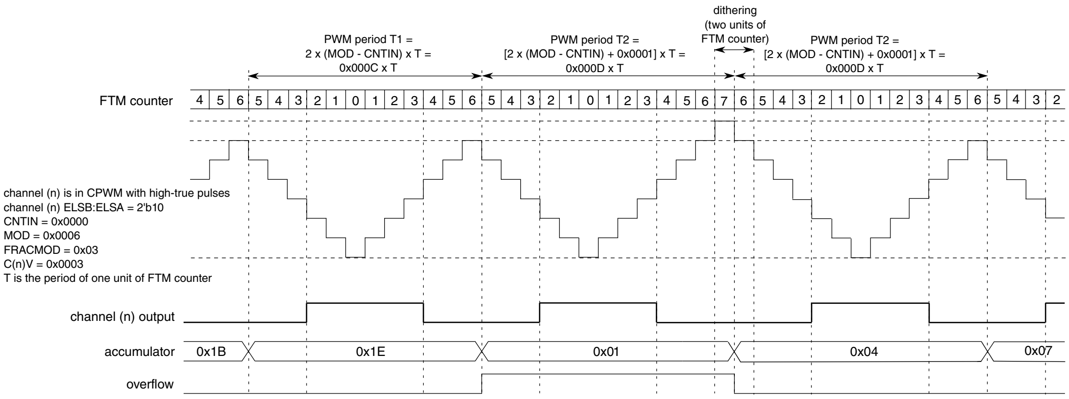


Figure 34-101. PWM Period Dithering with Up-Down Counting

NOTE

For the generation of 100% PWM signal in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using CPWM mode and PWM Period Dithering, it is recommended to use (C(n)V[15] = 0) and (C(n)V > MOD + 1) and (MOD ≠ 0x0000).

34.5.33.2 PWM Edge Dithering

The channel (n) internal accumulator used in the PWM edge dithering is reset when:

- the field VAL of the register C(n)V_MIRROR is updated with the value of its write buffer,
- the FRACVAL is updated with the value of its write buffer, or
- the FTM counter is stopped.

NOTE

For the PWM edge dithering, the register C(n)V_MIRROR should be used instead of the register C(n)V.

To avoid inconsistencies, the field FRACVAL is cleared when the field VAL of the register C(n)V is updated with the value of its write buffer.

The PWM edge dithering is not available:

- to the channel in input modes, and
- to the channel in output compare mode.

34.5.33.2.1 EPWM Mode

The PWM edge dithering for channel (n) in EPWM mode is enabled when a non-zero value is written to the channel (n) FRACVAL.

If the channel (n) is in EPWM mode and the PWM edge dithering is enabled, at the end of each EPWM period, the channel (n) FRACVAL value is added to the channel (n) internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

In this configuration, the initial edge of EPWM duty cycle happens when (FTM counter = CNTIN), its position is not modified by the PWM edge dithering. If there was not the overflow of the channel (n) accumulator in the current EPWM period, then the final edge of EPWM duty cycle happens on the channel (n) match (FTM counter = C(n)V), that is, its position is not modified by the edge dithering. However, if there was the overflow of the channel (n) accumulator in the current EPWM period, then the final edge of EPWM duty cycle happens when (FTM counter = C(n)V + 0x0001).

The following figures show some examples of PWM edge dithering when the channel (n) is in EPWM mode.

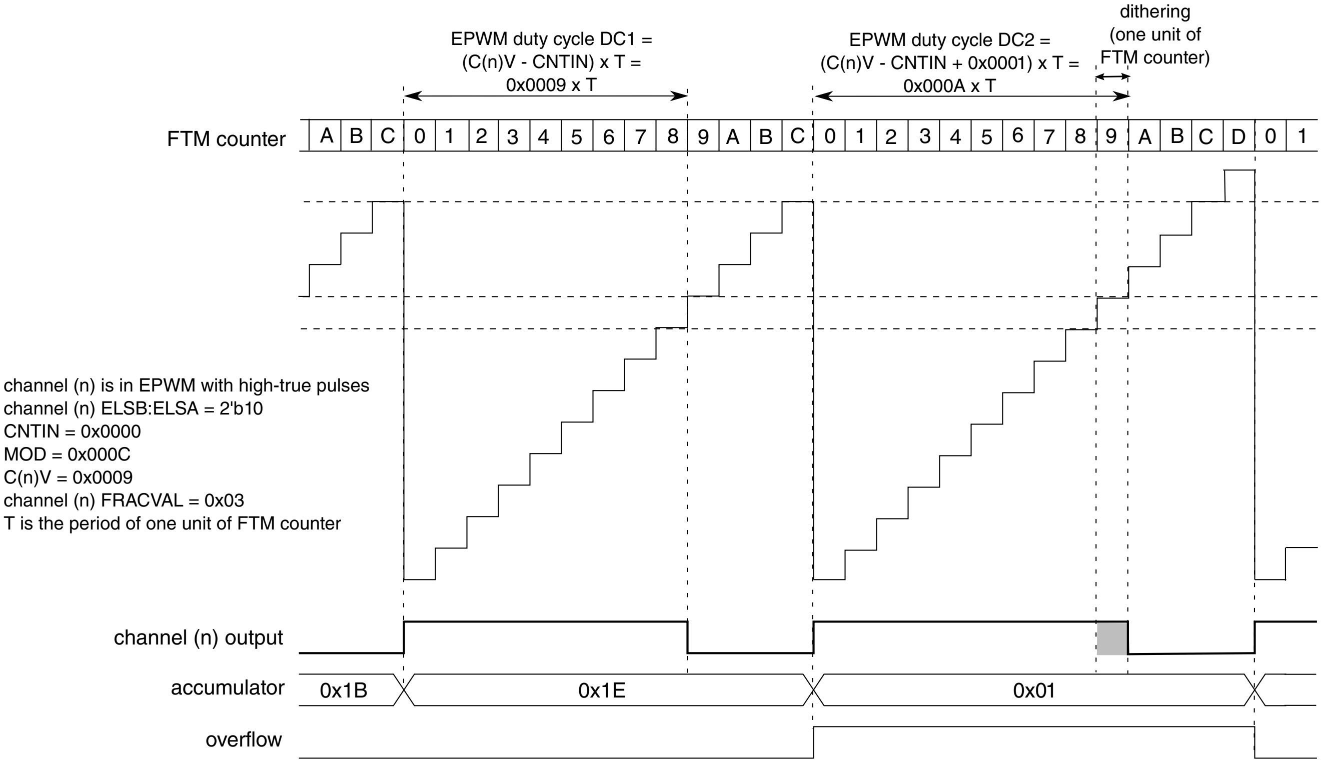

EPWM duty cycle DC1 =
(C(n)V - CNTIN) x T =
0x0009 x T
EPWM duty cycle DC2 =
(C(n)V - CNTIN + 0x0001) x T =
0x000A x T
dithering
(one unit of
FTM counter)
FTM counter
A B C 0 1 2 3 4 5 6 7 8 9 A B C 0 1 2 3 4 5 6 7 8 9 A B C D 0 1
channel (n) is in EPWM with high-true pulses
channel (n) ELSB:ELSA = 2'b10
CNTIN = 0x0000
MOD = 0x000C
C(n)V = 0x0009
channel (n) FRACVAL = 0x03
T is the period of one unit of FTM counter
channel (n) output
accumulator
0x1B
0x1E
0x01
overflow

Figure 34-102. Channel (n) is in EPWM Mode with PWM Edge Dithering

Assuming:

- the channel (n) is in EPWM mode,
- T is one unit of FTM counter,
- the EPWM duty cycle without edge dithering is [(C(n)V - CNTIN) x T],
- the number of PWM periods which duty cycle that has edge dithering is FRACVAL,
- the EWM duty cycle with edge dithering is [(C(n)V - CNTIN + 1) x T],

thus, the average duty cycle (in decimal) is [(C(n)V - CNTIN) + (FRACVAL/32)] x T, where the integer value is (C(n)V - CNTIN) and the fractional value is (FRACVAL/32). See the example below.

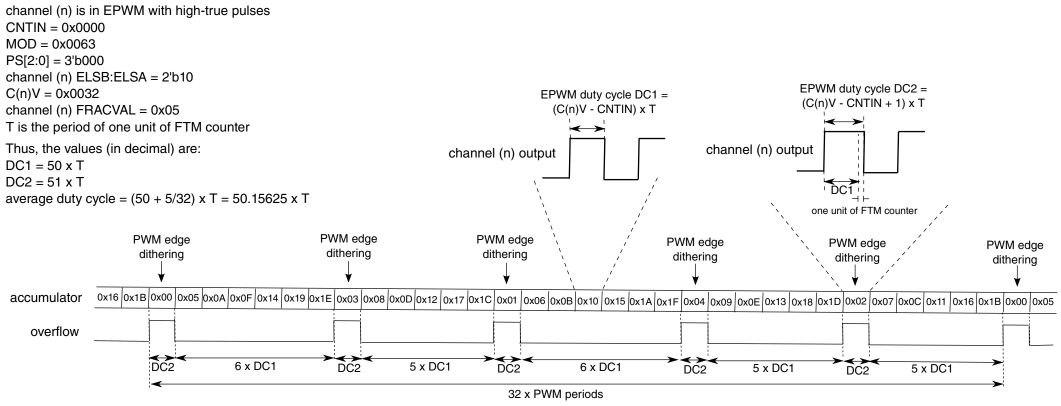


Figure 34-103. Example of Average Duty Cycle when the Channel (n) is in EPWM Mode with PWM Edge Dithering

34.5.33.2.2 CPWM Mode

The PWM edge dithering for channel (n) in CPWM mode is enabled when a non-zero value is written to the channel (n) FRACVAL.

If the channel (n) is in CPWM mode and the PWM edge dithering is enabled, at the end of each CPWM period, the channel (n) FRACVAL value is added to the channel (n) internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

In this configuration, if there was not the overflow of the channel (n) accumulator in the current CPWM period, then the duty cycle is not modified by the PWM edge dithering, that is, the initial edge of CPWM duty cycle happens on channel (n) match (FTM counter = C(n)V) when the FTM counter is decrementing, and the final edge of CPWM duty cycle on channel (n) match when the FTM counter is incrementing.

However, if there was the overflow of the channel (n) accumulator in the current CPWM period, then the initial edge of CPWM duty cycle happens when (FTM counter = C(n)V + 0x0001) and the FTM counter is decrementing, and the final edge of CPWM duty cycle when (FTM counter = C(n)V + 0x0001) and the FTM counter is incrementing.

The following figure shows an example of PWM edge dithering when the channel (n) is in CPWM mode.

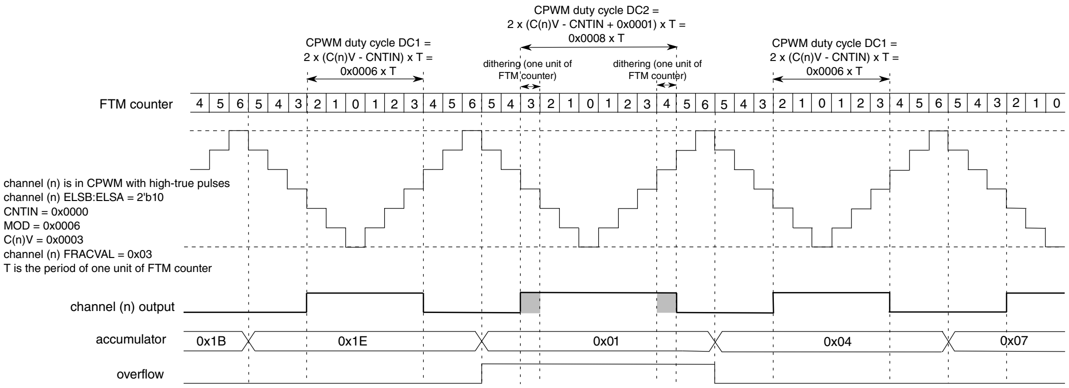


Figure 34-104. Channel (n) is in CPWM Mode with PWM Edge Dithering

34.5.33.2.3 Combine Mode

In the Combine mode, the PWM edge dithering can be done:

- in the channel (n) match (FTM counter = C(n)V) edge or
- in the channel (n+1) match (FTM counter = C(n+1)V) edge.

The channel (n) match edge dithering is enabled when a non-zero value is written to the channel (n) FRACVAL.

For the channel (n) match edge dithering, the channel (n) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n) FRACVAL value is added to the channel (n) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n) accumulator in the current PWM period, the channel (n) match edge is not modified, that is, it happens on channel (n) match. However, if there was the overflow of the channel (n) accumulator, the channel (n) match edge happens when (FTM counter = C(n)V + 0x0001).

The following figure shows an example of the channel (n) match edge dithering when the channels (n) and (n+1) are in Combine mode.

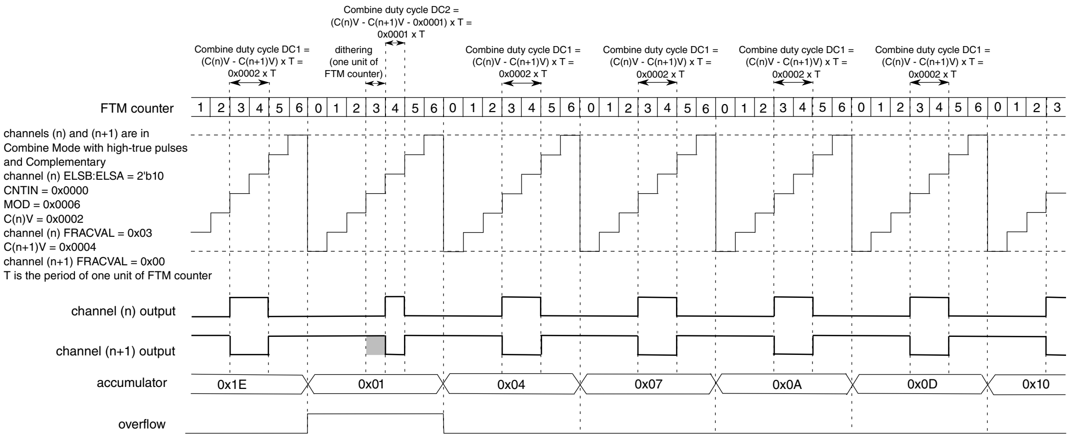


Figure 34-105. Channel (n) Match Edge Dithering in Combine Mode

The channel (n+1) match edge dithering is enabled when a non-zero value is written to the channel (n+1) FRACVAL.

For the channel (n+1) match edge dithering, the channel (n+1) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n+1) FRACVAL value is added to the channel (n+1) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n+1) accumulator in the current PWM period, the channel (n+1) match edge is not modified, that is, it happens on channel (n+1) match. However, if there was the overflow of the channel (n+1) accumulator, the channel (n+1) match edge happens when (FTM counter = C(n+1)V + 0x0001).

The following figure shows an example of the channel (n+1) match edge dithering when the channels (n) and (n+1) are in Combine mode.

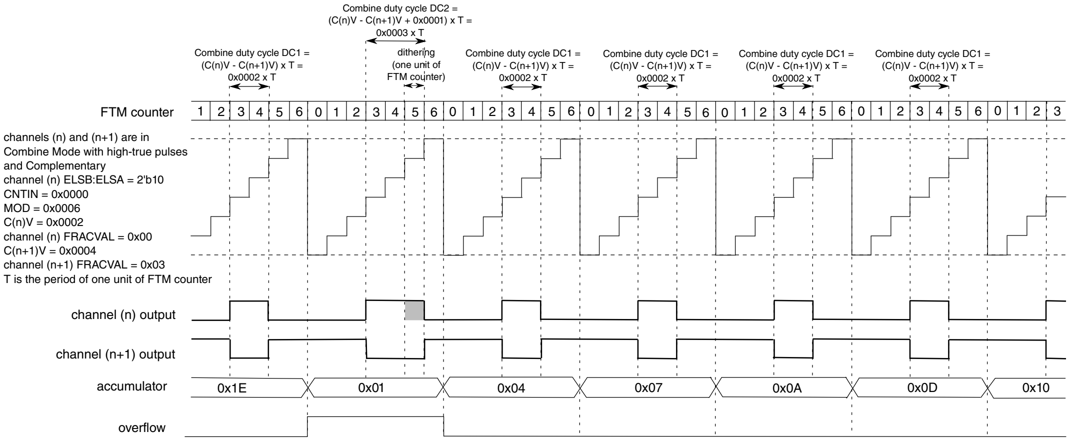


Figure 34-106. Channel (n+1) Match Edge Dithering in Combine Mode

NOTE

It is recommended to use only one PWM Edge Dithering (channel (n) PWM Edge Dithering or channel (n+1) PWM Edge Dithering) at a time.

For the generation of 0% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V < CNTIN or C(n)V > MOD) and (channel (n) FRACVAL is zero) and
- (channel (n+1) FRACVAL is zero).

For the generation of 100% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V = CNTIN) and (channel (n) FRACVAL is zero) and
- (C(n+1)V < CNTIN or C(n+1)V > MOD) and (channel (n +1) FRACVAL is zero).

34.5.33.2.4 Modified Combine PWM Mode

In the Modified Combine PWM mode, the PWM edge dithering can be done:

- in the channel (n) match (FTM counter = C(n)V) edge or
- in the channel (n+1) match (FTM counter = C(n+1)V) edge.

The channel (n) match edge dithering is enabled when a non-zero value is written to the channel (n) FRACVAL.

For the channel (n) match edge dithering, the channel (n) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n) FRACVAL value is added to the channel (n) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n) accumulator in the current PWM period, the channel (n) match edge is not modified, that is, it happens on channel (n) match. However, if there was the overflow of the channel (n) accumulator, the channel (n) match edge happens when (FTM counter = C(n)V + 0x0001).

The following figure shows an example of the channel (n) match edge dithering when the channels (n) and (n+1) are in Modified Combine PWM mode.

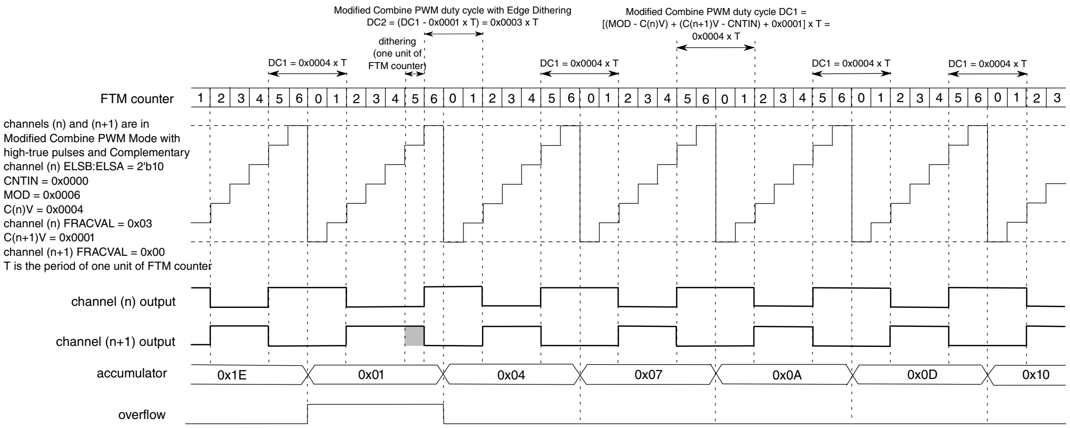

Modified Combine PWM duty cycle with Edge Dithering
DC2 = (DC1 - 0x0001 x T) = 0x0003 x T
Modified Combine PWM duty cycle DC1 =
[(MOD - C(n)V) + (C(n+1)V - CNTIN) + 0x0001] x T =
0x0004 x T
dithering
(one unit of
FTM counter)
DC1 = 0x0004 x T
DC1 = 0x0004 x T
DC1 = 0x0004 x T
DC1 = 0x0004 x T
FTM counter
1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3
channels (n) and (n+1) are in
Modified Combine PWM Mode with
high-true pulses and Complementary
channel (n) ELSB:ELSA = 2'b10
CNTIN = 0x0000
MOD = 0x0006
C(n)V = 0x0004
channel (n) FRACVAL = 0x03
C(n+1)V = 0x0001
channel (n+1) FRACVAL = 0x00
T is the period of one unit of FTM counter
channel (n) output
channel (n+1) output
accumulator
0x1E
0x01
0x04
0x07
0x0A
0x0D
0x10
overflow

Figure 34-107. Channel (n) Match Edge Dithering in Modified Combine PWM Mode

The channel (n+1) match edge dithering is enabled when a non-zero value is written to the channel (n+1) FRACVAL.

For the channel (n+1) match edge dithering, the channel (n+1) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n+1) FRACVAL value is added to the channel (n+1) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n+1) accumulator in the current PWM period, the channel (n+1) match edge is not modified, that is, it happens on channel (n+1) match. However, if there was the overflow of the channel (n+1) accumulator, the channel (n+1) match edge happens when (FTM counter = C(n+1)V + 0x0001).

The following figure shows an example of the channel (n+1) match edge dithering when the channels (n) and (n+1) are in Modified Combine PWM mode.

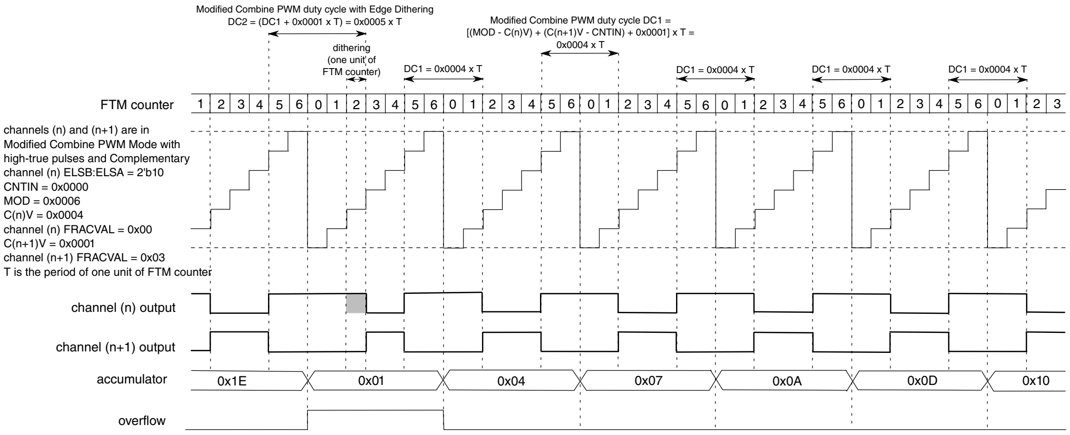

Modified Combine PWM duty cycle with Edge Dithering
DC2 = (DC1 + 0x0001 x T) = 0x0005 x T
Modified Combine PWM duty cycle DC1 =
[(MOD - C(n)V) + (C(n+1)V - CNTIN) + 0x0001] x T =
0x0004 x T
dithering
(one unit of
FTM counter)
DC1 = 0x0004 x T
DC1 = 0x0004 x T
DC1 = 0x0004 x T
DC1 = 0x0004 x T
FTM counter
1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3 4 5 6 0 1 2 3
channels (n) and (n+1) are in
Modified Combine PWM Mode with
high-true pulses and Complementary
channel (n) ELSB:ELSA = 2'b10
CNTIN = 0x0000
MOD = 0x0006
C(n)V = 0x0004
channel (n) FRACVAL = 0x00
C(n+1)V = 0x0001
channel (n+1) FRACVAL = 0x03
T is the period of one unit of FTM counter
channel (n) output
channel (n+1) output
accumulator
0x1E
0x01
0x04
0x07
0x0A
0x0D
0x10
overflow

Figure 34-108. Channel (n+1) Match Edge Dithering in Modified Combine PWM Mode

NOTE

It is recommended to use only one PWM Edge Dithering (channel (n) PWM Edge Dithering or channel (n+1) PWM Edge Dithering) at a time.

For the generation of 0% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Modified Combine PWM mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V < CNTIN or C(n)V > MOD) and (channel (n) FRACVAL is zero) and
- (CNTIN ≤ C(n+1)V ≤ MOD) and (channel (n+1) FRACVAL is zero).

For the generation of 100% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Modified Combine PWM mode and PWM Edge Dithering, it is recommended to use:

- (CNTIN ≤ C(n)V ≤ MOD) and (channel (n) FRACVAL is zero) and
- (C(n+1)V < CNTIN or C(n+1)V > MOD) and (channel (n +1) FRACVAL is zero).

34.6 Reset Overview

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

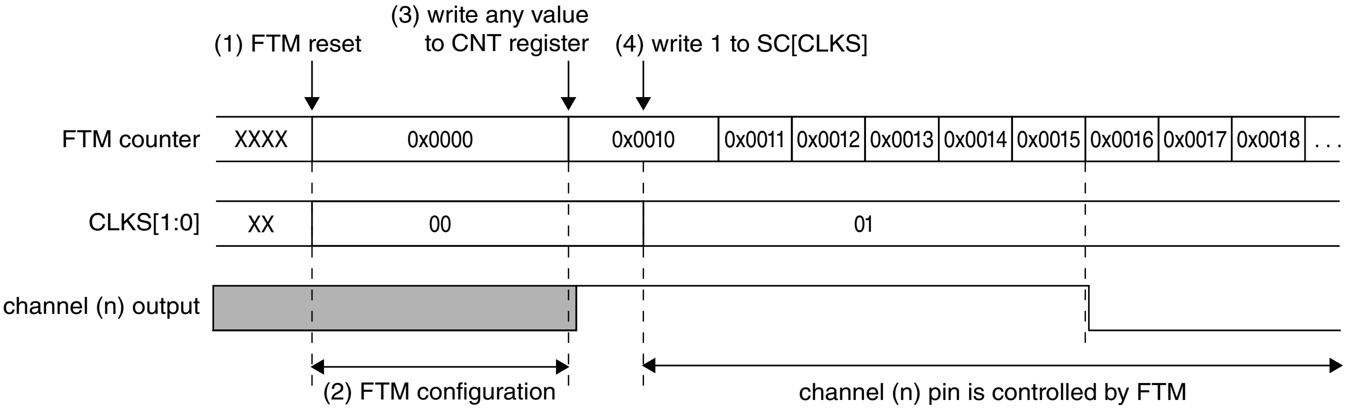
- the FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 2'b00);
- the timer overflow interrupt is zero (Timer Overflow Interrupt);
- the channels interrupts are zero (Channel (n) Interrupt);
- the fault interrupt is zero (Fault Interrupt);
- the channels are in input capture mode (Input Capture Mode);
- the channels outputs are zero;
- the channels ELSB:ELSA = 0:0 (Channel Modes) and PWMEN = 0 (External Control of Channels Output).

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (CLKS[1:0] = 2'b00), its value is updated to zero and the pins are not controlled by FTM (Channel Modes).

After the reset, the FTM should be configured (item 2). It is necessary to define the FTM counter mode, the FTM counting limits (MOD and CNTIN registers value), the channels mode and CnV registers value according to the channels mode.

Thus, it is recommended to write any value to CNT register (item 3). This write updates the FTM counter with the CNTIN register value and the channels output with its initial value (except for channels in output compare mode) (Counter reset).

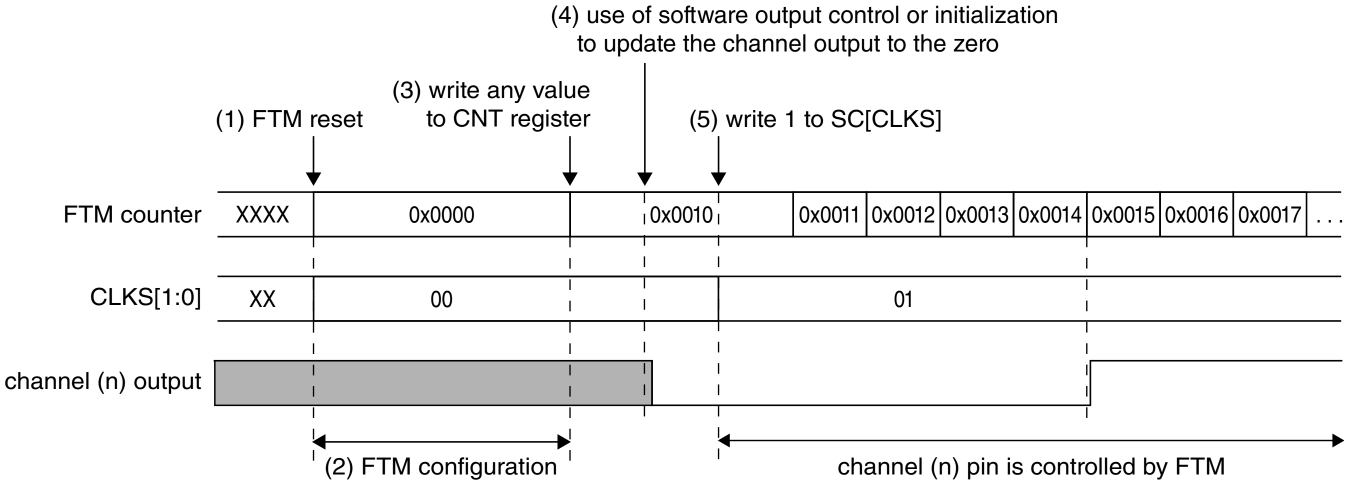
The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are only controlled by FTM when CLKS[1:0] bits are different from zero.



NOTES:
- CNTIN = 0x0010
- Channel (n) is in low-true combine mode with CNTIN < C(n)V < C(n+1)V < MOD
- C(n)V = 0x0015

Figure 34-109. FTM behavior after reset when the channel (n) is in Combine mode

The following figure shows an example when the channel (n) is in Output Compare mode and the channel (n) output is toggled when there is a match. In the Output Compare mode, the channel output is not updated to its initial value when there is a write to CNT register (item 3). In this case, use the software output control (Software Output Control Mode) or the initialization (Initialization) to update the channel output to the selected value (item 4).


(4) use of software output control or initialization to update the channel output to the zero
(3) write any value to CNT register
(1) FTM reset
(5) write 1 to SC[CLKS]
FTM counter
XXXX
0x0000
0x0010
0x0011
0x0012
0x0013
0x0014
0x0015
0x0016
0x0017
. . .
CLKS[1:0]
XX
00
01
channel (n) output
(2) FTM configuration
channel (n) pin is controlled by FTM

NOTES:
- CNTIN = 0x0010
- Channel (n) is in output compare and the channel (n) output is toggled when there is a match
- C(n)V = 0x0014

Figure 34-110. FTM behavior after reset when the channel (n) is in Output Compare mode

34.7 FTM Interrupts

34.7.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

34.7.2 Reload Point Interrupt

The Reload Point interrupt is generated when (RIE = 1) and (RF = 1).

34.7.3 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHIE = 1) and (CHF = 1).

34.7.4 Fault Interrupt

The fault interrupt is generated when (FAULTIE = 1) and (FAULTF = 1).

34.8 Initialization Procedure

The following initialization procedure is recommended to configure the FlexTimer. This procedure can also be used to do a new configuration.

1. Define the POL bits.
2. Mask the channels outputs using SYNCHOM = 0. Two clocks after the write to OUTMASK, the channels outputs are in the safe value.
3. (Re)Configuration FTM counter and channels to generation of periodic signals:
   a. Disable the clock.
   b. Examples of (re)configuration:
   - Write to MOD
   - Write to CNTIN
   - Configure the channels that will be used
   - Write to CnV for the channels in output modes
   - (Re)Configure deadtime and fault control
   - Do not use the SWOC without SW synchronization (see item 6)
   - Do not use the Inverting without SW synchronization (see item 6)
   - Do not use the Initialization
   - Do not change the polarity control
   - Do not configure the HW synchronization
4. Write any value to CNT. The FTM Counter is reset and the channels outputs are updated according to new configuration.
5. Enable the clock. Write to CLKS[1:0] bits a value different from zero.
6. Configure the SW synchronization for SWOC (if it is necessary), Inverting (if it is necessary) and Output Mask (always)
   a. Select synchronization for Output Mask
      - Write to SYNC (SWSYNC = 0, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)

   b. Write to SYNCONF
      - HW Synchronization can not be enabled (HWSOC = 0, HWINVC = 0, HWOM = 0, HWWRBUF = 0, HWRSTCNT = 0, HWTRIGMODE = 0)
      - SW Synchronization for SWOC (if it is necessary): SWSOC = [0/1] and SWOC = [0/1]
      - SW Synchronization for Inverting (if it is necessary): SWINVC = [0/1] and INVC = [0/1]
      - SW Synchronization for SWOM (always): SWOM = 1
      - No enable the SW Synchronization for write buffers (because the writes to registers with write buffer are done using CLKS[1:0] = 2’b00): SWWRBUF = 0 and CNTINC = 0

  - SW Synchronization for counter reset (always): SWRSTCNT = 1
  - Enhanced synchronization (always): SYNCMODE = 1

c. If the SWOC is used (SWSOC = 1 and SWOC = 1), then write to SWOCTRL register.

d. If the Inverting is used (SWINVC = 1 and INVC = 1), then write to INVCTRL register.

e. Write to OUTMASK to enable the masked channels.

7. Generate the Software Trigger
   - Write to SYNC (SWSYNC = 1, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)
8. Configure PWMEN bits (External Control of Channels Output).

34.9 Usage Guide

34.9.1 FTM Interrupts

The FlexTimer has multiple sources of interrupt. However, these sources are OR'd together to generate a single interrupt request to the interrupt controller. When an FTM interrupt occurs, read the FTM status registers (FMS, SC, and STATUS) to determine the exact interrupt source.

34.9.2 FTM Hall sensor support

For 3 phase motor control sensor-ed applications the use of Hall sensors, generally 3 sensors placed 120 degrees apart around the rotor, are deployed to detect position and speed. Each of the 3 sensors provides a pulse that applied to an input capture pin, can then be analyzed and both speed and position can be deduced. This device has two 2-channel FTMs (FTM1 and FTM2) and thus provides 4 input capture pins. To simplify the calculations required by the CPU on each hall sensor's input, if all 3 inputs are "exclusively OR'd" into one timer channel and the free running counter is refreshed on every edge then this can simplify the speed calculation.

Via the SIM module and SIM_FTMOPT1 register the FTM2CH1SEL bit provides the choice of normal FTM2_CH1 input or the XOR of FTM2_CH0, FTM2_CH1 and FTM1_CH1 pins that will be applied to FTM2_CH1.

NOTE

If the user utilizes FTM1_CH1 to be an input to FTM2_CH1, FTM1_CH0 can still be utilized for other functions.

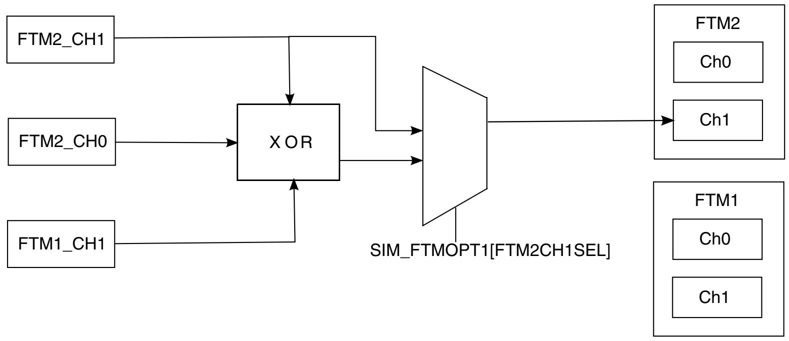


Figure 34-111. FTM Hall Sensor Configuration

34.9.3 FTM Modulation Implementation

FTM0 support a modulation function where the output channels when configured as PWM or Output Compare mode modulate another timer output when the channel signal is asserted. Any of the 8 channels of FTM0 can be configured to support this modulation function.

The SIM_FTMOPT1 register has control bits (FTMxCHySEL) that allow the user to select normal PWM/Output Compare mode on the corresponding FTM timer channel or modulate with FTM1_CH1. The diagram below shows the implementation for FTM0. See SIM Block Guide for further information.

When FTM1_CH1 is used to modulate an FTM0 channel, then the user must configure FTM1_CH1 to provide a signal that has a higher frequency than the modulated FTM0 channel output. Also it limits the use of the FTM1_CH0 function, as the FTM1_CH1 will be programmed to provide a 50% duty PWM signal and limit the start and modulus values for the free running counter.

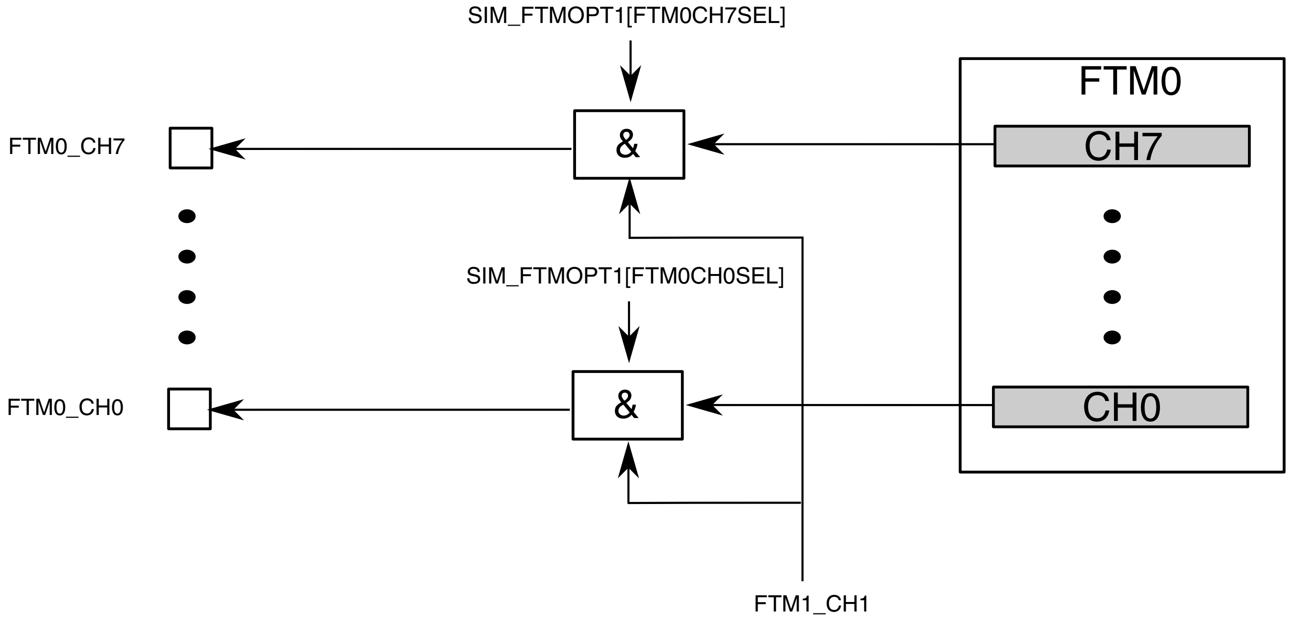
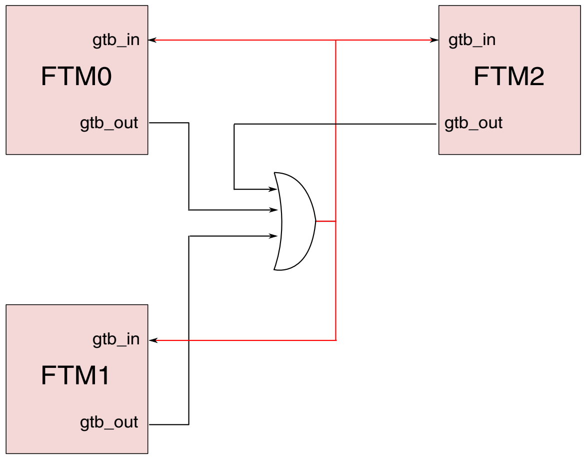


Figure 34-112. FTM Output Modulation

34.9.4 FTM Global Time Base

This chip provides the optional FTM global time base feature, see Global time base (GTB).

FTM supports global timer base through the GTB feature. Any of the FTM module could be used as the GTB_EN source. The global timer base only allows the FTM counters to start their operation synchronously, it does not automatically provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during misc FTM operation.



34.9.5 FTM BDM and debug halt mode

In the FTM chapter, references to the chip being in "BDM" are the same as the chip being in “debug halt mode".

Chapter 35
Low-power Periodic Interrupt Timer (LPIT)

35.1 Chip-specific Information for this Module

35.1.1 Instantiation Information

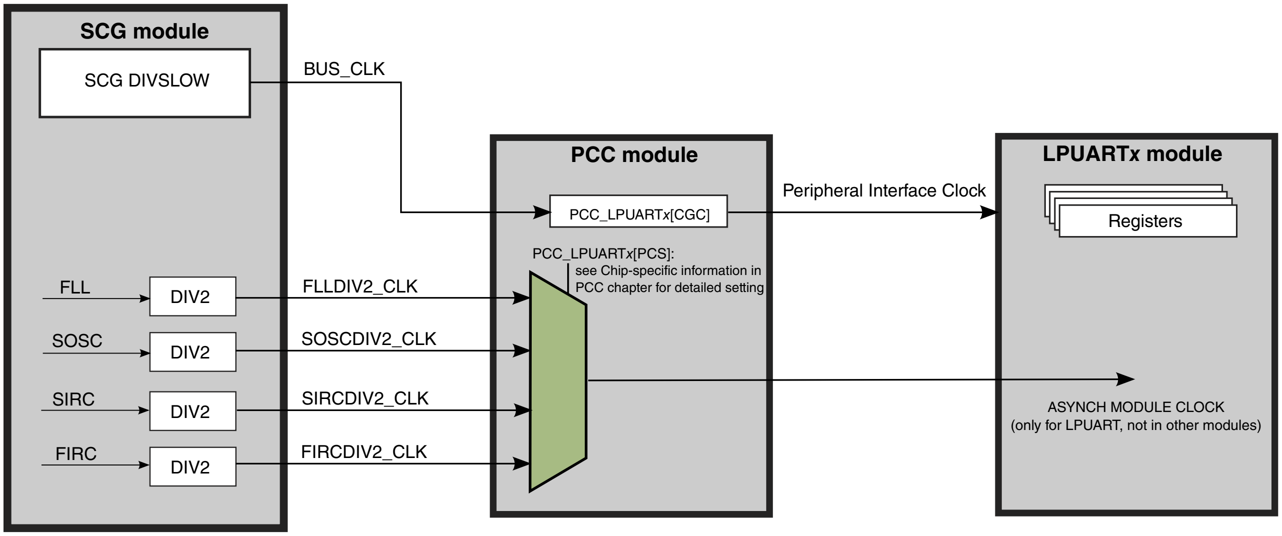
This device contains one LPIT module with four channels.

NOTE

The reset value of PARAM register is 0000_0404h, for this device.

35.1.2 LPIT Clocking Information

The LPIT module is only clocked by system clock shown in following diagram.

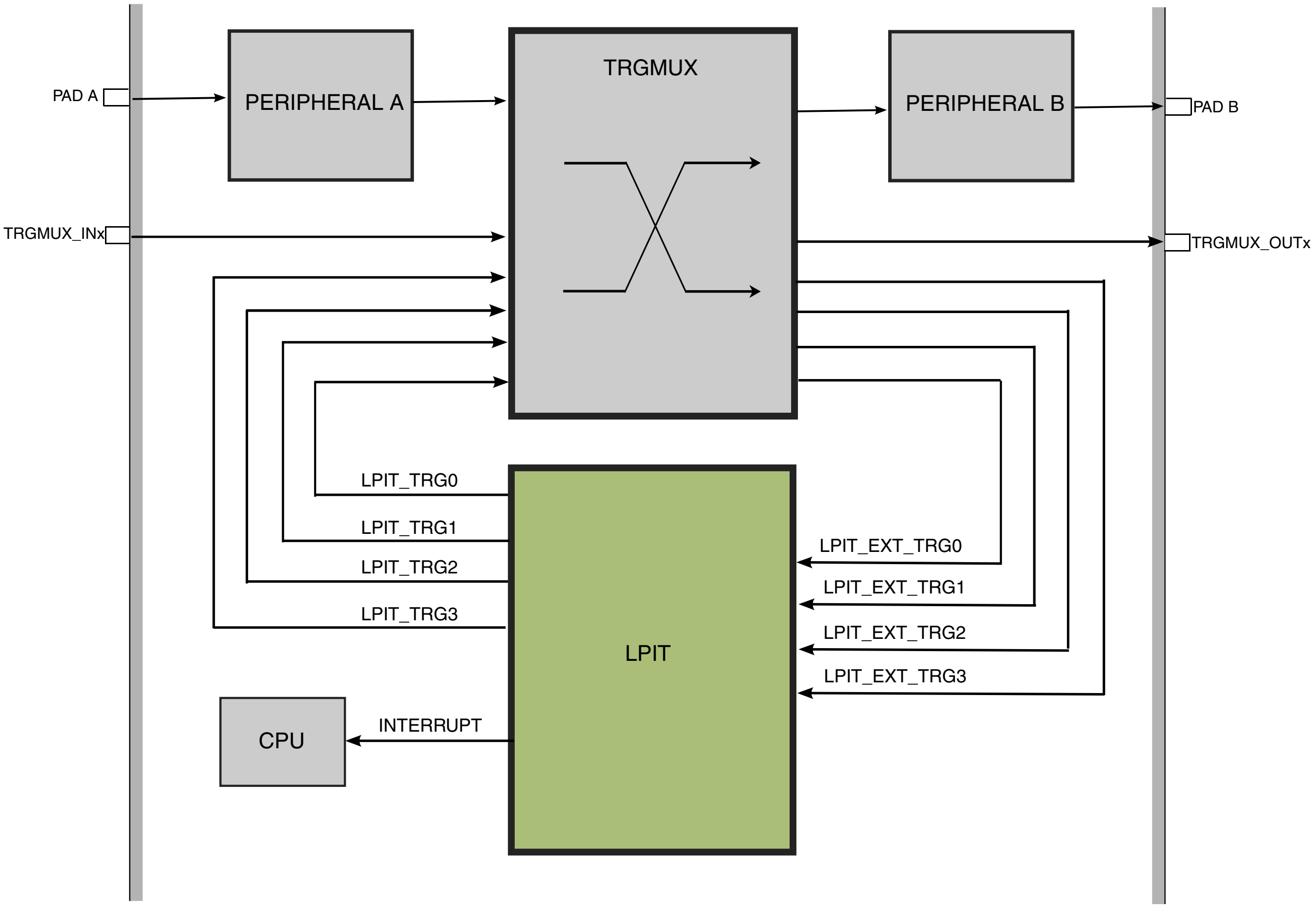


Peripheral Clocking - LPUART, etc.

Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, LPIT, FlexIO, etc.

35.1.3 Inter-connectivity Information

The LPIT module interconnectivity with other peripherals is based on the TRGMUX.



35.2 Overview

LPIT is a low-power periodic interrupt timer with multiple timer channels. After a timer reaches a programmed count, the respective timer channels generate pre-trigger and trigger output signals, and these outputs can be used to trigger other modules on the chip.

35.2.1 Block diagram

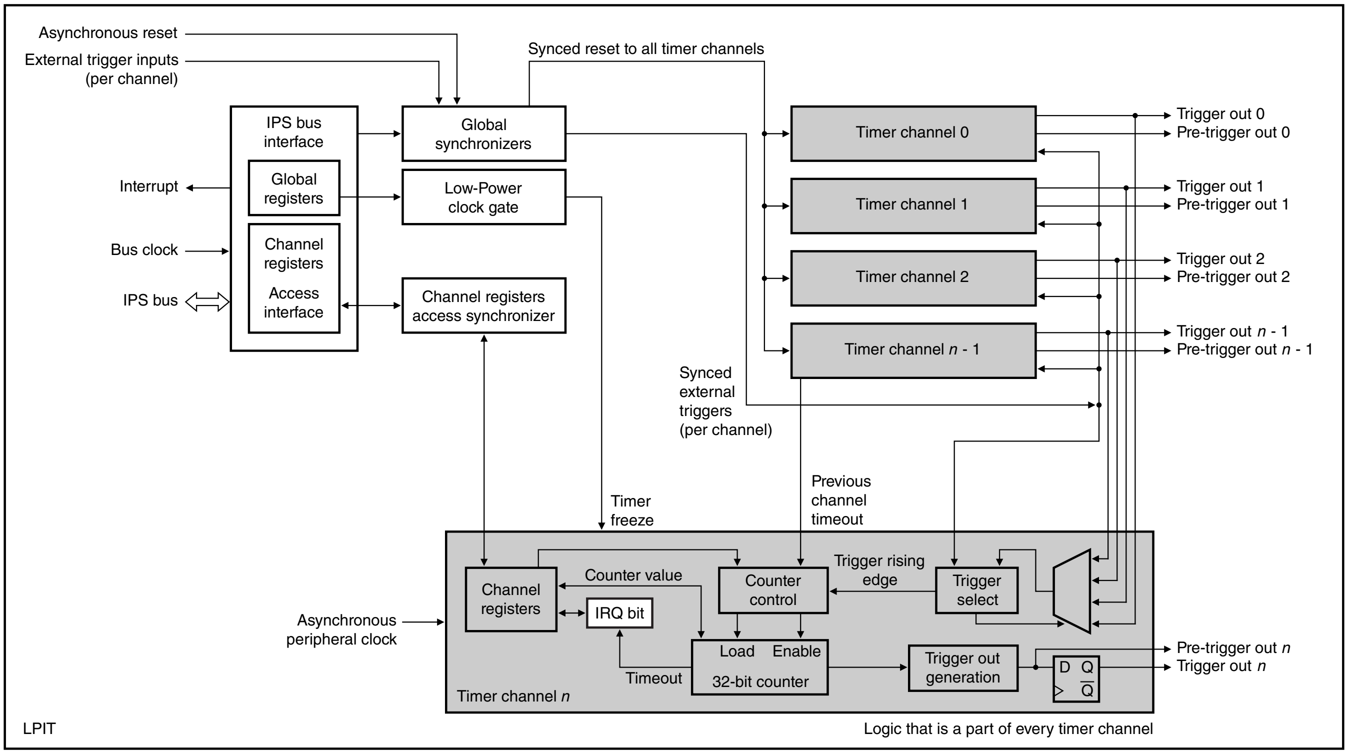


Figure 35-1. Block diagram

35.2.2 Features

LPIT allows you to configure timer channels in a way that they could be:

- Controlled using external triggers (triggers from outside LPIT).
- Controlled using internal triggers (triggers from other timer channels inside LPIT).
- Chained together, to form a larger width timer.
- Reloaded and counted again, or stopped after reaching the programmed count, depending on the timer modes used.

35.3 Functional description

35.3.1 Programming model

The LPIT programming model (see Figure 35-2) comprises:

- A global register set (common to all timer channels).
- Registers for each timer channel (that control their respective timer channels).

Access to these registers is synchronized with the asynchronous peripheral clock and then affects the timer channel registers:

- Each timer channel contains a 32-bit counter that loads the starting value and down counts after every peripheral clock's positive edge.
- After reaching a zero value (a channel timer timeout), a trigger output is generated.
- A timer enable register control field, external or internal triggers, or a previous channel timeout (when using timer chaining) control the counter enable.
- After a channel timer timeout, an interrupt flag is also set to tell the CPU about the timer timeout.

You must configure the following global registers only once:

- Module Control (MCR)
- Module Status (MSR)
- Module Interrupt Enable (MIER)
- Set Timer Enable (SETTEN)
- Clear Timer Enable (CLRTEN)

You must configure the following channel registers for each channel:

- Timer Value (TVAL0 - TVAL3)
- Timer Control (TCTRL0 - TCTRL3)

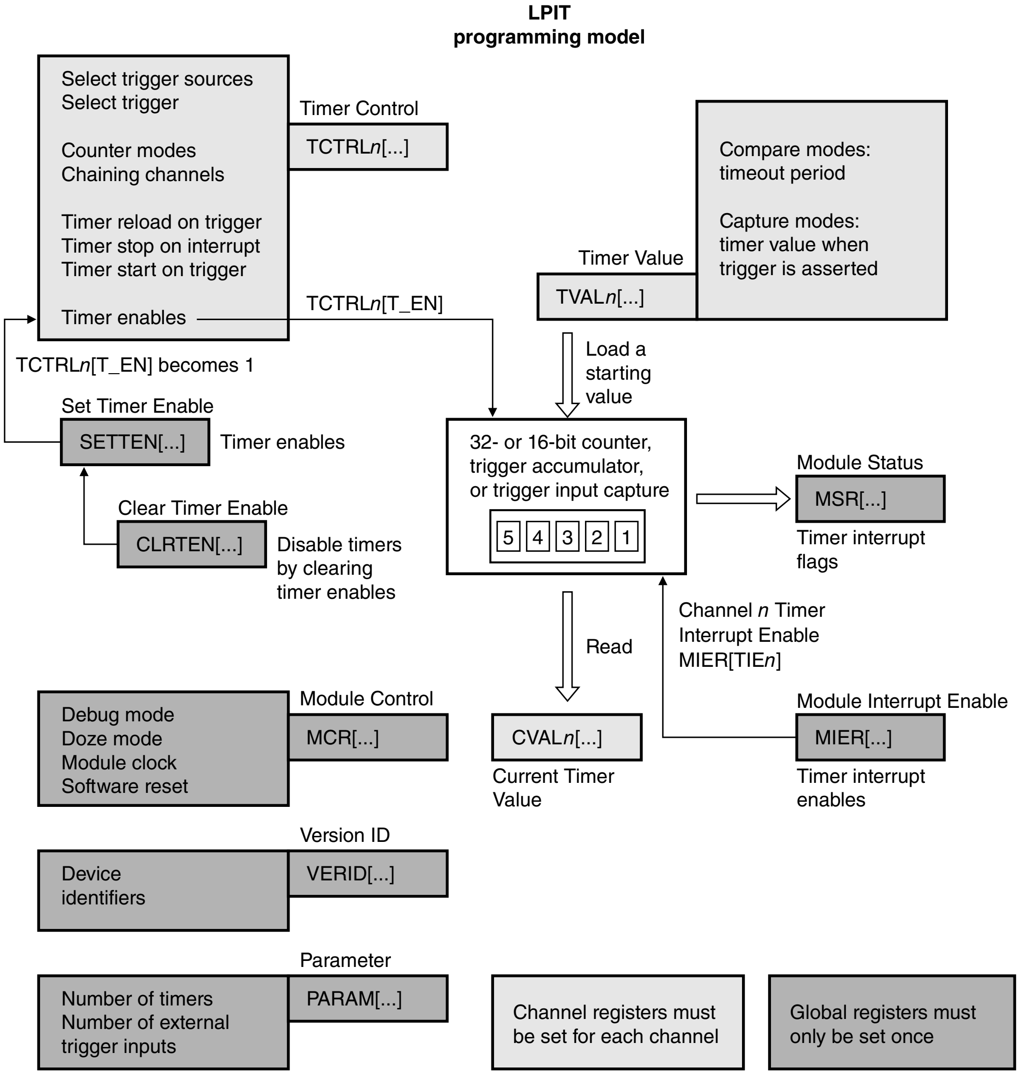


Figure 35-2. Programming model

35.3.2 Interfacing with other modules

The following figure shows the interface of an LPIT module with other modules on the chip:

- The CPU interface provides the clock, reset, and register read and write bus interface. It handles LPIT interrupts.
- The LPIT trigger output signals may trigger other modules on the chip, such as DMA and ADC.
- Similarly, other timer modules may provide trigger inputs to LPIT to control when an LPIT timer channel must start.

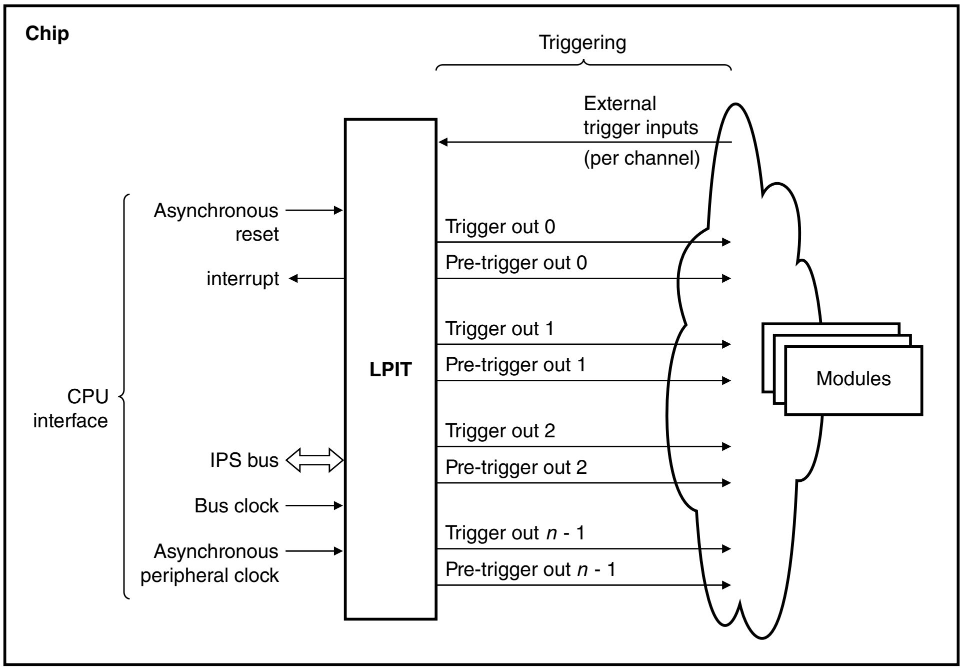


Figure 35-3. Interfacing with other modules

35.3.3 Chip power modes

LPIT supports the following chip power modes.

Table 35-1. Chip power modes

Chip mode	LPIT operation
Run	Normal operation
STOP and Wait	Can continue operating in this mode if MCR[DOZE_EN] = 1 and LPIT is using an external or internal clock source that remains operational during STOP and Wait modes.
Debug	Can continue operating in this mode if MCR[DBG_EN] = 1

35.3.4 Supported timer modes

To configure a timer mode, you must select an appropriate value using TCTRL*n*[MODE].

Table 35-2. Supported timer modes

Timer mode	Operation
32-bit periodic counter	The counter loads and decrements down to 0, and this sets the timer interrupt flag and asserts the output pretrigger.
Dual 16-bit periodic counter	• The counter loads and then the lower 16 bits decrement down to 0. This asserts the output pretrigger. • The upper 16 bits then decrement down to 0. This sets the timer interrupt flag and negates the output pretrigger.
32-bit trigger accumulator	The counter loads after the first trigger rising edge and then decrements down to 0 after each trigger rising edge. This sets the timer interrupt flag and asserts the output pretrigger.
32-bit trigger input capture	• The counter loads with a value of FFFF_FFFFh and then decrements down to 0. • If a trigger rising edge is detected, it stores the inverse of the current counter value in Timer Value (TVAL0 - TVAL3). This sets the timer interrupt flag and asserts the output pretrigger.

TCTRL*n*[TSOT], TCTRL*n*[TSOI], and TCTRL*n*[TROT] control the timer operation. These fields control the timer load, reload, start, and restart of the timers.

NOTE

- The trigger output is asserted one peripheral timer clock cycle after the pre-trigger output. The trigger and pre-trigger outputs deassert at the same time.
- The pre-trigger output is asserted for two clock cycles and the trigger output is asserted for one clock cycle (except in 16-bit Periodic Counter mode, where both pre-trigger and trigger outputs are asserted for many cycles depending on the value of TVAL*n*[TMR_VAL][31:16]).
- Timer changes (that are based on external triggers) take effect after four peripheral clocks, after the actual external trigger assertion. This is because of clock synchronization, rise edge detection, and timer update.

35.3.5 Timer channel modes

You can configure each timer channel in LPIT to work in either compare modes or capture modes.

The timer channels operate on an asynchronous clock, which is independent of the register read and write access clock. Clock synchronization between clock domains ensures normal operations.

Table 35-3. Timer channel modes

Mode	Function
Compare	The timers decrement when enabled and generate an output pretrigger and trigger output. The trigger output is one clock cycle delayed of the pre-trigger pulse. You can configure certain control fields to control each timer channel's start, reload, and restart (see Trigger control for timers for more information). You can also configure the timer to always decrement from a programmed start value, on selected trigger inputs, or previous channel timeout (when channels are chained). By chaining timer channels, applications can achieve larger timeout durations.
Capture	You can use the timer to perform measurements by using Timer Value (TVAL0 - TVAL3), as the timer value is captured when a selected trigger input is asserted. The timer can support once-off or multiple measurements, such as frequency measurements.

35.3.6 Trigger control for timers

You can configure various LPIT register fields to control how trigger inputs and the timer operate:

- TCTRL*n*[TRG_SEL] helps you select the input trigger for the channel from all other channels' trigger outputs.
- TCTRL*n*[TRG_SRC] helps you select between the internal and external trigger inputs to the channel.

The selected trigger affects how the timer operates, using the configuration of TCTRL*n*[TSOT], TCTRL*n*[TSOI], and TCTRL*n*[TROT] (see Table 35-4).

Table 35-4. Fields that control timer operations

If	=	Then
Timer stop on interrupt (TCTRL*n*[TSOI])	1	The counter stops after MSR[TIF*n*] assertion. To reload and decrement, it requires: • A trigger (if TCTRL*n*[TSOT] = 1). • A T_EN rising edge (if TCTRL*n*[TSOT] = 0).
	0	The counter does not stop after timeout.
Timer reload on trigger (TCTRL*n*[TROT])	1	The counter is loaded after each trigger.
	0	The counter is loaded after every T_EN rising edge or timeout rising edge (timeout is not used in Capture modes).
Timer start on trigger (TCTRL*n*[TSOT])	1	The counter starts decrementing after a trigger. Subsequent triggers are ignored until the counter times out.
	0	The counter decrements immediately after the next clock edge. When the channel is chained or is in Capture mode, TCTRL*n*[TSOT] has no effect.

In different timer modes, the programmable fields listed in Table 35-5 affect the timer operation differently.

Table 35-5. Timer modes and programmable fields

Mode (TCTRL*n*[MODE])	Fields affecting timer operations
32-bit periodic counter	TCTRL*n*[TSOT], TCTRL*n*[TSOI], and TCTRL*n*[TROT] affect the timer operation as described in Table 35-4.
Dual 16-bit periodic counter	
32-bit trigger accumulator	• Only TCTRL*n*[TSOI] controls the timer function. • TCTRL*n*[TROT] and TCTRL*n*[TSOT] have no effect on timer operations.
32-bit input trigger capture	• Only TCTRL*n*[TSOI] and TCTRL*n*[TROT] control the timer function. • TCTRL*n*[TSOT] has no effect on timer operations.

35.3.7 Channel chaining

You can chain individual timer channels together to achieve a larger value of timeout. Chaining enables these channels to work in a "nested loop" manner, thereby leading to an effective timeout value of $TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$.

To chain the channels together, write 1 to TCTRL*n*[CHAIN] for the corresponding channel. When a channel is chained, that channel's timer decrements after the previous channel's timeout pulse, regardless of the timer mode that TCTRL*n*[MODE] defines.

TCTRL*n*[TSOT] does not have any effect if you chain the channel timer (channel *n*) to the previous channel's timer (channel *n* - 1).

35.3.8 Detailed timing

The following table represents various timing diagrams. The diagrams are not "cycle-accurate," which means, they may not show some of the cycles, but they do show the timer channel behavior across several clock cycles.

Table 35-6. Timing diagrams

Mode (TCTRL*n*[MODE]) setting		Timing diagram	TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	TCTRL*n*[CHAIN]
32-bit periodic counter (Compare mode)	00	Case 1: TCTRL*n*[MODE] = 0 • For a use case requiring repeated interrupts with reload • Trigger outputs have equal periods	0	0	0	0
		Case 2: TCTRL*n*[MODE] = 0 • Useful for One-Shot Trigger mode • Trigger starts again after TCTRL*n*[T_EN] becomes 1	0	0	1	0

Table continues on the next page...

Table 35-6. Timing diagrams (continued)

Mode (TCTRL*n*[MODE]) setting		Timing diagram	TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	TCTRL*n*[CHAIN]
		Case 3: TCTRL*n*[MODE] = 0 • For a use case requiring repeated interrupts with reload • Trigger outputs have unequal periods	0	1	0	0
		Case 4: TCTRL*n*[MODE] = 0 • For a one-shot timer with reload before timeout of Timer mode • Timer starts again after you write 1 to TCTRL*n*[T_EN]	0	1	1	0
		Case 5: TCTRL*n*[MODE] = 0 • Useful for generating periodic interrupts after a predefined event (input trigger) • Output triggers have equal periods	1	0	0	0
		Case 6: TCTRL*n*[MODE] = 0 • Triggers One-Shot Timer mode • Output trigger period depends on the input trigger	1	0	1	0
		Case 7: TCTRL*n*[MODE] = 0 • Repeated interrupts with Reload Timer mode • Outputs triggers have unequal periods	1	1	0	0
		Case 8: TCTRL*n*[MODE] = 0 • Case shown for a nonperiodic input trigger (might not be a use case) • If input trigger is periodic and greater than timer timeout, it is the same as TCTRL*n*[TROT] = 0 • If input trigger is periodic and less than timer timeout, the timer never times out and always reloads on the input trigger (not a valid use case)	1	1	1	0
16-bit dual periodic counter (Compare mode)	01	Case 1: TCTRL*n*[MODE] = 1 • The effect of TCTRL*n*[TSOT], TCTRL*n*[TROT], and TCTRL*n*[TSOI] is the same as that of TCTRL*n*[MODE] = 0 (32-bit Counter Compare mode) • Both halves of the counter are affected in the same way	0	0	0	0
32-bit trigger accumulator mode	10	Case 1: TCTRL*n*[MODE] = 10	X	X	0	0

Table continues on the next page...

Table 35-6. Timing diagrams (continued)

Mode (TCTRL*n*[MODE]) setting		Timing diagram	TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	TCTRL*n*[CHAIN]
		• Useful for a continuous pulse counting mode • Trigger and timeout generated after programmed number of pulses are accumulated				
		Case 2: TCTRL*n*[MODE] = 10 • Useful for One-Shot Pulse Counting mode • Trigger and timeout generated after programmed number of pulses are accumulated	X	X	1	0
32-bit trigger capture mode	11	Case 1: TCTRL*n*[MODE] = 11 • Useful for determining the duration between pulses • Proper clock selection can ensure that the timer does not rollover more than once between two pulses	X	0	0	0
		Case 2: TCTRL*n*[MODE] = 11 • Useful for determining the duration between pulses • Selecting a fast timer clock provides accurate measurements but it can also cause timer rollover between pulses	X	1	0	0
		Case 3: TCTRL*n*[MODE] = 11 • One-Shot Timer Count mode • You can enable it again by writing 1 to TCTRL*n*[T_EN]	X	0	1	0
Timer chaining: effects on timing operations	XX	Timer chaining	X	X	X	1

35.3.8.1 Case 1: TCTRL*n*[MODE] = 0

The following figure represents case 1 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

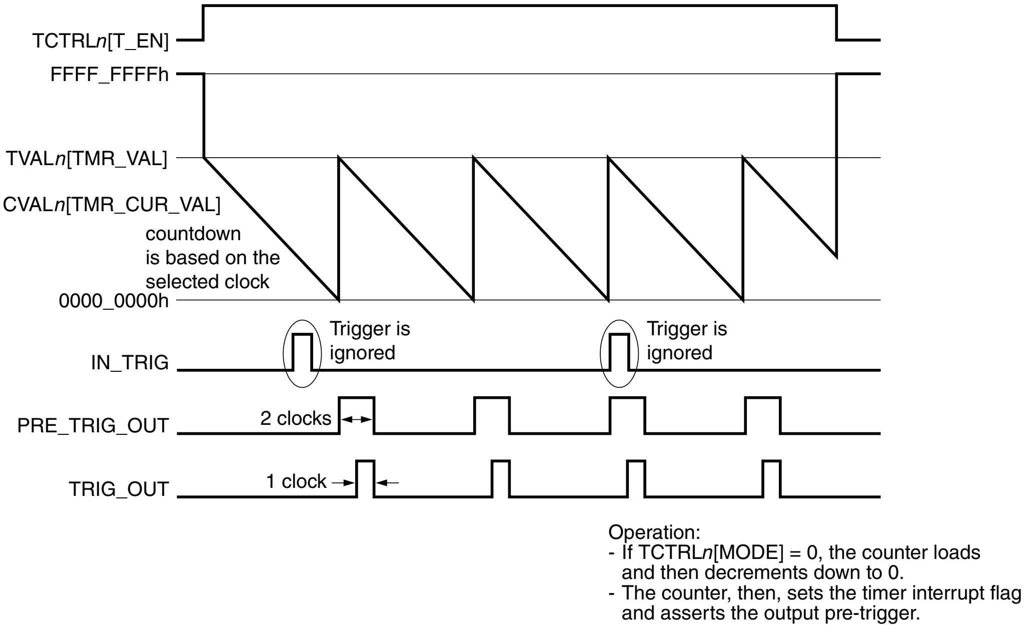

TCTRLn[T_EN]
FFFF_FFFFh
TVALn[TMR_VAL]
CVALn[TMR_CUR_VAL]
countdown is based on the selected clock
0000_0000h
IN_TRIG
Trigger is ignored
Trigger is ignored
PRE_TRIG_OUT
2 clocks
TRIG_OUT
1 clock
Operation:
- If TCTRLn[MODE] = 0, the counter loads and then decrements down to 0.
- The counter, then, sets the timer interrupt flag and asserts the output pre-trigger.

Figure 35-4. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

35.3.8.2 Case 2: TCTRL*n*[MODE] = 0

The following figure represents case 2 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

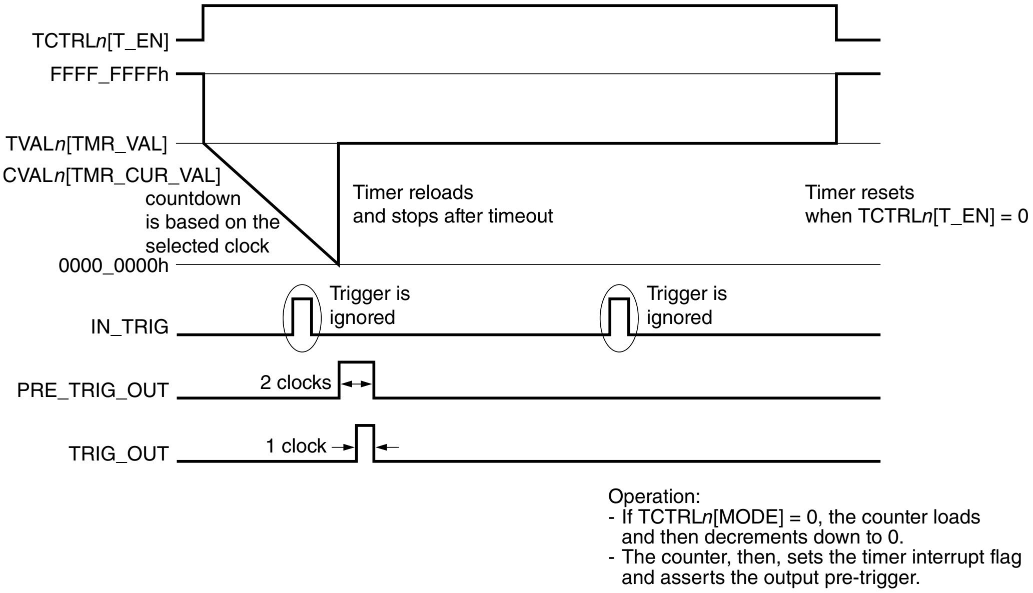


Figure 35-5. Case 2: TSOT = 0, TROT = 0, TSOI = 1, CHAIN = 0

35.3.8.3 Case 3: TCTRL*n*[MODE] = 0

The following figure represents case 3 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

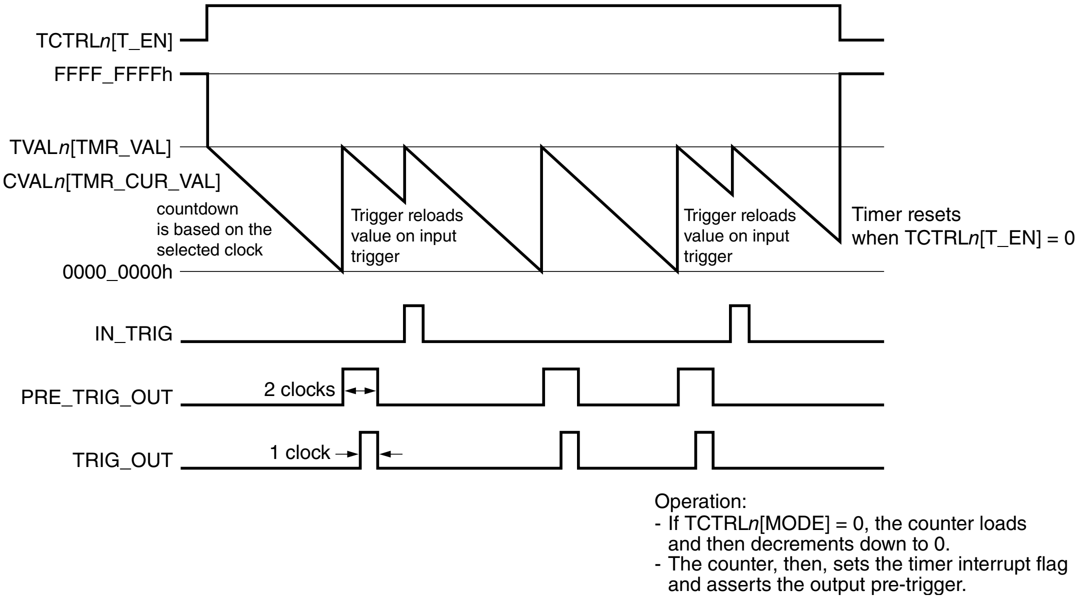
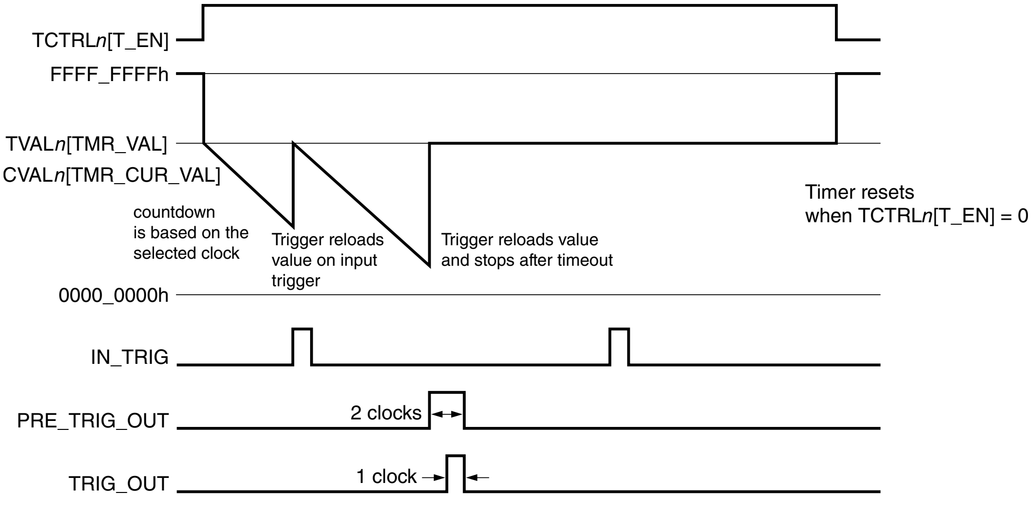


Figure 35-6. Case 3: TSOT = 0, TROT = 1, TSOI = 0, CHAIN = 0

35.3.8.4 Case 4: TCTRL*n*[MODE] = 0

The following figure represents case 4 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0



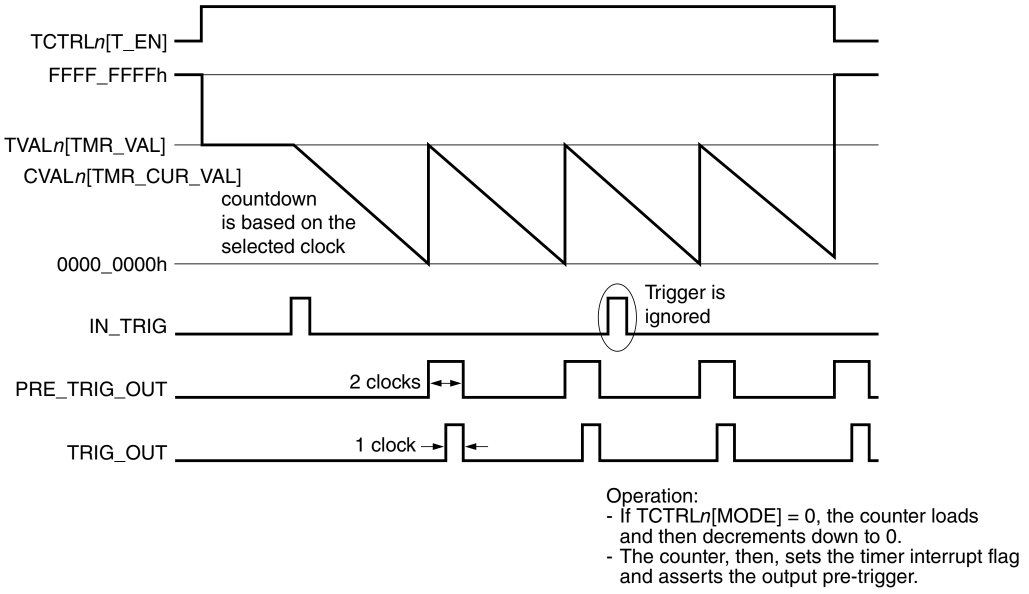
Operation:
- If TCTRL*n*[MODE] = 0, the counter loads and then decrements down to 0.
- The counter, then, sets the timer interrupt flag and asserts the output pre-trigger.

Figure 35-7. Case 4: TSOT = 0, TROT = 1, TSOI = 1, CHAIN = 0

35.3.8.5 Case 5: TCTRL*n*[MODE] = 0

The following figure represents case 5 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0



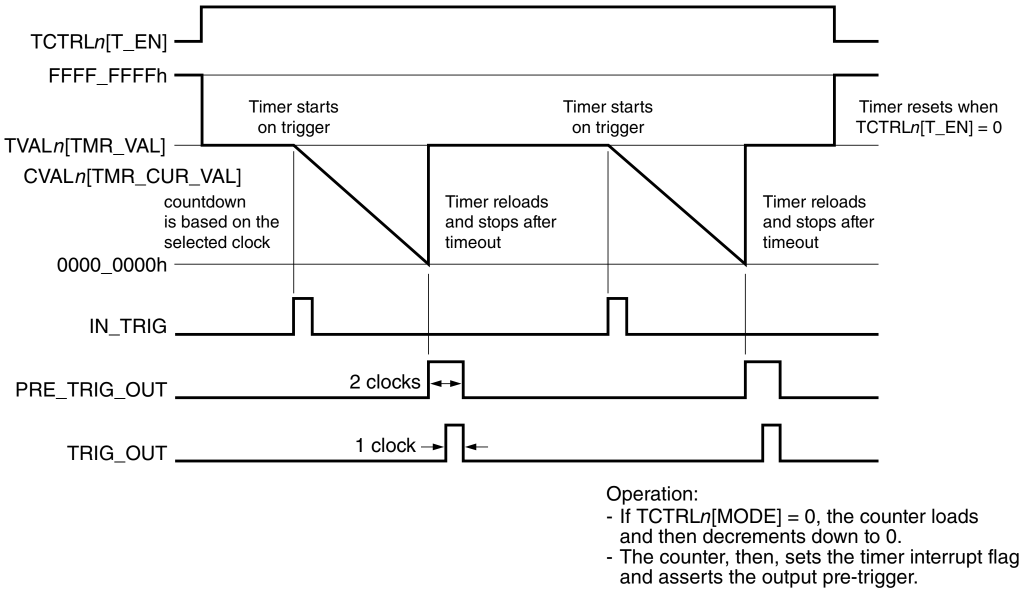
Operation:
- If TCTRLn[MODE] = 0, the counter loads and then decrements down to 0.
- The counter, then, sets the timer interrupt flag and asserts the output pre-trigger.

Figure 35-8. Case 5: TSOT = 1, TROT = 0, TSOI = 0, CHAIN = 0

35.3.8.6 Case 6: TCTRL*n*[MODE] = 0

The following figure represents case 6 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0



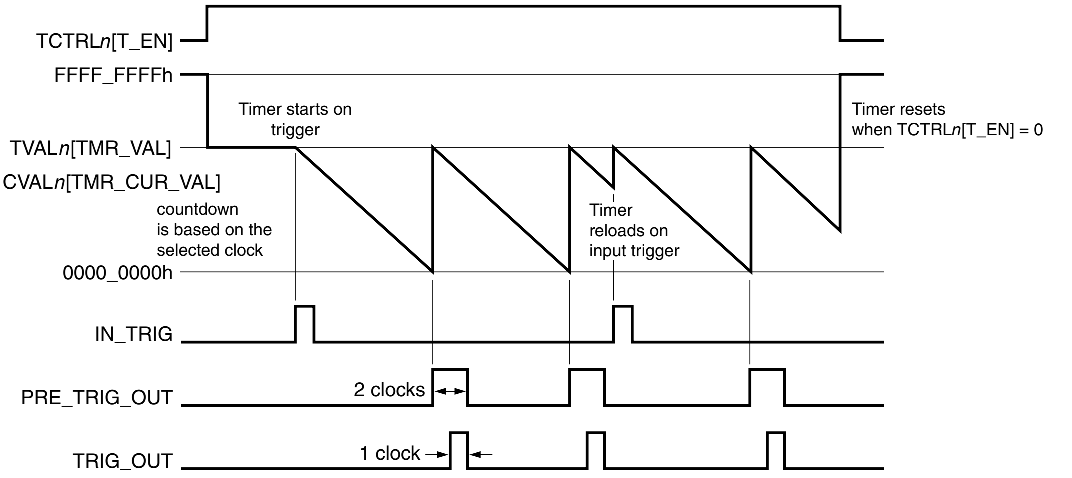
Operation:
- If TCTRLn[MODE] = 0, the counter loads and then decrements down to 0.
- The counter, then, sets the timer interrupt flag and asserts the output pre-trigger.

Figure 35-9. Case 6: TSOT = 1, TROT = 0, TSOI = 1, CHAIN = 0

35.3.8.7 Case 7: TCTRL*n*[MODE] = 0

The following figure represents case 7 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0



Operation:
- If TCTRL*n*[MODE] = 0, the counter loads and then decrements down to 0.
- The counter, then, sets the timer interrupt flag and asserts the output pre-trigger.

Figure 35-10. Case 7: TSOT = 1, TROT = 1, TSOI = 0, CHAIN = 0

35.3.8.8 Case 8: TCTRL*n*[MODE] = 0

The following figure represents case 8 in which TCTRL*n*[MODE] = 0 (32-bit periodic counter).

LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 1
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

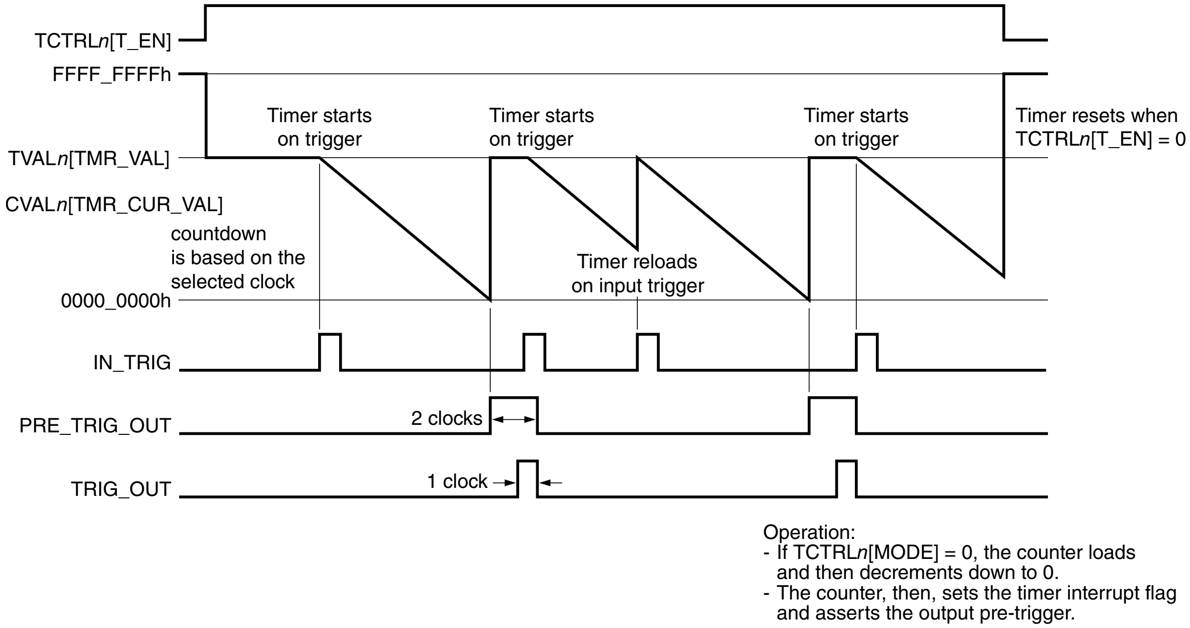


Figure 35-11. Case 8: TSOT = 1, TROT = 1, TSOI = 1, CHAIN = 0

35.3.8.9 Case 1: TCTRL*n*[MODE] = 1

The following figure represents case 1 in which TCTRL*n*[MODE] = 1 (16-bit dual periodic counter). LPIT works in Compare mode and is configured as follows:

- TCTRL*n*[TSOT] = 0
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0

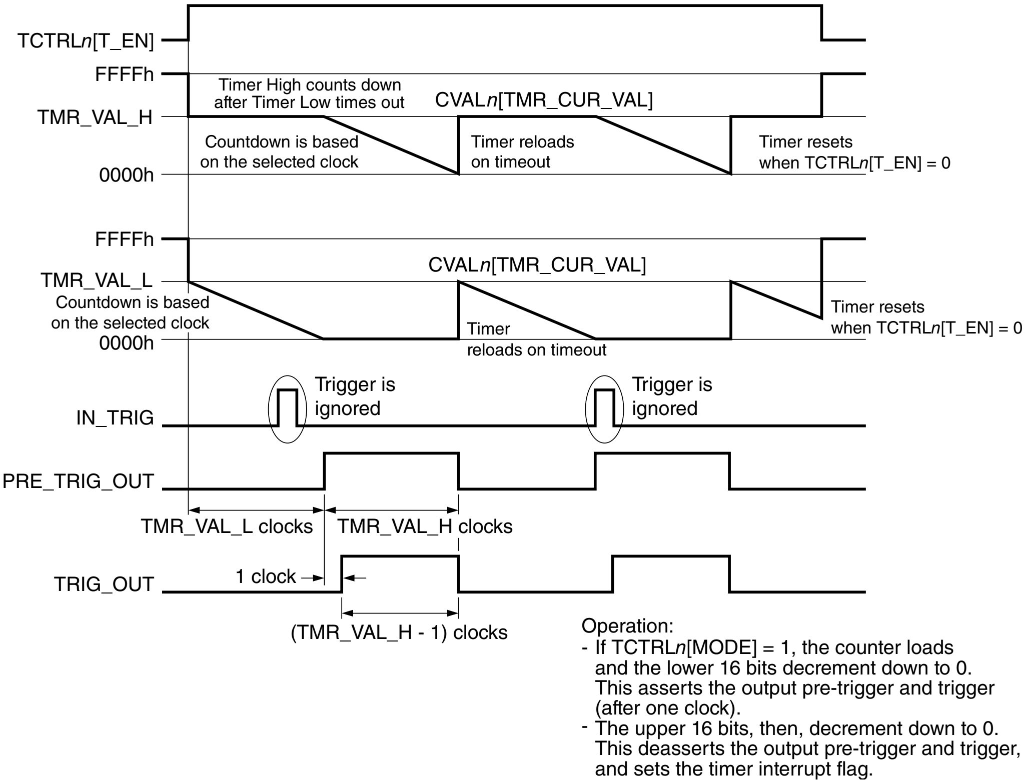

TCTRLn[T_EN]
FFFFh
TMR_VAL_H
0000h
Timer High counts down after Timer Low times out
CVALn[TMR_CUR_VAL]
Countdown is based on the selected clock
Timer reloads on timeout
Timer resets when TCTRLn[T_EN] = 0
FFFFh
TMR_VAL_L
Countdown is based on the selected clock
0000h
CVALn[TMR_CUR_VAL]
Timer reloads on timeout
Timer resets when TCTRLn[T_EN] = 0
IN_TRIG
Trigger is ignored
Trigger is ignored
PRE_TRIG_OUT
TMR_VAL_L clocks
TMR_VAL_H clocks
TRIG_OUT
1 clock
(TMR_VAL_H - 1) clocks
Operation:
- If TCTRLn[MODE] = 1, the counter loads and the lower 16 bits decrement down to 0. This asserts the output pre-trigger and trigger (after one clock).
- The upper 16 bits, then, decrement down to 0. This deasserts the output pre-trigger and trigger, and sets the timer interrupt flag.

Figure 35-12. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

If TCTRL*n*[MODE] = 1:

- The effect of timing control fields is similar to the effect of timer control fields when TCTRL*n*[MODE] = 0. See the individual timing diagrams for cases with TCTRL*n*[MODE] = 0 for more information.
- The timer interrupt (timeout) asserts when {TMR_H,TMR_L} = 0000_0000h.

See the following table for the behavior of timer control fields when TCTRL*n*[MODE] = 1.

Table 35-7. Timer control fields when TCTRL*n*[MODE] = 1

TCTRLn[TSOT]	TCTRLn[TROT]	TCTRLn[TSOI]	Function	Effect on timer
0	0	0	• For repeated interrupts with reload • Trigger outputs have equal periods	Similar to Case 1: TCTRL*n*[MODE] = 1.
0	0	1	One-shot mode	• Similar to Case 2: TCTRL*n*[MODE] = 0. • Both timers stop after first count down and then time out. • Timers do not count again until TCTRL*n*[T_EN] becomes 1.
0	1	0	• For repeated interrupts with reload • Trigger outputs have unequal periods	• Similar to Case 3: TCTRL*n*[MODE] = 0 .

Table continues on the next page...

Table 35-7. Timer control fields when TCTRL*n*[MODE] = 1 (continued)

TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	Function	Effect on timer
				• Both timers reload the value of TVAL*n*[TMR_VAL] after the trigger rising edge. • Output triggers clear after reload, if asserted.
0	1	1	Reloadable one-shot mode	• Similar to Case 4: TCTRL*n*[MODE] = 0. • If a trigger occurs before timeout, then both timers reload and count down (as shown); the timers stop after timeout. • A trigger assertion after timeout reloads the value of TVAL*n*[TMR_VAL] into the timers. • The timers do not count again until TCTRL*n*[T_EN] becomes 1 again.
1	0	0	• For generating periodic interrupts after a predefined event (input trigger) • Output triggers have equal periods	• Similar to Case 5: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] rises, the timers do not start until after the first trigger's rising edge. • Subsequent triggers have no effect.
1	0	1	• Triggered one-shot timer mode • Output trigger period depends on the input trigger	• Similar to Case 6: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] becomes 1, the timers do not start until after the first trigger's rising edge. • The timer stops counting after a timeout assertion. • The timer does not start counting again until a new trigger's rising edge is detected.
1	1	0	• For repeated interrupts with reload timer mode • Output triggers have unequal periods	• Similar to Case 7: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] becomes 1, the timers do not start until the first trigger's rising edge. • Subsequent triggers cause the timer to reload the value of TVAL*n*[TMR_VAL] into both counters. • The output triggers clear after a reload, if asserted.
1	1	1	• For a nonperiodic input trigger • If input trigger is periodic and greater than timer timeout, then it is the same as TCTRL*n*[TROT] = 0 (which is triggered one-shot timer mode; output trigger period depends on the input trigger)	• Similar to Case 8: TCTRL*n*[MODE] = 0. • After TCTRL*n*[T_EN] becomes 1, the timers do not start until the first trigger's rising edge.

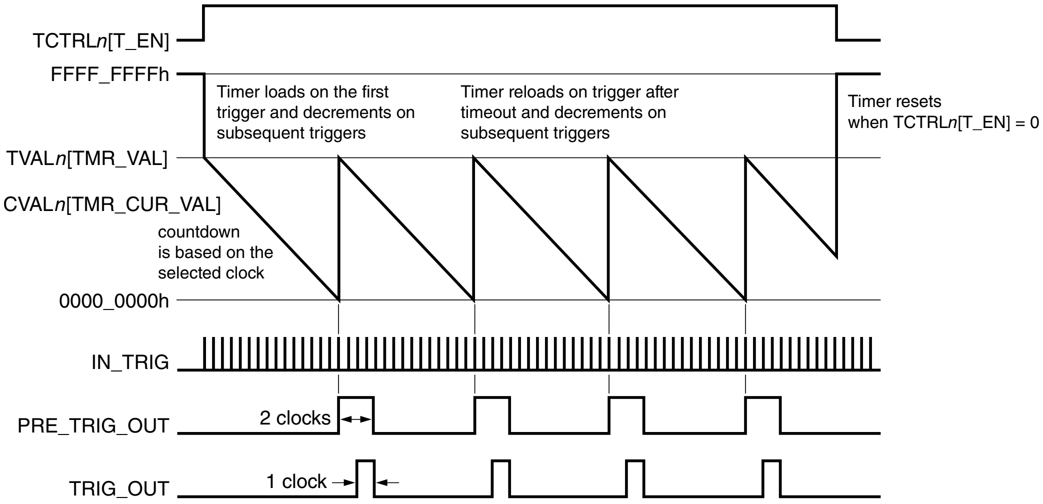
Table 35-7. Timer control fields when TCTRL*n*[MODE] = 1

TCTRL*n*[TSOT]	TCTRL*n*[TROT]	TCTRL*n*[TSOI]	Function	Effect on timer
				• The timers stop counting after a timeout assertion. • A trigger's rising edge causes the timers to reload and then count down.

35.3.8.10 Case 1: TCTRL*n*[MODE] = 10

The following figure represents case 1 in which TCTRL*n*[MODE] = 10. LPIT works in 32-bit trigger accumulator mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = X
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0


TCTRLn[T_EN]
FFFF_FFFFh
Timer loads on the first trigger and decrements on subsequent triggers
Timer reloads on trigger after timeout and decrements on subsequent triggers
Timer resets when TCTRLn[T_EN] = 0
TVALn[TMR_VAL]
CVALn[TMR_CUR_VAL]
countdown is based on the selected clock
0000_0000h
IN_TRIG
PRE_TRIG_OUT
2 clocks
TRIG_OUT
1 clock

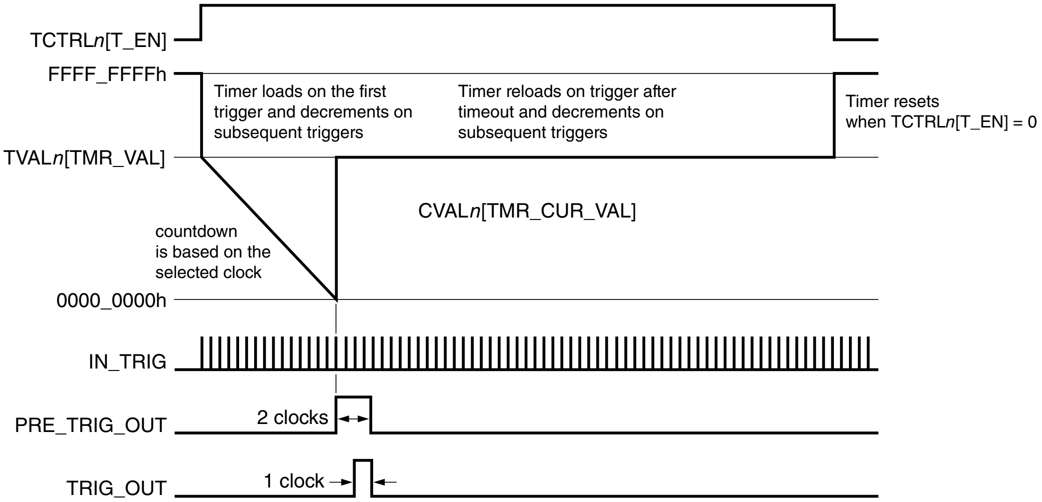
Operation:
- If TCTRL*n*[MODE] = 10, the counter loads on the first trigger rising edge, and then decrements down to 0 on each trigger's rising edge.
- The counter, then, sets the timer interrupt flag, and asserts the output pre-trigger and trigger.

Figure 35-13. Case 1: TSOT = X, TROT = X, TSOI = 0, CHAIN = 0

35.3.8.11 Case 2: TCTRL*n*[MODE] = 10

The following figure represents case 2 in which TCTRL*n*[MODE] = 10. LPIT works in 32-bit trigger accumulator mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = X
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0


TCTRL*n*[T_EN]
FFFF_FFFFh
TVAL*n*[TMR_VAL]
0000_0000h
IN_TRIG
PRE_TRIG_OUT
TRIG_OUT
Timer loads on the first trigger and decrements on subsequent triggers
Timer reloads on trigger after timeout and decrements on subsequent triggers
Timer resets when TCTRL*n*[T_EN] = 0
CVAL*n*[TMR_CUR_VAL]
countdown is based on the selected clock
2 clocks
1 clock

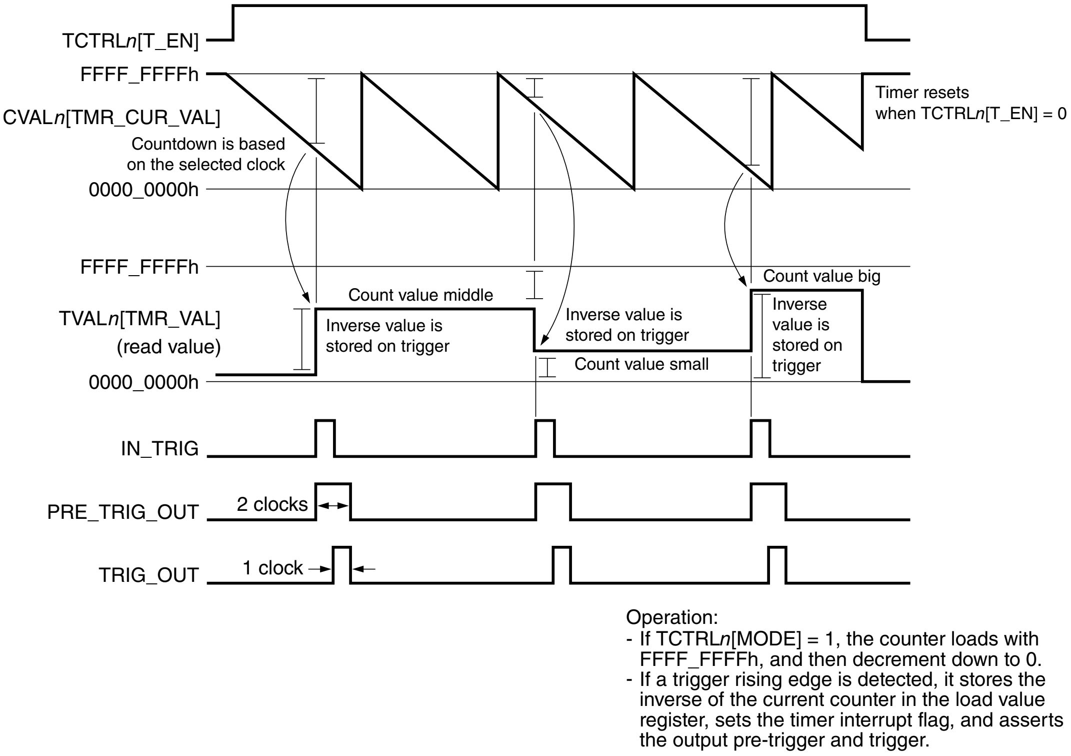
Operation:
- If TCTRL*n*[MODE] = 10, the counter loads on the first trigger rising edge, and then decrements down to 0 on each trigger's rising edge.
- This, then, sets the timer interrupt flag, and asserts the output pre-trigger and trigger.

Figure 35-14. Case 2: TSOT = X, TROT = X, TSOI = 1, CHAIN = 0

35.3.8.12 Case 1: TCTRL*n*[MODE] = 11

The following figure represents case 1 in which TCTRL*n*[MODE] = 11. LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0


TCTRLn[T_EN]
FFFF_FFFFh
CVALn[TMR_CUR_VAL]
Countdown is based on the selected clock
0000_0000h
Timer resets when TCTRLn[T_EN] = 0
FFFF_FFFFh
TVALn[TMR_VAL] (read value)
0000_0000h
Count value middle
Inverse value is stored on trigger
Inverse value is stored on trigger
Count value small
Count value big
Inverse value is stored on trigger
IN_TRIG
PRE_TRIG_OUT
2 clocks
TRIG_OUT
1 clock

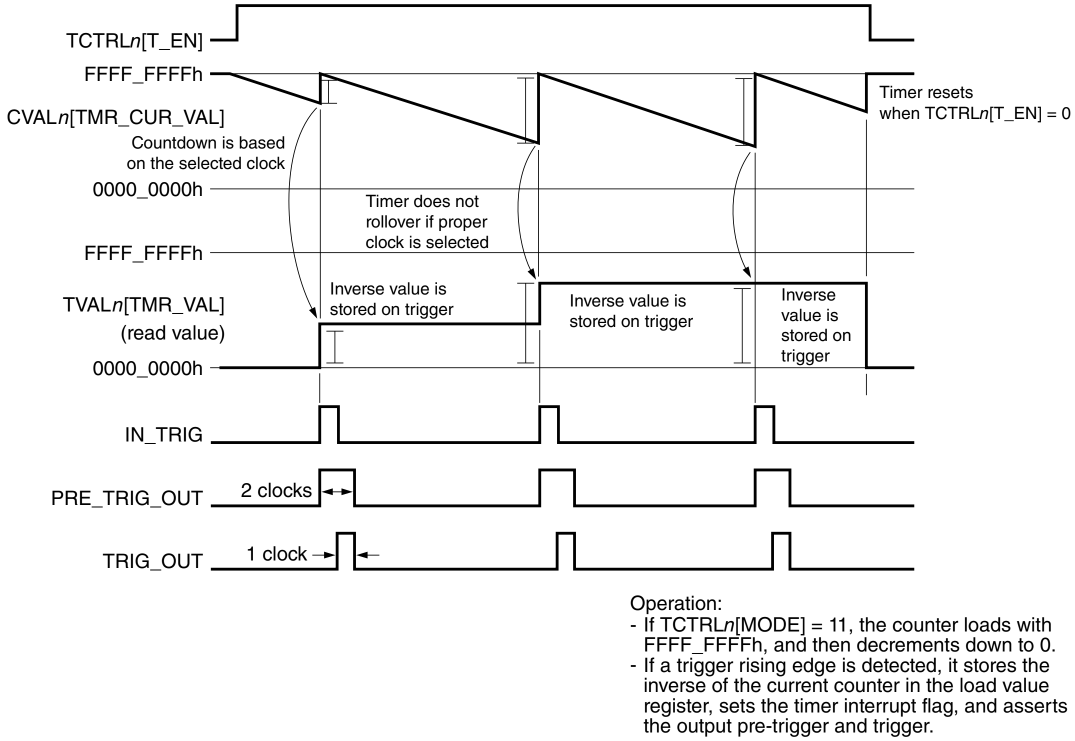
Operation:
- If TCTRLn[MODE] = 1, the counter loads with FFFF_FFFFh, and then decrement down to 0.
- If a trigger rising edge is detected, it stores the inverse of the current counter in the load value register, sets the timer interrupt flag, and asserts the output pre-trigger and trigger.

Figure 35-15. Case 1: TSOT = X, TROT = 0, TSOI = 0, CHAIN = 0

35.3.8.13 Case 2: TCTRL*n*[MODE] = 11

The following figure represents case 2 in which TCTRL*n*[MODE] = 11. LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 0
- TCTRL*n*[CHAIN] = 0


TCTRLn[T_EN]
FFFF_FFFFh
CVALn[TMR_CUR_VAL]
Countdown is based on the selected clock
0000_0000h
Timer does not rollover if proper clock is selected
FFFF_FFFFh
TVALn[TMR_VAL] (read value)
0000_0000h
Inverse value is stored on trigger
Inverse value is stored on trigger
Inverse value is stored on trigger
Timer resets when TCTRLn[T_EN] = 0
IN_TRIG
PRE_TRIG_OUT
2 clocks
TRIG_OUT
1 clock

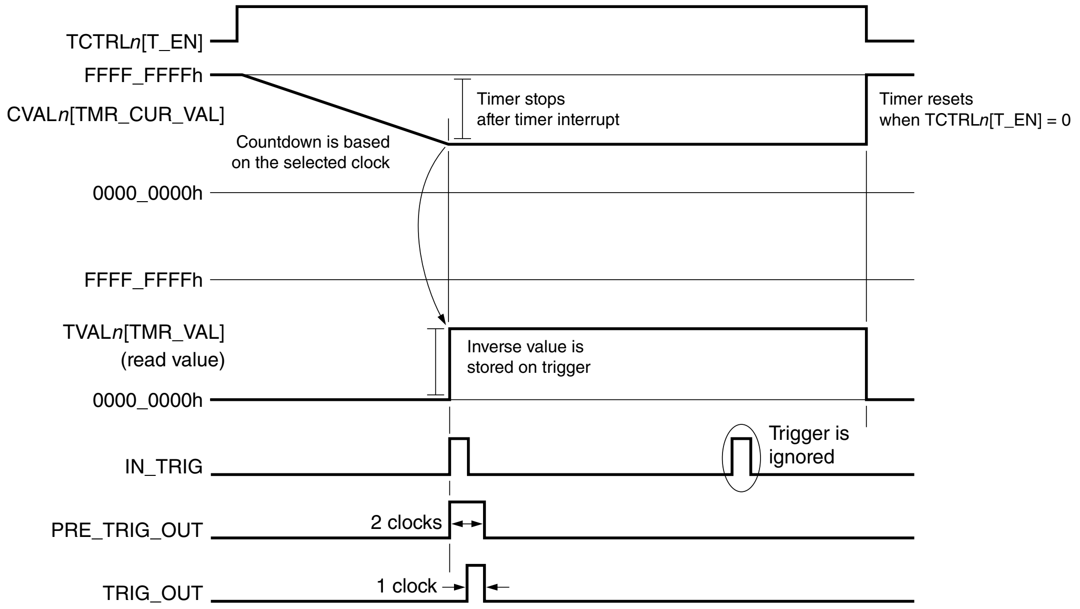
Operation:
- If TCTRLn[MODE] = 11, the counter loads with FFFF_FFFFh, and then decrements down to 0.
- If a trigger rising edge is detected, it stores the inverse of the current counter in the load value register, sets the timer interrupt flag, and asserts the output pre-trigger and trigger.

Figure 35-16. Case 2: TSOT = X, TROT = 1, TSOI = 0, CHAIN = 0

35.3.8.14 Case 3: TCTRL*n*[MODE] = 11

The following figure represents case 3 in which TCTRL*n*[MODE] = 11. LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 0
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0



Operation:
- If TCTRLn[MODE] = 11, the counter loads with FFFF_FFFFh, and then decrements down to 0.
- If a trigger rising edge is detected, it stores the inverse of the current counter in the load value register, sets the timer interrupt flag, and asserts the output pre-trigger and trigger.

Figure 35-17. Case 3: TSOT = X, TROT = 0, TSOI = 1, CHAIN = 0

35.3.8.15 Case 4: TCTRL*n*[MODE] = 11

Case 4, in which TCTRL*n*[MODE] = 11, is the same as case 3, except that the timer reloads to FFFF_FFFFh and then stops. The timer does not start until TCTRL*n*[T_EN] becomes 1 again. In this case, LPIT works in 32-bit trigger capture mode and is configured as follows:

- TCTRL*n*[TSOT] = X
- TCTRL*n*[TROT] = 1
- TCTRL*n*[TSOI] = 1
- TCTRL*n*[CHAIN] = 0

35.3.9 Timer chaining

Effect of chaining	- Chaining causes timer "*n*" to decrement on every timeout pulse (trigger output pulse) from timer "*n* - 1", regardless of the mode you configure in timer "*n*". - Timers "*n*" and "*n* - 1" effectively form a larger-width timer (64 bits). - You can chain more than two (or all) timer channels. - NXP recommends you to configure the same trigger source and timer controls for chained channels.

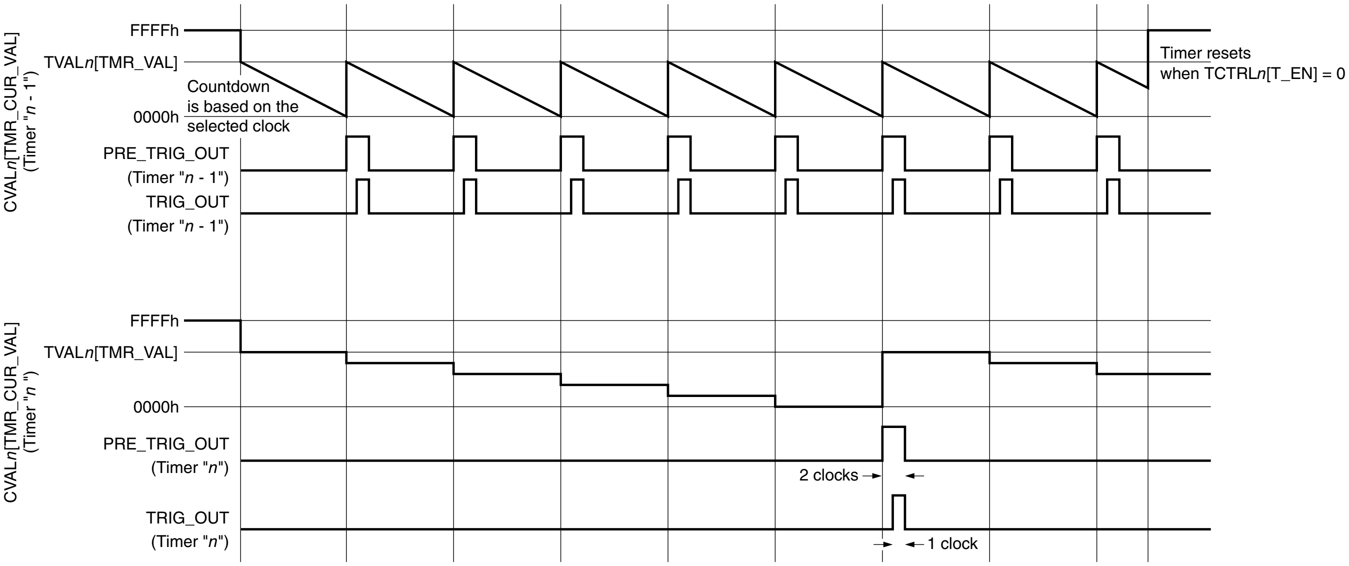

CVALn[TMR_CUR_VAL] (Timer "n - 1")
FFFFh
TVALn[TMR_VAL]
Countdown is based on the selected clock
0000h
Timer resets when TCTRLn[T_EN] = 0
PRE_TRIG_OUT (Timer "n - 1")
TRIG_OUT (Timer "n - 1")
CVALn[TMR_CUR_VAL] (Timer "n")
FFFFh
TVALn[TMR_VAL]
0000h
PRE_TRIG_OUT (Timer "n")
2 clocks
TRIG_OUT (Timer "n")
1 clock

Figure 35-18. Chaining effects

35.4 Initialization

Table 35-8. Initializing LPIT

Step	Action	How or why to perform the step
1	Enable the peripheral clock	By writing 1 to MCR[M_CEN]. **NOTE:** • Accessing certain registers (Module Status (MSR), Set Timer Enable (SETTEN), Clear Timer Enable (CLRTEN), Timer Value (TVAL0 - TVAL3), Current Timer Value (CVAL0 - CVAL3), and Timer Control (TCTRL0 - TCTRL3)) while MCR[M_CEN] = 0 leads to the assertion of a transfer error for that bus access. However, writing to CVAL*n* and reserved registers generates a transfer error. • There might be additional clock gating fields in the chip that gate the peripheral clock to this module. When enabling the clock to this module, you must configure those additional clock gating fields (in addition to configuring MCR[M_CEN]).
2	Wait for four peripheral clock cycles	To allow time for clock synchronization and reset deassertion.
3	Configure timer control fields	For each timer channel that is to be enabled: • Timer mode of operation fields, TCTRL*n*[MODE] • Trigger source selection fields, TCTRL*n*[TRG_SEL] and TCTRL*n*[TRG_SRC] • Trigger control fields, TCTRL*n*[TROT], TCTRL*n*[TSOT], and TCTRL*n*[TSOI]

Table continues on the next page...

Table 35-8. Initializing LPIT (continued)

Step	Action	How or why to perform the step
		NOTE: You must not update timer control fields when the timer is disabled. You can disable a timer by using any one of the following methods: • Write 1 to the specific timer's CLRTEN[CLR_T_EN_*n*] field. • Write 0 to TCTRL*n*[T_EN] for that channel.
4	Configure the channels that are to be chained	By writing 1 to TCTRL*n*[CHAIN] in the corresponding channel's Timer Control (TCTRL0 - TCTRL3).
5	Set the timer timeout value	By programming an appropriate value in TVAL*n*[TMR_VAL] for the channels that you configure in Compare mode.
6	Configure MIER[TIE*n*]	For those channels that are required to generate interrupts after timer timeouts.
7	Configure the low-power modes of the module	By writing 1 to MCR[DBG_EN] and MCR[DOZE_EN]. This is common to all timer channels.
8	Enable the channel timers	By writing 1 to the corresponding TCTRL*n*[T_EN].

NOTE

When you enable a timer channel in Compare mode, the first decrement takes an additional one or two clock cycles because of synchronization logic. This results in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster counter clock minimizes this impact.

Additionally,

- For channels that you configure in Capture mode, you can read the timer value from Timer Value (TVAL0 - TVAL3) when a channel timeout occurs.
- MSR[TIF*n*] are asserted after timer timeout. To clear these timer interrupt flags, write 1 to them.

35.5 Memory map and registers

35.5.1 LPIT register descriptions

The LPIT memory map comprises 32-bit aligned registers, which you can access via 8-bit, 16-bit, or 32-bit accesses. Read and write accesses to reserved locations generate a transfer error, and the read bus shows all 0s.

NOTE

- The memory map and complete module are in big-endian (BE) format.
- LPIT does not check whether programmed values in these registers are correct—you must write the correct values.

35.5.1.1 LPIT memory map

LPIT0 base address: 4003_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0000h
4h	Parameter (PARAM)	32	R	0000_0404h
8h	Module Control (MCR)	32	RW	0000_0000h
Ch	Module Status (MSR)	32	RW	0000_0000h
10h	Module Interrupt Enable (MIER)	32	RW	0000_0000h
14h	Set Timer Enable (SETTEN)	32	RW	0000_0000h
18h	Clear Timer Enable (CLRTEN)	32	RW	0000_0000h
20h	Timer Value (TVAL0)	32	RW	0000_0000h
24h	Current Timer Value (CVAL0)	32	R	FFFF_FFFFh
28h	Timer Control (TCTRL0)	32	RW	0000_0000h
30h	Timer Value (TVAL1)	32	RW	0000_0000h
34h	Current Timer Value (CVAL1)	32	R	FFFF_FFFFh
38h	Timer Control (TCTRL1)	32	RW	0000_0000h
40h	Timer Value (TVAL2)	32	RW	0000_0000h
44h	Current Timer Value (CVAL2)	32	R	FFFF_FFFFh
48h	Timer Control (TCTRL2)	32	RW	0000_0000h
50h	Timer Value (TVAL3)	32	RW	0000_0000h
54h	Current Timer Value (CVAL3)	32	R	FFFF_FFFFh
58h	Timer Control (TCTRL3)	32	RW	0000_0000h

35.5.1.2 Version ID (VERID)

35.5.1.2.1 Offset

Register	Offset
VERID	0h

35.5.1.2.2 Function

Contains design version specification numbers.

35.5.1.2.3 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	0

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	0

35.5.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module design specification.
23-16 MINOR	Minor Version Number Indicates the minor version number for the module design specification.
15-0 FEATURE	Feature Number Indicates the feature set number.

35.5.1.3 Parameter (PARAM)

35.5.1.3.1 Offset

Register	Offset
PARAM	4h

35.5.1.3.2 Function

Provides parameter settings that are used when incorporating this module into the chip.

35.5.1.3.3 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	EXT_TRIG								CHANNEL							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

35.5.1.3.4 Fields

Field	Function
31-16 —	Reserved
15-8 EXT_TRIG	Number of External Trigger Inputs Specifies the number of external triggers implemented in this chip.
7-0 CHANNEL	Number of Timer Channels Specifies the number of timer channels implemented in this chip.

35.5.1.4 Module Control (MCR)

35.5.1.4.1 Offset

Register	Offset
MCR	8h

35.5.1.4.2 Function

Contains software reset, clock enable, and mode enable fields.

35.5.1.4.3 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												DBG_EN	DOZE_EN	SW_RST	M_CEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

35.5.1.4.4 Fields

Field	Function
31-4 —	Reserved
3 DBG_EN	Debug Mode Enable Stops the timer channels when the chip enters Debug mode. 0b - Stops timer channels 1b - Allows timer channels to continue running
2 DOZE_EN	DOZE Mode Enable Stops the timer channels when the chip enters Doze mode. 0b - Stops timer channels 1b - Allows timer channels to continue running
1 SW_RST	Software Reset Resets all timer channels and registers, except Module Status (MSR). This field remains 1 until software clears it. Before clearing this field, software must wait for four peripheral clocks (for clock synchronization and reset propagation). 0b - Does not reset 1b - Resets
0 M_CEN	Module Clock Enable Enables the peripheral clock to LPIT module timers. This field must become 1 when accessing the following registers: • Module Status (MSR) • Set Timer Enable (SETTEN) • Clear Timer Enable (CLRTEN) • Timer Value (TVAL0 - TVAL3) • Current Timer Value (CVAL0 - CVAL3) • Timer Control (TCTRL0 - TCTRL3) The following considerations apply when using this field:

Field	Function
	• You must enable both the bus and peripheral clocks to allow clock synchronization and update of the aforementioned registers. Accessing these registers when MCR[M_CEN] = 0 asserts a transfer error for that bus cycle. • Writing to Current Timer Value (CVAL0 - CVAL3) and reserved registers always generates a transfer error. **NOTE:** There may be additional clock gating fields available in this chip that gate the peripheral clock to LPIT. You must configure those additional clock gating fields appropriately to enable the peripheral clock to LPIT. 0b - Disable 1b - Enable

35.5.1.5 Module Status (MSR)

35.5.1.5.1 Offset

Register	Offset
MSR	Ch

35.5.1.5.2 Function

Contains channel timer interrupt flags.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

35.5.1.5.3 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												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

35.5.1.5.4 Fields

Field	Function
31-4 —	Reserved
3 TIF3	Channel 3 Timer Interrupt Flag Specifies whether the channel 3 timer has timed out. In compare modes, at the end of the timer period, this flag becomes 1. In capture modes, when the trigger asserts, this flag becomes 1. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag
2 TIF2	Channel 2 Timer Interrupt Flag Specifies whether the channel 2 timer has timed out. In compare modes, at the end of the timer period, this flag becomes 1. In capture modes, when the trigger asserts, this flag becomes 1. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag
1 TIF1	Channel 1 Timer Interrupt Flag Specifies whether the channel 1 timer has timed out. In compare modes, this flag becomes 1 at the end of the timer period. In capture modes, this flag becomes 1 when the trigger asserts. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag
0 TIF0	Channel 0 Timer Interrupt Flag Specifies whether the channel 0 timer has timed out.

Field	Function
	In compare modes, this flag becomes 1 at the end of the timer period. In capture modes, this flag becomes 1 when the trigger asserts. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not timed out 1b - Timed out When writing 0b - No effect 1b - Clear the flag

35.5.1.6 Module Interrupt Enable (MIER)

35.5.1.6.1 Offset

Register	Offset
MIER	10h

35.5.1.6.2 Function

Contains channel timer interrupt enable fields.

35.5.1.6.3 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												TIE3	TIE2	TIE1	TIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

35.5.1.6.4 Fields

Field	Function
31-4 —	Reserved
3 TIE3	Channel 3 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF3] = 1. 0b - Disable 1b - Enable
2 TIE2	Channel 2 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF2] = 1. 0b - Disable 1b - Enable
1 TIE1	Channel 1 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF1] = 1. 0b - Disable 1b - Enable
0 TIE0	Channel 0 Timer Interrupt Enable Enables interrupt generation when: • This field = 1. • The corresponding timer interrupt flag, MSR[TIF0] = 1. 0b - Disable 1b - Enable

35.5.1.7 Set Timer Enable (SETTEN)

35.5.1.7.1 Offset

Register	Offset
SETTEN	14h

35.5.1.7.2 Function

Allows the simultaneous enabling of timer channels.

You can enable timer channels by using either of the following ways:

- Writing 1 to TCTRL*n*[T_EN] in the respective TCTRL*n* register.
- Writing 1 to the corresponding SETTEN[SET_T_EN_*n*] field.

To disable timer channels simultaneously, use Clear Timer Enable (CLRTEN). Writing 0 to the fields of this register has no effect.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

35.5.1.7.3 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												SET_T_EN_3	SET_T_EN_2	SET_T_EN_1	SET_T_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

35.5.1.7.4 Fields

Field	Function
31-4 —	Reserved
3 SET_T_EN_3	Set Timer 3 Enable Works with TCTRL3[T_EN] and enables timer channel 3. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL3[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_3]

Table continues on the next page...

Field	Function
	0b - No effect 1b - Enables timer channel 3
2 SET_T_EN_2	Set Timer 2 Enable Works with TCTRL2[T_EN] and enables timer channel 2. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL2[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_2] 0b - No Effect 1b - Enables timer channel 2
1 SET_T_EN_1	Set Timer 1 Enable Works with TCTRL1[T_EN] and enables timer channel 1. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL1[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_1] 0b - No Effect 1b - Enables timer channel 1
0 SET_T_EN_0	Set Timer 0 Enable Works with TCTRL0[T_EN] and enables timer channel 0. Writing 0 to this field does not disable the counter; rather it has no effect. This field becomes 0 if any of the following conditions is true: • TCTRL0[T_EN] = 0 • You write 1 to CLRTEN[CLR_T_EN_0] 0b - No effect 1b - Enables timer channel 0

35.5.1.8 Clear Timer Enable (CLRTEN)

35.5.1.8.1 Offset

Register	Offset
CLRTEN	18h

35.5.1.8.2 Function

Allows the simultaneous disabling of timer channels.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

35.5.1.8.3 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												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

35.5.1.8.4 Fields

Field	Function
31-4 —	Reserved
3 CLR_T_EN_3	Clear Timer 3 Enable Works with TCTRL3[T_EN] and disables timer channel 3. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 3. It also turns TCTRL3[T_EN] = 0 for timer channel 3. 0b - No action 1b - Turns TCTRL3[T_EN] = 0 for timer channel 3
2 CLR_T_EN_2	Clear Timer 2 Enable Works with TCTRL2[T_EN] and disables timer channel 2. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 2. It also turns TCTRL2[T_EN] = 0 for timer channel 2. 0b - No action 1b - Turns TCTRL2[T_EN] = 0 for timer channel 2
1 CLR_T_EN_1	Clear Timer 1 Enable Works with TCTRL1[T_EN] and disables timer channel 1. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 1. It also turns TCTRL1[T_EN] = 0 for timer channel 1.

Table continues on the next page...

Field	Function
	0b - No action 1b - Turns TCTRL1[T_EN] = 0 for timer channel 1
0 CLR_T_EN_0	Clear Timer 0 Enable Works with TCTRL0[T_EN] and disables timer channel 0. Writing 1 to this self-clearing field does not enable the counter. It disables timer channel 0. It also turns TCTRL0[T_EN] = 0 for timer channel 0. 0b - No action 1b - Turns TCTRL0[T_EN] = 0 for timer channel 0

35.5.1.9 Timer Value (TVAL0 - TVAL3)

35.5.1.9.1 Offset

Register	Offset
TVAL0	20h
TVAL1	30h
TVAL2	40h
TVAL3	50h

35.5.1.9.2 Function

Contains timer values:

- In compare modes, this register selects the timeout period for the timer channels.
- In capture modes, this register is loaded with the value of the counter when the trigger asserts.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

35.5.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TMR_VAL															
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	TMR_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

35.5.1.9.4 Fields

Field	Function
31-0 TMR_VAL	Timer Value Specifies the timer value and whether it is valid. • In Compare mode, this field specifies the timer channel start value: • The timer counts down for (TVAL*n* + 1) cycles until the timer reaches 0, then the timer generates an interrupt and loads the TVAL*n* value again. • Writing a new value to TVAL*n* does not restart the timer channel; instead, the new value is loaded "after the timer expires." • To abort the current timer cycle and start a timer period with a new value, you must disable the timer channel and then enable it again. • In Capture mode, whenever the trigger asserts, this register stores the inverse of the counter value. 0000_0000_0000_0000_0000_0000_0000_0000b, 0000_0000_0000_0000_0000_0000_0000_0001b - Invalid load value in Compare mode 0000_0000_0000_0000_0000_0000_0000_0010b-1111_1111_1111_1111_1111_1111_1111_1111b - In Compare mode: the value to be loaded; in Capture mode, the value of the timer

35.5.1.10 Current Timer Value (CVAL0 - CVAL3)

35.5.1.10.1 Offset

Register	Offset
CVAL0	24h
CVAL1	34h
CVAL2	44h
CVAL3	54h

35.5.1.10.2 Function

Indicates the current timer counter value.

NOTE

While the timer is enabled and incrementing, reading the CVAL*n* register may not return the correct value.

35.5.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMR_CUR_VAL															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	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

35.5.1.10.4 Fields

Field	Function
31-0 TMR_CUR_VAL	Current Timer Value Represents the current timer value, if the timer is enabled.

35.5.1.11 Timer Control (TCTRL0 - TCTRL3)

35.5.1.11.1 Offset

Register	Offset
TCTRL0	28h
TCTRL1	38h
TCTRL2	48h
TCTRL3	58h

35.5.1.11.2 Function

Contains control fields for each timer channel:

- TRG_SEL for trigger selection
- TRG_SRC for trigger source selection
- TROT for timer reload
- TSOI for timer stoppage
- TSOT for timer decrementing
- MODE for timer operation mode selection
- CHAIN for channel chaining

You must update timer controls when the timer is disabled, and use either of the following ways to disable a timer:

- By writing 1 to the specific timer's CLRTEN[CLR_T_EN_*n*] field.
- By writing 0 to T_EN for that channel.

NOTE

Unless the peripheral clock to the timers is enabled (MCR[M_CEN] = 1), reading or writing to this register generates a transfer error.

35.5.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				TRG_SEL				TRG_SRC	0				TROT	TSOI	TSOT
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												MODE		CHAIN	T_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

35.5.1.11.4 Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Field	Function
27-24 TRG_SEL	Trigger Select Selects the trigger to use for starting and/or reloading the LPIT timer. This field selects one trigger from the set of internal or external triggers that TRG_SRC provides. TRG_SRC helps you make a choice between internal and external trigger signals for each channel. **NOTE:** You must change the value of this field when the LPIT timer channel is disabled. 0000b-0011b - Timer channel 0–3 trigger source 0100b-1111b - Reserved
23 TRG_SRC	Trigger Source Selects whether to use internal or external trigger sources. You can select the trigger to be used by using this field or the TRG_SEL field. If a channel does not have an associated external trigger, write 1 to TRG_SRC. See LPIT chip-specific information for the available external trigger options. 0b - External 1b - Internal
22-19 —	Reserved
18 TROT	Timer Reload on Trigger Specifies whether the timer reloads after the selected trigger. If this field = 1, the LPIT timer reloads when a rising edge is detected on the selected trigger input. The trigger input is ignored if LPIT is disabled during Debug mode (MCR[DBG_EN] = 0) or Doze mode (MCR[DOZE_EN] = 0). 0b - Does not reload 1b - Reloads
17 TSOI	Timer Stop on Interrupt Controls whether the channel timer stops after being timed out. If this field = 0, the channel timer does not stop after timeout. If this field = 1, the channel timer stops after a timeout and then restarts based on the configuration of TSOT. If TSOT = 0, the channel timer restarts after a rising edge on T_EN is detected, which means that the timer channel is disabled and then enabled. If TSOT = 1, the channel timer restarts after a rising edge on the selected trigger is detected. 0b - Does not stop 1b - Stops
16 TSOT	Timer Start on Trigger Controls when the timer starts decrementing. If this field = 0, the timer starts decrementing immediately based on the restart condition (controlled by TSOI). If this field = 1, the timer starts decrementing when a rising edge on a selected trigger is detected. 0b - Immediately 1b - When a rising edge is detected
15-4 —	Reserved
3-2 MODE	Timer Operation Mode Configures the channel timer's mode of operation. This field controls how the timer decrements. 00b - 32-bit periodic counter 01b - Dual 16-bit periodic counter 10b - 32-bit trigger accumulator 11b - 32-bit trigger input capture

Table continues on the next page...

Field	Function
1 CHAIN	Chain Channel Specifies whether channel chaining is enabled. If channel chaining is enabled, the timer channel decrements after the previous channel's timeout (when timer channel N-1 trigger asserts). Timer channel 0 cannot be chained. If channel chaining is disabled, the channel timer runs independently. 0b - Disabled 1b - Enabled
0 T_EN	Timer Enable Enables the timer channel. 0b - Disable 1b - Enable

35.6 Usage Guide

35.6.1 Periodic timer/counter

LPIT typical usage is to generate periodic trigger pulses and interrupts.

Example: LPIT channel0 trigger a periodic interrupt every 1 second

- Enable the LPIT module clock;
- Reset the timer channels and registers;
- Setup timer operation in debug and doze modes and enable the module;
- Setup the channel counters operation mode to "32-bit Periodic Counter",and keep default values for the trigger source;
- Set timer period for channel 0 as 1 second;
- Enable channel0 interrupt;
- Starts the timer counting afer all configuration;
- In the channel interrupt rountine, clear the channel flag every 1 second.

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(LPIT0);
LPIT0_MCR |= LPIT_MCR_SW_RST_MASK;
LPIT0_MCR &= ~LPIT_MCR_SW_RST_MASK;
LPIT0_MCR |= (LPIT_MCR_DBG_EN(1) | LPIT_MCR_DOZE_EN(1) | LPIT_MCR_M_CEN_MASK);
LPIT0_TCTRL0 |= LPIT_TCTRL_MODE(0);
LPIT0_TVAL0 = ONE_SECOND_VALUE;
LPIT0_MIER |= LPIT_MIER_TIE0_MASK;
NVIC_EnableIRQ(LPIT0_IRQ);
LPIT0_SETTEN |= LPIT_SETTEN_SET_T_EN_0_MASK;
```

35.6.2 LPIT/ADC Trigger

The LPIT could be used as an alternate ADC hardware trigger source, whose implementation is via TRGMUX. Each LPIT channel supports one pre-trigger and one trigger. The LPIT channels are implemented based on independent counters. When used as ADC trigger source, the channel outputs are ORed together to generate the ADC hardware trigger. The following diagram shows an example of using LPIT triggering ADC0.

Example: LPIT hardware trigger via TRGMUX for ADC conversion

- ADC module initialization and enable its hardware trigger;
- Enable the LPIT module clock;
- Reset the LPIT timer channels and registers;
- Setup timer operation in debug and doze modes and enable LPIT module;
- Setup the LPIT_CH0 and LPIT_CH1 counters mode to "32-bit Periodic Counter",and keep default values for the trigger source;
- Set timer period for LPIT_CH0 and LPIT_CH1,they are used as ADC pre-trigger delay;
- Starts the timer counting after all configuration;
- In SIM register, select TRGMUX output as ADC pre-trigger and trigger source;
- Configure LPIT_CH0 and LPIT_CH1 as ADC hardware trigger by TRGMUX;
- In the ADC interrupt routine, clear the COCO flag and read the conversion value. (If Rn is read, the COCO flag will be cleared automatically.)

The following pseudo-code matches the described setup above:

```
ADC_Config();
CLOCK_EnableClock(LPIT0);
LPIT0_MCR |= LPIT_MCR_SW_RST_MASK;
LPIT0_MCR &= ~LPIT_MCR_SW_RST_MASK;
LPIT0_MCR |= (LPIT_MCR_DBG_EN(1) | LPIT_MCR_DOZE_EN(1) | LPIT_MCR_M_CEN_MASK);
LPIT0_TCTRL0 |= LPIT_TCTRL_MODE(0);
LPIT0_TCTRL1 |= LPIT_TCTRL_MODE(0);
LPIT0_TVAL0 = ADC_PRETRG_DELAY_VALUE1;
LPIT0_TVAL1 = ADC_PRETRG_DELAY_VALUE2;
LPIT0_SETTEN |= LPIT_SETTEN_SET_T_EN_0_MASK|LPIT_SETTEN_SET_T_EN_1_MASK;
SIM_ADCOPT |= SIM_ADCOPT_ADC0TRGSEL(1)|SIM_ADCOPT_ADC0PRETRGSEL(1);
TRGMUX0_ADC0 = TRGMUX_TRGCFG_SEL0(7)|TRGMUX_TRGCFG_SEL1(8);
```

Chapter 36
Pulse Width Timer (PWT)

36.1 Chip-specific information for this module

36.1.1 Instantiation Information

The Pulse Width Timer (PWT) module on this device consists of one 16-bit counter, which can be used to capture or measure the pulse width mapping on its input channels.

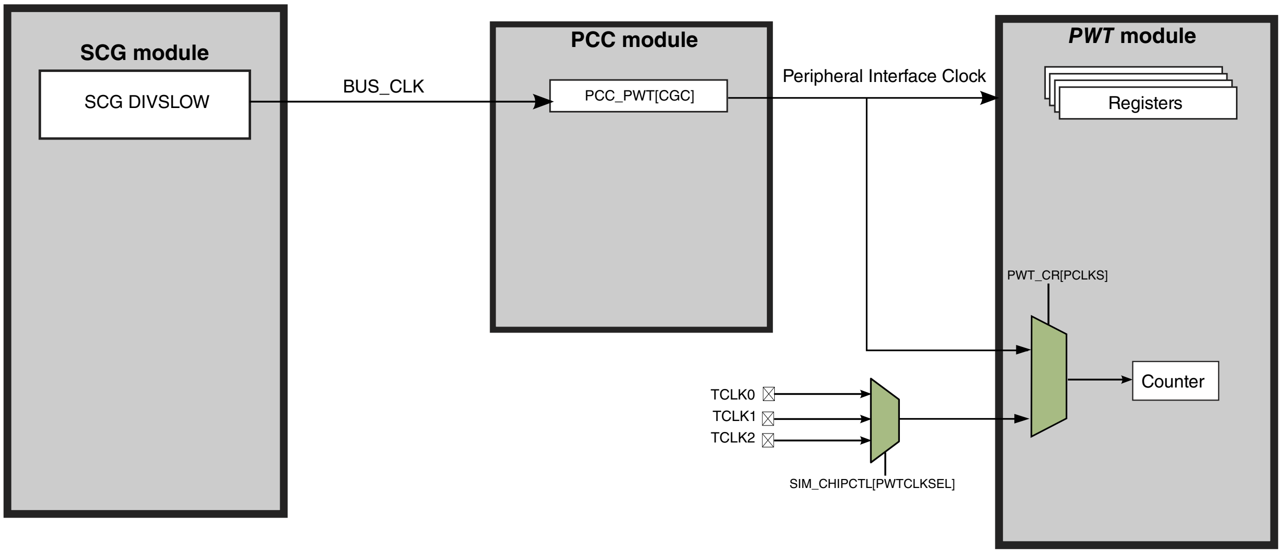
The counter of PWT has two selectable clocks sources, and support up to BUS_CLK with internal timer clock. PWT module supports programmable positive or negative pulse edges, and programmable interrupt generation upon pulse width values or counter overflow.

36.1.2 PWT Clocking Information

Two software selectable clock sources are available for input to pre-scaler divider of PWT module:

- Bus clock
- External clock from pins (TCLKx)

Peripheral Clocking - PWT

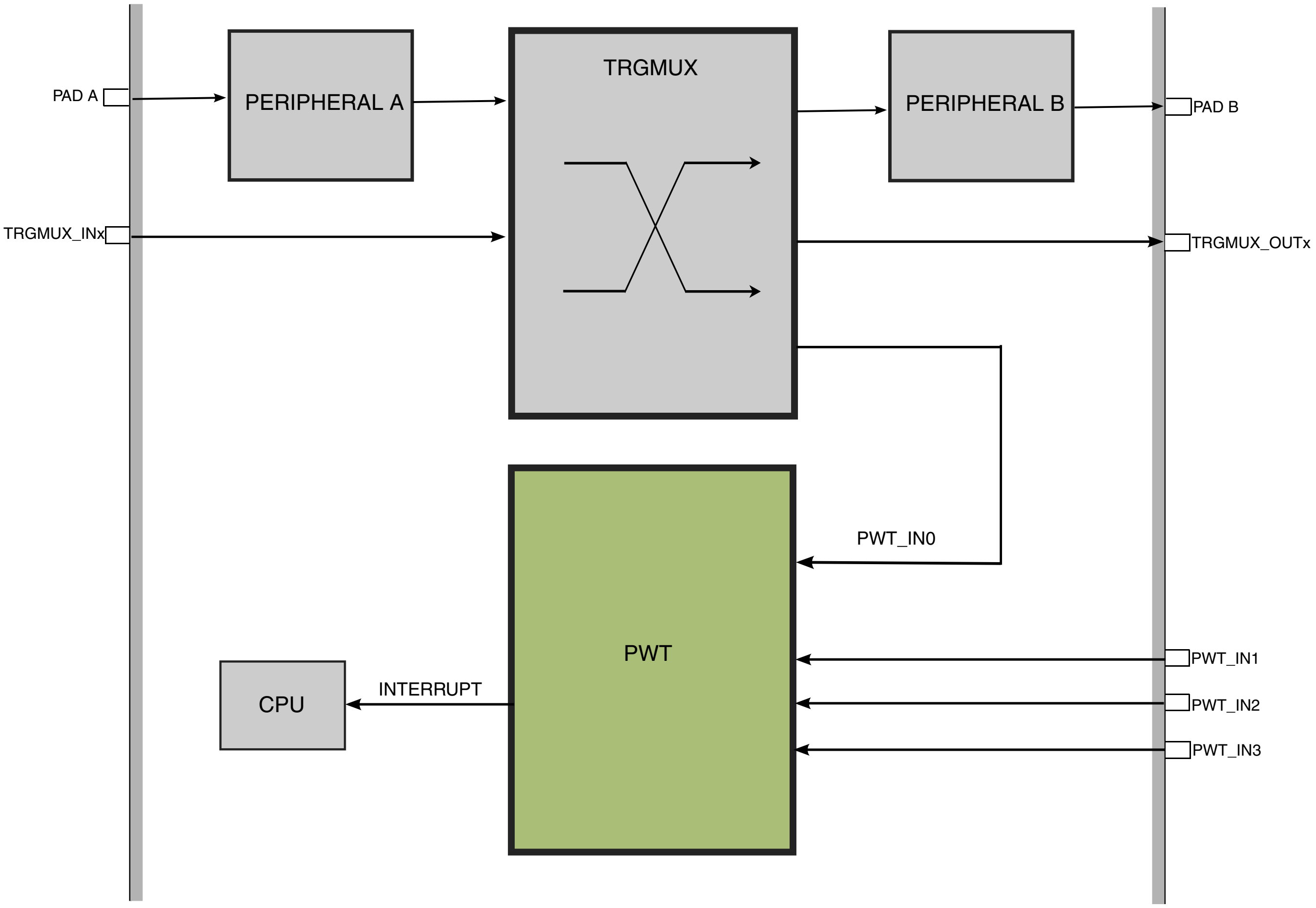


36.1.3 Inter-connectivity Information

PWT module has four input channels, which is connected as shown in the following table:

Table 36-1. PWT input connections

PWT input channel	Connection
0	TRGMUX output
1	PWT_IN1 pin
2	PWT_IN2 pin
3	PWT_IN3 pin



36.2 Overview

PWT measures the duration of a pulse or the period of a signal input using a 16-bit free-running counter, and divides clock frequency using a clock prescaler.

36.2.1 Block diagram

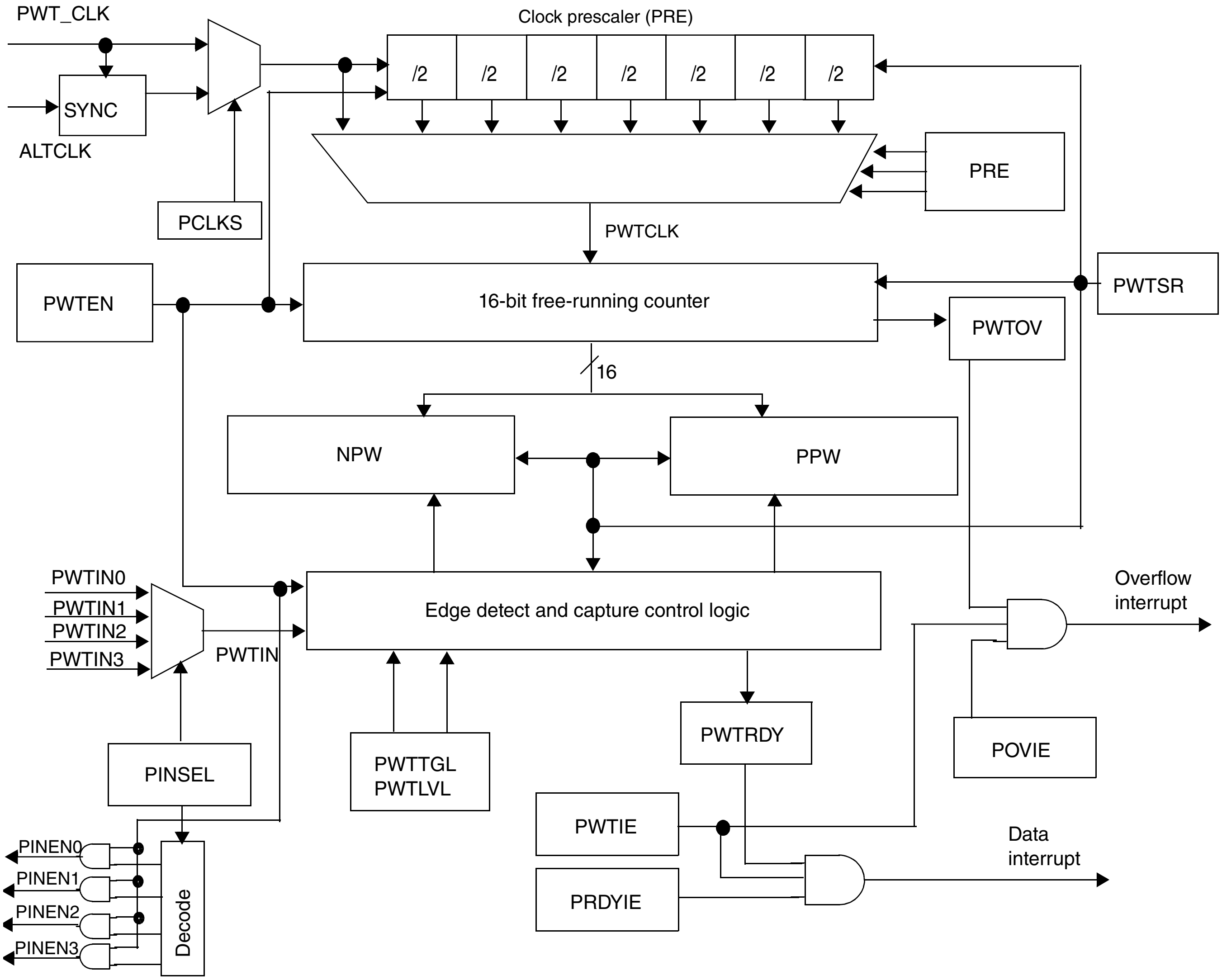


Figure 36-1. PWT block diagram

NOTE

PWT_CLK depends on the Chip input clock.

36.2.2 Features

- Automatic measurement of pulse width with 16-bit resolution
- Separate positive and negative pulse width measurements
- Programmable triggering edge for starting measurement
- Programmable measuring time between successive alternating edges, rising edges or falling edges
- Programmable prescaler from clock input as 16-bit counter time base

- Two selectable clock sources—PWT_CLK and alternative clock
- Four selectable pulse inputs
- Programmable interrupt generation upon pulse width value updated and counter overflow

36.3 Functional description

36.3.1 PWT counter and PWT clock prescaler

PWT measures the duration of a pulse or the period of a signal input to the PWTIN using a 16-bit free-running counter (CNTH[PWTH] and CNTL[PWTL]). A clock prescaler of CLKPRE(CR[PRE]) in PWT provides the frequency divided clock to CNTH[PWTH] and CNTL[PWTL]. The clock prescaler can select clock input from bus clock and alternative clock by CR[PCLKS].

The counter uses the frequency divided clock from CR[PRE] for counter advancing. The frequency of the prescaler is programmable as the clock frequency divided by 1, 2, 4, 8, 16, 32, 64, 128 (depending on the setting of CR[PRE]).

When the counter is enabled, it starts counting using the selected and divided clock source. The counter is cleared without loading to the registers when the first valid edge (trigger edge) is detected. If no valid trigger edge is detected for a long time, it is possible for the counter to overflow. When 16-bit free-running counter is running, any edge to be measured after the trigger edge causes the value of CNTH[PWTH] and CNTL[PWTL] to be uploaded to the appropriate pulse width registers. At the same time, CNTH[PWTH] and CNTL[PWTL] are reset to 0x00 and the clock prescaler output is also reset. CNTH[PWTH] and CNTL[PWTL] start advancing again with the input clock. If the counter runs exceeding 0xFFFF (then it re-counts from 0), the CS[PWTOV] bit is set.

36.3.2 Edge detection and capture control

The edge detection and capture control logic detects measurement trigger edges and controls which pulse width register(s) is/are updated and when to update.

PWTIN can be selected from one of four sources by configuring CR[PINSEL].

As soon as PWT is enabled, the counter starts counting up until an edge transitions on the selected PWTIN. Determined by PWT_CS[FCTLE] and PWTIN state, the counter's contents can be uploaded to the corresponding registers.

If PWT_CS[FCTLE] is cleared to 0, the first 16-bit free counter content will just be ignored and not uploaded to neither PWT_PPH:L nor PWT_NPH:L. Otherwise, determined by current PWTIN state(as signalized by PWT_CR[LVL]), the counter content will be uploaded to PWT_PPH:L if PWT_CR[LVL] is 1 and PWT_NPH:L if PWT_CR[LVL] is 0.

In normal measurement, when the PWT_CS[PWTRDY] is set, software can then read out the positive pulse width and negative pulse width values from PWT_PPH:L and PWT_NPH:L respectively and the selected PWTIN duty ratio can then be calculated. The exception is when overflow happens, software needs to check PWT_CR[TGL] and PWT_CR[LVL] to determine if it is low overflow(0 duty ratio) ,high overflow(100% duty ratio), toggled low overflow or toggled high overflow. Below table 1 shows the meaning:

Table 36-2. Abnormal PWTIN duty ratio

Flag	PWT_CR[TGL]	PWT_CR[LVL]	Description
PWT_CS[PWTOV]	0	0	Low overflow
	0	1	High overflow
	1	0	Toggled low overflow
	1	1	Toggled high overflow

The following figure illustrates the trigger edge detection and pulse width registers update of PWT.

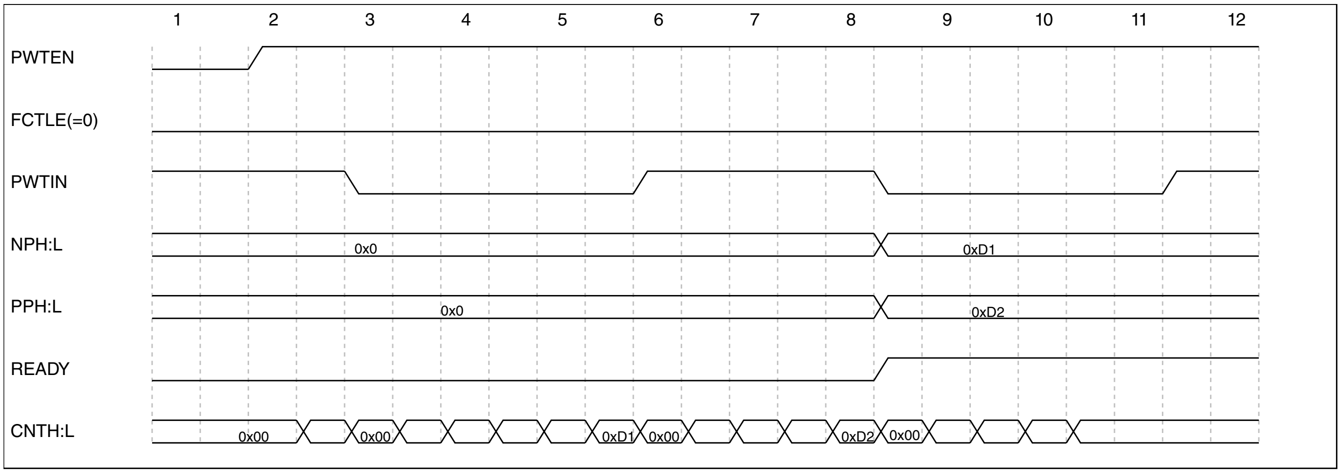


Figure 36-2. PWT normal measurement with FCTLE = 0

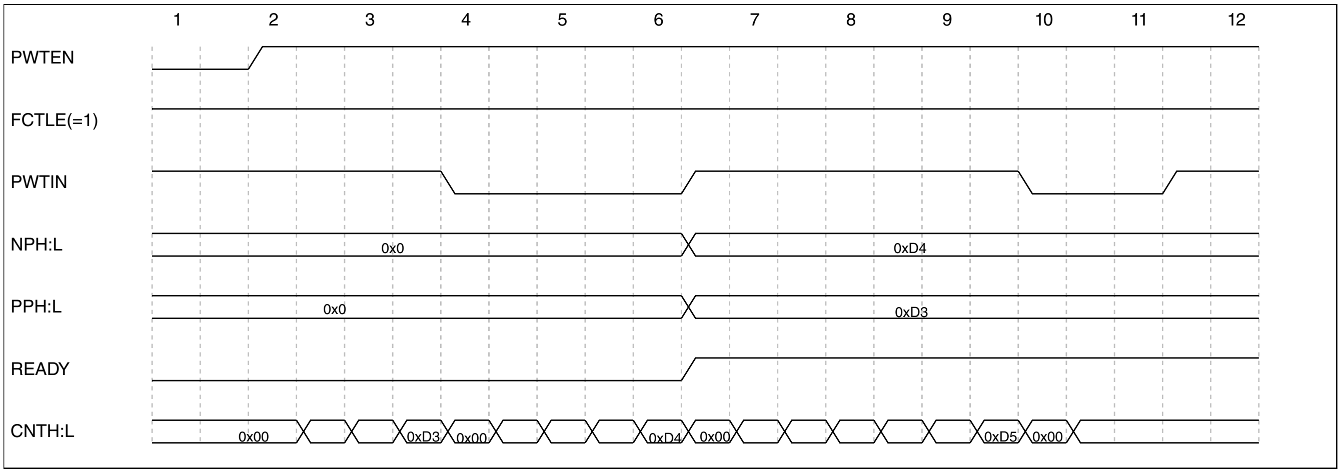


Figure 36-3. PWT normal measurement with FCTLE = 1

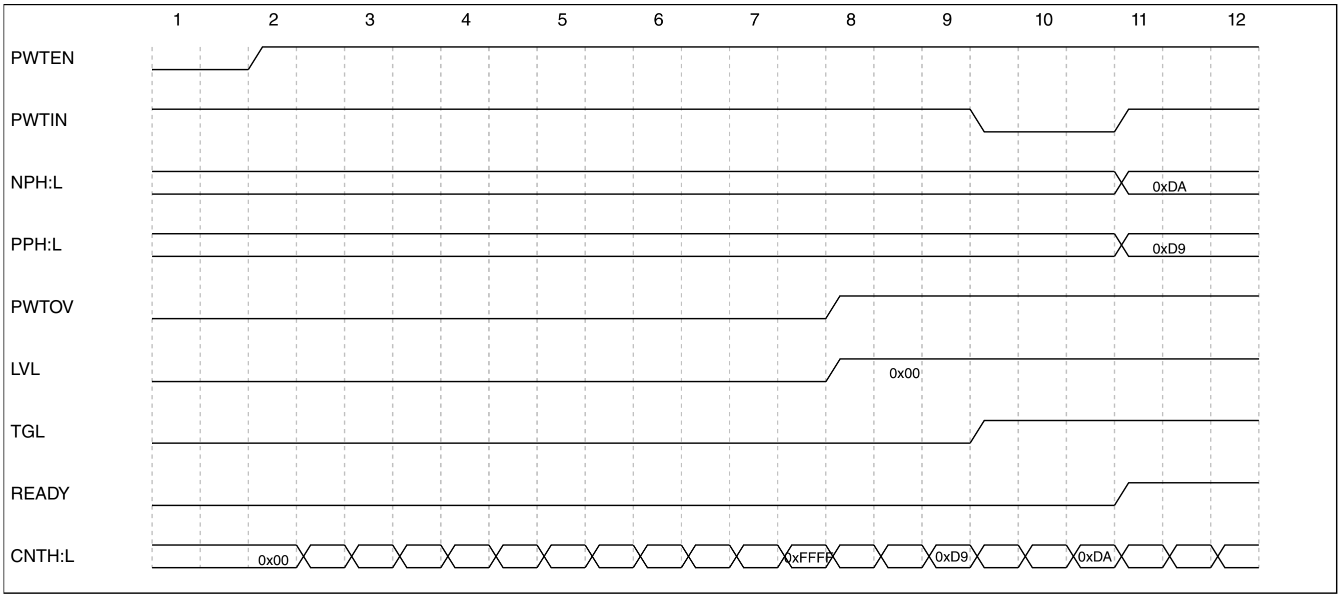


Figure 36-4. PWT measurement overflows at high level with FCTLE = 1

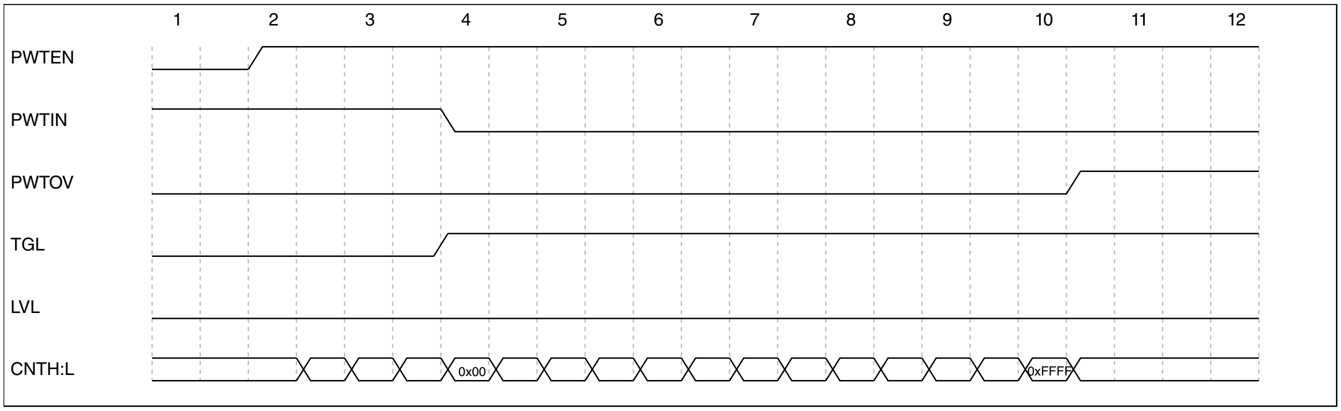


Figure 36-5. PWT measurement overflows with PWTIN toggles

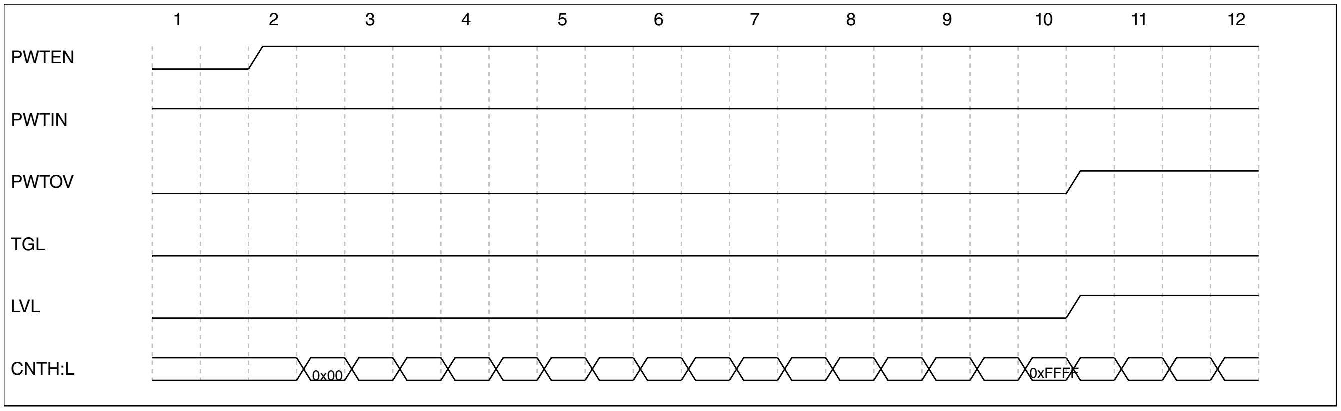


Figure 36-6. PWT measurement overflows without PWTIN toggles

The PWTRDY flag bit indicates that the data can be read in PPH[PPWH], PPL[PPWL] and/or NPH[NPWH], NPL[NPWL], whenever there is a valid edge transition happened on the selected PWTIN.

When CS[PWTRDY] is set, the updated pulse width register(s) transfers the data to corresponding 16-bit read buffer(s). The read value of pulse width registers actually comes from the corresponding read buffers, whenever the chip is in normal run mode or debug mode. Reading followed by writing 0 to CS[PWTRDY] flag clears this bit. Until CS[PWTRDY] is cleared, the 16-bit read buffer(s) cannot be updated. But this does not affect the upload of pulse width registers from the PWT counter.

If another pulse measurement is completed and the pulse width registers are updated, the clearing of the CS[PWTRDY] flag fails, that is, CS[PWTRDY] remains set, but the 16-bit read buffer(s) is updated again as long as the action is cleared. You must complete the pulse width data reading before clearing CS[PWTRDY] to avoid missing data. This mechanism makes sure that the second pulse measurement is not lost in case the MCU does not have enough time to read the first one ready for reading. This mechanism is automatically restarted by an MCU reset, which is done by writing a 1 to CS[PWTSR] or writing a 0 to CS[PWTEN] followed by writing a 1 to it.

The following figure illustrates the buffering mechanism of pulse width register:

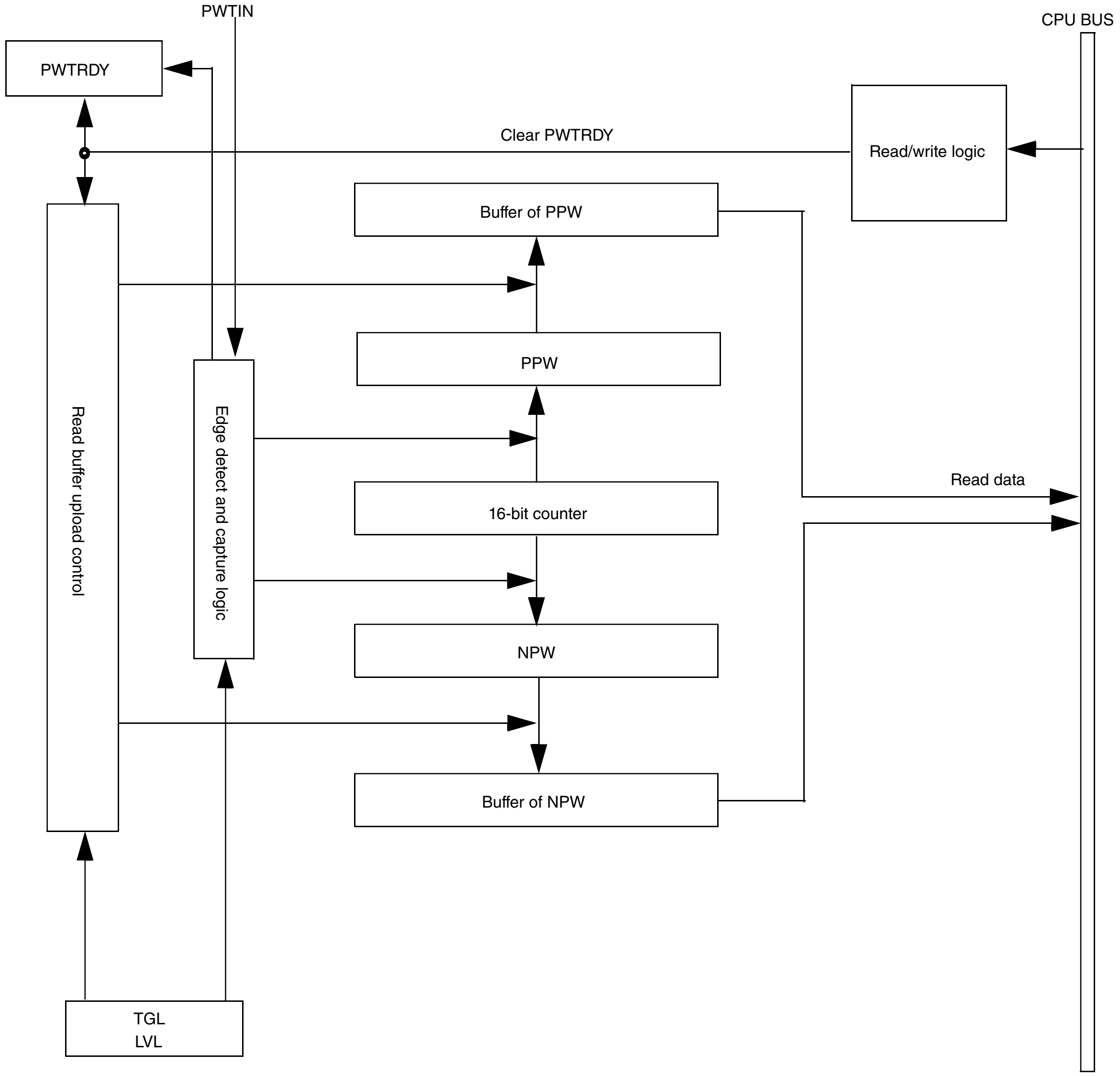


Figure 36-7. Buffering mechanism of pulse width register

When PWT completes any pulse width measurement, a signal is generated to reset CNTH[PWTH] and CNTL[PWTL], and the clock prescaler's output after the data has been uploaded to the pulse width registers. To make sure that there is no missing count, CNTH[PWTH], CNTL[PWTL], and the clock prescaler's output are reset in a bus clock cycle after the completion of a pulse width measurement.

36.3.3 Counter overflow function

After PWT counter is enabled, the counter overflow occurs if no valid trigger edge is detected for a long time.

The following figures illustrate the counter overflow in different PWTIN pin states.

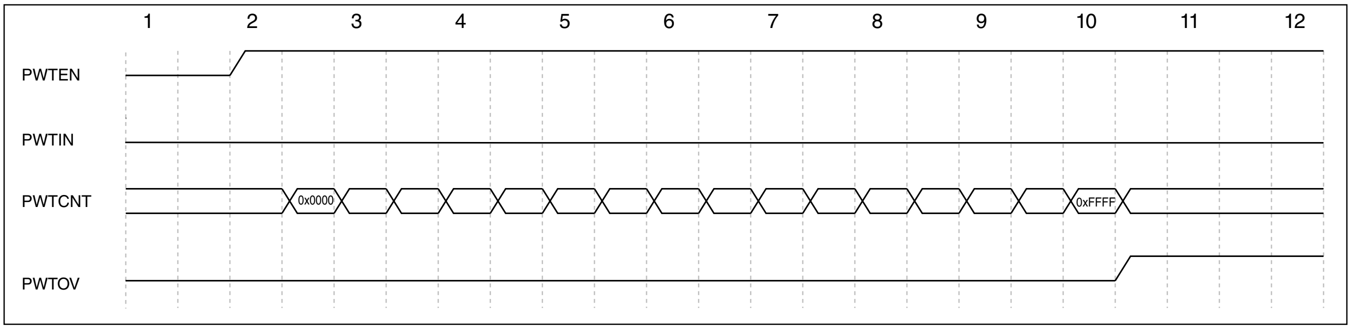


Figure 36-8. PWT counter overflow with low PWTIN

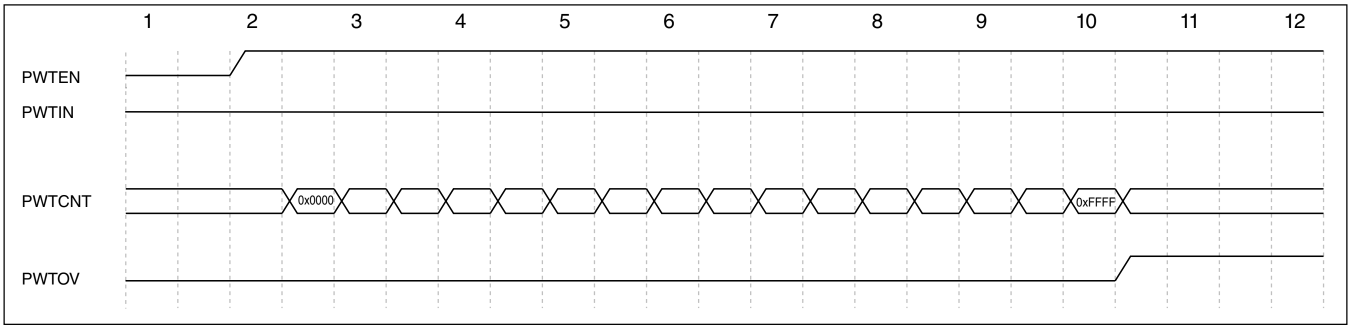


Figure 36-9. PWT counter overflow with high PWTIN

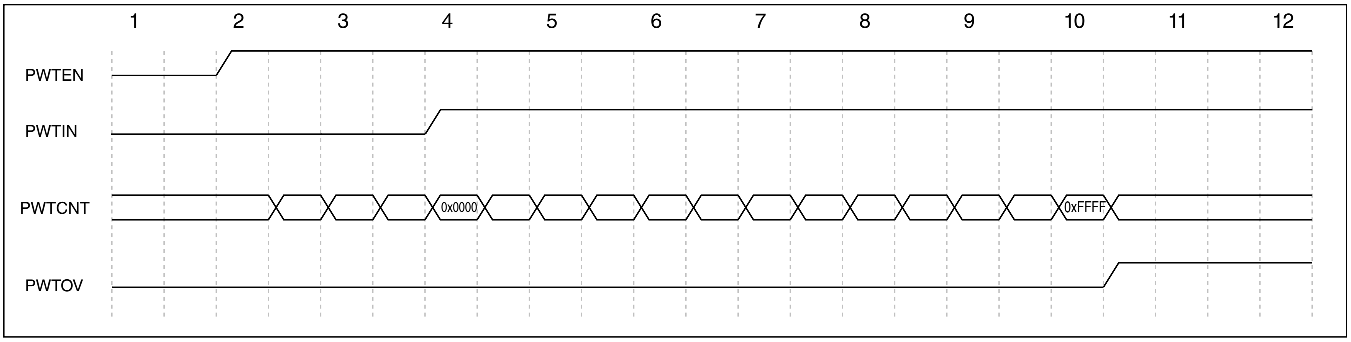


Figure 36-10. PWT counter overflow with PWTIN positive edge

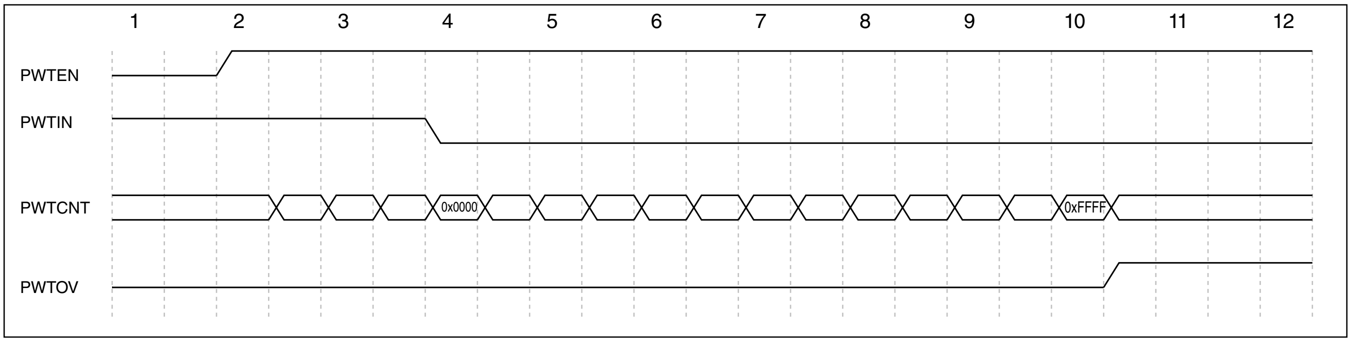


Figure 36-11. PWT counter overflow with PWTIN negative edge

36.3.4 Modes of operation

The following table describes the operation of the PWT module in various modes.

Modes	Description
Run	When enabled, PWT is active.
Wait	When enabled, PWT is active and can perform the wake-up function if the corresponding interrupt is enabled.
Stop	When enabled, PWT is halted. Contents and operating status of registers are preserved. If stop exits with reset then the module resets. If stop exits with another source, the module resumes operation based on module status upon exit.
Active background	Upon entering Debug mode, PWT suspends all counting and pulse edge detection until the MCU returns to normal user operating mode. Counting and edge detection resume from the suspended value when normal user operating mode returns as long as the PWTSR bit (PWT software reset) is not written to 1 and the PWT module is still enabled.

36.3.5 Clocking

PWT has two clock sources, one is PWT_CLK and the other is ALTCLK. The clock source is configured by the CR[PCLKS] bit.

NOTE

> The ALTCLK input must be synchronized by the bus clock. Variations in duty cycle and clock jitter must also been accommodated so that the ALTCLK input does not exceed one-fourth of the bus frequency.

36.3.6 Reset

PWT soft reset is built into PWT as a mechanism used to reset or restart this module. PWT soft reset is triggered by writing a 1 to CS[PWTSR]. Unlike reset by the CPU, PWT soft reset does not restore everything in this module to its reset state. The following steps explain how a PWT soft reset is completed.

1. The PWT counter is set to 0x0000.
2. The 16-bit buffer of PWT counter is reset.
3. The PWT clock prescaler output is reset.
4. The edge detection logic is reset.
5. The capture logic is reset and the latching mechanism of pulse width registers is also restarted.
6. PPH[PPWH], PPL[PPWL], NPH[NPWH], and NPL[NPWL] are set to 0x00.

7. CS[PWTOV] and CS[PWTRDY] are set to 0.
8. All other PWT register settings are not changed.

Writing a 0 to CS[PWTEN] also resets PWT, but the reset state is held until CS[PWTEN] is set to 1.

36.3.7 Interrupts

The other major component of PWT is the interrupts control logic. When CS[PWTOV] and CS[POVIE] are set, a PWT overflow interrupt can be generated. When CS[PWTRDY] bit and CS[PRDYIE] are set, a pulse width data ready interrupt can be generated. CS[PWTIE] controls the interrupt generation of PWT. The functionality of PWT is not affected while the interrupt is being generated.

36.4 External signals

Table 36-3. PWT signal description

Signal	I/O	Pullup	Description
PWTIN3:0	I	No	Pulse width timer capture inputs
PINEN3:0	O	No	Enable signals for PWTIN3:0 inputs
ALTCLK	I	No	Alternative clock source for the counter

36.4.1 PWTIN3:0

PWTIN3:0 can come from internal or external sources. The minimum pulse width to be measured is 1 PWTCLK cycle, any pulse narrower than this value is ignored by PWT module. The PWTCLK cycle time depends on the PWT clock source selection and prescaler rate setting.

36.4.2 ALTCLK

The ALTCLK input can be selected as the clock source of the PWT counter when CR[PCLKS] is set. The ALTCLK pin can be shared with a general-purpose port pin. See the Pins and Connections chapter for the pin location and priority of this function.

36.5 Initialization

To initialize PWT:

1. Configure the PCLKS, PRE, PINSEL, and FCTLE bits to select clock source, set pre-scaler rate, select PWT input pin, and enable first counter load.
2. Set the PWTIE, PRDYIE, and POVIE bits if corresponding interrupt is desired to be generated.
3. Set the PWTEN bit to enable the pulse width measurement.

NOTE

Steps 1 and 2 can be done sequentially or not, but they must be completed before step 3 to make sure that all settings are ready before pulse width measurement is enabled.

36.6 Application

The period of PWTIN input signal is measured by the following logic:

1. PWT is initialized by configuring CR[PRE] and CR[FCTLE]. See Configuration examples for specific configuration examples.
2. The counter starts counting from 0 using PWT when the first rising trigger edge of PWTIN is detected.
3. Values of the CNTH[PWTH] bit and CNTL[PWTL] bit are uploaded to the PPH[PPWH] and PPL[PPWL] bits respectively and reset to 0x0000. The counter starts counting again upon subsequent falling edges.
4. The counter value is uploaded to the NPH[NPWH] and NPL[NPWL] bits and reset to 0x0000 upon successive rising edges.

36.6.1 Configuration examples

36.6.1.1 Configuration example when PWTCLK is bus clock divided by 1

Write 000b to CR[PRE] and 1 to CR[FCTLE] to initialize PWT.

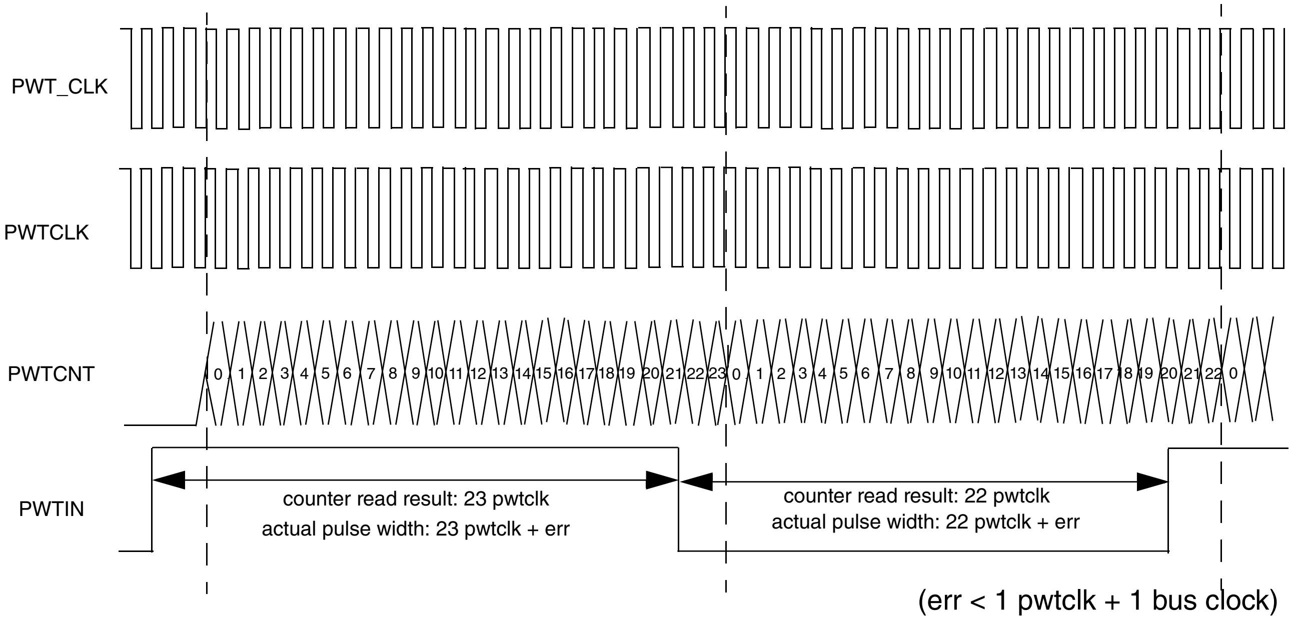

PWT_CLK
PWTCLK
PWTCNT
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 0
PWTIN
counter read result: 23 pwtclk
actual pulse width: 23 pwtclk + err
counter read result: 22 pwtclk
actual pulse width: 22 pwtclk + err
(err < 1 pwtclk + 1 bus clock)

Figure 36-12. Configuration example when PWTCLK is bus clock divided by 1

36.6.1.2 Configuration example when PWTCLK is bus clock divided by 2

Write 001b to CR[PRE] and 1 to CR[FCTLE] to initialize PWT.

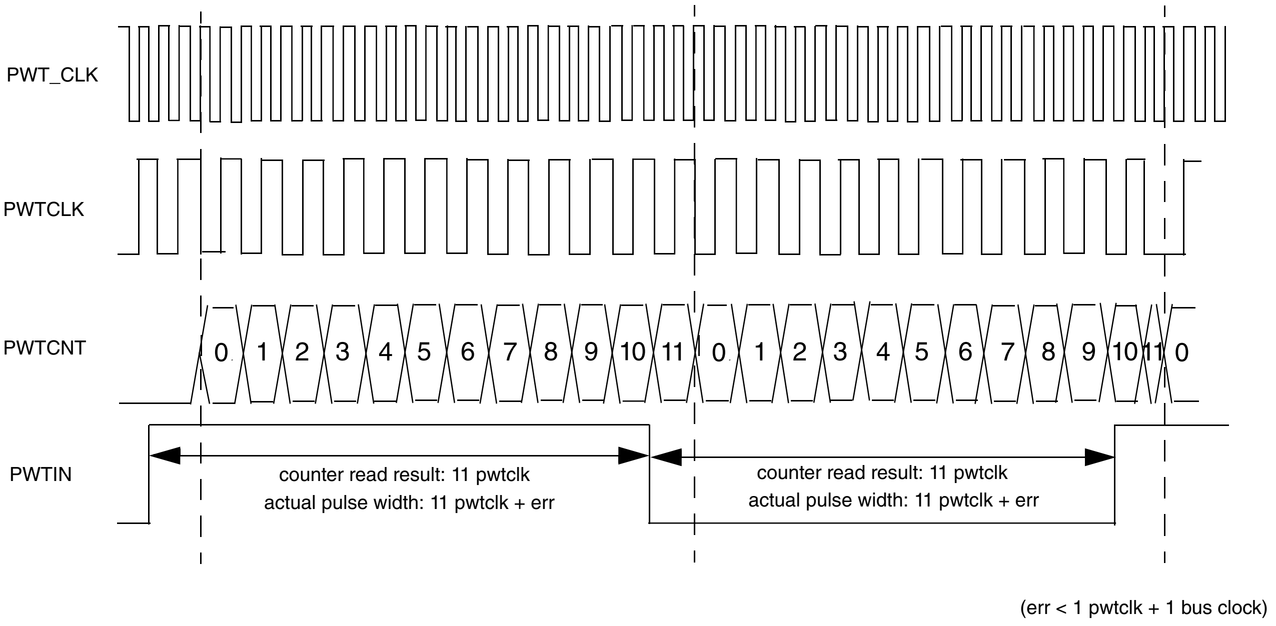

PWT_CLK
PWTCLK
PWTCNT
0 1 2 3 4 5 6 7 8 9 10 11 0 1 2 3 4 5 6 7 8 9 10 11 0
PWTIN
counter read result: 11 pwtclk
actual pulse width: 11 pwtclk + err
counter read result: 11 pwtclk
actual pulse width: 11 pwtclk + err
(err < 1 pwtclk + 1 bus clock)

Figure 36-13. Configuration example when PWTCLK is bus clock divided by 2

36.6.1.3 Configuration example when PWTCLK is bus clock divided by 4

Write 010b to CR[PRE] and 1 to CR[FCTLE] to initialize PWT.

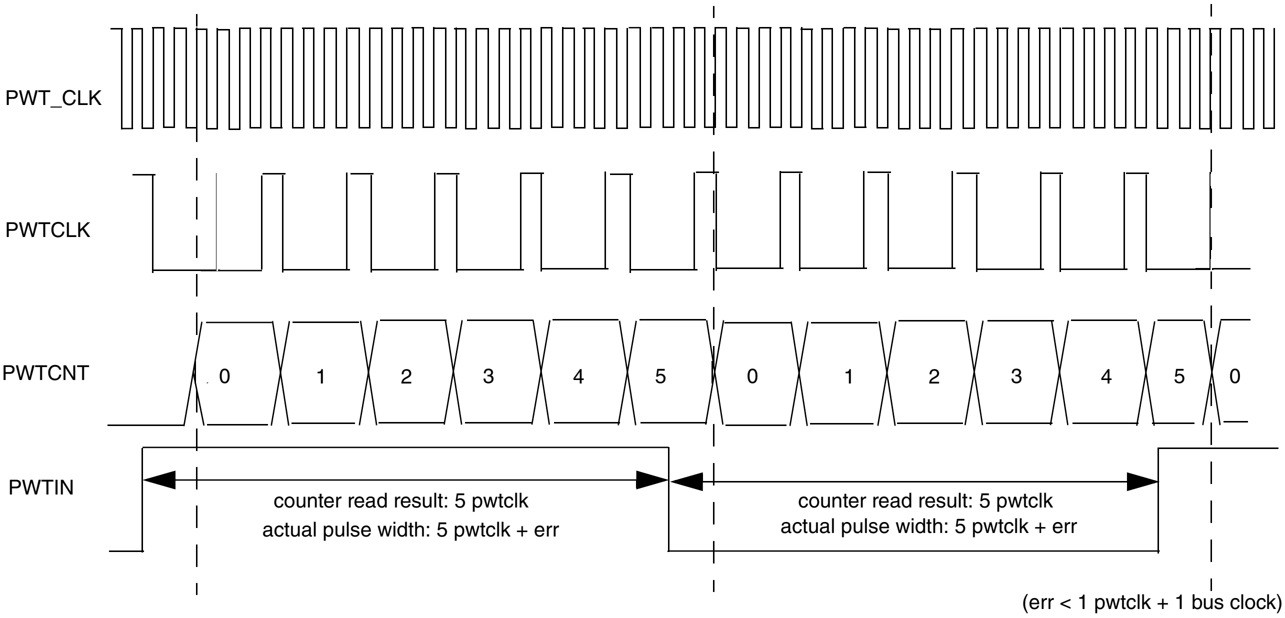

PWT_CLK
PWTCLK
PWTCNT
0
1
2
3
4
5
0
1
2
3
4
5
0
PWTIN
counter read result: 5 pwtclk
actual pulse width: 5 pwtclk + err
counter read result: 5 pwtclk
actual pulse width: 5 pwtclk + err
(err < 1 pwtclk + 1 bus clock)

Figure 36-14. Configuration example when PWTCLK is bus clock divided by 4

36.6.1.4 Configuration example when PWTCLK is bus clock divided by 8

Write 011b to CR[PRE] and 1 to CR[FCTLE] to initialize PWT.

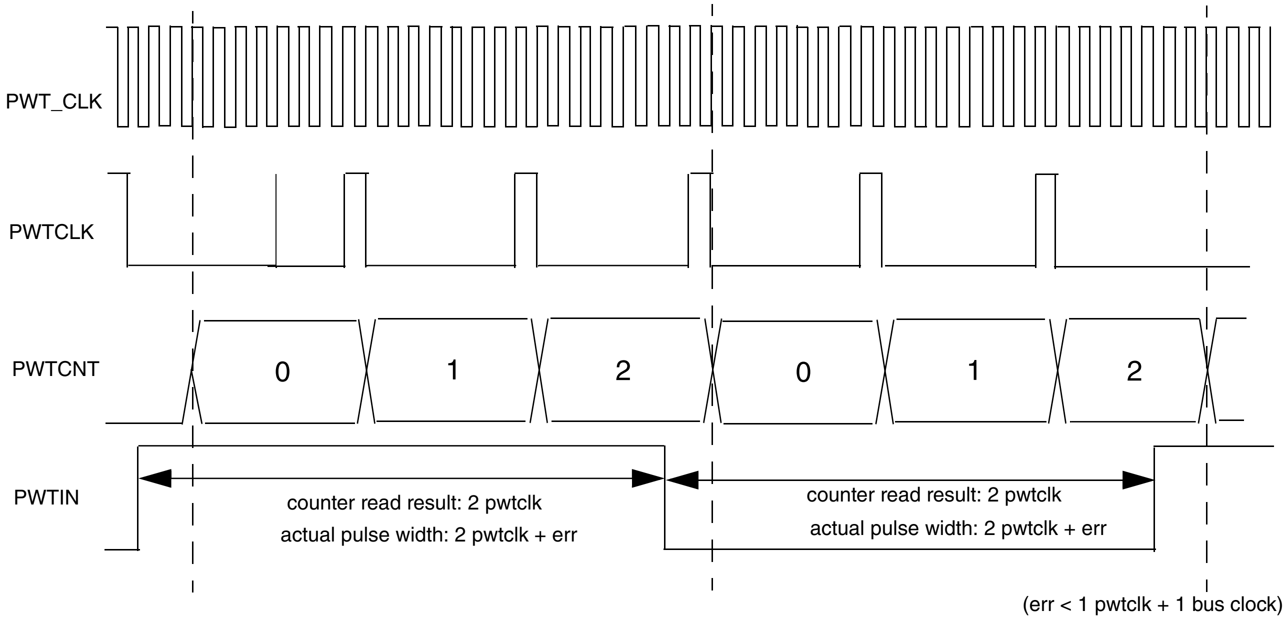

PWT_CLK
PWTCLK
PWTCNT
0
1
2
0
1
2
PWTIN
counter read result: 2 pwtclk
actual pulse width: 2 pwtclk + err
counter read result: 2 pwtclk
actual pulse width: 2 pwtclk + err
(err < 1 pwtclk + 1 bus clock)

Figure 36-15. Configuration example when PWTCLK is bus clock divided by 8

36.7 Register descriptions

36.7.1 PWT register descriptions

36.7.1.1 PWT memory map

PWT base address: 4005_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Pulse Width Timer Control and Status (CS)	8	RW	00h
1h	Pulse Width Timer Control (CR)	8	RW	00h
2h	Pulse Width Timer Positive Pulse Width (High) (PPH)	8	RO	00h
3h	Pulse Width Timer Positive Pulse Width (Low) (PPL)	8	RO	00h
4h	Pulse Width Timer Negative Pulse Width (High) (NPH)	8	RO	00h
5h	Pulse Width Timer Negative Pulse Width (Low) (NPL)	8	RO	00h
6h	Pulse Width Timer Counter (High) (CNTH)	8	RO	00h
7h	Pulse Width Timer Counter (Low) (CNTL)	8	RO	00h

36.7.1.2 Pulse Width Timer Control and Status (CS)

36.7.1.2.1 Offset

Register	Offset
CS	0h

36.7.1.2.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PWTEN	PWTIE	PRDYIE	POVIE	0	FCTLE	PWTRDY	PWTOV
W					PWTSR			
Reset	0	0	0	0	0	0	0	0

36.7.1.2.3 Fields

Field	Function
7 PWTEN	PWT Module Enable Specifies whether to enable or disable PWT. **NOTE:** To avoid unexpected behavior, do not change any PWT configurations as long as PWTEN is set. 0b - Disables 1b - Enables
6 PWTIE	PWT Module Interrupt Enable Specifies whether to enable or disable PWT to generate an interrupt. 0b - Disables 1b - Enables
5 PRDYIE	PWT Pulse Width Data Ready Interrupt Enable Specifies whether to enbale or disable PWT to generate an interrupt when PWTRDY is set as long as PWTIE is set. 0b - Disables 1b - Enables
4 POVIE	PWT Counter Overflow Interrupt Enable Specifies whetehr to enbale or disable PWT to generate an interrupt when PWTOV is set due to PWT counter overflow. 0b - Disables 1b - Enables
3 PWTSR	PWT Soft Reset Specifies whether to perform a soft reset to PWT.

Table continues on the next page...

Field	Function
	0b - No action taken. 1b - Performs a soft reset to PWT.
2 FCTLE	First counter load enable after enable Specifies whether to load the counter value to the corresponding PWTx_PPW[H,L], PWTx_NPW[H,L] after first enable. 0b - Not to load the first counter values to the corresponding registers 1b - Load the first coutner values to the corresponding registers depending on the PWTIN level
1 PWTRDY	PWT Pulse Width Valid Indicates whether PWT register(s) is updated and ready to be read. This field is cleared by reading PWTRDY and then writing 0 to PWTRDY bit when PWTRDY is set. Writing 1 to this field has no effect. 0b - PWT register(s) is not up-to-date. 1b - PWT register(s) is updated.
0 PWTOV	PWT Counter Overflow Indicates whether the PWT counter runs from 0xFFFF to 0x0000. This field is cleared by writing 0 to PWTOV when PWTOV is set. Writing 1 to this field has no effect. If another overflow occurs when this field is being cleared, the clearing fails. 0b - PWT counter has no overflow. 1b - PWT counter runs from 0xFFFF to 0x0000.

36.7.1.3 Pulse Width Timer Control (CR)

36.7.1.3.1 Offset

Register	Offset
CR	1h

36.7.1.3.2 Diagram

Bits	7	6	5	4	3	2	1	0
R	PCLKS	PINSEL		TGL	LVL	PRE		
W				W1C				
Reset	0	0	0	0	0	0	0	0

36.7.1.3.3 Fields

Field	Function
7	PWT Clock Source Selection

Table continues on the next page...

Field	Function
PCLKS	Controls the selection of clock source for the PWT counter. 0b - PWT_CLK is selected as the clock source of PWT counter. 1b - Alternative clock is selected as the clock source of PWT counter.
6-5 PINSEL	PWT Pulse Inputs Selection Enables the corresponding PWT input port, if this PWT input comes from an external source. 00b - PWTIN0 is enabled. 01b - PWTIN1 is enabled. 10b - PWTIN2 is enabled. 11b - PWTIN3 is enabled.
4 TGL	PWTIN states Toggled from last state This flag indicates if the selected PWTIN has toggled its state since last time this bit has cleared to 0. 0b - The selected PWTIN hasn't changed its original states from last time. 1b - The selected PWTIN has toggled its states.
3 LVL	PWTIN Level when Overflows This Read Only bit signalizes the selected PWTIN states when the coutner overflows to read out.
2-0 PRE	PWT Clock Prescaler (CLKPRE) Setting Selects the value by which the clock is divided to clock the PWT counter. 000b - Clock divided by 1. 001b - Clock divided by 2. 010b - Clock divided by 4. 011b - Clock divided by 8. 100b - Clock divided by 16. 101b - Clock divided by 32. 110b - Clock divided by 64. 111b - Clock divided by 128.

36.7.1.4 Pulse Width Timer Positive Pulse Width (High) (PPH)

36.7.1.4.1 Offset

Register	Offset
PPH	2h

36.7.1.4.2 Diagram

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

36.7.1.4.3 Fields

Field	Function
7-0 PPWH	Positive Pulse Width[15:8] High byte of captured positive pulse width value.

36.7.1.5 Pulse Width Timer Positive Pulse Width (Low) (PPL)

36.7.1.5.1 Offset

Register	Offset
PPL	3h

36.7.1.5.2 Diagram

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

36.7.1.5.3 Fields

Field	Function
7-0 PPWL	Positive Pulse Width[7:0] Low byte of captured positive pulse width value.

36.7.1.6 Pulse Width Timer Negative Pulse Width (High) (NPH)

36.7.1.6.1 Offset

Register	Offset
NPH	4h

36.7.1.6.2 Diagram

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

36.7.1.6.3 Fields

Field	Function
7-0 NPWH	Negative Pulse Width[15:8] High byte of captured negative pulse width value.

36.7.1.7 Pulse Width Timer Negative Pulse Width (Low) (NPL)

36.7.1.7.1 Offset

Register	Offset
NPL	5h

36.7.1.7.2 Diagram

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

36.7.1.7.3 Fields

Field	Function
7-0 NPWL	Negative Pulse Width[7:0] Low byte of captured negative pulse width value.

36.7.1.8 Pulse Width Timer Counter (High) (CNTH)

36.7.1.8.1 Offset

Register	Offset
CNTH	6h

36.7.1.8.2 Diagram

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

36.7.1.8.3 Fields

Field	Function
7-0 PWTH	PWT counter[15:8] High byte of PWT counter register.

36.7.1.9 Pulse Width Timer Counter (Low) (CNTL)

36.7.1.9.1 Offset

Register	Offset
CNTL	7h

36.7.1.9.2 Diagram

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

36.7.1.9.3 Fields

Field	Function
7-0 PWTL	PWT counter[7:0] Low byte of PWT counter register.

36.8 Usage Guide

PWT provides an accurate signal frequency measurement for both the positive and negative portions of a periodic signal, useful for applications such as motor control. In conjunction with a Pulse Width Modulated signal it can effectively be used to implement a highly accurate closed loop motor control system, or any other system in which it might be necessary to measure a periodic signal frequency and duty cycle, providing not only accuracy but also high flexibility.

36.8.1 Edge detection, capture control and period measurement

PWT typical usage is external signal input capture and time period measurement.

Example: PWT input channel 1 capture external signal and measure its time period

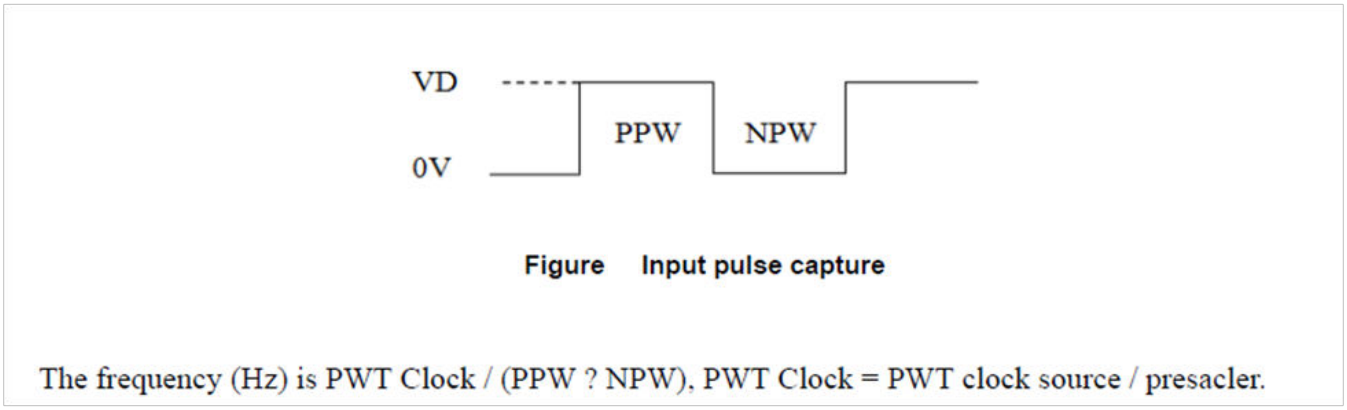


Figure Input pulse capture

The frequency (Hz) is PWT Clock / (PPW ? NPW), PWT Clock = PWT clock source / presacler.

- Enable the PWT module clock;
- Reset the timer channels and registers;
- Configure not to load the first counter values to corresponding registers, enable the PWT interrupt;
- Select bus clock as clock source and enable PWT_IN1 as input source;

- Set the module enable bit to start PWT;
- Wait for the pulse width valid flag (PWTRDY) in interrupt routinue, then get the positive and negative value(PPW, NPW) to calculate the period.

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(PWT);
PWT_CS|= PWT_CS_PWTSR_MASK;
PWT_CS|= PWT_CS_FCTLE(0)|PWT_CS_PWTIE_MASK|PWT_CS_PRDYIE_MASK;
PWT_CR|= PWT_CR_PCLKS(0)|PWT_CR_PRE(0)|PWT_CR_PINSEL(1);
PWT_CS|= PWT_CS_PWTEN_MASK;
EnableIRQ(PWT_IRQ);
```

Chapter 37
Low Power Timer (LPTMR)

37.1 Chip-specific information for this module

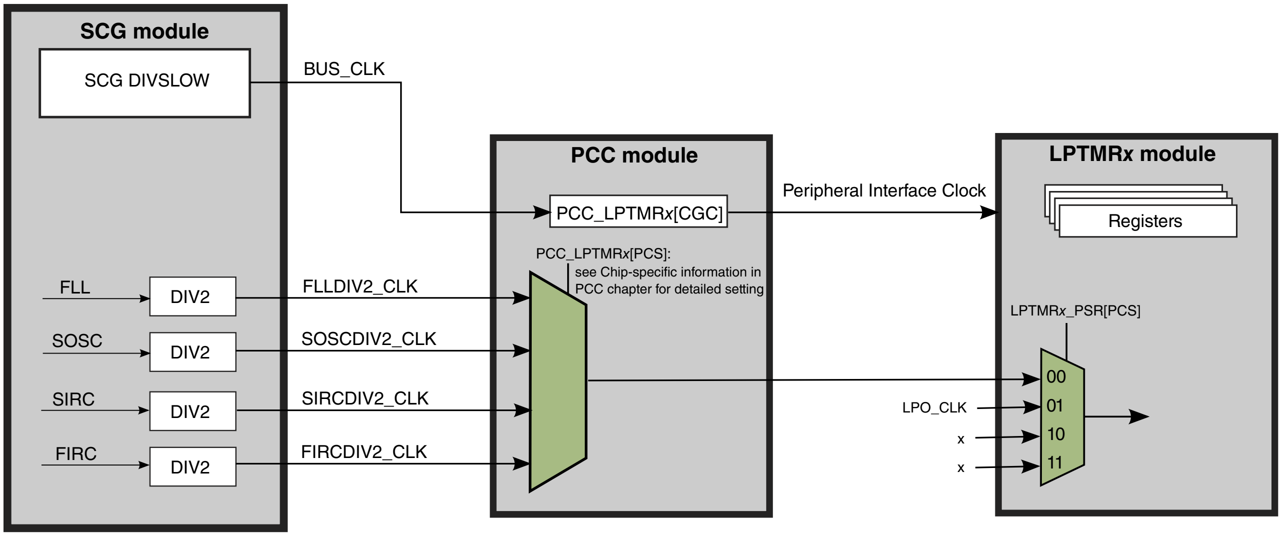
37.1.1 Instantiation Information

This device contains one LPTMR module with 1-channel, 16-bit pulse counter.

37.1.2 LPTMR Clocking Information

The following figure shows the input clock sources available for this module.

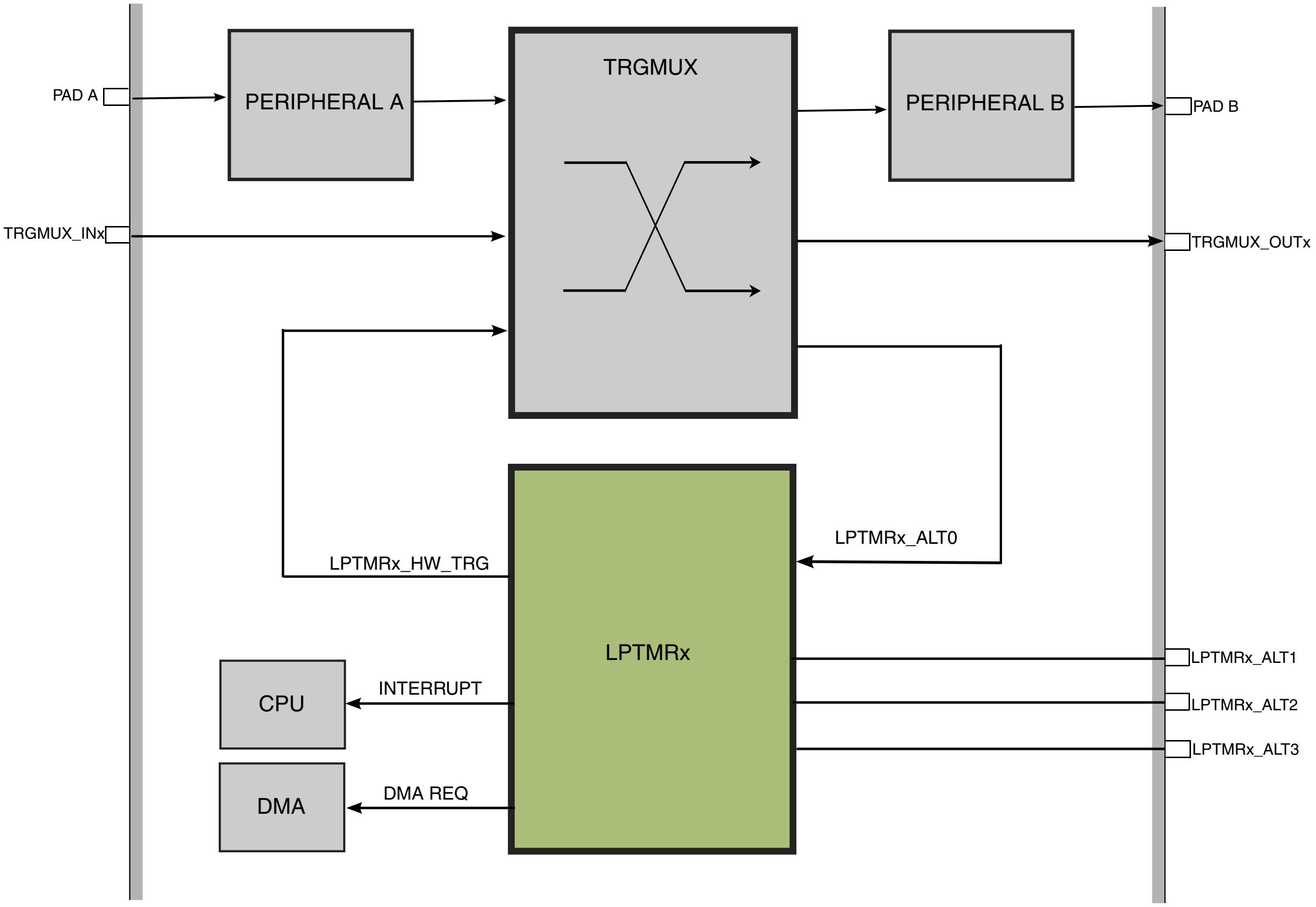
Peripheral Clocking - LPTMR



37.1.3 Inter-connectivity Information

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

LPTMRx_CSR[TPS]	Pulse counter input number	Chip input
00	0	TRGMUX output
01	1	LPTMR0_ALT1 pin
10	2	LPTMR0_ALT2 pin
11	3	LPTMR0_ALT3 pin



37.2 Overview

You can configure LPTMR to operate as a time counter with an optional prescaler, or as a pulse counter with an optional glitch filter, across all power modes, including low-power modes. It is reset only on POR or LVD, allowing it to be used as a time-of-day counter.

37.2.1 Block diagram

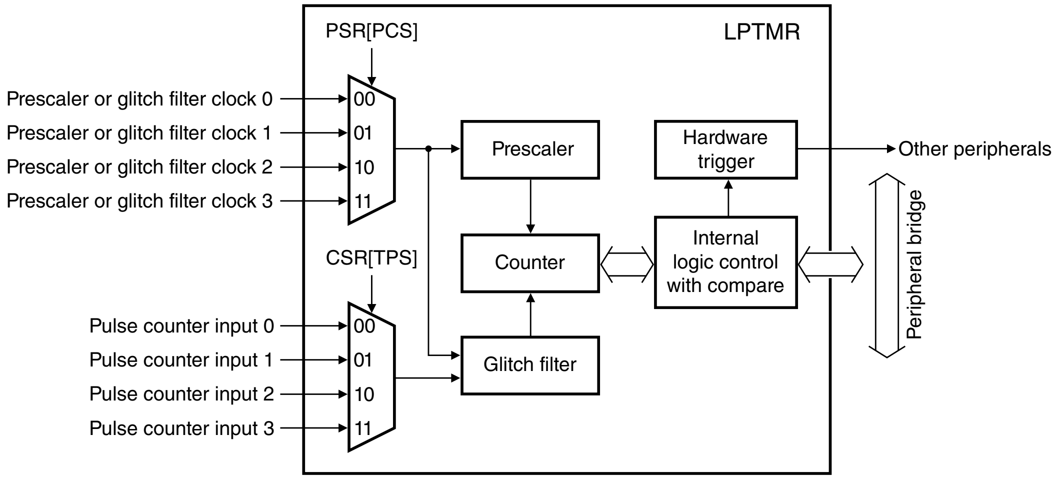


Figure 37-1. Block diagram

37.2.2 Features

- 16-bit time counter or pulse counter with compare:
  - Optional interrupt that can generate an asynchronous wake-up from any low-power mode
  - Hardware trigger output
  - Counter that supports a free-running mode or reset on compare
- Configurable clock source for prescaler and glitch filter
- Configurable input source for pulse counter (rising-edge or falling-edge)

37.3 Functional description

37.3.1 Low-power modes

In low-power modes, LPTMR continues to operate normally. You can configure LPTMR to exit a low-power mode by generating either an interrupt or a DMA request.

37.3.2 Clocks

The LPTMR prescaler and glitch filter can be clocked by one of the clocks that you configure by using PSR[PCS]. You must enable the clock source before you enable LPTMR.

In Pulse Counter mode, with the glitch filter bypassed, the selected input source directly clocks Counter (CNR), and no other clock source is required. To minimize power in this case, configure the glitch filter clock source for a clock that is disabled.

NOTE

- You may need to configure the clock source that you select in PSR[PCS] for it to remain enabled in low-power modes. Otherwise, LPTMR does not operate in low-power modes.
- The clock source or pulse input source selected for LPTMR must not exceed the maximum frequency of f_{LPTMR} defined in the chip data sheet.

37.3.3 Reset

LPTMR is reset only on POR or LVD. When configuring LPTMR registers, you must initially write to Control Status (CSR) with LPTMR disabled, before configuring Prescaler and Glitch Filter (PSR) and Compare (CMR). Then, you must write 1 to CSR[TEN] as the last step in the initialization. Doing so ensures that LPTMR is configured correctly and the LPTMR counter is reset to zero following a POR or LVD.

37.3.4 Prescaler and 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

You must not alter the prescaler and glitch filter configuration when LPTMR is enabled.

37.3.4.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks Counter (CNR). When LPTMR is enabled, CNR increments every 2^1 to 2^{16} prescaler clock cycles. After LPTMR is enabled, the first increment of CNR takes an additional one or two prescaler clock cycles because of the synchronization logic.

37.3.4.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments Counter (CNR) on every clock cycle. When LPTMR is enabled, the first increment takes an additional one or two prescaler clock cycles because of the synchronization logic.

37.3.4.3 Glitch filter enabled

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks Counter (CNR). When 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.

Table 37-1. 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 also deasserts.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output also asserts.

NOTE

The input is sampled only on the rising clock edge.

The value of CNR increments each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which CNR can increment is once every 2^2 to 2^{16} glitch filter clock edges. When first enabled, the glitch filter waits for an additional one or two glitch filter clock edges because of the synchronization logic.

37.3.4.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the value of Counter (CNR) every time it asserts. Before LPTMR is first enabled, the selected input source is forced to be asserted. This prevents CNR from incrementing if the selected input source is already asserted when LPTMR is first enabled.

37.3.5 Counter

The value of Counter (CNR) increments by 1 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.

CNR is reset when LPTMR is disabled or if the counter register overflows. If CSR[TFC] = 0, then CNR is also reset whenever CSR[TCF] = 1.

When the core is halted in Debug mode:

- CNR continues incrementing if configured for Pulse Counter mode.
- CNR stops incrementing if configured for Time Counter mode.

You cannot initialize CNR but can read it at any time. On each read of CNR, you must first write a value to it. This synchronizes and registers the current value of CNR into a temporary register. The contents of the temporary register are returned on each read of CNR.

When reading 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.

37.3.6 Compare

After the next Counter (CNR) increment (when its value is equal to that of Compare (CMR)), the following events occur:

- CSR[TCF] is read as 1b.
- LPTMR generates an interrupt if CSR[TIE] is 1 as well.
- LPTMR generates a hardware trigger.
- LPTMR writes 0 to CNR if CSR[TFC] is 0.

When LPTMR is enabled, you can modify the value of CMR only when CSR[TCF] is 1. When updating CMR, you must write to it and clear CSR[TCF] before the LPTMR counter increments past the new LPTMR compare value.

NOTE

When LPTMR is enabled in Time Counter mode, the first increment takes an additional one or two clock cycles because of the synchronization logic. This results in the first compare (and therefore interrupt and hardware trigger) occurring slightly later. A faster prescaler clock or larger prescaler value minimizes this impact.

37.3.7 Interrupt

LPTMR generates an interrupt whenever CSR[TIE] and CSR[TCF] are 1. CSR[TCF] is cleared by disabling LPTMR or writing a logic 1 to it.

You can modify the value of CSR[TIE] and write 1 to CSR[TCF] when LPTMR is enabled.

LPTMR generates an interrupt asynchronously to the system clock. The interrupt can be used to generate a wake-up from any low-power mode, provided LPTMR is enabled as a wake-up source.

37.3.8 Hardware trigger

The LPTMR hardware trigger asserts at the same time CSR[TCF] is set and can be used to trigger hardware events in other peripherals without your intervention. The hardware trigger is always enabled.

Table 37-2. Hardware trigger

When	Then
CMR[COMPARE] and CSR[TFC] are 0	The LPTMR hardware trigger asserts on the first compare and does not deassert.
CMR[COMPARE] is set to a nonzero value, or if CSR[TFC] = 1	The LPTMR hardware trigger asserts on each compare and deasserts on the following increment of Counter (CNR).

37.4 External signals

Table 37-3. External signals

Signal	Description		Direction
LPTMR_ALT*n*	Pulse Counter Input LPTMR can select one of the input pins to be used in Pulse Counter mode.		Input
	State meaning	Assertion—If configured for Pulse Counter mode with an active-high input, assertion causes Counter (CNR) to increment. Deassertion—If configured for Pulse Counter mode with an active-low input, deassertion causes CNR to increment.	
	Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.	

37.5 Initialization

Perform the following procedure to initialize LPTMR:

1. Configure Control Status (CSR) for the selected mode and pin configuration, when CSR[TEN] is 0. This resets the counter and clears the flag.
2. Configure Prescaler and Glitch Filter (PSR) with the selected clock source and prescaler or glitch filter configuration.
3. Configure Compare (CMR) with the selected compare point.
4. Write 1 to CSR[TEN] to enable LPTMR.

37.6 Application information

37.6.1 Application 1: Generate an interrupt every 100 ms using 32.768 kHz clock source

1. Disable LPTMR by writing 0 to CSR[TEN].
2. Select a 32.768 kHz clock source by configuring PSR[PCS].
3. Bypass the prescaler and glitch filter by writing 1 to PSR[PBYP].
4. Assert an interrupt every 3277 cycles by configuring CMR[COMPARE] = 0CCCh.
5. Enable LPTMR by writing 1 to CSR[TEN].
6. Enable the LPTMR interrupt by writing 1 to CSR[TIE].

NOTE

This is just an example. See the chip-specific LPTMR information for the clocks supported on a given chip.

37.6.2 Application 2: Generate an interrupt once a minute using 32.768 kHz clock source

1. Disable LPTMR by writing 0 to CSR[TEN].
2. Select a 32.768 kHz clock source by configuring PSR[PCS].
3. Select the prescaler to divide the prescaler clock by 32768 to increment the counter once a second by configuring PSR[PRESCALE] = 0Eh.
4. Assert an interrupt every 60 seconds by configuring CMR[COMPARE] = 003Bh.
5. Enable LPTMR by writing 1 to CSR[TEN].
6. Enable the LPTMR interrupt by writing 1 to CSR[TIE].

NOTE

This is just an example. See the chip-specific LPTMR information for the clocks supported on a given chip.

37.7 Memory map and register definition

NOTE

The LPTMR registers are reset only on POR or LVD. See Reset for more information.

37.7.1 LPTMR register descriptions

37.7.1.1 LPTMR memory map

LPTMR0 base address: 4004_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Status (CSR)	32	RW	0000_0000h
4h	Prescaler and Glitch Filter (PSR)	32	RW	0000_0000h
8h	Compare (CMR)	32	RW	0000_0000h
Ch	Counter (CNR)	32	RW	0000_0000h

37.7.1.2 Control Status (CSR)

37.7.1.2.1 Offset

Register	Offset
CSR	0h

37.7.1.2.2 Function

Controls various features of LPTMR.

37.7.1.2.3 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							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

37.7.1.2.4 Fields

Field	Function
31-9 —	Reserved
8 TDRE	Timer DMA Request Enable Enables the timer DMA request. When TDRE is 1, the LPTMR DMA request is generated whenever CSR[TCF] is also set. Then, CSR[TCF] is cleared after the DMA controller completes execution. 0b - Disable 1b - Enable
7 TCF	Timer Compare Flag Compares the timer. TCF sets on the next Counter (CNR) increment when LPTMR is enabled and Counter (CNR) equals Compare (CMR). TCF is cleared when LPTMR is disabled or a logic 1 is written to it.

Table continues on the next page...

Field	Function
	NOTE: You must clear this flag before enabling the timer interrupt or DMA request. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - CNR ≠ (CMR + 1) 1b - CNR = (CMR + 1) When writing 0b - No effect 1b - Clear the flag
6 TIE	Timer Interrupt Enable Enables the timer interrupt. If TIE is 1, then LPTMR generates an interrupt if CSR[TCF] is 1 as well. 0b - Disable 1b - Enable
5-4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. The input connections vary by chip. For details, see the chip configuration information about connections to these inputs. You must modify this field only when LPTMR is disabled. 00b - Input 0 01b - Input 1 10b - Input 2 11b - Input 3
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. If TPP is 0, then the pulse counter input source is active-high, and Counter (CNR) increments on the rising edge. If TPP is 1, then the pulse counter input source is active-low, and CNR increments on the falling edge. You must modify this field only when LPTMR is disabled. 0b - Active-high 1b - Active-low
2 TFC	Timer Free-Running Counter Specifies when the counter resets. If TFC is 0, Counter (CNR) resets on the count cycle following Counter (CNR) becoming equal to Compare (CMR). If TFC is 1, CNR resets on overflow. In both cases, CSR[TCF] sets to 1 on the cycle after CNR and CMR match. You must modify this field only when LPTMR is disabled. 0b - Reset when TCF asserts 1b - Reset on overflow
1 TMS	Timer Mode Select Configures the mode of LPTMR. You must modify this field only when LPTMR is disabled. 0b - Time Counter 1b - Pulse Counter
0 TEN	Timer Enable Enables the LPTMR timer. If TEN is 0, it resets the LPTMR internal logic, including CNR[COUNTER] and CSR[TCF]. If TEN is 1, LPTMR is enabled. Do not alter CSR[5:1] when writing 1 to this field. 0b - Disable

Field	Function
	1b - Enable

37.7.1.3 Prescaler and Glitch Filter (PSR)

37.7.1.3.1 Offset

Register	Offset
PSR	4h

37.7.1.3.2 Function

Configures features related to the prescaler and glitch filter.

37.7.1.3.3 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									PRESCALE				PBYP	PCS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

37.7.1.3.4 Fields

Field	Function
31-7 —	Reserved
6-3 PRESCALE	Prescaler and Glitch Filter Value Configures the size of the prescaler in Time Counter mode and the width of the glitch filter in Pulse Counter mode. The width of the glitch filter can vary by one cycle because of the pulse counter input synchronization.

Table continues on the next page...

Field	Function
	You must modify this field only when LPTMR is disabled. 0000b - Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration 0001b - Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after two rising clock edges 0010b - Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after four rising clock edges 0011b - Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after eight rising clock edges 0100b - Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges 0101b - Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges 0110b - Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges 0111b - Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges 1000b - Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges 1001b - Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges 1010b - Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges 1011b - Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges 1100b - Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges 1101b - Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges 1110b - Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges 1111b - Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges
2 PBYP	Prescaler and Glitch Filter Bypass Controls the clocking of Counter (CNR). If PBYP is 0, the output of the prescaler or glitch filter clocks CNR. If PBYP is 1, the selected prescaler clock in Time Counter mode, or else the selected input source in Pulse Counter mode, directly clocks CNR. You must modify this field only when LPTMR is disabled. 0b - Prescaler and glitch filter enable 1b - Prescaler and glitch filter bypass
1-0 PCS	Prescaler and Glitch Filter Clock Select Selects the clock to be used by the LPTMR prescaler and glitch filter. In Time Counter mode, PCS selects the input clock to the prescaler. In Pulse Counter mode, PCS selects the input clock to the glitch filter. See the chip configuration details for information on connections to these inputs. You must modify this field only when LPTMR is disabled. 00b - Clock 0 01b - Clock 1 10b - Clock 2 11b - Clock 3

37.7.1.4 Compare (CMR)

37.7.1.4.1 Offset

Register	Offset
CMR	8h

37.7.1.4.2 Function

Configures the compare values to Counter (CNR) (see Compare for more information).

37.7.1.4.3 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 W	COMPARE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

37.7.1.4.4 Fields

Field	Function
31-16 —	Reserved
15-0 COMPARE	Compare Value Configures the compare values to Counter (CNR). On the next CNR increment, if LPTMR is enabled and Counter (CNR) equals Compare (CMR), then: 1. LPTMR writes 1 to CSR[TCF]. 2. The hardware trigger asserts until the next time CNR increments. If CMR = 0, the hardware trigger remains asserted until LPTMR is disabled. If LPTMR is enabled, you must modify the value of CMR only if CSR[TCF] is 1.

37.7.1.5 Counter (CNR)

37.7.1.5.1 Offset

Register	Offset
CNR	Ch

37.7.1.5.2 Function

Specifies counter values (see Counter for more information).

37.7.1.5.3 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 W	COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

37.7.1.5.4 Fields

Field	Function
31-16 —	Reserved
15-0 COUNTER	Counter Value Contains the current value of the LPTMR counter at the time you last wrote to this register.

37.8 Usage Guide

LPTMR is very useful in low power situations. It can be used as a wake-up timer to wake the MCU out of sleep modes after a certain amount of time. If used as pulse counter mode with the glitch filter enabled, then there is no need for a clock to be on. The MCU can wakeup based on counting pulses.

37.8.1 Time Counter mode

The typical usage of LPTMR is as Time Counter mode to generate periodic trigger pulses and interrupts.

Example: LPTMR trigger a periodic interrupt every 1 second

- Enable the LPTMR module clock;
- Configure LPTMR to Timer counter mode by default, use LPO 128K as clock source, bypass the prescaler;
- Set the compare value register to 1 second value;
- Enable timer interrupt ;
- Starts the timer counting after all configuration;
- In the interrupt routine, clear the channel compare flag TCF every 1 second.

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(LPTMR0);
LPTMR0_CSR = 0;
LPTMR0_PSR |= LPTMR_PSR_PBYP_MASK|LPTMR_PSR_PCS(1);
LPTMR0_CMR = ONE_SECOND_VALUE;
LPTMR0_CSR |= LPTMR_CSR_TIE_MASK;
EnableIRQ(LPTMR0_IRQn);
LPTMR0_CSR |= LPTMR_CSR_TEN_MASK;
```

37.8.2 Pulse Counter mode

LPTMR another option is used as Pulse Counter mode to count the input pulses.

Example: LPTMR count the input pulses on LPTMR0_ALT1 pin

- Enable the LPTMR module clock;
- Configure LPTMR to Pulse counter mode, use LPO 128K as clock source, bypass the glitch filter
- Set the compare value register to the value you want to compare the numbers of pulse
- Enable the pulse counter input enable on LPTMR0_ALT1
- Enable timer interrupt
- Starts the pulse counting after all configuration;
- In the interrupt routine, clear the channel compare flag TCF when the counter reaches the value in compare register;

The following pseudo-code matches the described setup above:

```
CLOCK_EnableClock(LPTMR0);
LPTMR0_CSR |= LPTMR_CSR_TPS(1)|LPTMR_CSR_TMS_MASK;
LPTMR0_PSR |= LPTMR_PSR_PBYP_MASK|LPTMR_PSR_PCS(1);
LPTMR0_CMR = PULSE_COMPARE_VALUE;
LPTMR0_CSR |= LPTMR_CSR_TIE_MASK;
EnableIRQ(LPTMR0_IRQn);
LPTMR0_CSR |= LPTMR_CSR_TEN_MASK;
```

Chapter 38
Real Time Clock (RTC)

38.1 Chip-specific information for this module

38.1.1 RTC Instantiation

NOTE

There is no integrated capacitor for this device, therefore no tunable capacitors (included in the crystal oscillator) can be configured by software.

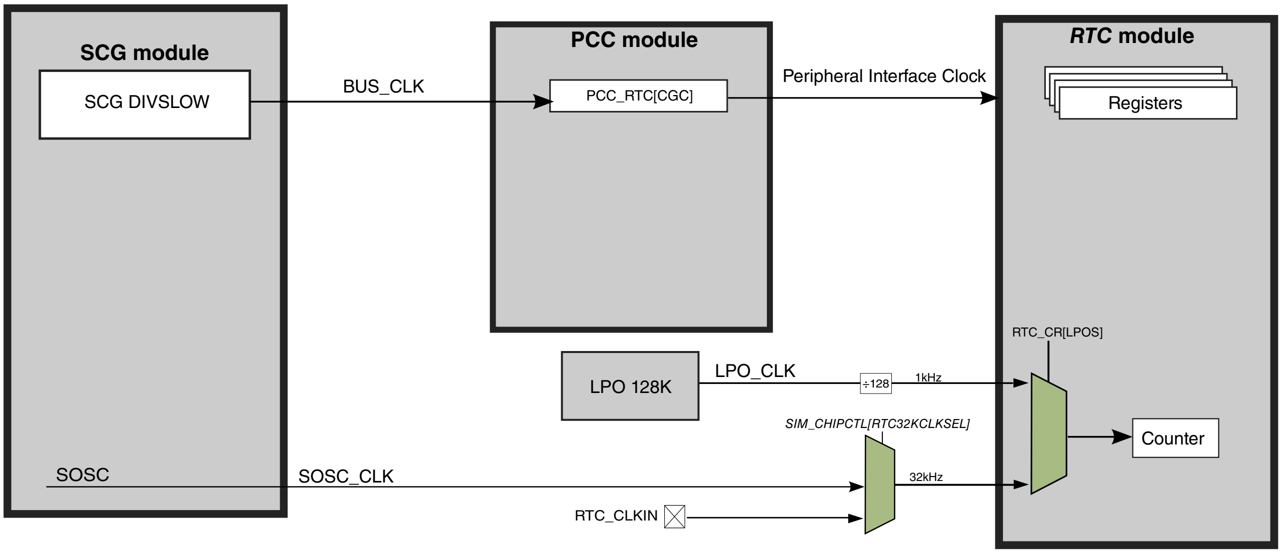
38.1.2 RTC Clocking Information

The following figure shows the input clock sources available for this module.

NOTE

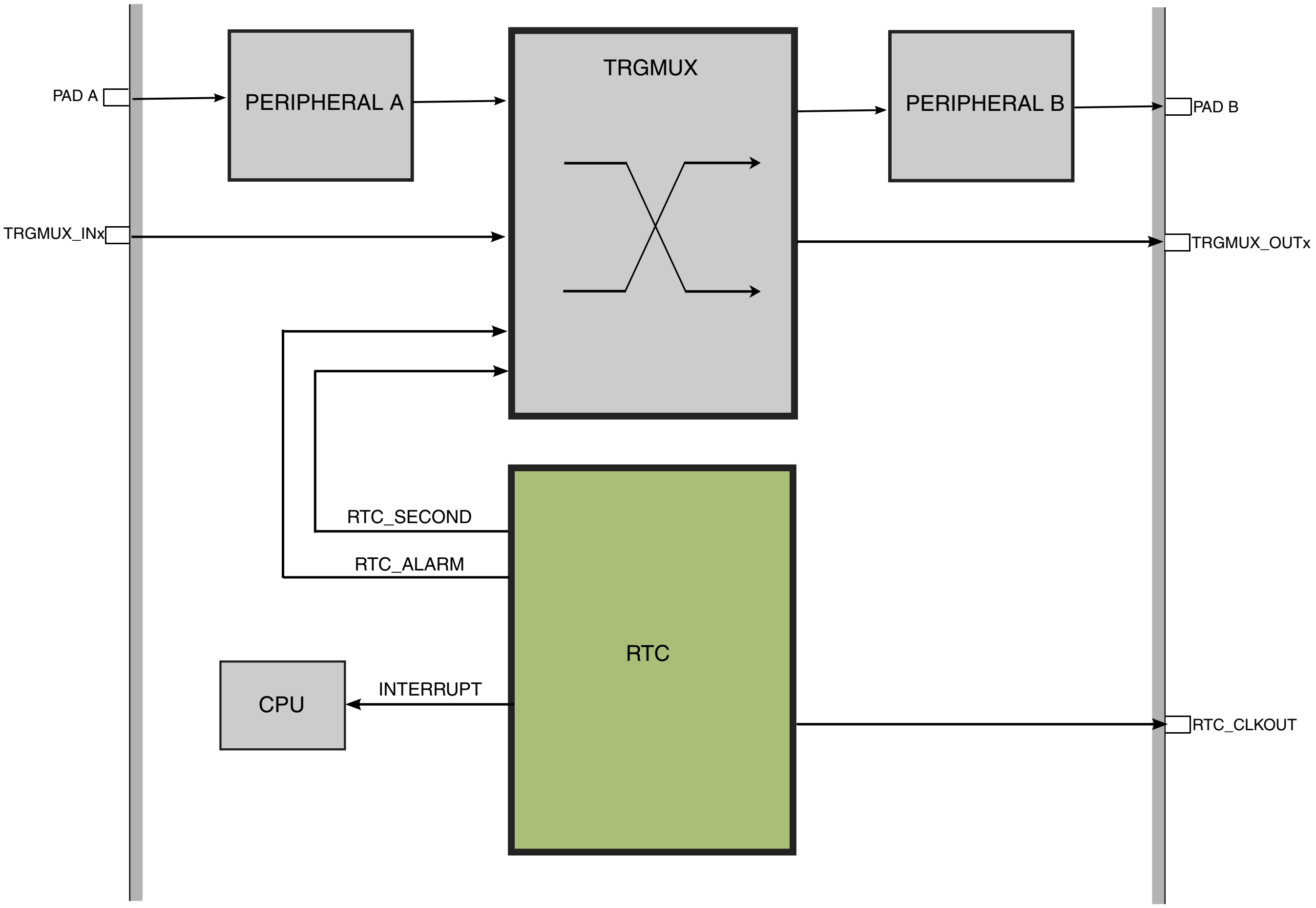
No 32 kHz crystal in this device. See the clocking figure below, for more details about RTC clock source.

Peripheral Clocking - RTC



38.1.3 Inter-connectivity Information

The RTC inter-connectivity is shown in following diagram.



38.2 Overview

38.2.1 Block diagram

The following figure is the block diagram of this module.

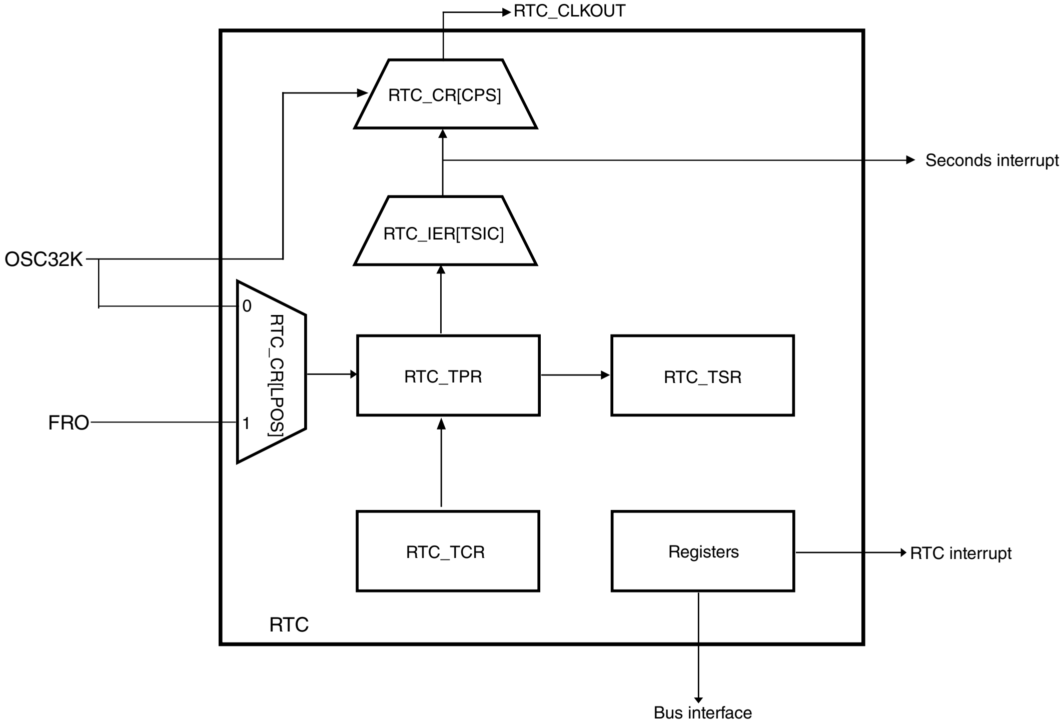


Figure 38-1. RTC module block diagram

38.2.2 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 a 1 kHz 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

38.3 Functional description

38.3.1 Power

The RTC is an always powered block that remains active in all low power modes.

38.3.2 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, SR[TOF] is set and the time prescaler register stops 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.

38.3.3 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 do 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.

38.3.4 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] is set when TAR equals TSR and TSR increments.

SR[TAF] is cleared by writing TAR. This is usually the next alarm value, although writing a value that is less than TSR (such as 0) prevents SR[TAF] from setting again. SR[TAF] cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

38.3.5 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].

38.3.6 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) disables the software reset. Locking LR blocks future updates to LR.

Write accesses to a locked register are ignored and do not generate a bus error.

38.3.7 Modes of operation

The RTC remains functional in all low power modes and can generate an interrupt to exit any low power mode.

38.3.8 Clocking

The time counter within the RTC is clocked by default from a 32.768 kHz clock. Alternatively, the time counter can be clocked by a LPO 1 kHz clock and the prescaler increments by 32 for each LPO clock.

The 32.768 kHz crystal oscillator is disabled at POR and must be enabled by software. After enabling the crystal 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.

38.3.9 Reset

The power-on-reset signal initializes all RTC registers to their default state. A software reset bit can also initialize all RTC registers.

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.

38.3.10 Interrupts

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 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.

38.4 External signals

Table 38-1. RTC external signals description

Signal	Description	I/O
RTC_CLKOUT	Prescaler square-wave output or RTC 32.768 kHz clock	O

38.4.1 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 RTC 32.768 kHz clock.

38.5 Initialization

To initialize the RTC time counter:

1. Enable the desired clock source for the RTC and configure RTC_CR[LPOS] to select that clock.
2. Initialize the time prescaler register by writing RTC_TPR[TPR] with the appropriate time value between 0x0 and 0x7FFF_FFF.
3. Initialize the time second register by writing RTC_TSR[TSR] with the appropriate time value that is greater than 0x0.
4. Enable the time counter by writing RTC_SR[SR]=0x10.

38.6 Register descriptions

All registers must be accessed using 32-bit writes and all register accesses incur three wait states.

When the supervisor access control bit is clear, only supervisor mode software can write to the RTC registers. Writes to the RTC registers by non-supervisor mode software generate a bus error. Both supervisor and non-supervisor mode software can always read the RTC registers.

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 writes can not be completed.

38.6.1 RTC register descriptions

38.6.1.1 RTC memory map

RTC base address: 4003_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RTC Time Seconds (TSR)	32	RW	0000_0000h
4h	RTC Time Prescaler (TPR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8h	RTC Time Alarm (TAR)	32	RW	0000_0000h
Ch	RTC Time Compensation (TCR)	32	RW	0000_0000h
10h	RTC Control (CR)	32	RW	0000_0000h
14h	RTC Status (SR)	32	RW	0000_0001h
18h	RTC Lock (LR)	32	RW	0000_00FFh
1Ch	RTC Interrupt Enable (IER)	32	RW	0000_0007h

38.6.1.2 RTC Time Seconds (TSR)

38.6.1.2.1 Offset

Register	Offset
TSR	0h

38.6.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TSR															
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	TSR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.6.1.2.3 Fields

Field	Function
31-0 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] is not set. The time counter reads as zero when SR[TOF] or SR[TIF] is set. When the time counter is disabled, the TSR can be read or written. Writing to the TSR when the time counter is disabled clears SR[TOF] or SR[TIF]. Writing to TSR with zero is supported, but not recommended because TSR reads as zero when SR[TIF] or SR[TOF] is set (indicating the time is invalid).

38.6.1.3 RTC Time Prescaler (TPR)

38.6.1.3.1 Offset

Register	Offset
TPR	4h

38.6.1.3.2 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 W	TPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.6.1.3.3 Fields

Field	Function
31-16 —	Reserved
15-0 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 reads 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.

38.6.1.4 RTC Time Alarm (TAR)

38.6.1.4.1 Offset

Register	Offset
TAR	8h

38.6.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TAR															
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	TAR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.6.1.4.3 Fields

Field	Function
31-0 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].

38.6.1.5 RTC Time Compensation (TCR)

38.6.1.5.1 Offset

Register	Offset
TCR	Ch

38.6.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CIC								TCV							
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 W	CIR								TCR							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.6.1.5.3 Fields

Field	Function
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. Reading this register at the same time as the seconds counter is incrementing can result in an incorrect value being read.
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). Reading this register at the same time as the seconds counter is incrementing can result in an incorrect value being read.
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.
7-0 TCR	Time Compensation Register Configures the number of 32.768 kHz clock cycles in each second, equal to 32,768 - TCR (where TCR is a twos complement sign extended value). This register is double buffered and writes do not take affect until the end of the current compensation interval. Some example values are show below. 0000_0000b - Time Prescaler Register overflows every 32768 clock cycles. 0000_0001b - Time Prescaler Register overflows every 32767 clock cycles. 0111_1110b - Time Prescaler Register overflows every 32642 clock cycles. 0111_1111b - Time Prescaler Register overflows every 32641 clock cycles. 1000_0000b - Time Prescaler Register overflows every 32896 clock cycles. 1000_0001b - Time Prescaler Register overflows every 32895 clock cycles. 1111_1111b - Time Prescaler Register overflows every 32769 clock cycles.

38.6.1.6 RTC Control (CR)

38.6.1.6.1 Offset

Register	Offset
CR	10h

38.6.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						Reserved	CPE	0						Reserved	
W							0								0	
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	Reserved	Reserved	Reserved	Reserved	Reserved	CLKO	Reserved	LPOS	0	CPS	Reserved	UM	SUP	Reserved	SWR
W	0	0	0	0	0	0		0				0			0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.6.1.6.3 Fields

Field	Function
31-26 —	Reserved
25 —	Reserved
24 CPE	Clock Pin Enable Specifies whether to enable or disable RTC_CLKOUT: 0b - Disables 1b - Enables
23-18 —	Reserved
17-16 —	Reserved
15 —	Reserved
14	Reserved

Table continues on the next page...

Field	Function
—	
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 CLKO	Clock Output 0b - The 32 kHz clock is output to other peripherals. 1b - The 32 kHz clock is not output to other peripherals.
8 —	Reserved
7 LPOS	LPO Select When set, the RTC prescaler increments using the LPO 1 kHz clock and not the RTC 32.768 kHz 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). 0b - RTC prescaler increments using 32.768 kHz clock. 1b - RTC prescaler increments using 1 kHz LPO, bits [4:0] of the prescaler are ignored.
6 —	Reserved
5 CPS	Clock Pin Select 0b - The prescaler output clock (as configured by TSIC) is output on RTC_CLKOUT. 1b - The RTC 32.768 kHz clock is output on RTC_CLKOUT, provided it is output to other peripherals.
4 —	Reserved
3 UM	Update Mode Allows SR[TCE] to be written even when the Status Register is locked. When set, the SR[TCE] can be written if the SR[TIF] or SR[TOF] are set or if the SR[TCE] is clear. Whenever both SR[TCE] and CR[UM] are set, then SR[TCE] should only be written once either SR[TIF] or SR[TOF] are set. 0b - Registers cannot be written when locked. 1b - Registers can be written when locked under limited conditions.
2 SUP	Supervisor Access 0b - Non-supervisor mode write accesses are not supported and a bus error is generated. 1b - Non-supervisor mode write accesses are supported.
1 —	Reserved
0 SWR	Software Reset 0b - No effect. 1b - Resets all RTC registers except for the SWR bit . The SWR bit is cleared by POR and by software explicitly clearing it.

38.6.1.7 RTC Status (SR)

38.6.1.7.1 Offset

Register	Offset
SR	14h

38.6.1.7.2 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								0	0		TCE	0	TAF	TOF	TIF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

38.6.1.7.3 Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6-5 —	Reserved
4 TCE	Time Counter Enable Specifies whether to enable or disable time counter: 0b - Disables. When time counter is disabled the TSR register and TPR register are writeable, but do not increment. 1b - Enables. When time counter is enabled the TSR register and TPR register are not writeable, but increment.
3 —	Reserved
2 TAF	Time Alarm Flag • Is set when the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments.

Table continues on the next page...

Field	Function
	• Is cleared by writing the TAR register. • Indicates whether time alarm has occured: 0b - Time alarm has not occurred. 1b - Time alarm has occurred.
1 TOF	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. • Is cleared by writing the TSR register when the time counter is disabled. • Indicates whether time overflow has occured: 0b - Time overflow has not occurred. 1b - Time overflow has occurred and time counter reads 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. • Indicates whether the time is valid or invalid: 0b - Time is valid. 1b - Time is invalid and time counter is read as zero.

38.6.1.8 RTC Lock (LR)

38.6.1.8.1 Offset

Register	Offset
LR	18h

38.6.1.8.2 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								1	LRL	SRL	CRL	TCL	1		
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

38.6.1.8.3 Fields

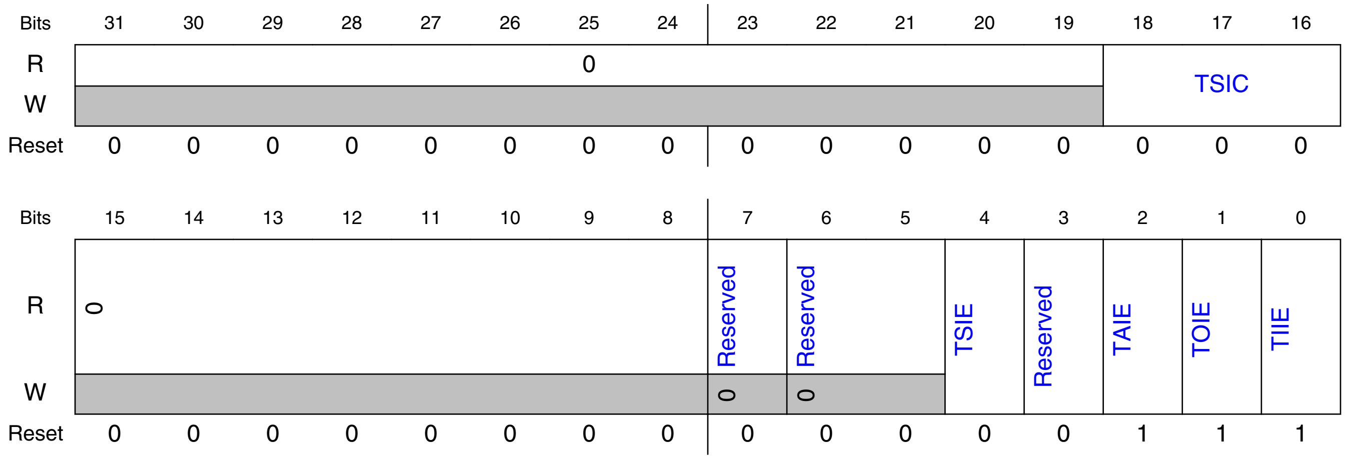
Field	Function
31-8 —	Reserved
7 —	Reserved
6 LRL	Lock Register Lock After being cleared, this bit can be set only by POR or software reset. 0b - Lock Register is locked and writes are ignored. 1b - Lock Register is not locked and writes complete as normal.
5 SRL	Status Register Lock After being cleared, this bit can be set only by POR or software reset. 0b - Status Register is locked and writes are ignored. 1b - 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. 0b - Control Register is locked and writes are ignored. 1b - 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. 0b - Time Compensation Register is locked and writes are ignored. 1b - Time Compensation Register is not locked and writes complete as normal.
2-0 —	Reserved

38.6.1.9 RTC Interrupt Enable (IER)

38.6.1.9.1 Offset

Register	Offset
IER	1Ch

38.6.1.9.2 Diagram



38.6.1.9.3 Fields

Field	Function
31-19 —	Reserved
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. 000b - 1 Hz. 001b - 2 Hz. 010b - 4 Hz. 011b - 8 Hz. 100b - 16 Hz. 101b - 32 Hz. 110b - 64 Hz. 111b - 128 Hz.
15-8 —	Reserved
7 —	Reserved
6-5 —	Reserved
4 TSIE	Time Seconds Interrupt Enable • 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. • Specifies whether to enable or disable the time seconds interrupt: 0b - Disables 1b - Enables
3	Reserved

Table continues on the next page...

Field	Function
—	
2 TAIE	Time Alarm Interrupt Enable Specifies whether an interrupt is generated by the time alarm flag: 0b - No interrupt is generated. 1b - An interrupt is generated.
1 TOIE	Time Overflow Interrupt Enable Specifies whether an interrupt is generated by the time overflow flag: 0b - No interrupt is generated. 1b - An interrupt is generated.
0 TIIE	Time Invalid Interrupt Enable Specifies whether an interrupt is generated by the time invalid flag: 0b - No interrupt is generated. 1b - An interrupt is generated.

38.7 Usage Guide

38.7.1 Clock source information

To get an accuracy clock for RTC, an external 32.768 kHz crystal should be connected to EXTAL32/XTAL32 pin, or a 32.768 kHz clock signal to RTC_CLKIN pin. Alternatively, the time counter can be clocked by the LPO 1 kHz and the prescaler will increment by 32 for each LPO clock, which is not that precisely.

38.7.2 Usage examples

This section shows the application examples of initializing the RTC module, setting the data time and alarm.

RTC Module Initialization

The RTC module is reset by a POR or a software reset (The access control registers are not affected by either VBAT POR or the software reset).

Before using the RTC module, a software reset is recommend by setting the RTC_CR[SWR] bit. And the 32.768 kHz external crystal should be enabled to provide clock to RTC.

```
// Reset the RTC by set RTC_CR[SWR] bit, and wait
// for the TIF flag cleared by writing the TSR
while (RTC_SR & RTC_SR_TIF_MASK)
{
    RTC_CR |= RTC_CR_SWR_MASK;
    RTC_CR &= ~RTC_CR_SWR_MASK;
    RTC_TSR = 1;
}

// Setup the update mode and supervisor access mode
// enable 32.768 kHz oscillator timer
RTC_CR = RTC_CR_CPE(0) | RTC_CR_LPOS(0) | RTC_CR_CPS(1) |
         RTC_CR_UM(0) | RTC_CR_SUP(0)| RTC_CR_OSCE(1);
// disable all the interrupts first
RTC_IER = 0;
// stop timer first
RTC_SR &= ~RTC_SR_TCE_MASK;
```

Set Date Time

After RTC initialized, user can set the date time before starting the timer. Please make sure the timer is stopped when setting the date time by RTC_TSR register.

```
// stop timer first
RTC_SR &= ~RTC_SR_TCE_MASK;
// convert the date time to secs first, then write to RTC_TSR register
RTC_TSR = datetime_in_secs;
// start the timer
RTC_SR |= RTC_SR_TCE_MASK;
```

Set Alarm

To set an alarm and trigger alarm interrupt, user should enable the alarm interrupt, write the alarm seconds into RTC_TAR.

```
uint32_t datetime_in_secs;

// assume the timer is running
// enable the interrupt
RTC_IER |= RTC_IER_TAIE_MASK;
// enable the RTC IRQ in NVIC
NVIC_EnableIRQ(RTC_IRQn);

// get the current date time in secs
datetime_in_secs = RTC_TSR;
datatime_in_secs += 10;
// set alarm 10s later
RTC_TAR = datetime_in_secs;
```

After 10 seconds, the RTC Alarm IRQ would be triggered and IRQ Handler called. In the IRQ Handler, user should first clear the interrupt status:

```
if (RTC_SR & RTC_SR_TAF_MASK)
{
    // clear the TAF flag by writing the RTC_TAR register
    RTC_TAR = 0;
}
// Then doing the alarm task in this IRQ Handler
………
```

38.7.3 RTC_CLKOUT signal

When the RTC is enabled and the port control module selects the RTC_CLKOUT function, the RTC_CLKOUT signal output either a square wave prescaler output (configurable to 1, 2, 4, 8, 16, 32, 64 or 128 Hz) or 32 kHz output derived from RTC oscillator as shown below.

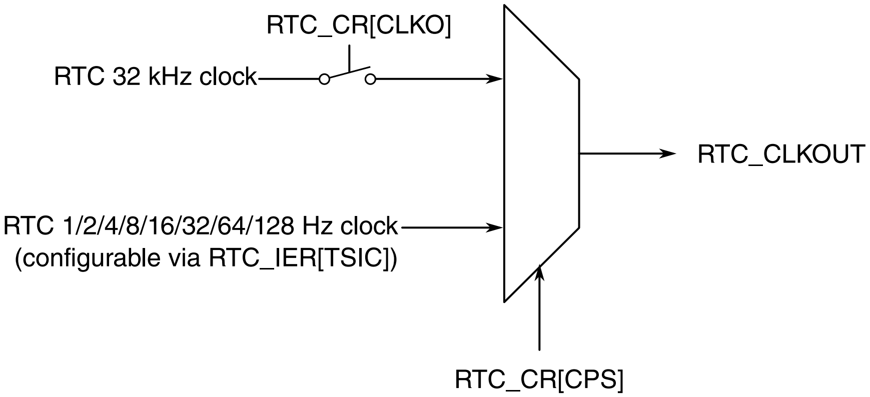

RTC_CR[CLKO]
RTC 32 kHz clock
RTC_CLKOUT
RTC 1/2/4/8/16/32/64/128 Hz clock
(configurable via RTC_IER[TSIC])
RTC_CR[CPS]

Figure 38-2. RTC_CLKOUT generation

NOTE

When using LPO 1kHz as RTC clock source, it cannot directly output to pad. But RTC can normally output 1/2/4/8/…/64/128 Hz clock using prescaler.

Chapter 39
Low Power Serial Peripheral Interface (LPSPI)

39.1 Chip-specific information for this module

39.1.1 Instantiation Information

This device contains two LPSPI modules. The LPSPI can remain functional in Stop and VLPS mode provided the clock it is using remains enabled.

Table 39-1. LPSPI Configuration

	TX FIFO (word/32bit)	RX FIFO (word/32bit)	Chip Selects
LPSPI0	4	4	4
LPSPI1	4	4	4

NOTE

The TX/RX FIFO "word" does not refer to system bus width 32-bit, and it varies for different communication module. For example:

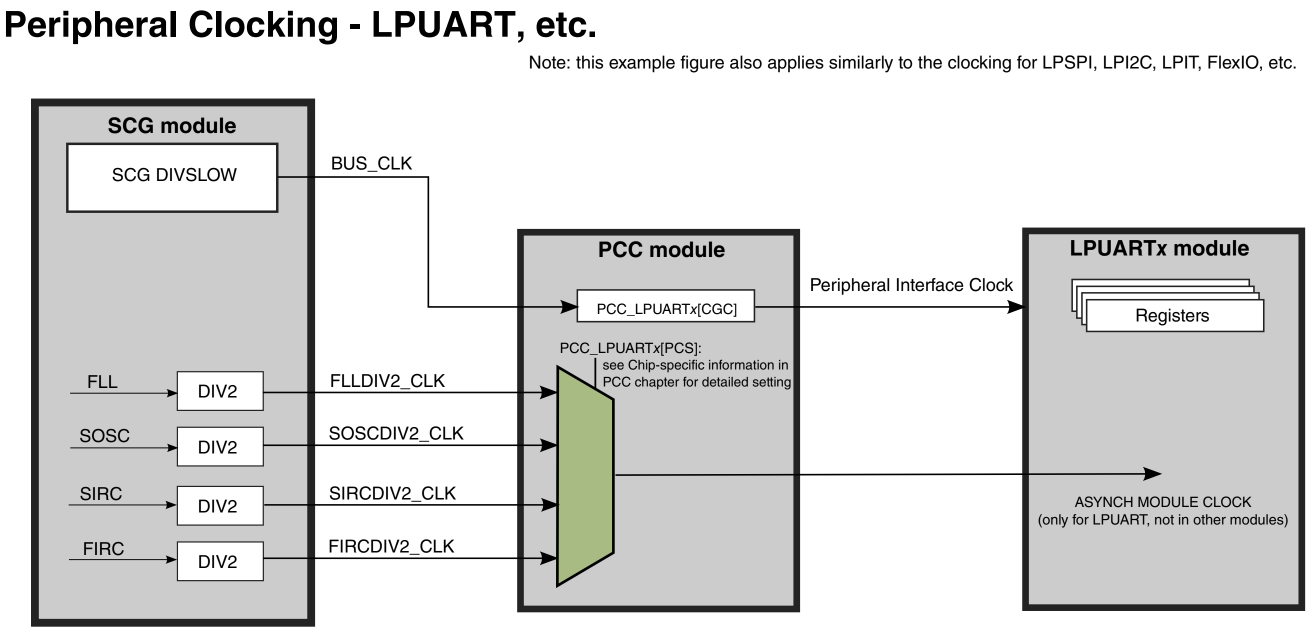
- LPSPI: 32-bit
- LPI2C: 8-bit (except CMD)
- LPUART: 10-bit

NOTE

The exact number of chip select for each module is depending on the package, not all of the chip selects are available on different packages.

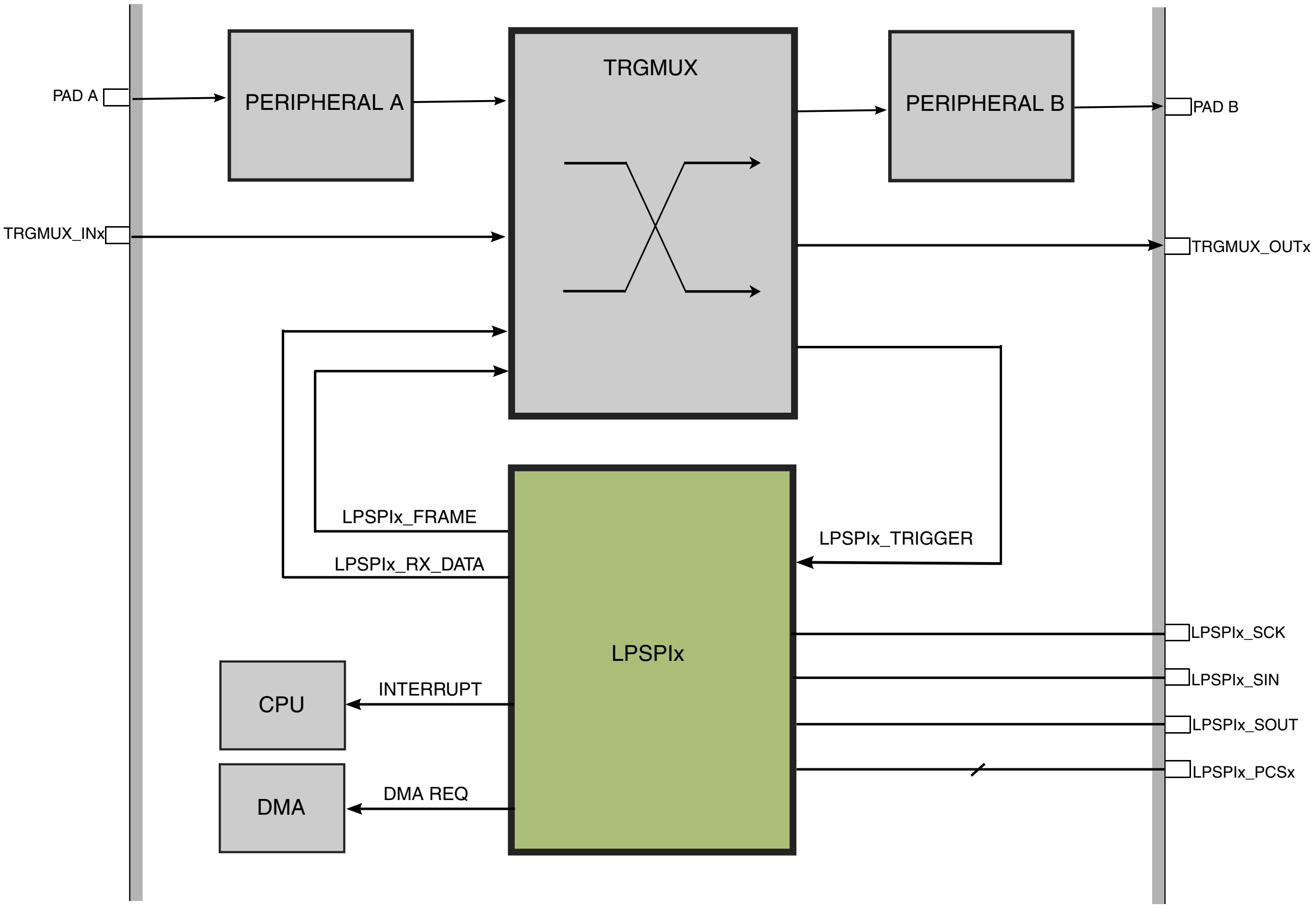
39.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

The following figure shows the input clock sources available for this module.


Peripheral Clocking - LPUART, etc.
Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, LPIT, FlexIO, etc.
SCG module
SCG DIVSLOW
BUS_CLK
FLL
DIV2
FLLDIV2_CLK
SOSC
DIV2
SOSCDIV2_CLK
SIRC
DIV2
SIRCDIV2_CLK
FIRC
DIV2
FIRCDIV2_CLK
PCC module
PCC_LPUARTx[CGC]
PCC_LPUARTx[PCS]:
see Chip-specific information in
PCC chapter for detailed setting
Peripheral Interface Clock
LPUARTx module
Registers
ASYNCH MODULE CLOCK
(only for LPUART, not in other modules)

39.1.3 Inter-connectivity Information

The LPSPI inter-connectivity is shown in following diagram.



39.2 Overview

LPSPI provides an efficient interface (either as a master or slave) to an SPI bus, which is a synchronous serial communication interface used in embedded systems. It is typically used to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing with secure digital cards and LCD displays.

39.2.1 Block diagram

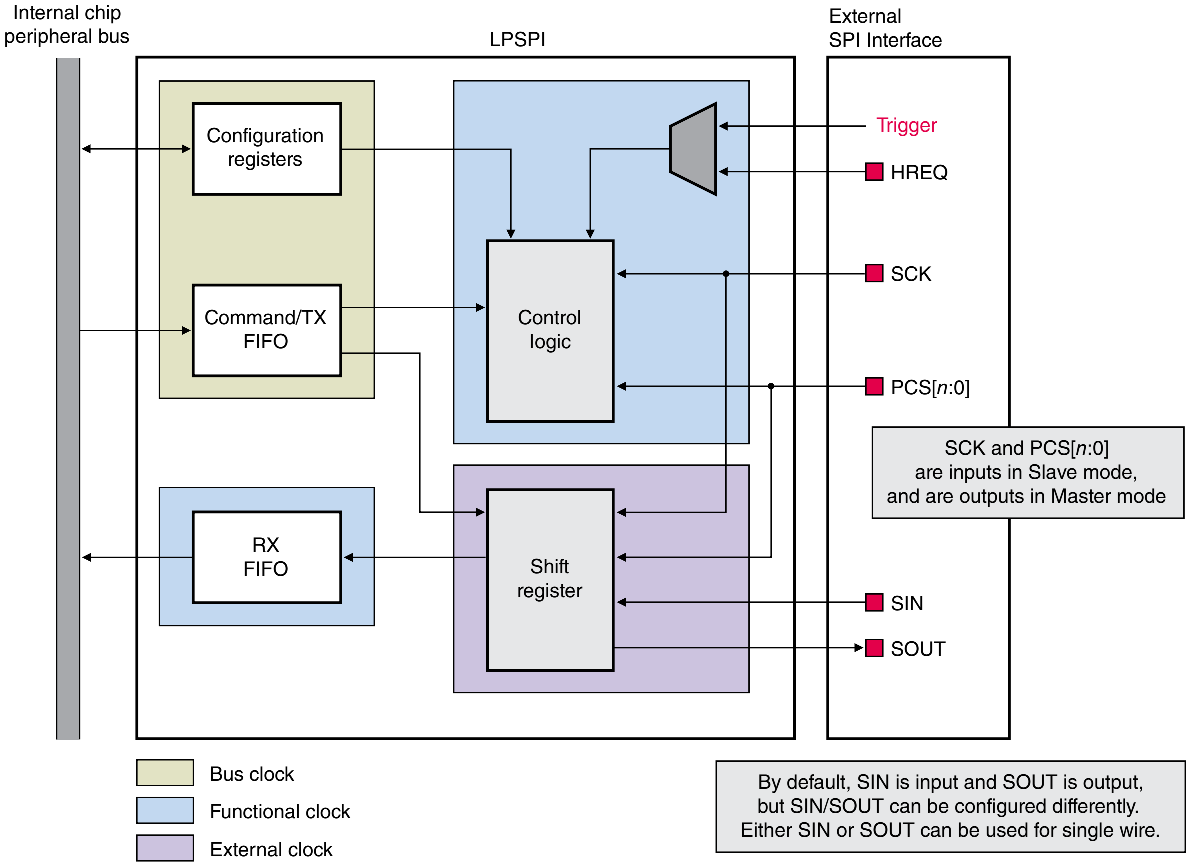

Internal chip peripheral bus
LPSPI
External SPI Interface
Configuration registers
Command/TX FIFO
Control logic
RX FIFO
Shift register
Trigger
HREQ
SCK
PCS[n:0]
SCK and PCS[n:0] are inputs in Slave mode, and are outputs in Master mode
SIN
SOUT
Bus clock
Functional clock
External clock
By default, SIN is input and SOUT is output, but SIN/SOUT can be configured differently. Either SIN or SOUT can be used for single wire.

Figure 39-1. Block diagram

39.2.2 Features

- Minimal CPU overhead, with DMA transmit and receive requests supporting FIFO register accesses
- Operation continues in STOP mode, if configured to do so and an appropriate clock is available
- Support available for 32-bit word size
- Configurable clock polarity and phase
- Support available for 4 peripheral chip selects in Master mode
- Support available for Slave mode
- 4-word transmit and command FIFO
- 4-word receive FIFO
- Flexible timing parameters in Master mode, including SCK frequency and delays between PCS and SCK edges
- Full-duplex transfers that support 1-bit transmit and receive on each clock edge
- Half-duplex transfers that support:

  - 1-bit transmit or receive on each clock edge
  - 2-bit transmit or receive on each clock edge (Master mode only)
  - 4-bit transmit or receive on each clock edge (Master mode only)
- Option to use host request to control the start of an SPI bus transfer (Master mode only)
- Receive data match logic that discards nonmatching data and interrupt on data match

39.3 Functional description

39.3.1 Master mode

39.3.1.1 Transmit and command FIFO commands

The transmit and command FIFO is a combined FIFO that includes both transmit data words and command words. You store:

- Transmit data words in the transmit and command FIFO, by writing to Transmit Data (TDR).
- Command words in the transmit and command FIFO, by writing to Transmit Command (TCR).

When a command word is at the top of the transmit and command FIFO, the actions that can occur depend on whether LPSPI is busy or between frames (see TCR[CONT] and TCR[CONTC]). See Table 39-2 for conditions and possible corresponding actions when a command word is at the top of the transmit and command FIFO.

Table 39-2. Possible actions when a command word is at the top of the transmit and command FIFO

Condition	Action
LPSPI is enabled and idle.	The command word is pulled from the FIFO, and this command word controls all subsequent transfers.
LPSPI is busy and TCR[CONTC] is 0.	The SPI frame completes at the end of the existing word, ignoring TCR[FRAMESZ]. The command word is then pulled from the FIFO and that command word controls all subsequent transfers (or until the next update to the command word). Note that a command word with TCR[CONTC] = 0 always terminates the existing transfer regardless of the previous TCR[CONT] value.
LPSPI is busy; the existing TCR[CONT] value is 1 and the new TCR[CONTC] value is 1.	The command word must be updated at the frame boundary. The command word is pulled from the FIFO during the last SCK pulse of the existing frame (based on the value of FRAMESZ), 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 TCR[CONTC] = 1, only the lower 24 bits of the command word are updated. If the command word is updated at a word boundary, then the transfer halts (stops) after that word. TCR[CONTC] is ignored when not at a frame boundary, so the frame ends prematurely.

TCR[CONT] = 1 keeps PCS asserted at end of frame, allowing the transfer to continue.

TCR[CONTC] = 1 specifies that this command word must not terminate the existing frame, and the transfer can continue using the new command word.

TCR[CONTC] = 1 is restricted in the sense that the new command must load on a frame boundary, and the only way for a transfer to continue from a frame boundary is when the previous command has TCR[CONT] = 1.

You can read the current state of the existing command word from Transmit Command (TCR). It requires at least three LPSPI functional clock cycles for Transmit Command (TCR) to update after you write to it (assuming an empty FIFO), and LPSPI must be enabled (CR[MEN] = 1).

Writing to Transmit Command (TCR) does not initiate an SPI bus transfer, unless TCR[TXMSK] = 1. When TCR[TXMSK] = 1, a new command word is not loaded until the end of the existing frame (based on the value of TCR[FRAMESZ]); at the end of the transfer, TCR[TXMSK] transitions to 0.

In Master mode, the LPSPI command word in Transmit Command (TCR) controls SPI attributes based on the selections in register fields. See Table 39-3 for TCR fields and associated functionality related to data transfer.

Table 39-3. Command word in Master mode

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
CPOL	Clock polarity	Specifies the polarity of the SCK pin. Any change of CPOL value causes a transition on the SCK pin.	N
CPHA	Clock phase	Specifies the clock phase of the transfer.	N
PRESCALE	Prescaler value	Specifies a prescaler used to divide the LPSPI functional clock, to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS enables LPSPI to connect to different slave devices at different frequencies.	N
PCS	Peripheral chip select	Specifies which PCS pin asserts for the transfer; the polarity of PCS is static and specified by CFGR1[PCSPOL]. If CFGR1[PCSCFG] = 1, do not select PCS[3:2].	N
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.	Y
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing with devices that organize data as big-endian.	Y
CONT	Continuous transfer	Configures LPSPI for a continuous transfer that keeps PCS asserted between frames (as specified by FRAMESZ). You must write a new	Y

Table continues on the next page...

Table 39-3. Command word in Master mode (continued)

Transmit Command (TCR)		Description	Can this field be modified during a data transfer?
Field	Name		
		command word to cause PCS to negate. Also, this field supports changing the command word at frame size boundaries.	
CONTC	Continuing command	Indicates that this is a new command word for the existing continuous transfer. When CONTC = 1, the command word must only be written to the transmit and command FIFO on a frame boundary.	Y
RXMSK	Receive data mask	Masks the receive data and does not store the masked receive data in the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.	Y
TXMSK	Transmit data mask	Masks the transmit data; masked transmit data is not pulled from the transmit FIFO, and the output data pin is 3-stated (unless otherwise configured by CFGR1[OUTCFG]). This option is useful for half-duplex transfers.	Y
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse: • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit half-duplex transfers are useful for interfacing with QuadSPI memory devices, and either TXMSK or RXMSK must also be 1.	Y
FRAMESZ	Frame size	Configures the frame size in number of bits equal to (FRAMESZ + 1): • The minimum frame size is 8 bits. • The minimum word size is 2 bits; a frame size of 33 bits (or similar) is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remaining bits. For example, a 72-bit transfer consists of three words: the first and second words are 32 bits, and the third word is 8 bits.	Y

39.3.1.1.1 SPI bus transfers

LPSPI initiates an SPI bus transfer when all these conditions are true:

- Data is written to the transmit FIFO.
- The HREQ pin is asserted (or the HREQ function is disabled).
- LPSPI is enabled.

To perform the SPI bus transfer, LPSPI uses the attributes configured in Transmit Command (TCR) and the timing parameters defined in Clock Configuration (CCR).

The SPI bus transfer ends after the number of bits indicated by the value of FRAMESZ have been transferred (provided CONT = 0), or at the end of a word when a new transmit command word is at the top of the transmit and command FIFO. When LPSPI is disabled, the SPI bus transfers end after the transmit FIFO is empty and LPSPI is idle.

The HREQ input is only checked the next time LPSPI goes idle (LPSPI completes the current transmit FIFO and there are no attribute updates in Transmit Command (TCR)).

39.3.1.1.2 Circular FIFO

The transmit and command FIFO supports a circular FIFO feature. This feature enables the LPSPI master to (periodically) repeat a short data transfer that fits within the transmit and command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled (CFGR0[CIRFIFO] = 1), the current state of the FIFO read pointer is saved and the status flags are not updated. After the FIFO is empty and LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit and command FIFO are not permanently pulled from the FIFO when Circular FIFO mode is enabled.

39.3.1.2 Receive FIFO and data match

The receive FIFO stores received data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of being stored in the receive FIFO:

- Received data is written to the receive FIFO at the end of the frame.
- During a multiple-word or continuous transfer, the received data is also written to the receive FIFO at the same time the new transmit data is read from the transmit FIFO.
- During a continuous transfer, if the transmit FIFO is empty, then the received data is only written to the receive FIFO after the transmit FIFO is written or after TCR is written to end the frame.

LPSPI provides a receive data match function that can match received data against one of the two words in DMR0 and DMR1, or against a masked data word. You can also configure the received data match function to compare only the first one or two received data words since the start of the frame:

- Received data that is already discarded because of TCR[RXMSK] cannot cause the data match flag to set, and delays the receive data match on the first received data word, until all discarded data is received.
- You can configure the receive data match function to discard all received data until a data match is detected, using CFGR0[RDMO].
- After a receive data match, to allow all subsequent data to be received, write 0 to CFGR0[RDMO], and then write 0 to SR[DMF].

39.3.1.3 Timing parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the selection specified in TCR[PRESCALE]. Although you cannot change Clock Configuration (CCR) when LPSPI is busy, to support interfacing with different slave devices at different frequencies, you can change the TCR[PRESCALE] selection between SPI bus transfers by using Transmit Command (TCR).

NOTE

The minimum value shown in Table 39-4 is the minimum counter value, but the values of Clock Configuration (CCR) must also satisfy the data sheet specs based on the LPSPI functional clock frequency and prescaler value.

Table 39-4. Timing parameters

Clock Configuration (CCR)		Description	Minimum value	Maximum value
Field	Name			
SCKDIV	SCK divider	Configures the SCK clock period to (SCKDIV + 2) cycles. When you configure the SCK divider to an odd number of cycles, the first half of the SCK cycle is one cycle longer than the second half of the SCK cycle.	0 (2 cycles)	255 (257 cycles)
DBT	Delay between transfers (PCS negation to PCS assertion)	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 PCS pins.	0 (2 cycles)	255 (257 cycles)
DBT	Delay between transfers (last SCK edge to first SCK edge)	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 when the external slave requires a large delay between different words of an SPI bus transfer.	0 (1 cycle)	255 (256 cycles)
PCSSCK	PCS-to-SCK delay	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	SCK-to-PCS delay	Configures the minimum delay between the last SCK edge and the PCS negation to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

Figure 39-2 shows the timing settings controlled by:

- TCR[CPHA]
- TCR[CPOL]
- CCR[SCKPCS]
- CCR[PCSSCK]

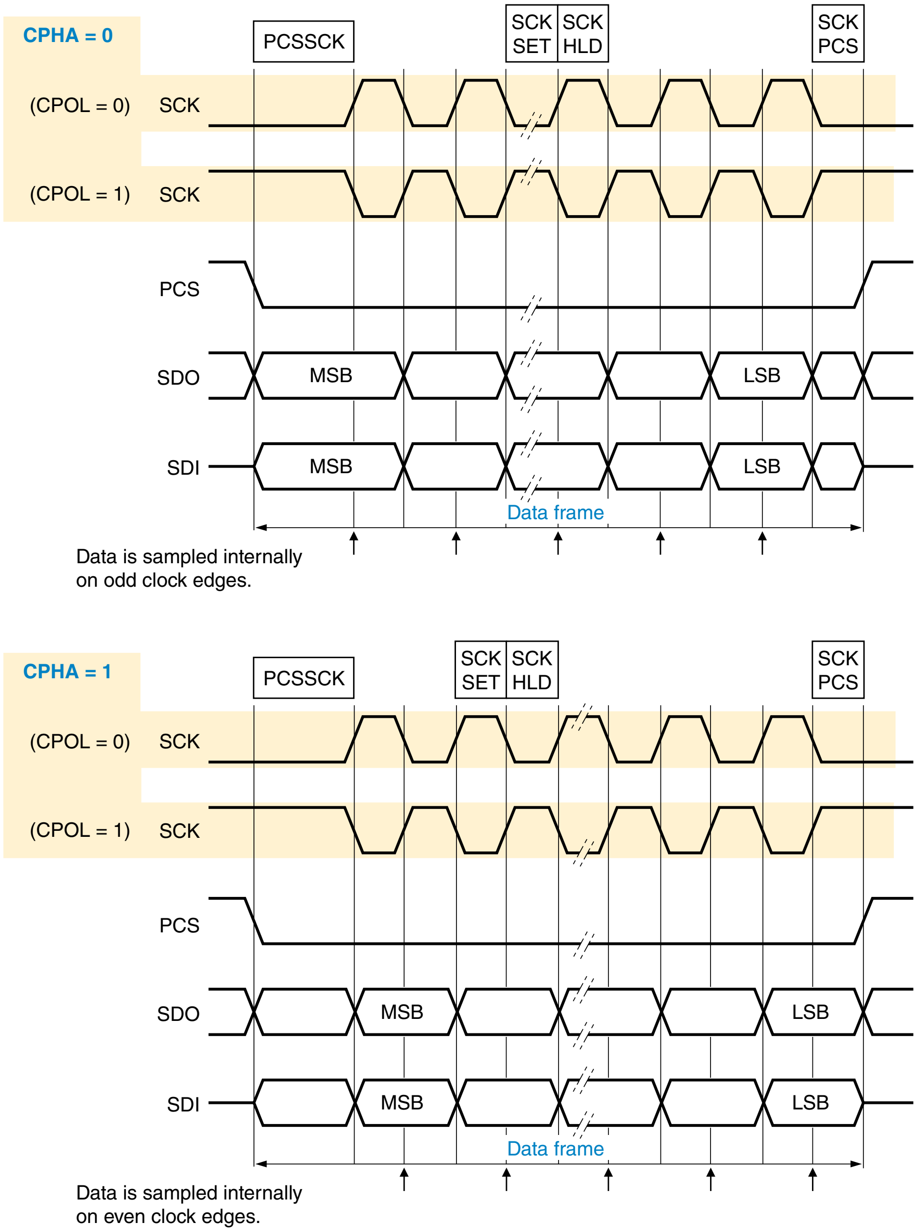


Figure 39-2. Clock phase (TCR[CPHA]) timing diagram example

39.3.1.4 Pin configuration

Following are the pin configuration settings for half-duplex transfers:

- To swap directions or to support half-duplex transfers on the same pin, you can configure the SIN and SOUT pins using CFGR1[PINCFG].
- To specify whether an output data pin (SOUT, for example) 3-states when PCS is negated, or if the output data pin retains the last value, use CFGR1[OUTCFG].

- When configuring half-duplex transfers, you must configure the output data pins to 3-state when PCS is negated (CFGR1[OUTCFG] = 1).
- When performing half-duplex 2-bit transfers, you can write any value to CFGR1[PCSCFG].
- When performing half-duplex 4-bit transfers, you must write 1h to CFGR1[PCSCFG].

39.3.1.5 Clock loopback

Configure the LPSPI master to use one of the following clocks to sample the input data:

- The SCK output clock
- A delayed version of the SCK output clock

The delayed version of the SCK is chosen by the SCK pin output delay, plus the SCK pin input delay, and is selected by writing 1 to CFGR1[SAMPLE]. Enabling the loopback version of the SCK pin can improve the setup time of the input data from the slave.

See the chip data sheet for the specific input setup time in Master Loopback mode.

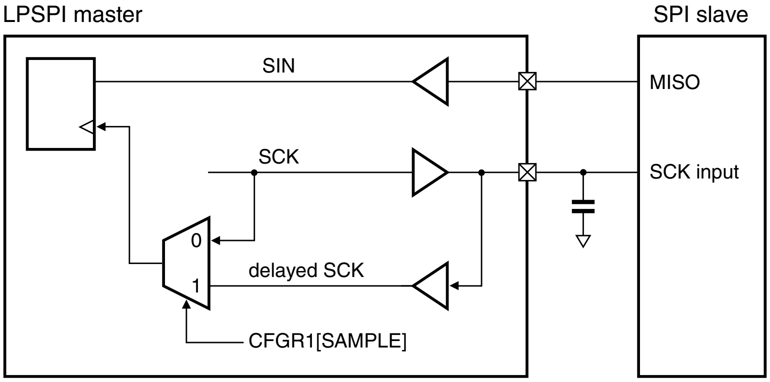


Figure 39-3. Clock loopback

39.3.2 Slave mode

LPSPI Slave mode:

- Uses the same shift register and logic that Master mode uses.
- Does not use Clock Configuration (CCR).
- Requires Transmit Command (TCR) to remain static (unchanged) during SPI bus transfers.

39.3.2.1 Transmit and command FIFO commands

You must initialize Transmit Command (TCR) before enabling LPSPI in Slave mode, although this register is not updated until after LPSPI is enabled. After LPSPI is enabled, you must make changes to this register only when LPSPI is idle. In Slave mode, the LPSPI command word in this register controls SPI attributes. Before the PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag sets.

Table 39-5. Command word in Slave mode

Transmit Command (TCR)		Description
Field	**Name**	
CPOL	Clock polarity	Specifies the polarity of the external SCK input.
CPHA	Clock phase	Specifies the clock phase of transfer.
PRESCALE	Prescaler value	Specifies the LPSPI functional clock prescaler.
PCS	Peripheral chip select	Specifies which PCS is used. The polarity of PCS is static and configured by CFGR1[PCSPOL]. If CFGR1[PCSCFG] is not equal to zero, then do not select the PCS[3:2] pins.
LSBF	LSB first	Specifies whether LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.
BYSW	Byte swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing with devices that organize data as big-endian.
CONT	Continuous transfer	When continuous transfer is selected in Slave mode, after the number of bits indicated by FRAMESZ are transferred, LPSPI passes through and transmits the received data until the next PCS negation. Whatever is shifted in on the receive data is shifted out as transmit data considering that there is a 32-bit shift register.
CONTC	Continuing command	This field is reserved in Slave mode.
RXMSK	Receive data mask	Masks the receive data; LPSPI does not store masked receive data in the receive FIFO or perform receive data matching. This option is useful for half-duplex transfers or to specify which fields are compared during receive data matching.
TXMSK	Transmit data mask	Masks the transmit data so that the masked transmit data is not pulled from transmit FIFO, and the output data pin is 3-stated (unless otherwise specified in CFGR1[OUTCFG]). This option is useful for half-duplex transfers.
WIDTH	Transfer width	Specifies the number of bits shifted on each SCK pulse: • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats.
FRAMESZ	Frame size	Specifies the frame size in number of bits equal to (FRAMESZ + 1): • The minimum frame size is 8 bits. • The minimum word size is 2 bits; a frame size of 33 bits (or similar) is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contain the remainder bits. For example, a 72-bit transfer consists of three words: the first and second words are 32 bits, and the third word is 8 bits.

39.3.2.2 Receive FIFO and data match

The receive FIFO stores receive data during SPI bus transfers. When TCR[RXMSK] = 1, the received data is discarded instead of storing the received data in the receive FIFO.

Receive data supports a receive data match function that can match received data against one of the two words in DMR0 and DMR1 or against a masked data word. You can also configure the data match function to compare only the first one or two received data words since the start of the frame:

- Received data that is already discarded because TCR[RXMSK] = 1 cannot cause the data match to set, and delays the match on the first received data word, until all discarded data is received.
- By using CFGR0[RDMO], you can also configure the receiver match function to discard all received data until a data match is detected.
- After a receive data match, to allow all subsequent data to be received, first write 0 to CFGR0[RDMO], then clear SR[DMF].

39.3.2.3 Clocked interface

LPSPI supports interfacing with external masters that provide only clock and data pins (PCS is not required). This interface requires:

- Writing 1 to TCR[CPHA] (data is changed on the leading edge of SCK and captured on the following edge).
- Configuring the PCS input to be always asserted (CFGR1[PCSPOL*n*] = 1). For example, to configure PCS[0] to be always asserted, write 1 to PCSPOL[0], and do not configure PCS[0] in the pin muxing. The chip-level drives PCS to a certain value (ideally 1); you could use CFGR1[PCSPOL*n*] to invert that value.
- Writing 1 to CFGR1[AUTOPCS] to enable automatic PCS generation. When CFGR1[AUTOPCS] = 1, a minimum of four LPSPI functional clock cycles (divided by the selection specified in TCR[PRESCALE]) is required between the last SCK edge of one word and the first SCK edge of the next word.

39.3.3 Low-power modes

Table 39-6. Low-power modes

Chip mode	LPSPI operation
Run	Normal operation
STOP or Wait	Can continue operating in STOP mode if CR[DOZEN] = 0 and LPSPI is using an external or internal clock source that remains operating during STOP or Wait modes

39.3.4 Debug mode

Table 39-7. Debug mode

Chip mode	LPSPI operation
Debug (the core is in Debug or Halted mode)	Can continue operating in Debug mode, if CR[DBGEN] = 1

39.3.5 Clocking

Table 39-8. LPSPI clocks

Type of clock	Description
Functional	• Asynchronous to the bus clock. • If the LPSPI functional clock remains enabled in low-power modes, then LPSPI can perform SPI bus transfers and low-power wakeups in both Master and Slave modes. • LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least two times faster than the SPI external clock frequency (SCK).
External	• The LPSPI shift register is clocked directly by the SCK clock. • How the SCK clock is generated or supplied depends on the mode (Master or Slave): • In Master mode, the SCK clock is generated internally. • In Slave mode, the SCK clock is supplied externally.
Bus	The bus clock is only used for bus accesses to the LPSPI control and configuration registers. The bus clock frequency must be high enough to support the data bandwidth requirements of the LPSPI registers, including the FIFOs.

See the chip-specific LPSPI information for more.

39.3.6 Reset

Table 39-9. LPSPI resets

Type of reset	Description
Chip	Resets the LPSPI logic and registers to their default states.
Software	• Resets the LPSPI logic and registers to their default states, except for the Control register. • The LPSPI software reset is controlled using CR[RST].
FIFO	• Resets the transmit and command FIFO and the receive FIFO. • CR[RTF] and CR[RRF] are write-only. • After being reset, FIFO is empty.

39.3.7 Interrupts and DMA requests

The following table lists Slave mode sources (status flags) that can generate LPSPI interrupts and LPSPI slave transmit and receive DMA requests.

Table 39-10. Interrupts and DMA requests

Status (SR)		Description	Can generate		
Status flag	Name		Interrupt?	DMA request?	Low-power wake-up?
TDF	Transmit data flag	Indicates that data can be written to transmit FIFO, as configured by the transmit FIFO watermark, FCR[TXWATER].	Y	TX	Y
RDF	Receive data flag	Indicates that data can be read from the receive FIFO, as configured by the receive FIFO watermark, FCR[RXWATER].	Y	RX	Y
WCF	Word complete flag	Indicates that the word is complete and the last bit of the word has been sampled.	Y	N	Y
FCF	Frame complete flag	Indicates that the frame is complete and PCS is deasserted.	Y	N	Y
TCF	Transfer complete flag	Indicates that transfer is complete, PCS is deasserted, and the transmit and command FIFO is empty.	Y	N	Y
TEF	Transmit error flag	Indicates a transmit and command FIFO underrun. In Master mode, when CFGR1[NOSTALL] = 0 (transfers stall when transmit FIFO is empty), TEF cannot be set.	Y	N	Y
REF	Receive error flag	Indicates a receive FIFO overflow.	Y	N	Y
DMF	Data match flag	Indicates that the received data matches the configured data match value.	Y	N	Y
MBF	Module busy flag	Indicates that LPSPI is busy performing an SPI bus transfer.	N	N	N

39.3.8 Peripheral triggers

The connection of the LPSPI peripheral triggers with other peripherals depends on the device that is used.

Table 39-11. Peripheral triggers

Type of trigger	Description	Additional information
Frame output	The frame output trigger is an idle high signal, which: • Asserts at the end of each frame (when PCS deasserts). • Remains asserted until PCS next asserts.	LPSPI generates two output triggers that can be connected to other peripherals on the chip.
Word output	The word output trigger: • Asserts at the end of each received word. • Remains asserted for one SCK period.	
Input	To control the start of an LPSPI bus transfer, the LPSPI input trigger can be selected instead of the HREQ input: • The LPSPI input trigger is synchronized, and must assert for at least two cycles of the LPSPI functional clock divided by the configuration defined in TCR[PRESCALE] so that the input trigger can be detected. • When LPSPI is busy, the HREQ input (and therefore the LPSPI input trigger) is ignored. • When LPSPI is busy, both the HREQ and LPSPI input triggers are ignored. They are used to start a new transfer when LPSPI is idle.	

39.4 External signals

Table 39-12. External signals

Signal	Name	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 CFGR0[HREN] = 1 and CFGR0[HRSEL] = 0: • Input in either Slave mode or when used as master host request • Output in Master mode	I/O
PCS[2]/DATA[2]	Peripheral chip select or data pin 2 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Slave mode • Output in Master mode When CFGR1[PCSCFG] = 1:	I/O

Table continues on the next page...

Table 39-12. External signals (continued)

Signal	Name	Description	I/O
		• Input in half-duplex parallel data receive transfers • Output in half-duplex parallel data transmit transfers	
PCS[3]/DATA[3]	Peripheral chip select or data pin 3 during parallel data transfers	When CFGR1[PCSCFG] = 0: • Input in Slave mode • Output in Master mode When CFGR1[PCSCFG] = 1: • Input in half-duplex parallel data receive transfers • Output in half-duplex parallel 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 half-duplex parallel data transfers)	I/O
SIN/DATA[1]	Serial data input	Can be configured as serial data output signal (used as data pin 1 in half-duplex parallel data transfers)	I/O

39.5 Initialization

This module does not require initialization.

39.6 Memory map and registers

NOTE

- Writing to a read-only register or reading a write-only register can cause bus errors.
- LPSPI does not check values programmed in registers for validity, so you must take care to write valid values only.

39.6.1 LPSPI register descriptions

LPSPI provides an efficient interface to an SPI bus, either as a master or slave. An SPI bus is a synchronous serial communication interface used in embedded systems. It is typically used to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing with secure digital cards and LCD displays.

39.6.1.1 LPSPI memory map

LPSPI0 base address: 4002_C000h

LPSPI1 base address: 4002_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0004h
4h	Parameter (PARAM)	32	R	0000_0202h
10h	Control (CR)	32	RW	0000_0000h
14h	Status (SR)	32	RW	0000_0001h
18h	Interrupt Enable (IER)	32	RW	0000_0000h
1Ch	DMA Enable (DER)	32	RW	0000_0000h
20h	Configuration 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match 0 (DMR0)	32	RW	0000_0000h
34h	Data Match 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration (CCR)	32	RW	0000_0000h
58h	FIFO Control (FCR)	32	RW	0000_0000h
5Ch	FIFO Status (FSR)	32	R	0000_0000h
60h	Transmit Command (TCR)	32	RW	0000_001Fh
64h	Transmit Data (TDR)	32	W	0000_0000h
70h	Receive Status (RSR)	32	R	0000_0002h
74h	Receive Data (RDR)	32	R	0000_0000h

39.6.1.2 Version ID (VERID)

39.6.1.2.1 Offset

Register	Offset
VERID	0h

39.6.1.2.2 Function

Contains version numbers for the module design and feature set.

39.6.1.2.3 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	0

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	1	0	0

39.6.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the module specification.
23-16 MINOR	Minor Version Number Indicates the minor version number of the module specification.
15-0 FEATURE	Module Identification Number Indicates the feature set number 0000_0000_0000_0100b - Standard feature set supporting a 32-bit shift register. All other values are reserved.

39.6.1.3 Parameter (PARAM)

39.6.1.3.1 Offset

Register	Offset
PARAM	4h

39.6.1.3.2 Function

Contains:

- Number of PCS pins.
- Receive FIFO size.
- Transmit FIFO size.

39.6.1.3.3 Diagram

Bits	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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

39.6.1.3.4 Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size Indicates the maximum number of words in the receive FIFO. The maximum number of words is 2^{RXFIFO}.
7-0 TXFIFO	Transmit FIFO Size Indicates the maximum number of words in the transmit FIFO. The maximum number of words is 2^{TXFIFO}.

39.6.1.4 Control (CR)

39.6.1.4.1 Offset

Register	Offset
CR	10h

39.6.1.4.2 Function

Contains fields that control the module operation.

39.6.1.4.3 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								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

39.6.1.4.4 Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Deletes all entries in the receive FIFO. This field always reads 0. 0b - No effect 1b - Reset
8 RTF	Reset Transmit FIFO Deletes all entries in the transmit FIFO. This field always reads 0. 0b - No effect 1b - Reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables LPSPI when the CPU is in Debug mode. If this field is 0, LPSPI is disabled when the CPU is halted; the PCS pin is deasserted after the transmit FIFO is empty regardless of the state of Transmit Command (TCR). You must update this field only when LPSPI is disabled (MEN = 0). 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables LPSPI when the chip is in Doze mode. You must update this field only when LPSPI is disabled (MEN = 0). 0b - Enable 1b - Disable
1 RST	Software Reset

Table continues on the next page...

Field	Function
	Resets all internal logic and registers, except Control (CR). The reset takes effect immediately and remains asserted until you write 0 to it. There is no minimum delay required before clearing the software reset by writing 0. 0b - Not reset 1b - Reset
0 MEN	Module Enable Enables the module. After writing 0, MEN remains set until LPSPI has completed the current transfer and is idle. 0b - Disable 1b - Enable

39.6.1.5 Status (SR)

39.6.1.5.1 Offset

Register	Offset
SR	14h

39.6.1.5.2 Function

Contains data flow status.

39.6.1.5.3 Diagram

Bits	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

Bits	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

39.6.1.5.4 Fields

Field	Function
31-25	Reserved

Table continues on the next page...

Field	Function
—	
24 MBF	Module Busy Flag Indicates, in Master mode, whether there is data to transmit and LPSPI is able to transmit (for example, the HREQ pin is asserted). The HREQ pin deasserts after the PCS pin deasserts and the LPSPI master has waited for half the time specified in CCR[DBT] with no new data to transmit. Slave mode sets this flag when LPSPI is enabled and PCS is asserted. 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates whether the received data matches DMR0[MATCH0] and/or DMR1[MATCH1] (as configured by CFGR1[MATCFG]). **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No match 1b - Match When writing 0b - No effect 1b - Clear the flag
12 REF	Receive Error Flag Indicates a receive FIFO overflow error. When this flag is set: 1. End the transfer. 2. Empty the receive FIFO. 3. Clear this flag. 4. Restart the transfer from the beginning. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No overflow 1b - Overflow When writing 0b - No effect 1b - Clear the flag
11 TEF	Transmit Error Flag Indicates a transmit FIFO underrun error. When this flag is set: 1. End the transfer. 2. Clear this flag. 3. Restart the transfer from the beginning. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No underrun 1b - Underrun When writing

Table continues on the next page...

Field	Function
	0b - No effect 1b - Clear the flag
10 TCF	Transfer Complete Flag Indicates, in Master mode, whether all transfers are complete and LPSPI has returned to the Idle state and the transmit FIFO is empty. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
9 FCF	Frame Complete Flag Indicates whether a frame transfer is complete after PCS deasserts. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
8 WCF	Word Complete Flag Indicates whether the last bit of a received word is sampled. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not complete 1b - Complete When writing 0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the number of words in the receive FIFO is greater than the value in FCR[RXWATER]. 0b - Receive data not ready 1b - Receive data ready
0 TDF	Transmit Data Flag Indicates whether the number of words in the transmit FIFO is equal to or less than the value in FCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data requested

39.6.1.6 Interrupt Enable (IER)

39.6.1.6.1 Offset

Register	Offset
IER	18h

39.6.1.6.2 Function

Enables interrupts based on data flow and errors.

39.6.1.6.3 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		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

39.6.1.6.4 Fields

Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable Enables the data match interrupt. 0b - Disable 1b - Enable
12 REIE	Receive Error Interrupt Enable Enables the receive complete interrupt. 0b - Disable 1b - Enable
11 TEIE	Transmit Error Interrupt Enable Enables the transmit complete interrupt. 0b - Disable

Table continues on the next page...

Field	Function
	1b - Enable
10 TCIE	Transfer Complete Interrupt Enable Enables the transfer complete interrupt. 0b - Disable 1b - Enable
9 FCIE	Frame Complete Interrupt Enable Enables the frame complete interrupt. 0b - Disable 1b - Enable
8 WCIE	Word Complete Interrupt Enable Enables the word complete interrupt. 0b - Disable 1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables the receive data interrupt. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables the transmit data interrupt. 0b - Disable 1b - Enable

39.6.1.7 DMA Enable (DER)

39.6.1.7.1 Offset

Register	Offset
DER	1Ch

39.6.1.7.2 Function

Enables the DMA data flow.

39.6.1.7.3 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														RDDE	TDDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.7.4 Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables the receive data DMA. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables the transmit data DMA. 0b - Disable 1b - Enable

39.6.1.8 Configuration 0 (CFGR0)

39.6.1.8.1 Offset

Register	Offset
CFGR0	20h

39.6.1.8.2 Function

Includes fields to configure LPSPI.

39.6.1.8.3 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						RDMO	CIRFIFO	0				0	HRSEL	HRPOL	HREN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.8.4 Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only Enables receive data match. When enabled, all received data that does not cause SR[DMF] to assert is discarded: • Write 1 to this field when LPSPI is idle and SR[DMF] = 0. • After SR[DMF] = 1, this field is ignored. • To ensure that no receive data is lost when disabling RDMO, write 0 to this field before clearing SR[DMF]. See CFGR1[MATCFG] for the received data matching options. When disabled, all received data is stored in the receive FIFO. 0b - Disable 1b - Enable
8 CIRFIFO	Circular FIFO Enable Enables circular FIFO. When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO is emptied as in normal operation, but when LPSPI is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This restoring of the read pointer causes the contents of the transmit FIFO to be cycled through repeatedly. **NOTE:** The read pointer is restored for as long as this field is 1. Writing additional words to the FIFO when this field is 1 adds them to the end of the FIFO, up to the size of the transmit FIFO. 0b - Disable 1b - Enable
7-4 —	Reserved
3	Reserved

Table continues on the next page...

Field	Function
—	
2 HRSEL	Host Request Select Specifies the source of the host request input. When the host request function is enabled with the HREQ pin, the PCS[1] function is disabled. 0b - HREQ pin 1b - Input trigger
1 HRPOL	Host Request Polarity Specifies the polarity of the HREQ pin. 0b - Active low 1b - Active high
0 HREN	Host Request Enable Enables LPSPI, in Master mode, to start a new SPI bus transfer only if the host request input is asserted. When LPSPI is busy, the host request input is ignored. 0b - Disable 1b - Enable

39.6.1.9 Configuration 1 (CFGR1)

39.6.1.9.1 Offset

Register	Offset
CFGR1	24h

39.6.1.9.2 Function

Includes fields to configure LPSPI. You must write to this register only when LPSPI is disabled.

In addition to pin and output configurations, this register contains match configuration details; the following table shows match conditions specified in MATCFG.

Table 39-13. Match conditions for CFGR1[MATCFG]

Condition	Description
Match first data word with compare word	Match if first data word equals MATCH0 logically ORed with MATCH1 first_data_word == (MATCH0 || MATCH1)
Match any data word with compare word	Match if any data word equals MATCH0 logically ORed with MATCH1 any_data_word == (MATCH0 || MATCH1)
Sequential match, first data word	Match if first data word equals MATCH0, and second data word equals MATCH1 (first_data_word == MATCH0) && (second_data_word == MATCH1)

Table continues on the next page...

Table 39-13. Match conditions for CFGR1[MATCFG] (continued)

Condition	Description
Sequential match, any data word	Match if any data word equals MATCH0, and the next data word equals MATCH1 (any_data_word == MATCH0) && (next_data_word == MATCH1)
Match first data word (masked) with compare word (masked)	Match if first data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1 (first_data_word && MATCH1) == (MATCH0 && MATCH1)
Match any data word (masked) with compare word (masked)	Match if any data word logically ANDed with MATCH1 equals MATCH0 logically ANDed with MATCH1 (any_data_word && MATCH1) == (MATCH0 && MATCH1)

39.6.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			0	PCSCFG	OUTCFG	PINCFG		0					MATCFG		
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				PCSPOL				0			0	NOSTALL	AUTOPCS	SAMPLE	MASTER
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.9.4 Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 PCSCFG	Peripheral Chip Select Configuration Specifies PCS pin configuration. When performing parallel transfers, you must configure this field to enable the desired transfer. 0b - PCS[3:2] configured for chip select function 1b - PCS[3:2] configured for half-duplex 4-bit transfers (PCS[3:2] = DATA[3:2])
26 OUTCFG	Output Configuration Specifies whether the output data is 3-stated between accesses (when PCS is deasserted). When performing half-duplex transfers, this field must be 1.

Table continues on the next page...

Field	Function
	0b - Retain last value 1b - 3-stated
25-24 PINCFG	Pin Configuration Specifies the pins used for input and output data during serial transfers. **NOTE:** When performing parallel transfers, this field must be 0. 00b - SIN is used for input data; SOUT is used for output data 01b - SIN is used for both input and output data; only half-duplex serial transfers are supported 10b - SOUT is used for both input and output data; only half-duplex serial transfers are supported 11b - SOUT is used for input data; SIN is used for output data
23-19 —	Reserved
18-16 MATCFG	Match Configuration Specifies the condition that causes SR[DMF] to assert. See the match conditions listed in Table 1 for more information. **NOTE:** When writing to this field, either the old value or new value must be in the disabled state (0). You cannot transition from a nonzero value to another nonzero value. 000b - Match is disabled 001b - Reserved 010b - Match first data word with compare word 011b - Match any data word with compare word 100b - Sequential match, first data word 101b - Sequential match, any data word 110b - Match first data word (masked) with compare word (masked) 111b - Match any data word (masked) with compare word (masked)
15-12 —	Reserved
11-8 PCSPOL	Peripheral Chip Select Polarity Specifies the polarity of each PCS pin. Bit *n* in this field (the least-significant bit is bit 0) corresponds to PCS[*n*]. **NOTE:** The entire PCSPOL field is not fully supported in every LPSPI module instance. See the LPSPI chip-specific information. 0000b - Active low 0001b - Active high
7-5 —	Reserved
4 —	Reserved
3 NOSTALL	No Stall Disables a normal operating feature that causes LPSPI, when in Master mode, to stall transfers when the transmit FIFO is empty. This feature prevents transmit FIFO underruns. Writing 1 to this field disables this functionality. 0b - Disable 1b - Enable
2 AUTOPCS	Automatic PCS Enables automatic PCS generation. For correct operation in Slave mode, LPSPI requires the PCS signal to deassert between frames. Writing 1 to this field generates an internal PCS signal at the end of each transfer word when TCR[CPHA] = 1.

Table continues on the next page...

Field	Function
	When this field is 1, SCK must remain idle for at least four LPSPI functional clock cycles, divided by the prescaler (see TCR[PRESCALE]) selected between each word to ensure correct operation. This field is ignored in Master mode. 0b - Disable 1b - Enable
1 SAMPLE	Sample Point Specifies the SCK clock edge on which LPSPI, when in Master mode, samples input data. Writing 1 to this field causes LPSPI to sample input data on a delayed loopback SCK clock edge, which improves the setup time when sampling data (see Clock loopback). In this configuration, the input data setup time in Master mode is equal to the input data setup time in Slave mode. In Slave mode, this field is ignored. **NOTE:** When SAMPLE = 1, both the input and output buffers must be enabled for the SCK pin. 0b - SCK edge 1b - Delayed SCK edge
0 MASTER	Master Mode Specifies the LPSPI operating mode, Master or Slave. This field directly controls the direction of the SCK and PCS pins. 0b - Slave mode 1b - Master mode

39.6.1.10 Data Match 0 (DMR0)

39.6.1.10.1 Offset

Register	Offset
DMR0	30h

39.6.1.10.2 Function

Specifies the match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register when CFGR1[MATCFG] > 0.

39.6.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH0															
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	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.10.4 Fields

Field	Function
31-0 MATCH0	Match 0 Value Specifies the MATCH0 value to be compared against received data.

39.6.1.11 Data Match 1 (DMR1)

39.6.1.11.1 Offset

Register	Offset
DMR1	34h

39.6.1.11.2 Function

Specifies the match data to be used when data matching is enabled. See CFGR1[MATCFG] for the received data matching options.

NOTE

Do not change the value in this register while CFGR1[MATCFG] > 0.

39.6.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH1															
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	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.11.4 Fields

Field	Function
31-0 MATCH1	Match 1 Value Specifies the MATCH1 value to be compared against received data.

39.6.1.12 Clock Configuration (CCR)

39.6.1.12.1 Offset

Register	Offset
CCR	40h

39.6.1.12.2 Function

Contains clock configuration fields that are used only in Master mode; you can only change them when LPSPI is disabled (CR[MEN] = 0).

39.6.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKPCS								PCSSCK							
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	DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.12.4 Fields

Field	Function
31-24 SCKPCS	SCK-to-PCS Delay Configures SCK-to-PCS delay. In Master mode, this field helps you configure the delay from the last SCK edge to PCS negation: • The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. See Figure 39-2 for more information.
23-16 PCSSCK	PCS-to-SCK Delay Configures PCS-to-SCK delay. In Master mode, this field helps you configure the delay from PCS assertion to the first SCK edge: • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle. See Figure 39-2 for more information.
15-8 DBT	Delay Between Transfers Configures the delay between transfers. In Master mode, this field: • Configures the delay 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 selected prescaler (see TCR[PRESCALE]). • The minimum delay is two cycles. • Half of the delay occurs before PCS assertion and the other half of the delay occurs after PCS negation. If the command word is updated between two transfers, then the command word is updated halfway between the PCS negation of the last transfer and PCS assertion of the next transfer. • The command word sets which PCS signal is used (of PCS[3:0]}, the polarity and phase of the SCK signal, and the selected prescaler. • Configures the delay 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 selected prescaler (see TCR[PRESCALE]). • The minimum delay is one cycle.
7-0	SCK Divider

Field	Function
SCKDIV	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 selected prescaler (see TCR[PRESCALE]). • The minimum SCK period is two cycles. • If the SCK period is an odd number of cycles, then the first half of the SCK period is one cycle longer than the second half of the SCK period. Baud rate = function clock ÷ (2^PRESCALE × (SCKDIV + 2))

39.6.1.13 FIFO Control (FCR)

39.6.1.13.1 Offset

Register	Offset
FCR	58h

39.6.1.13.2 Function

Contains the receive FIFO and transmit FIFO watermark values.

39.6.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0						RXWATER	
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								0						TXWATER	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.13.4 Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Field	Function
—	
23-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark Causes LPSPI to set SR[RDF] when the number of words in the receive FIFO is greater than RXWATER. Writing a value equal to or greater than the FIFO size truncates the written value.
15-8 —	Reserved
7-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark Causes LPSPI to set SR[TDF] when the number of words in the transmit FIFO is equal to or less than TXWATER. Writing a value equal to or greater than the FIFO size truncates the written value.

39.6.1.14 FIFO Status (FSR)

39.6.1.14.1 Offset

Register	Offset
FSR	5Ch

39.6.1.14.2 Function

Contains fields that indicate the number of words currently stored in the receive and transmit FIFOs.

39.6.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0					RXCOUNT		
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								0					TXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.14.4 Fields

Field	Function
31-24 —	Reserved
23-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Indicates the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Indicates the number of words currently stored in the transmit FIFO.

39.6.1.15 Transmit Command (TCR)

39.6.1.15.1 Offset

Register	Offset
TCR	60h

39.6.1.15.2 Function

Pushes the data into the transmit FIFO, in the same order as written.

When you write to either this register or to Transmit Data (TDR), each write pushes data into the transmit FIFO. You must write to this register only using 32-bit writes, which are tagged and cause the command register to update; after that the entry reaches the top of the FIFO and LPSPI is enabled. This allows changes to the command word and the transmit data itself to be interleaved. That is, writes to the two registers can be interleaved (write command word, then data word, then command word, and so on). Changing the command word causes all subsequent SPI bus transfers 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 1. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK = 1). Hardware writes 0 to TXMSK when PCS deasserts.
- In Master mode, if the command word is changed before an existing frame has completed, then the existing frame terminates and the command word updates. The command word can be changed during a continuous transfer, if CONTC of the new command word is 1 and the command word is written on a frame size boundary.
- In Slave mode, the command word must be changed only when LPSPI is idle and there is no SPI bus transfer.

Avoid resetting the transmit FIFO after writing to this register; wait for the command register to update from the FIFO first.

Avoid register reading problems: Reading this register returns the current state of the register. Reading this register at the same time that it is loaded from the transmit FIFO can return an incorrect register value. It is recommended to:

- Read this register when the transmit FIFO is empty.
- Read this register more than once and then compare the returned values.

39.6.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CPOL	CPHA	PRESCALE			Reserved	PCS		LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
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				FRAMESZ											
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

39.6.1.15.4 Fields

Field	Function
31 CPOL	Clock Polarity Specifies the value of SCK when it is idle. You can update this field only when PCS is deasserted. See Figure 39-2 for more information. 0b - Inactive low

Table continues on the next page...

Field	Function
	1b - Inactive high
30 CPHA	Clock Phase Indicates whether data is captured or changed on the leading edge of SCK and captured or changed on the following edge of SCK. You can update this field only when PCS is deasserted. See Figure 39-2 for more information. 0b - Captured 1b - Changed
29-27 PRESCALE	Prescaler Value Specifies the division of the LPSPI functional clock. For all SPI bus transfers, this value is applied to Clock Configuration (CCR). You can update this field only when PCS is deasserted. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128
26 —	Reserved
25-24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. This field is updated only when PCS is deasserted. **NOTE:** This entire field is not fully supported in every LPSPI module instance. See the chip-specific LPSPI information. 00b - Transfer using PCS[0] 01b - Transfer using PCS[1] 10b - Transfer using PCS[2] 11b - Transfer using PCS[3]
23 LSBF	LSB First Indicates whether data is transferred with MSB first or LSB first. 0b - MSB first 1b - LSB first
22 BYSW	Byte Swap Swaps the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers). 0b - Disable byte swap 1b - Enable byte swap
21 CONT	Continuous Transfer Enables continuous transfer: • In Master mode, this field keeps PCS asserted at the end of the frame size until a command word is received that starts a new frame. • In Slave mode, when this field is enabled, LPSPI only transmits the first FRAMESZ bits, after which LPSPI transmits received data (assuming a 32-bit shift register) until the next PCS negation. 0b - Disable 1b - Enable
20	Continuing Command

Table continues on the next page...

Field	Function
CONTC	Enables the command word to be changed within a continuous transfer in Master mode: • The initial command word must enable continuous transfer (CONT = 1). • The continuing command must have CONTC = 1. • 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. 0b - Command word for start of new transfer 1b - Command word for continuing transfer
19 RXMSK	Receive Data Mask Masks receive data (receive data is not stored in the receive FIFO). 0b - Normal transfer 1b - Mask receive data
18 TXMSK	Transmit Data Mask Masks transmit data (no data is loaded from the transmit FIFO and the output pin is 3-stated). In Master mode, TXMSK initiates a new transfer that cannot be aborted by another command word. TXMSK automatically transitions to 0 at the end of the transfer. 0b - Normal transfer 1b - Mask transmit data
17-16 WIDTH	Transfer Width Configures serial (1-bit) or parallel transfers. For half-duplex parallel transfers, either RXMSK or TXMSK must be 1. 00b - 1-bit transfer 01b - 2-bit transfer 10b - 4-bit transfer 11b - Reserved
15-12 —	Reserved
11-0 FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1): • The minimum frame size is 8 bits. • The minimum word size is 2 bits; a frame size of 33 bits is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32 bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO contains the remainder bits. For example, a 72-bit transfer consists of three words: the first and second words are 32 bits, and the third word is 8 bits.

39.6.1.16 Transmit Data (TDR)

39.6.1.16.1 Offset

Register	Offset
TDR	64h

39.6.1.16.2 Function

Pushes the data into the transmit FIFO, in the same order that the data is written. You can write to this register using 32-, 16-, or 8-bit writes.

When you write to this register or to Transmit Command (TCR), each write pushes data into the FIFO with zero pushed in unwritten bytes.

39.6.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	DATA															
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	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.16.4 Fields

Field	Function
31-0 DATA	Transmit Data Indicates transmit data. Both 8-bit and 16-bit writes of transmit data zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8-bit and 16-bit writes (to 32 bits) means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

39.6.1.17 Receive Status (RSR)

39.6.1.17.1 Offset

Register	Offset
RSR	70h

39.6.1.17.2 Function

Contains data flow status fields for receive FIFO.

39.6.1.17.3 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														RXEMPTY	SOF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

39.6.1.17.4 Fields

Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty Indicates whether the receive FIFO is empty. 0b - Not empty 1b - Empty
0 SOF	Start of Frame Indicates whether this is the first data word received after PCS assertion. 0b - Subsequent data word 1b - First data word

39.6.1.18 Receive Data (RDR)

39.6.1.18.1 Offset

Register	Offset
RDR	74h

39.6.1.18.2 Function

Pulls the first entry from the receive FIFO.

39.6.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
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	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.6.1.18.4 Fields

Field	Function
31-0 DATA	Receive Data

Chapter 40
Low Power Inter-Integrated Circuit (LPI2C)

40.1 Chip-specific information for this module

40.1.1 Instantiation Information

This device has one LPI2C modules. The LPI2C can remain functional in Stop and VLPS mode provided the clock it is using remains enabled.

Table 40-1. LPI2C Configuration

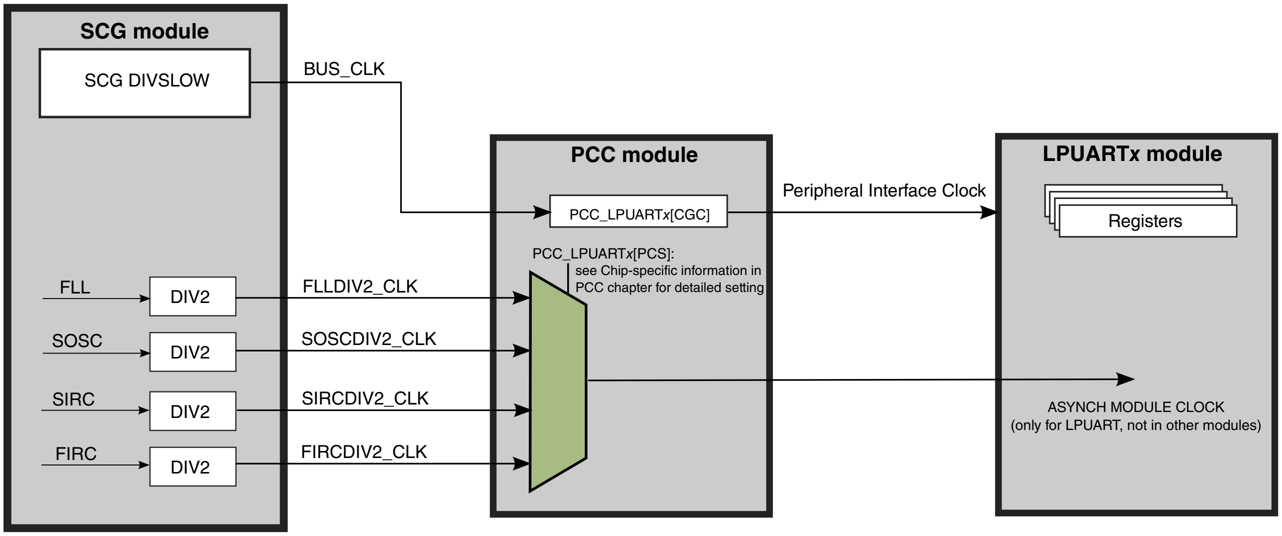
	TX FIFO (word/8bit)	RX FIFO (word/8bit)	SMBus	Slave mode enable
LPI2C0	4	4	Yes	Yes
LPI2C1	4	4	Yes	Yes

40.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

The following figure shows the input clock sources available for this module.

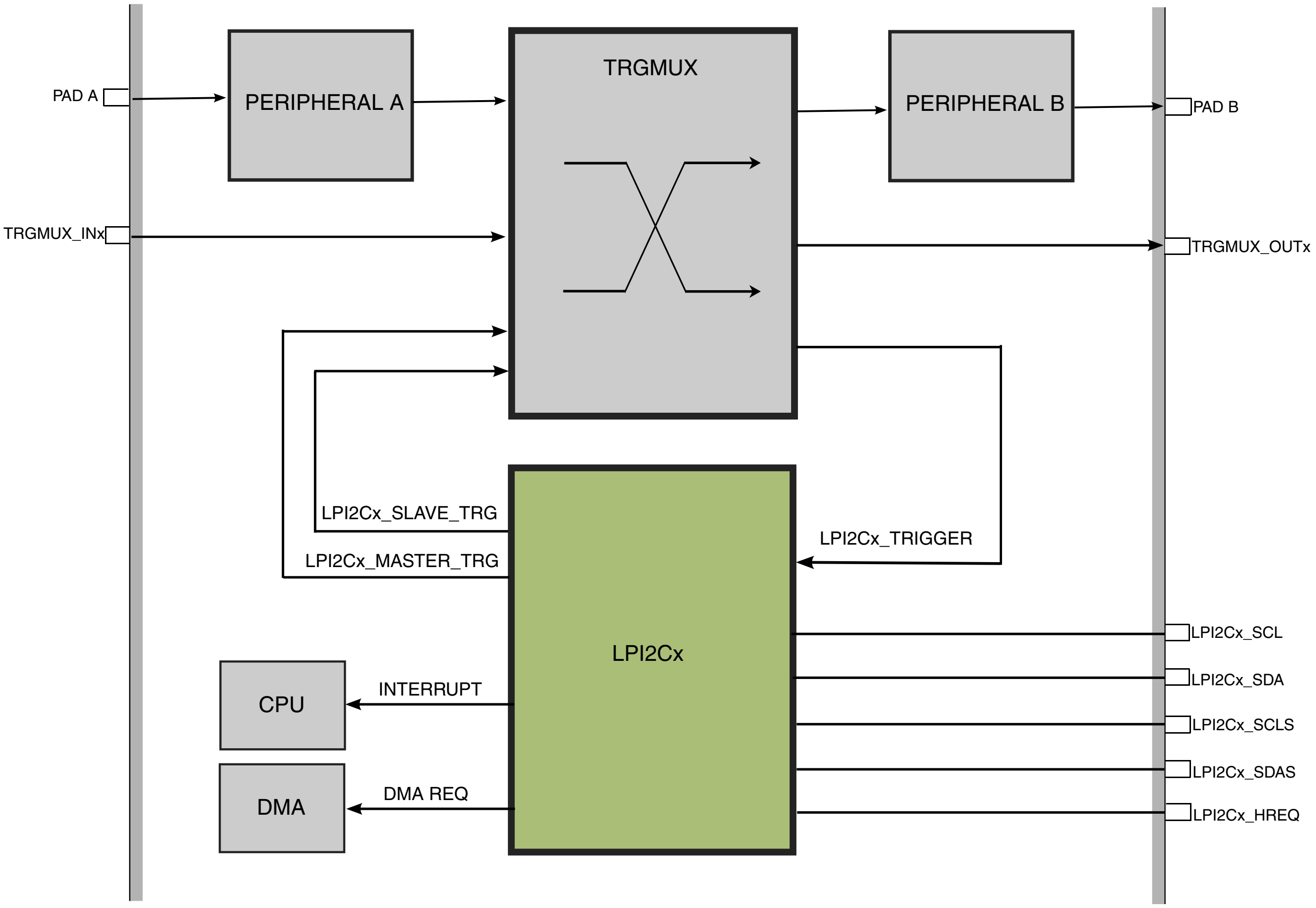
Peripheral Clocking - LPUART, etc.

Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, LPIT, FlexIO, etc.



40.1.3 Inter-connectivity Information

The LPI2C inter-connectivity is shown in following diagram.



40.2 Overview

LPI2C supports an efficient interface to an I2C bus as a controller and target:

- Implements logic support for Standard, Fast, Fast+, Ultra Fast, and High-Speed (HS) modes of operation
- Uses little CPU overhead, with DMA offloading of FIFO register accesses
- Continues operating in STOP modes if an appropriate clock is available

LPI2C also complies with the System Management Bus (SMBus) Specification, version 3. The SMBus is a single-ended simple two-wire bus, which is typically used for low-bandwidth communications.

The Inter-Integrated Circuit (I^2C) serial bus is multi-controller, multi-target, packet-switched, and single-ended, and is often used to attach microcontroller ICs to lower-speed peripheral ICs.

NOTE

Terminology in this chapter has been updated to align with I^{2}C-bus specification, Rev. 7.0, as shown in Table 40-2.

Table 40-2. Updated terms

Updated term	Deprecated term
Controller	Master
Target	Slave

40.2.1 Block diagram

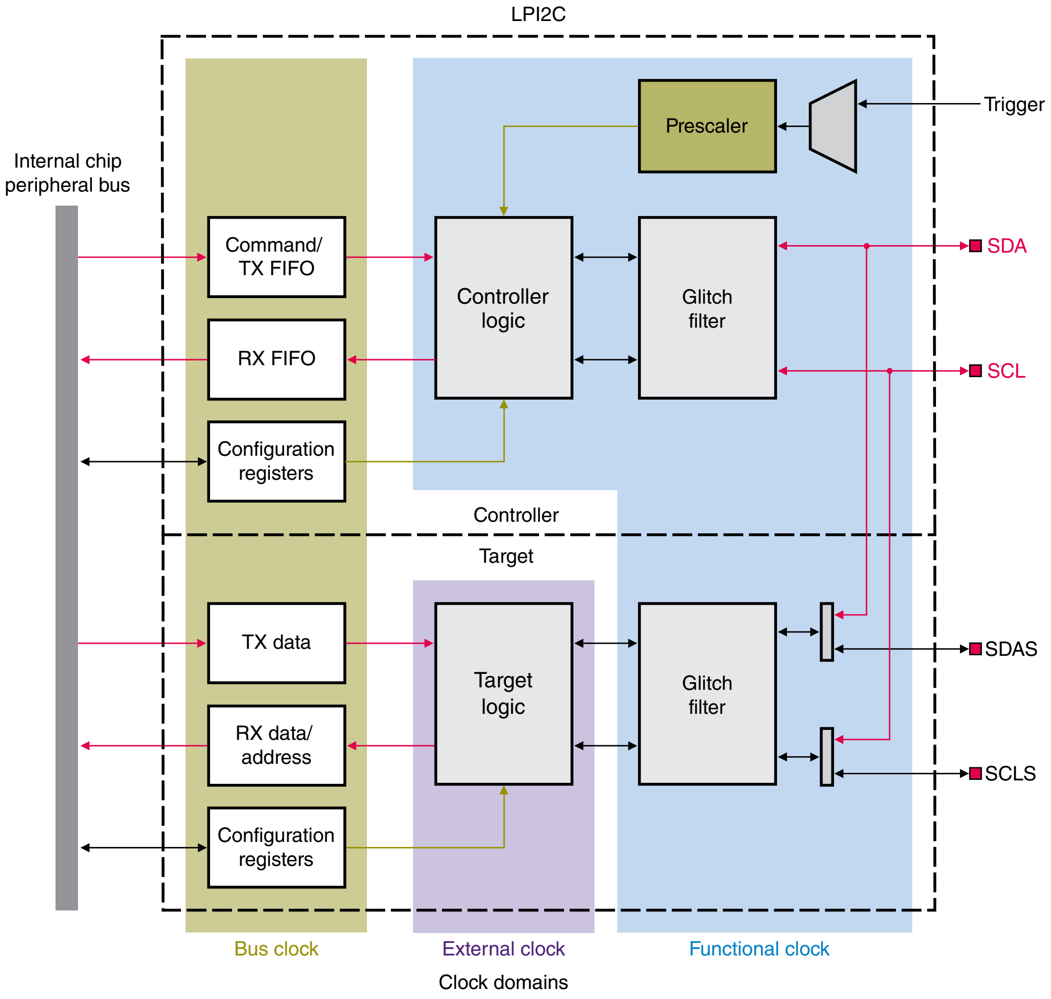


Figure 40-1. Block diagram

40.2.2 Features

LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes
- HS mode in target mode and controller mode
- Multicontroller, including synchronization and arbitration, means that any number of controller nodes can be present. Also, controller and target roles can be changed between messages (after a Stop signal is sent).
- Clock stretching. Used on the SCL line, as an I2C flow control mechanism.
- Arbitration for when the system has more than one controller. When used on the SDA line, ensures that there is only one I2C transmitter at a time.

- General call, seven-bit addressing, and ten-bit addressing
- Software reset, Start byte, and device ID (also require software support)

The LPI2C controller supports:

- Command and transmit FIFO of 4 words (8-bit transmit data + 3-bit command)
- Receive FIFO of 4 words (8-bit receive data).
- Command FIFO waiting for an I2C idle bus before initiating a transfer
- Initiation of repeated Start and Stop conditions and one or more controller-receiver transfers by command FIFO
- Stop condition generation from command FIFO, or automatic generation of Stop condition when the transmit FIFO is empty
- Interrupt generation on data match and unwanted data rejection, via flexible receive data match
- Flags and optional interrupt signals at repeated Start condition, Stop condition, loss of arbitration, unexpected NACK, and command word errors
- Configurable bus idle timeout and pin-stuck-low timeout

The LPI2C target supports:

- Separate I2C target registers to minimize software overhead because of controller or target switching
- 7-bit or 10-bit addressing, address range, SMBus alert, and general call address.
- Transmit data register that supports interrupt or DMA requests
- Receive data register that supports interrupt or DMA requests
- Software-controllable ACK or NACK, with optional clock stretching on ACK or NACK field
- Configurable clock stretching, to avoid transmit-FIFO-underrun and receive-FIFO-overrun errors
- Flags and optional interrupt at end of packet, Stop condition, or bit error detection

40.3 Functional description

40.3.1 Controller mode

The LPI2C controller logic operates independently from the target logic to perform all controller-mode transfers on the I2C bus.

40.3.1.1 Transmit and Command FIFO commands

The transmit FIFO stores command data to initiate various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- Start or repeated Start condition with address byte, expecting ACK or NACK.
- Transmit data. This operation is the default for zero-extended-byte writes to the transmit FIFO.
- Receive 1-256 bytes of data. You can configure this operation to discard received data and not to store it in the receive FIFO.
- Stop condition. You can configure this operation to send a Stop condition when the transmit FIFO is empty.

Multiple transmit and receive commands can be inserted between the Start and Stop conditions. To comply with the I2C specification, transmit and receive commands must not be interleaved. The receive data command and the receive data and discard commands can be interleaved. This interleaving ensures that only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C controller automatically transmits a NACK on the last byte of a receive data command. It transmits the NACK unless the next command in the FIFO is also a receive data command. If the transmit FIFO is empty when a receive data command completes, a NACK is also automatically transmitted.

The LPI2C controller supports 10-bit addressing via a (repeated) Start condition, followed by a transmit data byte containing the second address byte, followed by any number of data bytes with the controller transmit data.

A Start or repeated Start condition expecting a NACK (for example, HS mode controller code) must be followed by a Stop or (repeated) Start condition.

40.3.1.2 Controller operations

When LPI2C is enabled, it monitors the I2C bus to detect when the I2C is idle (MSR[BBF]). If either SCL or SDA are low, the I2C bus is no longer considered idle. The bus becomes idle if a Stop condition is detected or if a bus idle timeout is detected (as configured by MCFGR2[BUSIDLE]). After the bus is idle, if the transmit FIFO is not empty and the host request is asserted or disabled, the LPI2C controller initiates a transfer on the bus. This transfer involves the following steps:

1. Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler (MCFGR1[PRESCALE]).
2. Transmit a Start condition and address byte using the timing configuration in Controller Clock Configuration 0 (MCCR0). If an HS mode transfer is configured,

the timing configuration from Controller Clock Configuration 1 (MCCR1) is used instead.

3. Perform controller transmit or controller receive transfers, as configured by the transmit FIFO.
4. Transmit NACK on the last byte of a controller receive transfer. This action is performed unless the next command in the transmit FIFO is also a receive data command and the transmit FIFO is not empty.
5. Transmit a repeated Start or Stop condition as configured by the transmit FIFO or MCFGR1[AUTOSTOP]. A repeated Start can change which timing configuration register is used.

When the LPI2C controller is disabled, LPI2C continues emptying the transmit FIFO until a Stop condition is transmitted. (The controller could be disabled due to MCR[MEN] being 0, or automatically due to mode entry.) However, LPI2C no longer stalls the I2C bus by waiting for the transmit or receive FIFO. After the transmit FIFO is empty, LPI2C generates a Stop condition automatically.

The LPI2C controller can stall the I2C bus under certain conditions. This stalling results in SCL pulled low continuously on the first bit of a byte, until these conditions change:

- The LPI2C controller is enabled and busy, the transmit FIFO is empty, and MCFGR1[AUTOSTOP] is 0. The LPI2C controller continues to stall the bus until the transmit FIFO is loaded with more data.
- The LPI2C controller is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and the receive FIFO is full. The LPI2C controller continues to stall the I2C bus until the receive FIFO is emptied.

40.3.1.3 Receive FIFO and data matching

The receive FIFO stores receive data during controller-receiver transfers. You can configure the LPI2C controller to discard received data instead of storing it in the receive FIFO. This option is configured via the command word in the transmit FIFO.

Received data supports a receive data match function that can match received data against one of two bytes, or against a masked data byte. You can configure the data match function to compare only the first one or two data words received since the last (repeated) Start condition. Received data that is already discarded due to the command word cannot cause a data match. It delays the match on the first data word received until after the discarded data is received.

You can configure the receiver match function to discard all received data until a data match is detected, using MCFGR0[RDMO]. Following a data match, write 0 to MCFGR0[RDMO] before writing 0 to MSR[DMF] to allow all subsequent data to be received.

40.3.1.4 Timing parameters

The LPI2C controller can configure the following timing parameters. Parameters are configured separately for HS mode (Controller Clock Configuration 1 (MCCR1)) and other modes (Controller Clock Configuration 0 (MCCR0)). This separation allows the HS mode controller code to be sent using regular timing parameters. Then it allows a switch to HS mode timing (following a repeated Start) until the next STOP condition.

Configure the LPI2C controller timing parameters, measured in LPI2C functional clock cycles, as shown in Table 40-3. You must configure these parameters to meet the I2C timing specification for the required mode.

Table 40-3. Timing parameters

I2C specification timing parameter	I2C specification timing symbol	LPI2C timing parameter (in LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) × (2 ^ PRESCALE)
Hold time (repeated) Start condition	tHD:STA	(SETHOLD +1) × (2 ^ PRESCALE)
Low period of the SCL clock	tLOW	(CLKLO + 1) × (2 ^ PRESCALE)
High period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Setup time for a repeated Start condition or Stop condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) × (2 ^ PRESCALE)
Data hold time	tHD:DAT	(DATAVD + 1) × (2 ^ PRESCALE)
Data setup time	tSU:DAT	(SDA_LATENCY + 1) × (2 ^ PRESCALE)
Bus free time between a Stop and Start condition	tBUF	(CLKLO + 1 + SDA_LATENCY) × (2 ^ PRESCALE)
Data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) × (2 ^ PRESCALE)

Table 40-4 defines the latency parameters. These parameters assume that the risetime is less than one LPI2C functional clock cycle. The risetime depends on a number of factors, including the I/O propagation delay, the I2C bus loading, and the external pullup resistor sizing. A larger risetime increases the number of cycles that the signal takes to propagate through the synchronizer (and glitch filter), which increases the latency.

Table 40-4. Synchronization latency

Timing parameter	Timing definition
SCL_LATENCY	ROUNDDOWN ((2 + FILTSCL + SCL_RISETIME) ÷ (2 ^ PRESCALE))
SDA_LATENCY	ROUNDDOWN ((2 + FILTSDA + SDA_RISETIME) ÷ (2 ^ PRESCALE))

The following timing restrictions must be enforced to avoid unexpected Start or Stop conditions on the I2C bus. These restrictions also avoid unexpected Start or Stop conditions detected by the LPI2C controller. The timing restrictions can be summarized as "SDA cannot change when SCL is high outside a transmitted (repeated) Start or Stop condition."

Table 40-5. LPI2C timing parameter restrictions

Timing parameter	Minimum	Maximum	Comment
CLKLO	03h	—	CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	01h	—	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	02h	—	SETHOLD × (2 ^ PRESCALE) > SDA_LATENCY
DATAVD	01h	CLKLO – SDA_LATENCY – 1	Configure DATAVD to meet the data hold requirement in the I2C specification
FILTSCL	00h	[CLKLO × (2 ^ PRESCALE)] – 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)
FILTSDA	FILTSCL	[CLKLO × (2 ^ PRESCALE)] – 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO + SETHOLD + 2) × 2	—	Must also be greater than (CLKHI + 1)

See the UM10204, *I2C-bus specification and user manual*.

See Application information for example LPI2C timing configurations.

40.3.1.5 Error conditions

The LPI2C controller monitors errors while it is active. The following conditions generate an error flag and block a new Start condition from being sent, until the flag is cleared by software:

- A Start or Stop condition is detected and is not generated by the LPI2C controller (MSR[ALF] becomes 1).
- Transmitting data on SDA and different values are received (MSR[ALF] becomes 1).

- NACK is detected when transmitting data, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- NACK is detected and is expecting ACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- ACK is detected and is expecting NACK for the address byte, and MCFGR1[IGNACK] is 0 (MSR[NDF] becomes 1).
- Transmit FIFO is requesting to transmit or receive data without a Start condition (MSR[FEF] becomes 1).
- SCL (or SDA if MCFGR1[TIMECFG] is 1) is low for (MCFGR2[TIMELOW] × 256) prescaler cycles without a pin transition (MSR[PLTF] becomes 1).

You must respond to MSR[PLTF] to terminate the existing command. You can terminate the command cleanly by writing 0 to MCR[MEN], or you can terminate it abruptly by writing 1 to MCR[RST].

You can use MCFGR2[BUSIDLE] to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE + 1) prescaler cycles. The bus is considered idle when the LPI2C controller is first enabled. When BUSIDLE is configured greater than zero, then SCL or SDA must be high for (BUSIDLE + 1) prescaler cycles before the I2C bus is considered idle.

40.3.1.6 Pin configuration

Configuration	Description
Open-drain support	The LPI2C controller defaults to open-drain configuration of the SDA and SCL pins. Support for true open drain depends on the specific device, and requires the pins where LPI2C pins are muxed to support true open drain.
HS mode support	Support for HS mode depends on the specific device. This mode requires the SCL pin to support the current source pullup required in the I2C specification.
Ultra-Fast mode support	The LPI2C controller supports the output-only push-pull function required for I2C Ultra-Fast mode using the SDA and SCL pins. Support for Ultra-Fast mode also requires MCFGR1[IGNACK] to be 1.
Push-pull two-wire support	A push-pull two-wire configuration is available to the LPI2C controller. If LPI2C is the only controller and all I2C pins on the bus are at the same voltage, this configuration may support a partial HS mode. A partial HS mode, not a full HS mode, because this configuration actively drives high rather than enabling a current service pull-up. This configuration sets the SCL pin as push-pull for every clock except the ninth clock pulse, to allow HS-mode-compatible targets to perform clock stretching. In this mode, the SDA pin is tristated for controller-receive data bits and controller-transmit ACK/NACK bits, and is configured as push-pull at other times. To avoid the risk of contention when SDA is push-pull, configure the pin for open-drain operation, as part of the device-specific configuration.
Push-pull four-wire support	The push-pull four-wire configuration separates the SCL input data and output data into separate pins. It also separates the SDA input data and output data into separate pins. The SCL/SDA pins are used for input data; the SCLS/SDAS pins are used for output data, with configurable polarity. This configuration simplifies external connections when connecting the LPI2C to the I2C bus through external level shifters or discrete components. When using this four-wire configuration, the LPI2C controller logic and LPI2C target logic cannot connect to separate I2C buses.

40.3.2 Target mode

To perform all target mode transfers on the I2C bus, the LPI2C target logic operates independently from the LPI2C controller logic.

40.3.2.1 Address matching

You can configure the LPI2C target:

- To match one of two addresses, using either 7-bit or 10-bit addressing modes for each address.
- To match a range of addresses in either 7-bit or 10-bit addressing modes.
- To match the general call address and generate appropriate flags.
- To match the SMBus alert address and generate appropriate flags.
- To detect the HS mode controller code address, and to disable the digital filters and output valid delay time until the next Stop condition is detected.

After a valid address is matched, the LPI2C target automatically performs target-transmit or target-receive transfers until:

- A NACK is detected (unless SCFGR1[IGNACK] becomes 1).
- A bit error is detected (the LPI2C target is driving SDA, but a different value is sampled).
- A (repeated) Start or Stop condition is detected.

40.3.2.2 Transmit and receive data

Target Transmit Data (STDR) and Target Receive Data (SRDR) are double-buffered and only update during a target-transmit and target-receive transfer, respectively.

You can configure the target address that was received to be read from SRDR (for example, when using DMA to transfer data) or from Target Address Status (SASR).

You can configure STDR to request data only after a target-transmit transfer is detected. You can also configure it to request new data whenever STDR is empty.

Write to STDR only when SSR[TDF] is set.

Read SRDR only when SSR[RDF] is set, or when SSR[AVF] is set and SCFGR1[RXCFG] = 1.

Read SASR only when SSR[AVF] is set.

40.3.2.3 Clock stretching

The LPI2C target supports many configurable options for clock stretching. You can configure these conditions to perform clock stretching:

- SSR[AVF] is set during the ninth clock pulse of the address byte.
- SSR[TDF] is set during the ninth clock pulse of a target-transmit transfer.
- SSR[RDF] is set during the ninth clock pulse of a target-receive transfer.
- SSR[TAF] is set during the eighth clock pulse of an address byte or a target-receive transfer. In HS mode, this option is disabled.
- Clock stretching can be extended for a number of cycles equal to the value of SCFGR2[CLKHOLD] cycles. This stretching allows additional setup time to sample the SDA pin externally. In HS mode, this option is disabled.

When clock stretching is enabled, clock stretching extends for one peripheral bus clock cycle after SDA updates, unless extended by the SCFGR2[CLKHOLD] configuration.

40.3.2.4 Timing parameters

The LPI2C target can configure the following timing parameters:

- 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

These parameters are disabled when SCR[FILTEN] is 0, when SCR[FILTDZ] is 1 in Doze mode, and when LPI2C target detects HS mode. When disabled, the LPI2C target is clocked directly from the I2C bus. In this case, the target may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

The LPI2C target places the following restrictions on the timing parameters:

- You must configure SCFGR2[FILTSDA] to be greater than or equal to SCFGR2[FILTSCL] (unless compensating for board level skew between SDA and SCL).
- You must configure SCFGR2[DATAVD] to be less than the minimum SCL low period.

40.3.2.5 Error conditions

The LPI2C target can flag the following error conditions:

- SSR[BEF] is set when the LPI2C target is driving SDA but it samples a different value than what is expected.
- SSR[FEF] is set due to a transmit data underrun or a receive data overrun. To eliminate the possibility of underrun and overrun, enable clock stretching.
- SSR[FEF] is also set due to an address overrun, but only when SCFGR1[RXCFG] is 1. To eliminate the possibility of overrun, enable clock stretching.

The LPI2C target does not implement a timeout due to SCL or SDA being stuck low. If this detection is required, use the LPI2C controller logic so you can reset the LPI2C target when this condition is detected.

40.3.3 Low-power modes

LPI2C remains functional during low-power modes, if MCR[DOZEN] = 0 and LPI2C uses an external or internal clock source that remains enabled. LPI2C can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPI2C to be disabled in low-power modes when MCR[DOZEN] = 1. In this case, LPI2C waits for the current transfer to complete any pending operation.

NOTE

See the chip-specific information for low-power modes available on your chip.

40.3.4 Debug mode

Table 40-6. Debug mode

Mode	LPI2C operation
Debug	If MCR[DBGEN] = 1, can continue operating in Debug mode.

40.3.5 Peripheral triggers

The connection of the LPI2C peripheral triggers to other peripherals depends upon the specific device being used.

Table 40-7. LPI2C triggers

Trigger	Description
Controller output trigger	Generates an output trigger that can be connected to other peripherals on the device. The controller output trigger asserts on either a repeated Start or a Stop condition. The trigger remains asserted for one cycle of the LPI2C functional clock divided by MCFGR1[PRESCALE].
Target output trigger	Generates an output trigger that can be connected to other peripherals on the device. The target output trigger asserts on either a repeated Start or a Stop condition that occurs after a target address match. The target output trigger remains asserted until the next target SCL pin negation.
Input trigger	Controls the start of a LPI2C bus transfer. The input trigger is synchronized. To be detected, the input trigger must assert for at least two cycles of the LPI2C functional clock divided by the value of MCFGR1[PRESCALE].

40.3.6 Clocking

Table 40-8. LPI2C clocks

Clock	Description
LPI2C functional clock	The LPI2C functional clock is asynchronous to the bus clock. It can remain enabled in low-power modes to support I2C bus transfers by the LPI2C controller. The functional clock is also used by the LPI2C target to support digital filter and data hold time configurations. The LPI2C controller divides the functional clock by a prescaler (MCFGR1[PRESCALE]) and the resulting frequency must be at least eight times faster than the I2C bus bandwidth.
External clock	The LPI2C target logic is clocked directly from the external pins. These pins are SCL and SDA, or SCLS and SDAS if the controller and target are implemented on separate pins). This clocking allows the LPI2C target to remain operational, even when the LPI2C functional clock is disabled. **NOTE:** If the LPI2C functional clock is disabled, the LPI2C target digital filter must be disabled. This condition can affect compliance with some timing parameters of the I2C specification, such as data hold time.
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 controller and target registers.

For chip-specific clocking information, see the Clocking chapter.

40.3.7 Reset

Table 40-9. LPI2C resets

Reset	Description
Chip reset	The logic and registers for the LPI2C controller and target are reset to their default states after a chip reset.
Software reset	The LPI2C controller implements a software reset field in its control register. MCR[RST] resets all controller logic and registers to their default states, except for Controller Control (MCR) itself. The LPI2C target implements a software reset field in its control register. SCR[RST] resets all target logic and registers to their default states, except for Target Control (SCR) itself.
FIFO reset	The LPI2C controller implements write-only control fields that reset the transmit FIFO (MCR[RTF]) and receive FIFO (MCR[RRF]). After a FIFO is reset, that FIFO is empty. The LPI2C target implements write-only control fields that reset the transmit data register (SCR[RTF]) and receive data register (SCR[RRF]). After a data register is reset, that data register is empty.

40.3.8 Interrupts and DMA requests

Depending on the configuration, interrupts and DMA requests can be combined:

- LPI2C controller and target interrupts
- LPI2C controller and target transmit DMA requests
- LPI2C controller and target receive DMA requests

40.3.8.1 Controller mode

Table 40-10 lists the Controller mode sources that can generate LPI2C controller interrupts and LPI2C controller transmit and receive DMA requests.

Table 40-10. Controller interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (MSR[TDF])	Data can be written to transmit FIFO, as configured by MFCR[TXWATER].	Y	TX	Y
Receive Data Flag (MSR[RDF])	Data can be read from the receive FIFO, as configured by MFCR[RXWATER].	Y	RX	Y
End Packet Flag (MSR[EPF])	Controller has transmitted a repeated Start or Stop condition.	Y	N	Y
Stop Detect Flag (MSR[SDF])	Controller has transmitted a Stop condition .	Y	N	Y
NACK Detect Flag (MSR[NDF])	During an address byte, the controller expects an ACK but detects a NACK.	Y	N	Y

Table continues on the next page...

Table 40-10. Controller interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
	During an address byte, the controller expects a NACK but detects an ACK. During a controller-transmitter data byte, the controller detects a NACK.			
Arbitration Lost Flag (MSR[ALF])	The controller lost arbitration due to a Start or Stop condition detected at the wrong time, or the controller was transmitting data but received data different from the data that was transmitted.	Y	N	Y
FIFO Error Flag (MSR[FEF])	The controller expects a Start condition in the command FIFO, but the next entry in the command FIFO is not a Start condition.	Y	N	Y
Pin Low Timeout Flag (MSR[PLTF])	Pin low timeout is enabled and SCL (or SDA, if configured) is low for longer than the configured timeout.	Y	N	Y
Data Match Flag (MSR[DMF])	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry.	Y	N	Y
Controller Busy Flag (MSR[MBF])	LPI2C controller is busy transmitting or receiving data.	N	N	N
Bus Busy Flag (MSR[BBF])	LPI2C controller is enabled and activity is detected on the I2C bus, but no Stop condition is detected and no bus idle timeout (if enabled) occurred.	N	N	N

40.3.8.2 Target mode

Table 40-11 lists the target mode sources that can generate LPI2C target interrupts and LPI2C target transmit and receive DMA requests.

Table 40-11. Target interrupts and DMA requests

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Transmit Data Flag (SSR[TDF])	Data can be written to Target Transmit Data (STDR).	Y	TX	Y
Receive Data Flag (SSR[RDF])	Data can be read from Target Receive Data (SRDR).	Y	RX	Y
Address Valid Flag (SSR[AVF])	Address can be read from Target Address Status (SASR).	Y	RX	Y
Transmit ACK Flag (SSR[TAF])	ACK or NACK can be written to Target Transmit ACK (STAR).	Y	N	Y

Table continues on the next page...

Table 40-11. Target interrupts and DMA requests (continued)

Status flag	Description	Can generate		
		Interrupt?	DMA request?	Low-power wake-up?
Repeated Start Flag (SSR[RSF])	Target has detected an address match followed by a repeated Start condition.	Y	N	Y
Stop Detect Flag (SSR[SDF])	Target has detected an address match followed by a Stop condition.	Y	N	Y
Bit Error Flag (SSR[BEF])	Target was transmitting data, but received data is different from what was transmitted.	Y	N	Y
FIFO Error Flag (SSR[FEF])	This flag is set by: • Transmit data underrun • Receive data overrun • Address status overrun when SCFGR1[RXCFG] = 1 This flag can only be set when clock stretching is disabled.	Y	N	Y
Address Match 0 Flag (SSR[AM0F])	Target detected an address match SAMR[ADDR0].	Y	N	N
Address Match 1 Flag (SSR[AM1F])	Target detected an address match with SAMR[ADDR1] or using an address range.	Y	N	N
General Call Flag (SSR[GCF])	Target detected an address match with the general call address.	Y	N	N
SMBus Alert Response Flag (SSR[SARF])	Target detected an address match with the SMBus alert address.	Y	N	N
Target Busy Flag (SSR[SBF])	LPI2C target is busy receiving an address byte or is transmitting or receiving data.	N	N	N
Bus Busy Flag (SSR[BBF])	LPI2C target is enabled and a Start condition is detected on I2C bus, but no Stop condition detected.	N	N	N

40.4 External signals

Table 40-12. External signals

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
SCL	LPI2C clock line	SCL	In Four-Wire mode, this pin is the SCL input pin.	Input or output
SDA	LPI2C data line	SDA	In Four-Wire mode, this pin is the SDA input pin.	Input or output
SCLS	Secondary I2C clock line	Not used	In Four-Wire mode, this pin is the SCLS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SCL pin.	Input or output

Table continues on the next page...

Table 40-12. External signals (continued)

Signal	Name	Two-wire scheme	Four-wire scheme	Direction
SDAS	Secondary I2C data line	Not used	In Four-Wire mode, this pin is the SDAS output pin. If LPI2C controller/target are configured to use separate pins, then this pin is the LPI2C target SDA pin.	Input or output

Figure 40-2 shows the two-signal connection.

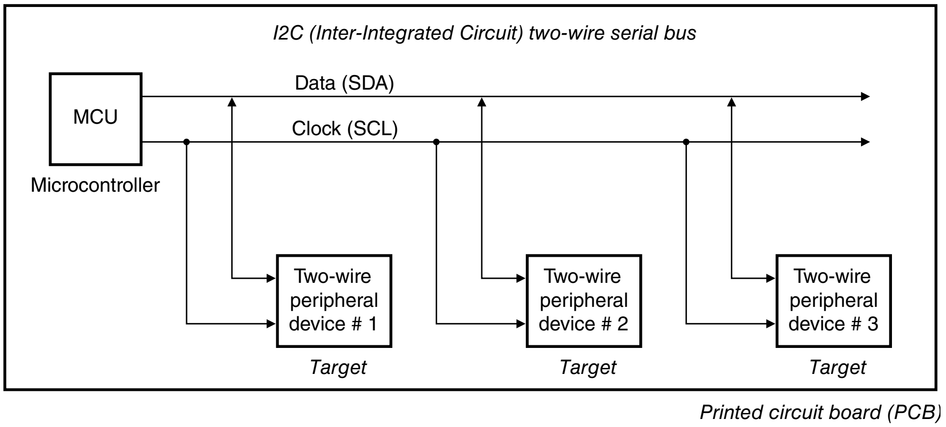

I2C (Inter-Integrated Circuit) two-wire serial bus
Data (SDA)
MCU
Clock (SCL)
Microcontroller
Two-wire peripheral device # 1
Target
Two-wire peripheral device # 2
Target
Two-wire peripheral device # 3
Target
Printed circuit board (PCB)

Figure 40-2. I^{2}C two-wire serial bus

Figure 40-3 shows a possible four-signal connection.

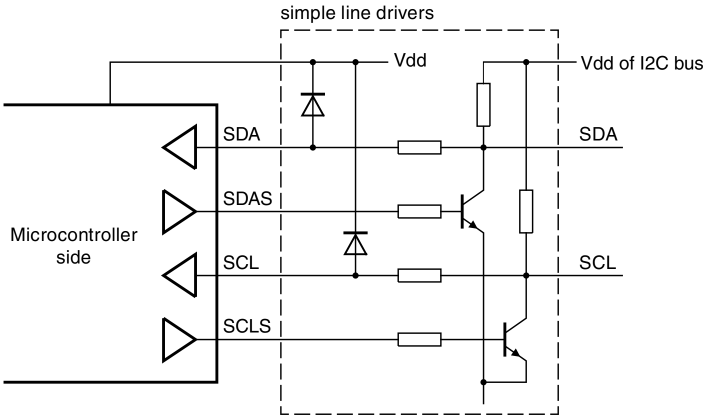


Figure 40-3. I^{2}C four-wire serial bus

40.5 Initialization

To initialize the LPI2C controller:

1. Configure Controller Configuration 0 (MCFGR0) –Controller Configuration 3 (MCFGR3) as required by the application.
2. Configure Controller Clock Configuration 0 (MCCR0) and Controller Clock Configuration 1 (MCCR1) to satisfy the timing requirements of the I2C mode supported by the application.
3. Enable controller interrupts and DMA requests as required by the application.
4. Enable the LPI2C controller by writing 1 to MCR[MEN].

To initialize the LPI2C target:

1. Configure Target Address Match (SAMR) with the I2C address of the target location on the I2C bus.
2. Configure Target Configuration 1 (SCFGR1) as required by the application.
3. Configure Target Configuration 2 (SCFGR2) to satisfy the timing requirements of the I2C mode supported by the application.
4. Enable target interrupts and DMA requests as required by the application.
5. Enable the LPI2C target by writing 1 to SCR[SEN].

40.6 Application information

Configure the I2C timing parameters to meet the requirements of the I2C specification. This configuration depends on the supported mode and LPI2C functional clock frequency. When switching between two modes using different clock configuration registers (for example, Fast mode and HS mode), MCFGR1[PRESCALE] must remain constant between the modes.

Table 40-13. Example timing configurations

I2C mode	Clock frequency	Baud rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Standard	8 MHz	100 kbit/s	0h	0h/0h	24h	28h	24h	02h
Standard	48 MHz	100 kbit/s	2h	1h/1h	37h	3Fh	37h	03h
Standard	60 MHz	100 kbit/s	2h	1h/1h	45h	50h	44h	04h
Fast	8 MHz	400 kbit/s	0h	0h/0h	04h	0Bh	05h	02h
Fast+	8 MHz	1 Mbit/s	0h	0h/0h	02h	03h	01h	01h
Fast	48 MHz	400 kbit/s	0h	1h/1h	1Dh	3Eh	35h	0Fh
Fast	48 MHz	400 kbit/s	2h	1h/1h	07h	11h	0Bh	03h
Fast+	48 MHz	1 Mbit/s	2h	1h/1h	03h	06h	04h	04h
HS	48 MHz	3.2 Mbit/s	0h	0h/0h	07h	08h	03h	01h
Fast	60 MHz	400 kbit/s	1h	2h/2h	11h	28h	1Fh	08h
Fast+	60 MHz	1 Mbit/s	1h	2h/2h	07h	0Fh	0Bh	01h
HS	60 MHz	3.33 Mbit/s	1h	0h/0h	04h	04h	02h	01h

40.7 Memory map and registers

40.7.1 LPI2C register descriptions

Writing to a read-only register or reading from a write-only register can cause bus errors. This module does not check whether programmed values in the registers are correct; you must ensure that valid programmed values are written to the registers.

40.7.1.1 LPI2C memory map

LPI2C0 base address: 4006_6000h

LPI2C1 base address: 4006_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0100_0003h
4h	Parameter (PARAM)	32	R	0000_0202h
10h	Controller Control (MCR)	32	RW	0000_0000h
14h	Controller Status (MSR)	32	RW	0000_0001h
18h	Controller Interrupt Enable (MIER)	32	RW	0000_0000h
1Ch	Controller DMA Enable (MDER)	32	RW	0000_0000h
20h	Controller Configuration 0 (MCFGR0)	32	RW	0000_0000h
24h	Controller Configuration 1 (MCFGR1)	32	RW	0000_0000h
28h	Controller Configuration 2 (MCFGR2)	32	RW	0000_0000h
2Ch	Controller Configuration 3 (MCFGR3)	32	RW	0000_0000h
40h	Controller Data Match (MDMR)	32	RW	0000_0000h
48h	Controller Clock Configuration 0 (MCCR0)	32	RW	0000_0000h
50h	Controller Clock Configuration 1 (MCCR1)	32	RW	0000_0000h
58h	Controller FIFO Control (MFCR)	32	RW	0000_0000h
5Ch	Controller FIFO Status (MFSR)	32	R	0000_0000h
60h	Controller Transmit Data (MTDR)	32	W	0000_0000h
70h	Controller Receive Data (MRDR)	32	R	0000_4000h
110h	Target Control (SCR)	32	RW	0000_0000h
114h	Target Status (SSR)	32	RW	0000_0000h
118h	Target Interrupt Enable (SIER)	32	RW	0000_0000h
11Ch	Target DMA Enable (SDER)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
124h	Target Configuration 1 (SCFGR1)	32	RW	0000_0000h
128h	Target Configuration 2 (SCFGR2)	32	RW	0000_0000h
140h	Target Address Match (SAMR)	32	RW	0000_0000h
150h	Target Address Status (SASR)	32	R	0000_4000h
154h	Target Transmit ACK (STAR)	32	RW	0000_0000h
160h	Target Transmit Data (STDR)	32	W	0000_0000h
170h	Target Receive Data (SRDR)	32	R	0000_4000h

40.7.1.2 Version ID (VERID)

40.7.1.2.1 Offset

Register	Offset
VERID	0h

40.7.1.2.2 Function

Contains version numbers for the module design and feature set.

40.7.1.2.3 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	0

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	1	1

40.7.1.2.4 Fields

Field	Function
31-24	Major Version Number

Table continues on the next page...

Field	Function
MAJOR	Returns the major version number for the module design specification.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification.
15-0 FEATURE	Feature Specification Number Returns the feature set number. 0000_0000_0000_0010b - Controller only, with standard feature set 0000_0000_0000_0011b - Controller and target, with standard feature set

40.7.1.3 Parameter (PARAM)

40.7.1.3.1 Offset

Register	Offset
PARAM	4h

40.7.1.3.2 Function

Contains parameter values implemented in the module.

40.7.1.3.3 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				MRXFIFO				0				MTXFIFO			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

40.7.1.3.4 Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Field	Function
15-12 —	Reserved
11-8 MRXFIFO	Controller Receive FIFO Size Configures the number of words in the controller receive FIFO to $2^{MRXFIFO}$.
7-4 —	Reserved
3-0 MTXFIFO	Controller Transmit FIFO Size Configures the number of words in the controller transmit FIFO to $2^{MTXFIFO}$.

40.7.1.4 Controller Control (MCR)

40.7.1.4.1 Offset

Register	Offset
MCR	10h

40.7.1.4.2 Function

Contains resets, debug enable, and other controller control settings.

40.7.1.4.3 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						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

40.7.1.4.4 Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Resets the receive FIFO. 0b - No effect 1b - Reset receive FIFO
8 RTF	Reset Transmit FIFO Resets the transmit FIFO. 0b - No effect 1b - Reset transmit FIFO
7-4 —	Reserved
3 DBGEN	Debug Enable Enables the controller in Debug mode. 0b - Disable 1b - Enable
2 DOZEN	Doze Mode Enable Enables the controller in Doze mode. 0b - Enable 1b - Disable
1 RST	Software Reset Resets all internal controller logic and registers except Controller Control (MCR). This field remains 1 (enabled) until you write 0 to it. The reset takes effect immediately and remains asserted until negated by software. There is no minimum delay required before clearing the software reset. 0b - No effect 1b - Reset
0 MEN	Controller Enable Enables the controller logic. 0b - Disable 1b - Enable

40.7.1.5 Controller Status (MSR)

40.7.1.5.1 Offset

Register	Offset
MSR	14h

40.7.1.5.2 Function

Contains status flags for transmit and receive data, for start and stop conditions, and for bus and controller busy or idle status.

40.7.1.5.3 Diagram

Bits	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

Bits	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

40.7.1.5.4 Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Specifies whether the I2C bus is busy. 0b - Idle 1b - Busy
24 MBF	Controller Busy Flag Specifies whether the I2C controller is busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 —	Reserved
14 DMF	Data Match Flag Indicates whether the received data matches MDMR[MATCH0] or MDMR[MATCH1] (as configured by MCFGR1[MATCFG]). Received data discarded due to MTDR[CMD] does not cause this flag to set. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Matching data not received

Table continues on the next page...

Field	Function
	1b - Matching data received When writing 0b - No effect 1b - Clear the flag
13 PLTF	Pin Low Timeout Flag Indicates whether pin low timeout has occurred. Sets when the SCL or SDA input is low for more than the number of PINLOW cycles defined by MCFGR3[PINLOW], even when the LPI2C controller is idle. You must resolve the pin low condition via software. PLTF cannot be cleared as long as the pin low timeout continues. Before LPI2C can initiate a Start condition, you must clear this flag. See MCFGR1[TIMECFG] for the SCL and/or SDA timeout settings. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Pin low timeout did not occur 1b - Pin low timeout occurred When writing 0b - No effect 1b - Clear the flag
12 FEF	FIFO Error Flag Detects the LPI2C controller's attempt to send or receive data without first generating a (repeated) Start condition. This error can occur when the transmit FIFO underflows when MCFGR1[AUTOSTOP] = 1. When this flag is set, the LPI2C controller sends a Stop condition (if busy). The controller does not initiate a new Start condition until the flag is cleared. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
11 ALF	Arbitration Lost Flag Indicates whether arbitration is lost. Either of these conditions sets this flag: • The LPI2C controller transmits a logic 1 and detects a logic 0 on the I2C bus. • The LPI2C controller detects a Start or Stop condition when the LPI2C controller is transmitting data. When ALF is set, the LPI2C controller releases the I2C bus (goes idle), and the LPI2C controller does not initiate a new Start condition until the ALF is cleared. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Controller did not lose arbitration 1b - Controller lost arbitration When writing 0b - No effect

Table continues on the next page...

Field	Function
	1b - Clear the flag
10 NDF	NACK Detect Flag Indicates whether an unexpected NACK has been detected. This flag is set when the LPI2C controller detects a NACK it was not expecting when transmitting an address or data. When set, the controller does not initiate a new Start condition until this flag is cleared. If a NACK is expected for a given address (as configured by the command word), this flag is set if a NACK is not generated. When this flag is set, the LPI2C controller automatically transmits a Stop condition if MCFGR1[AUTOSTOP] = 1, or if the transmit FIFO is not empty. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No unexpected NACK detected 1b - Unexpected NACK detected When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C controller has generated a Stop condition. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No Stop condition generated 1b - Stop condition generated When writing 0b - No effect 1b - Clear the flag
8 EPF	End Packet Flag Indicates whether the LPI2C controller has generated a repeated Start condition or a Stop condition. When the controller first generates a Start condition, this flag is not set. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No Stop or repeated Start generated 1b - Stop or repeated Start generated When writing 0b - No effect 1b - Clear the flag
7-2 —	Reserved
1 RDF	Receive Data Flag Indicates whether the receive data is ready. This flag is set when the number of words in the receive FIFO is greater than MFCR[RXWATER]. 0b - Receive data not ready 1b - Receive data ready
0	Transmit Data Flag

Field	Function
TDF	Indicates whether transmit data is requested. This flag is set when the number of words in the transmit FIFO is equal or less than MFCR[TXWATER]. 0b - Transmit data not requested 1b - Transmit data requested

40.7.1.6 Controller Interrupt Enable (MIER)

40.7.1.6.1 Offset

Register	Offset
MIER	18h

40.7.1.6.2 Function

Contains enables for:

- Transmit and receive data interrupts
- Start, Stop, and NACK detection interrupts
- DMA interrupts

40.7.1.6.3 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	DMIE	PLTIE	FEIE	ALIE	NDIE	SDIE	EPIE	0						RDIE	TDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.6.4 Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Field	Function
15 —	Reserved
14 DMIE	Data Match Interrupt Enable Enables interrupt for data match. 0b - Disable 1b - Enable
13 PLTIE	Pin Low Timeout Interrupt Enable Enables interrupt for pin-low timeout. 0b - Disable 1b - Enable
12 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable
11 ALIE	Arbitration Lost Interrupt Enable Enables interrupt for arbitration lost. 0b - Disable 1b - Enable
10 NDIE	NACK Detect Interrupt Enable Enables interrupt for NACK detection. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 EPIE	End Packet Interrupt Enable Enables interrupt for end packet. 0b - Disable 1b - Enable
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

40.7.1.7 Controller DMA Enable (MDER)

40.7.1.7.1 Offset

Register	Offset
MDER	1Ch

40.7.1.7.2 Function

Contains DMA transmit, request, and receive enables.

40.7.1.7.3 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														RDDE	TDDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.7.4 Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable Enables DMA receive data. 0b - Disable 1b - Enable
0 TDDE	Transmit Data DMA Enable Enables DMA transmit data. 0b - Disable 1b - Enable

40.7.1.8 Controller Configuration 0 (MCFGR0)

40.7.1.8.1 Offset

Register	Offset
MCFGR0	20h

40.7.1.8.2 Function

Contains host settings and other receive and transfer settings.

40.7.1.8.3 Diagram

Bits	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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						RDMO	CIRFIFO	0				0	HRSEL	HRPOL	HREN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.8.4 Fields

Field	Function
31-18 —	Reserved
17-16 —	Reserved
15-10 —	Reserved
9 RDMO	Receive Data Match Only Determines whether all received data that does not set MSR[DMF] is discarded. After MSR[DMF] is set, the RDMO configuration is ignored. When disabling RDMO, write 0 to this field before writing 0 to MSR[DMF] to ensure that no receive data is lost. 0b - Received data is stored in the receive FIFO 1b - Received data is discarded unless MSR[DMF] is set
8 CIRFIFO	Circular FIFO Enable Enables the transmit FIFO read pointer to be saved to a temporary register. The transmit FIFO empties as normal. After the LPI2C controller is idle and the transmit FIFO is empty, the read pointer value is restored from the temporary register. This setting causes the contents of the transmit FIFO to be cycled

Table continues on the next page...

Field	Function
	through repeatedly. If MCFGR1[AUTOSTOP] is 1, then a Stop condition is sent whenever the transmit FIFO is empty and the read pointer is restored. 0b - Disable 1b - Enable
7-4 —	Reserved
3 —	Reserved
2 HRSEL	Host Request Select Selects the source of the host request input. When host request is enabled, this field must not change. 0b - Reserved 1b - Host request input is input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request input. When host request is enabled, this field must not change. HRPOL sets the polarity for both the HREQ pin and the input trigger. • When HRPOL=0, the polarity is configured for active low, so host request is asserted if the HREQ pin or input trigger are logic 0. • When HRPOL=1, the polarity is configured for active high, so host request is asserted if the HREQ pin or input trigger are logic 1. 0b - Active low 1b - Active high
0 HREN	Host Request Enable Enables host request. When enabled, the LPI2C controller only initiates a Start condition if the host request input is asserted and the bus is idle. A repeated Start condition is not affected by the host request. 0b - Disable 1b - Enable

40.7.1.9 Controller Configuration 1 (MCFGR1)

40.7.1.9.1 Offset

Register	Offset
MCFGR1	24h

40.7.1.9.2 Function

Contains controls for pin configuration, clock prescaler, and various other control settings.

Write to this register only when the I2C controller is disabled.

40.7.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0	PINCFG			0					MATCFG		
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			0		TIMECFG	IGNACK	AUTOSTOP	0					PRESCALE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.9.4 Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26-24 PINCFG	Pin Configuration Configures the pin mode for LPI2C. 000b - Two-pin open drain mode. SCL/SDA pins: Bidirectional open drain for controller and target. SCLS/SDAS pins: Not used. 001b - Two-pin output only mode (Ultra-Fast mode). SCL/SDA pins: Output-only (Ultra-Fast mode) open drain for controller and target. SCLS/SDAS pins: Not used. 010b - Two-pin push-pull mode. SCL/SDA pins: Bidirectional push-pull for controller and target. SCLS/SDAS pins: Not used. 011b - Four-pin push-pull mode. SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Output-only push-pull for controller and target. 100b - Two-pin open-drain mode with separate LPI2C target. SCL/SDA pins: Bidirectional open drain for controller. SCLS/SDAS pins: Bidirectional open drain for target. 101b - Two-pin output only mode (Ultra-Fast mode) with separate LPI2C target. SCL/SDA pins: Output-only (Ultra-Fast mode) open drain for controller. SCLS/SDAS pins: Output-only open drain for target. 110b - Two-pin push-pull mode with separate LPI2C target. SCL/SDA pins: Bidirectional push-pull for controller. SCLS/SDAS pins: Bidirectional push-pull for target. 111b - Four-pin push-pull mode (inverted outputs). SCL/SDA pins: Input only for controller and target. SCLS/SDAS pins: Inverted output-only push-pull for controller and target.
23-19 —	Reserved
18-16	Match Configuration

Table continues on the next page...

Field	Function
MATCFG	Configures the condition that sets MSR[DMF]. See Controller Data Match (MDMR). 000b - Match is disabled 001b - Reserved 010b - Match is enabled: first data word equals MDMR[MATCH0] OR MDMR[MATCH1] 011b - Match is enabled: any data word equals MDMR[MATCH0] OR MDMR[MATCH1] 100b - Match is enabled: (first data word equals MDMR[MATCH0]) AND (second data word equals MDMR[MATCH1) 101b - Match is enabled: (any data word equals MDMR[MATCH0]) AND (next data word equals MDMR[MATCH1) 110b - Match is enabled: (first data word AND MDMR[MATCH1]) equals (MDMR[MATCH0] AND MDMR[MATCH1]) 111b - Match is enabled: (any data word AND MDMR[MATCH1]) equals (MDMR[MATCH0] AND MDMR[MATCH1])
15-13 —	Reserved
12-11 —	Reserved
10 TIMECFG	Timeout Configuration Configures which signals must be low for longer than the configured timeout to set MSR[PLTF]. When this field is 0, MSR[PLTF] is set when SCL is low for longer than the configured timeout. 0b - SCL 1b - SCL or SDA
9 IGNACK	Ignore NACK Determines whether the LPI2C controller ignores a received NACK and treats it as an ACK. This field must be 1 in Ultra-Fast mode. 0b - No effect 1b - Treat a received NACK as an ACK
8 AUTOSTOP	Automatic Stop Generation Determines whether a Stop condition is generated when the LPI2C controller is busy and the transmit FIFO is empty. A Stop condition can also be generated using a transmit FIFO command. When this field is 1, a Stop condition is automatically generated when the transmit FIFO is empty and the LPI2C controller is busy. 0b - No effect 1b - Stop automatically generated
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C controller logic except the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

40.7.1.10 Controller Configuration 2 (MCFGR2)

40.7.1.10.1 Offset

Register	Offset
MCFGR2	28h

40.7.1.10.2 Function

Contains the configuration for the bus idle timeout and glitch filters for SDA and SCL.

Write to this register only when the I2C controller is disabled.

40.7.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
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				BUSIDLE											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.10.4 Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C controller digital glitch filters for the SDA input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.

Table continues on the next page...

Field	Function
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C controller digital glitch filters for SCL input. The latency through the glitch filter is equal to the number of cycles defined by this field. The value of this field must be less than the minimum SCL low or high period. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. These cycles are based on the functional clock. Writing 0 to this field disables the glitch filter. MCFGR1[PRESCALE] does not affect the glitch filter cycle count. It is automatically bypassed in HS mode.
15-12 —	Reserved
11-0 BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period, in clock cycles. If both SCL and SDA are high for longer than the number of cycles defined by this field, the I2C bus is assumed to be idle and the controller can generate a Start condition. Writing 0 to this field disables the bus idle timeout.

40.7.1.11 Controller Configuration 3 (MCFGR3)

40.7.1.11.1 Offset

Register	Offset
MCFGR3	2Ch

40.7.1.11.2 Function

Configures the threshold value for the pin low timeout flag.

Write to this register only when the I2C controller is disabled.

40.7.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												PINLOW			
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	PINLOW								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.11.4 Fields

Field	Function
31-20 —	Reserved
19-8 PINLOW	Pin Low Timeout Configures the threshold value, in clock cycles, that sets MSR[PLTF]. If SCL or SDA (selected by MCFGR1[TIMECFG]) is low for longer than (PINLOW × 256) cycles, MSR[PLTF] is set. When this field is 0, the pin low timeout feature is disabled.
7-0 —	Reserved

40.7.1.12 Controller Data Match (MDMR)

40.7.1.12.1 Offset

Register	Offset
MDMR	40h

40.7.1.12.2 Function

Contains data match values.

Write to this register only when the I2C controller is disabled or idle.

40.7.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								MATCH1							
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								MATCH0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.12.4 Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Specifies match 1 value that is compared to the received data when receive data match is enabled.
15-8 —	Reserved
7-0 MATCH0	Match 0 Value Specifies match 0 value that is compared to the received data when receive data match is enabled.

40.7.1.13 Controller Clock Configuration 0 (MCCR0)

40.7.1.13.1 Offset

Register	Offset
MCCR0	48h

40.7.1.13.2 Function

Configures various clock controls.

You cannot make changes to this register when the I2C controller is enabled and is used for standard, fast, fast-mode plus, and ultra-fast transfers.

40.7.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DATAVD						0		SETHOLD					
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		CLKHI						0		CLKLO					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.13.4 Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) used by the controller for these conditions: • Hold time for a Start • Setup and hold time for a repeated Start • Setup time for a Stop 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 time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is

Field	Function
	extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.

40.7.1.14 Controller Clock Configuration 1 (MCCR1)

40.7.1.14.1 Offset

Register	Offset
MCCR1	50h

40.7.1.14.2 Function

Configures various clock controls.

You cannot makes changes to this register when the I2C controller is enabled and is used for HS mode transfers. The separate clock configuration for HS mode allows arbitration to take place in Fast mode (with timing configured by Controller Clock Configuration 0 (MCCR0)), before switching to HS mode (with timing configured by MCCR1).

40.7.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DATAVD						0		SETHOLD					
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		CLKHI						0		CLKLO					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.14.4 Fields

Field	Function
31-30 —	Reserved

Table continues on the next page...

Field	Function
29-24 DATAVD	Data Valid Delay Specifies the minimum number of cycles (minus one) used as the data hold time for SDA. This value must be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Specifies the minimum number of cycles (minus one) used by the controller for these conditions: • Hold time for a Start condition • Setup and hold time for a repeated Start condition • Setup time for a Stop condition The setup time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock high. The SCL high time is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Specifies the minimum number of cycles (minus one) that the controller drives the SCL clock low. This value is also used for the minimum bus free time between a Stop and a Start condition. This period is extended by the time needed to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) ÷ 2^PRESCALE cycles.

40.7.1.15 Controller FIFO Control (MFCR)

40.7.1.15.1 Offset

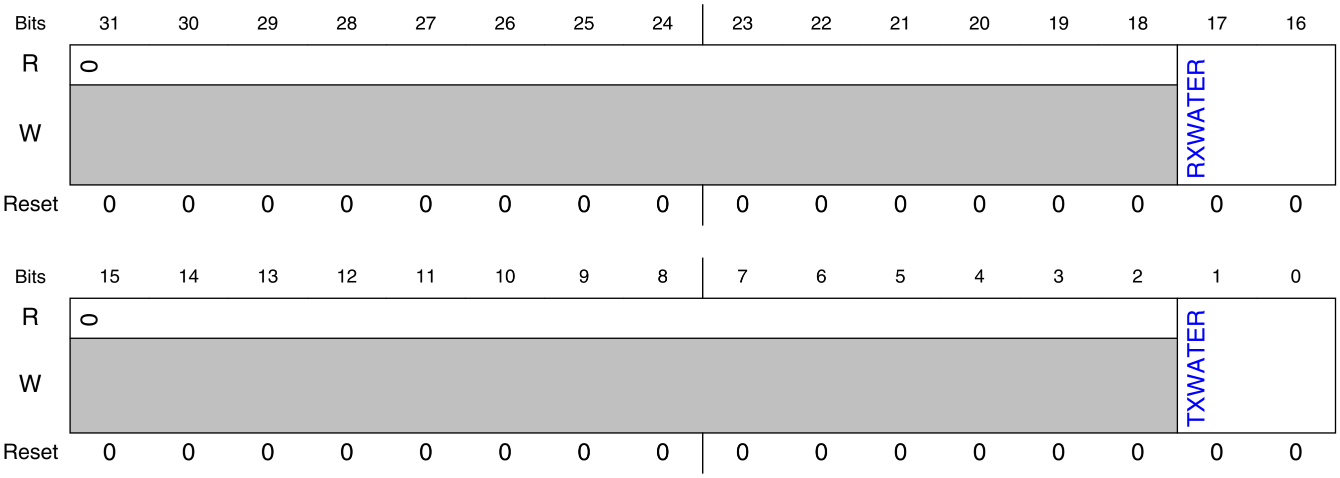
Register	Offset
MFCR	58h

40.7.1.15.2 Function

Controls the receive and transmit FIFO watermark values.

This register is used only in Stop mode, when this register is static (not changing).

40.7.1.15.3 Diagram



Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														RXWATER	
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														TXWATER	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.15.4 Fields

Field	Function
31-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark Determines the watermark for setting SSR[RDF]. That flag is set when the number of words in the receive FIFO is greater than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.
15-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark Determines the watermark for setting SSR[TDF]. That flag is set when the number of words in the transmit FIFO is equal or less than the value of this field. Writing a value equal to or greater than the FIFO size truncates the value.

40.7.1.16 Controller FIFO Status (MFSR)

40.7.1.16.1 Offset

Register	Offset
MFSR	5Ch

40.7.1.16.2 Function

Specifies the number of words in the transmit and receive FIFOs.

40.7.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													RXCOUNT		
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													TXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.16.4 Fields

Field	Function
31-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Specifies the number of words in the receive FIFO.
15-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Specifies the number of words in the transmit FIFO.

40.7.1.17 Controller Transmit Data (MTDR)

40.7.1.17.1 Offset

Register	Offset
MTDR	60h

40.7.1.17.2 Function

Configures transmit data:

- An 8-bit write to MTDR[CMD] is ignored and does not increment the FIFO write pointer.
- An 8-bit write to MTDR[DATA] zero-extends the value of MTDR[CMD] and increments the FIFO write pointer.
- A 16-bit or 32-bit write operation writes to both MTDR[CMD] and MTDR[DATA] and increments the FIFO write pointer.

40.7.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	0															
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	0					CMD			DATA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.17.4 Fields

Field	Function
31-11 —	Reserved
10-8 CMD	Command Data Selects command transmitted by controller. 000b - Transmit the value in DATA[7:0] 001b - Receive (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes and check each for a data match (if configured) before storing the received data in the receive FIFO. 010b - Generate Stop condition on I2C bus 011b - Receive and discard (DATA[7:0] + 1) bytes. DATA[7:0] is used as a byte counter. Receive that many bytes but do not check for a data match or store those bytes in the receive FIFO. 100b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] 101b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] (this transfer expects a NACK to be returned) 110b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode 111b - Generate (repeated) Start on the I2C bus and transmit the address in DATA[7:0] using HS mode (this transfer expects a NACK to be returned)
7-0 DATA	Transmit Data Contains data used by the commands listed in MTDR[CMD]. Performing an 8-bit write to this field zero-extends the value of MTDR[CMD].

40.7.1.18 Controller Receive Data (MRDR)

40.7.1.18.1 Offset

Register	Offset
MRDR	70h

40.7.1.18.2 Function

Contains the status of the receive FIFO and the data received by the I2C controller that has not been discarded.

40.7.1.18.3 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	RXEMPTY	0						DATA							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.18.4 Fields

Field	Function
31-15 —	Reserved
14 RXEMPTY	Receive Empty Indicates whether the controller receive data FIFO is empty. 0b - Not empty 1b - Empty
13-8 —	Reserved

Table continues on the next page...

Field	Function
7-0 DATA	Receive Data Contains data received by the I2C controller that has not been discarded. Received data can be discarded due to the command in MTDR[CMD], or the controller can be configured to discard nonmatching data.

40.7.1.19 Target Control (SCR)

40.7.1.19.1 Offset

Register	Offset
SCR	110h

40.7.1.19.2 Function

Contains resets and other target control settings.

40.7.1.19.3 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						0	0	0		FILTDZ	FILTEN	0		RST	SEN
W							RRF	RTF								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.19.4 Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO Empties the receive FIFO in Target Receive Data (SRDR). 0b - No effect

Table continues on the next page...

Field	Function
	1b - SRDR is now empty
8 RTF	Reset Transmit FIFO Empties the transmit FIFO in Target Transmit Data (STDR). 0b - No effect 1b - STDR is now empty
7-6 —	Reserved
5 FILTDZ	Filter Doze Enable Enables filter in Doze mode. Update this field only when the I2C target is disabled. 0b - Enable 1b - Disable
4 FILTEN	Filter Enable Enables digital filter and output delay counter for target mode. Update this field only when the I2C target is disabled. 0b - Disable 1b - Enable
3-2 —	Reserved
1 RST	Software Reset Resets target mode logic. The reset takes effect immediately. The value of this field remains 1 until you write 0 to it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 SEN	Target Enable Enables I2C Target mode. 0b - Disable 1b - Enable

40.7.1.20 Target Status (SSR)

40.7.1.20.1 Offset

Register	Offset
SSR	114h

40.7.1.20.2 Function

Contains status flags for transmit and receive data, for error conditions, and for bus and target busy or idle status.

40.7.1.20.3 Diagram

Bits	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

Bits	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

40.7.1.20.4 Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Indicates whether an I2C bus is idle or busy. 0b - Idle 1b - Busy
24 SBF	Target Busy Flag Indicates whether an I2C target is idle or busy. 0b - Idle 1b - Busy
23-16 —	Reserved
15 SARF	SMBus Alert Response Flag Indicates whether an SMBus alert response has been detected. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Disabled or not detected 1b - Enabled and detected
14 GCF	General Call Flag Indicates whether a target has detected the general call address. You can clear this flag by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - General call address disabled or not detected 1b - General call address detected
13 AM1F	Address Match 1 Flag

Table continues on the next page...

Field	Function
	Indicates whether the received address matches the value in ADDR1, or it falls within the ADDR0 to ADDR1 range as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - Matching address not received 1b - Matching address received
12 AM0F	Address Match 0 Flag Indicates whether the received address matches the ADDR0 field, as configured by SCFGR1[ADDRCFG]. This flag is cleared by reading Target Address Status (SASR). This flag cannot generate an asynchronous wakeup. 0b - ADDR0 matching address not received 1b - ADDR0 matching address received
11 FEF	FIFO Error Flag Indicates whether there is a FIFO error. This flag can only be set when clock stretching is disabled. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No FIFO error 1b - FIFO error When writing 0b - No effect 1b - Clear the flag
10 BEF	Bit Error Flag Indicates whether the LPI2C target has transmitted a logic 1 and detects a logic 0 on the I2C bus. The target ignores the rest of the transfer until the next (repeated) Start condition. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No bit error occurred 1b - Bit error occurred When writing 0b - No effect 1b - Clear the flag
9 SDF	Stop Detect Flag Indicates whether the LPI2C target detects a Stop condition, and if the LPI2C target matched the last address byte. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No Stop detected 1b - Stop detected When writing 0b - No effect 1b - Clear the flag

Table continues on the next page...

Field	Function
8 RSF	Repeated Start Flag Indicates whether the LPI2C target detects a repeated Start condition and if the LPI2C target matched the last address byte. This flag is not set when the target first detects a Start condition. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No repeated Start detected 1b - Repeated Start detected When writing 0b - No effect 1b - Clear the flag
7-4 —	Reserved
3 TAF	Transmit ACK Flag Indicates whether a transmit ACK or NACK is required. You can clear this flag by writing to Target Transmit ACK (STAR). 0b - Not required 1b - Required
2 AVF	Address Valid Flag Indicates whether the contents of Target Address Status (SASR) are valid. You can clear this flag by reading SASR. When SCFGR1[RXCFG] = 1, this flag is also cleared by reading Target Receive Data (SRDR). 0b - Not valid 1b - Valid
1 RDF	Receive Data Flag Indicates whether receive data is ready. You can clear this flag by reading Target Receive Data (SRDR). When SCFGR1[RXCFG] = 1, this flag is not cleared when reading Target Receive Data (SRDR) if SSR[AVF] = 1. 0b - Not ready 1b - Ready
0 TDF	Transmit Data Flag Indicates whether transmit data has been requested. This flag is cleared by writing to Target Transmit Data (STDR). When SCFGR1[TXCFG] = 0, if a NACK, repeated Start, or Stop condition is detected, this flag is also cleared. 0b - Transmit data not requested 1b - Transmit data is requested

40.7.1.21 Target Interrupt Enable (SIER)

40.7.1.21.1 Offset

Register	Offset
SIER	118h

40.7.1.21.2 Function

Contains transmit and receive data interrupt enables, start and stop detect interrupt enables, and other target interrupt enables.

40.7.1.21.3 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	SARIE	GCIE	AM1IE	AM0IE	FEIE	BEIE	SDIE	RSIE	0				TAIE	AVIE	RDIE	TDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.21.4 Fields

Field	Function
31-16 —	Reserved
15 SARIE	SMBus Alert Response Interrupt Enable Enables interrupt for SMBus alert response. 0b - Disable 1b - Enable
14 GCIE	General Call Interrupt Enable Enables interrupt for general call. 0b - Disabled 1b - Enabled
13 AM1IE	Address Match 1 Interrupt Enable Enables interrupt for address match 1. 0b - Disable 1b - Enable
12 AM0IE	Address Match 0 Interrupt Enable Enables interrupt for address match 0. 0b - Disable 1b - Enable
11 FEIE	FIFO Error Interrupt Enable Enables interrupt for FIFO error. 0b - Disable 1b - Enable
10	Bit Error Interrupt Enable

Table continues on the next page...

Field	Function
BEIE	Enables interrupt for bit error. 0b - Disable 1b - Enable
9 SDIE	Stop Detect Interrupt Enable Enables interrupt for Stop detection. 0b - Disable 1b - Enable
8 RSIE	Repeated Start Interrupt Enable Enables interrupt for repeated start. 0b - Disable 1b - Enable
7-4 —	Reserved
3 TAIE	Transmit ACK Interrupt Enable Enables interrupt for transmit ACK. 0b - Disable 1b - Enable
2 AVIE	Address Valid Interrupt Enable Enables interrupt for valid address. 0b - Disable 1b - Enable
1 RDIE	Receive Data Interrupt Enable Enables interrupt for receive data. 0b - Disable 1b - Enable
0 TDIE	Transmit Data Interrupt Enable Enables interrupt for transmit data. 0b - Disable 1b - Enable

40.7.1.22 Target DMA Enable (SDER)

40.7.1.22.1 Offset

Register	Offset
SDER	11Ch

40.7.1.22.2 Function

Contains the transmit, request, and receive enables for DMA.

40.7.1.22.3 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													AVDE	RDDE	TDDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.22.4 Fields

Field	Function
31-3 —	Reserved
2 AVDE	Address Valid DMA Enable Enables address valid DMA request. The address valid DMA request is shared with the receive data DMA request. If both are enabled, write 1 to SCFGR1[RXCFG] to allow the DMA to read the address from Target Receive Data (SRDR). 0b - Disable 1b - Enable
1 RDDE	Receive Data DMA Enable Enables receive data for DMA. 0b - Disable DMA request 1b - Enable DMA request
0 TDDE	Transmit Data DMA Enable Enables transmit data for DMA. 0b - Disable 1b - Enable

40.7.1.23 Target Configuration 1 (SCFGR1)

40.7.1.23.1 Offset

Register	Offset
SCFGR1	124h

40.7.1.23.2 Function

Configures various aspects of the target.

Write to this register only when the I2C target is disabled.

40.7.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					0			0					ADDRCFG		
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		HSMEN	IGNACK	RXCFG	TXCFG	SAEN	GCEN	0			0	ACKSTALL	TXDSTALL	RXSTALL	ADRSTALL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.23.4 Fields

Field	Function
31-27 —	Reserved
26-24 —	Reserved
23-19 —	Reserved
18-16 ADDRCFG	Address Configuration Configures the condition that causes an address to match. 000b - Address match 0 (7-bit) 001b - Address match 0 (10-bit) 010b - Address match 0 (7-bit) or address match 1 (7-bit) 011b - Address match 0 (10-bit) or address match 1 (10-bit) 100b - Address match 0 (7-bit) or address match 1 (10-bit) 101b - Address match 0 (10-bit) or address match 1 (7-bit) 110b - From address match 0 (7-bit) to address match 1 (7-bit) 111b - From address match 0 (10-bit) to address match 1 (10-bit)
15-14 —	Reserved
13 HSMEN	HS Mode Enable

Table continues on the next page...

Field	Function
	Enables detection of the HS mode controller code of target address 0000_1XX, but does not cause an address match on this code. When this field is 1 and any HS mode controller code is detected, SCR[FILTEN] and SCFGR1[ACKSTALL] are ignored until the next Stop condition is detected. 0b - Disable 1b - Enable
12 IGNACK	Ignore NACK Determines whether the target ends transfer when a NACK condition is detected. When this field is 1, the LPI2C target continues transfers after a NACK is detected. This field is required to be 1 in Ultra-Fast mode. 0b - End transfer on NACK 1b - Do not end transfer on NACK
11 RXCFG	Receive Data Configuration Configures which data is returned and which flags are cleared when reading Target Receive Data (SRDR). When this field is 0, reading SRDR returns received data and clears MSR[RDF]. When this field is 1, reading SRDR: • Returns the value of Target Address Status (SASR) and clears SSR[AVF] when SSR[AVF] is set. • Returns received data and clears MSR[RDF] when SSR[AVF] is not set. 0b - Return received data, clear MSR[RDF] 1b - Return SASR and clear SSR[AVF] when SSR[AVF] is set, return received data and clear MSR[RDF] when SSR[AFV] is not set
10 TXCFG	Transmit Flag Configuration Determines which conditions set MSR[TDF]. This field always becomes 1 before a NACK is detected at the end of a target-transmit transfer. This change can cause an extra word to be written to the transmit data FIFO. When this field is 0, Target Transmit Data (STDR) is automatically emptied when a target-transmit transfer is detected. MSR[TDF] is set when a target-transmit transfer is detected, and MSR[TDF] is cleared at the end of the target-transmit transfer. When this field is 1, MSR[TDF] is set when STDR is empty, and MSR[TDF] is cleared when STDR is full. This setting allows STDR to be filled before a target-transmit transfer is detected. However, it can cause STDR to be written before a NACK is detected on the last byte of a target-transmit transfer. 0b - MSR[TDF] is set only during a target-transmit transfer when STDR is empty 1b - MSR[TDF] is set whenever STDR is empty
9 SAEN	SMBus Alert Enable Enables a match on an SMBus alert. 0b - Disable 1b - Enable
8 GCEN	General Call Enable Enables a general call address. 0b - Disable 1b - Enable
7-5 —	Reserved
4 —	Reserved
3	ACK SCL Stall

Table continues on the next page...

Field	Function
ACKSTALL	Enables SCL clock stretching during target-transmit address bytes and target-receiver address and data bytes, so you can write to Target Transmit ACK (STAR) before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the ninth bit, and is therefore not compatible with HS mode. If this field is 1: • You do not need to write 1 to SCFGR1[RXSTALL] or SCFGR1[ADRSTALL]. 0b - Disable 1b - Enable
2 TXDSTALL	Transmit Data SCL Stall Enables SCL clock stretching when SSR[TDF] = 1 during a target-transmit transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when SSR[RDF] = 1 during a target-receive transfer. Clock stretching occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when SSR[AVF] = 1. Clock stretching only occurs following the ninth bit, and is therefore compatible with HS mode. 0b - Disable 1b - Enable

40.7.1.24 Target Configuration 2 (SCFGR2)

40.7.1.24.1 Offset

Register	Offset
SCFGR2	128h

40.7.1.24.2 Function

Configures data valid delay, clock hold time, and glitch filters for SDA and SCL.

Write to this register only when the I2C target is disabled.

40.7.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
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		DATAVD						0				CLKHOLD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.24.4 Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C target digital glitch filters for SDA input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C target digital glitch filters for SCL input. Writing 0 to this field disables the glitch filter. Glitches equal to or less than the number of cycles defined by this field are filtered out and ignored. The latency through the glitch filter is equal to the number of cycles defined by this field + 3. The latency must be configured to be less than the minimum SCL low or high period. MCFGR1[PRESCALE] does not affect the glitch filter cycle count, and the glitch filter cycle count is disabled in HS mode.
15-14 —	Reserved
13-8 DATAVD	Data Valid Delay Configures the SDA data valid delay time for the I2C target, which is equal to FILTSCL + DATAVD + 3 cycles. The data valid delay must be configured to be less than the minimum SCL low period.

Table continues on the next page...

Field	Function
	MCFGR1[PRESCALE] does not affect the I2C target data valid delay time, and the I2C target data valid delay time is disabled in HS mode.
7-4 —	Reserved
3-0 CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C target, when clock stretching is enabled. The minimum hold time is equal to the number of cycles defined by this field + 3. MCFGR1[PRESCALE] does not affect the I2C target clock hold time, and the I2C target clock hold time is disabled in HS mode.

40.7.1.25 Target Address Match (SAMR)

40.7.1.25.1 Offset

Register	Offset
SAMR	140h

40.7.1.25.2 Function

Contains address values for received target match comparison.

Write to this register only when the I2C target is disabled.

40.7.1.25.3 Diagram

Bits	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

Bits	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

40.7.1.25.4 Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 Value Contains the value of address 1, which is compared to the received address to detect the target address. In 10-bit mode, the first address byte is compared to {11110, ADDR1[26:25]} and the second address byte is compared to ADDR1[24:17]. In 7-bit mode, the address is compared to ADDR1[23:17].
16-11 —	Reserved
10-1 ADDR0	Address 0 Value Contains the value of address 0, which is compared to the received address to detect the target 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

40.7.1.26 Target Address Status (SASR)

40.7.1.26.1 Offset

Register	Offset
SASR	150h

40.7.1.26.2 Function

Contains the received address and its validity.

40.7.1.26.3 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	ANV	0			RADDR										
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.26.4 Fields

Field	Function
31-15 —	Reserved
14 ANV	Address Not Valid Indicates whether SASR[RADDR] is valid. 0b - Valid 1b - Not valid
13-11 —	Reserved
10-0 RADDR	Received Address Contains the received address. Updates whenever SSR[AM0F] or SSR[AM1F] is set. Reading Target Address Status (SASR) clears SSR[AM0F] and SSR[AM1F]. In 7-bit mode, the address byte is stored in RADDR[7:0]. In 10-bit mode, the first address byte is {11110, RADDR[10:9], RADDR[0]} and the second address byte is RADDR[8:1]. The Read-or-Write bit is therefore always stored in RADDR[0].

40.7.1.27 Target Transmit ACK (STAR)

40.7.1.27.1 Offset

Register	Offset
STAR	154h

40.7.1.27.2 Function

Configures choice of ACK or NACK on each received word.

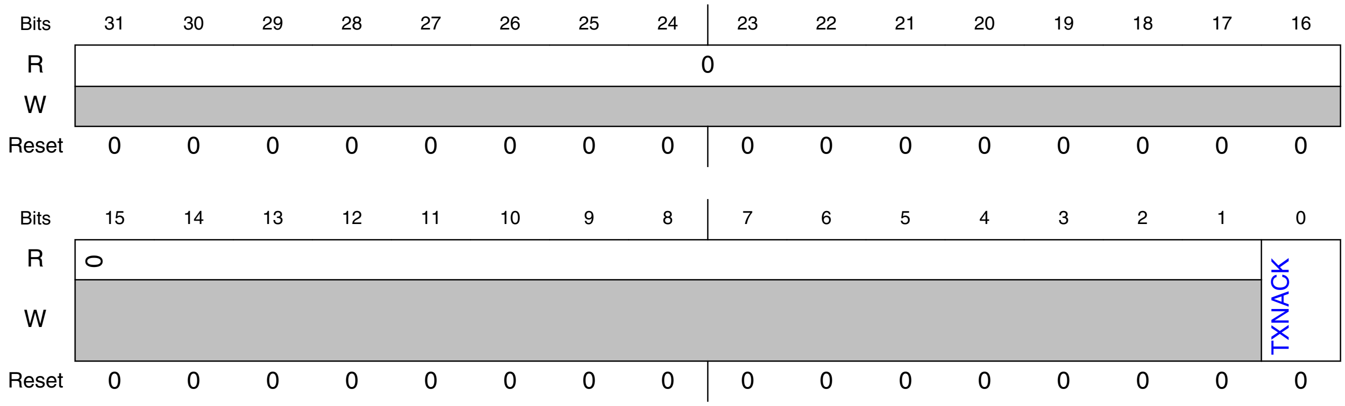
You can write to this register only when SCFGR1[ACKSTALL] = 1.

SCFGR1[ACKSTALL] enables clock stretching during the ACK-or-NACK bit slot. During this time, you can write to this register.

The logic ensures that the clock stretching continues for at least one bus clock cycle after this register is updated.

This clock stretching time can be extended via SCFGR2[CLKHOLD].

40.7.1.27.3 Diagram



40.7.1.27.4 Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK Selects whether transmit ACK (logic 0) or NACK (logic 1) is returned on the bus by the I2C target after receiving each word. • When SCFGR1[ACKSTALL] = 1, a transmit NACK signal must be written once for each matching address byte and each received word. SCFGR1[ACKSTALL] must be 1, because that setting stalls the data transfer until software reads the received word (and determines whether to respond with an ACK or NACK). • To configure the default (ACK or NACK), you can write to this field when LPI2C target is disabled or idle. 0b - Transmit ACK 1b - Transmit NACK

40.7.1.28 Target Transmit Data (STDR)

40.7.1.28.1 Offset

Register	Offset
STDR	160h

40.7.1.28.2 Function

Contains the I2C target data to transmit.

Clock stretching (enabled or disabled) affects when the transmit data is transferred. SCFGR1[TXDSTALL] enables clock stretching during the first data bit of a target-transmit transfer.

If clock stretching is enabled (SCFGR1[TXDSTALL] = 1), the transmit data transfer is stalled until this register is updated. Clock stretching is extended by at least 1 bus clock cycle after this register is updated. Clock stretching can be delayed further by using SCFGR2[CLKHOLD].

If clock stretching is disabled (SCFGR1[TXDSTALL] = 0), the transmit data must be written before the start of the target-transmit transfer, otherwise SSR[FEF] is set.

40.7.1.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	0															
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	0								DATA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.28.4 Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Field	Function
—	
7-0 DATA	Transmit Data Contains the I2C target data to transmit. Writing data to this register stores I2C target transmit data in this register.

40.7.1.29 Target Receive Data (SRDR)

40.7.1.29.1 Offset

Register	Offset
SRDR	170h

40.7.1.29.2 Function

Contains status of target receive data transfer.

40.7.1.29.3 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	SOF	RXEMPTY	0			RADDR			DATA							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.7.1.29.4 Fields

Field	Function
31-16 —	Reserved
15	Start of Frame

Table continues on the next page...

Field	Function
SOF	Indicates whether this data word is the first data word since a (repeated) Start or Stop condition. 0b - Not first 1b - First
14 RXEMPTY	Receive Empty Indicates whether this register is empty. 0b - Not empty 1b - Empty
13-11 —	Reserved
10-8 RADDR	Received Address Contains the address received by the IC2 target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [10:8] of SASR[RADDR] are returned. Otherwise, this field returns zero.
7-0 DATA	Received Data Contains the data received by the I2C target. When both SCFGR1[RXCFG] and SSR[AVF] are 1, bits [7:0] of SASR[RADDR] are returned.

40.8 Usage Guide

For master (controller mode):

- Configure functional clock source
- Reset LPI2C module by LPI2C0_MCR[RST]
- Configure baudrate
- Set Tx/Rx FIFO watermark by LPI2C0_MFCR
- Enable Master mode by set LPI2C0_MCR[MEN]

For slave (target mode):

- Configure functional clock source
- Set the slave address into LPI2C0_SAMR
- Configure the TDF only be set in the Slave-Transmit condition by LPI2C0_SCFGR1[TXCFG]
- Enable the TX Data SCL Stall and RX SCL Stall for clock stretching on SCL
- Enable Slave mode by set LPI2C0_SCR[SEN]

Chapter 41
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

41.1 Chip-specific information for this module

41.1.1 Instantiation Information

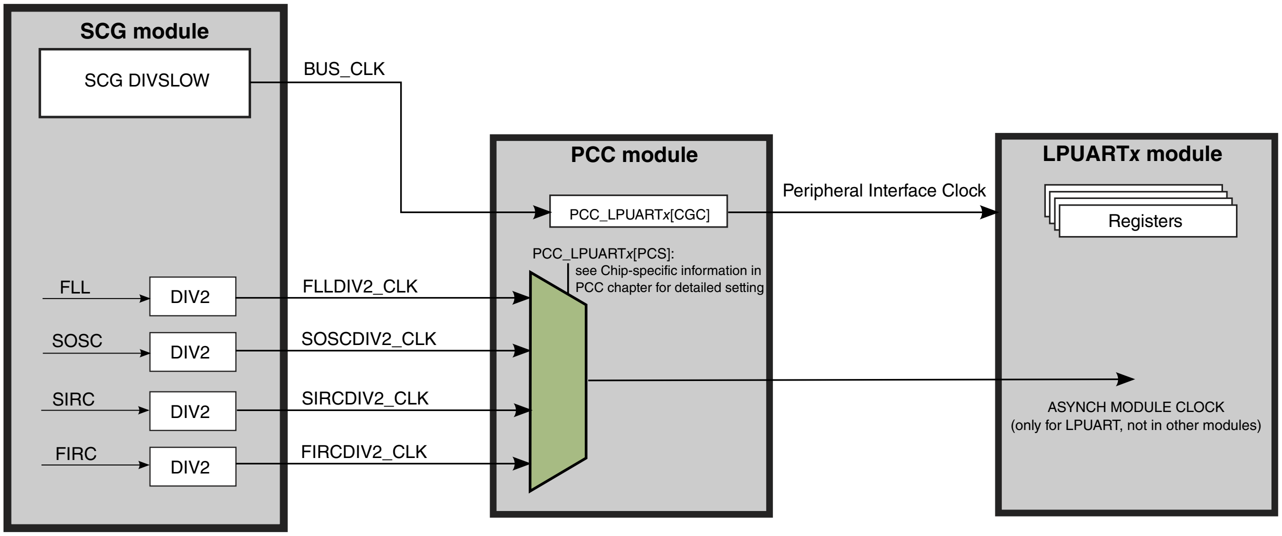
This device has three LPUART modules. The LPUART can remain functional in Stop and VLPS mode provided the clock it is using remains enabled.

Table 41-1. LPUART Configuration

	TX FIFO (word/10bit)	RX FIFO (word/10bit)	Single-wire mode
LPUART0	4	4	Yes
LPUART1	4	4	Yes
LPUART2	4	4	Yes

41.1.2 Module Clocking Information for LPUART, LPSPI, LPI2C, FlexIO and LPIT

The following figure shows the input clock sources available for this module.

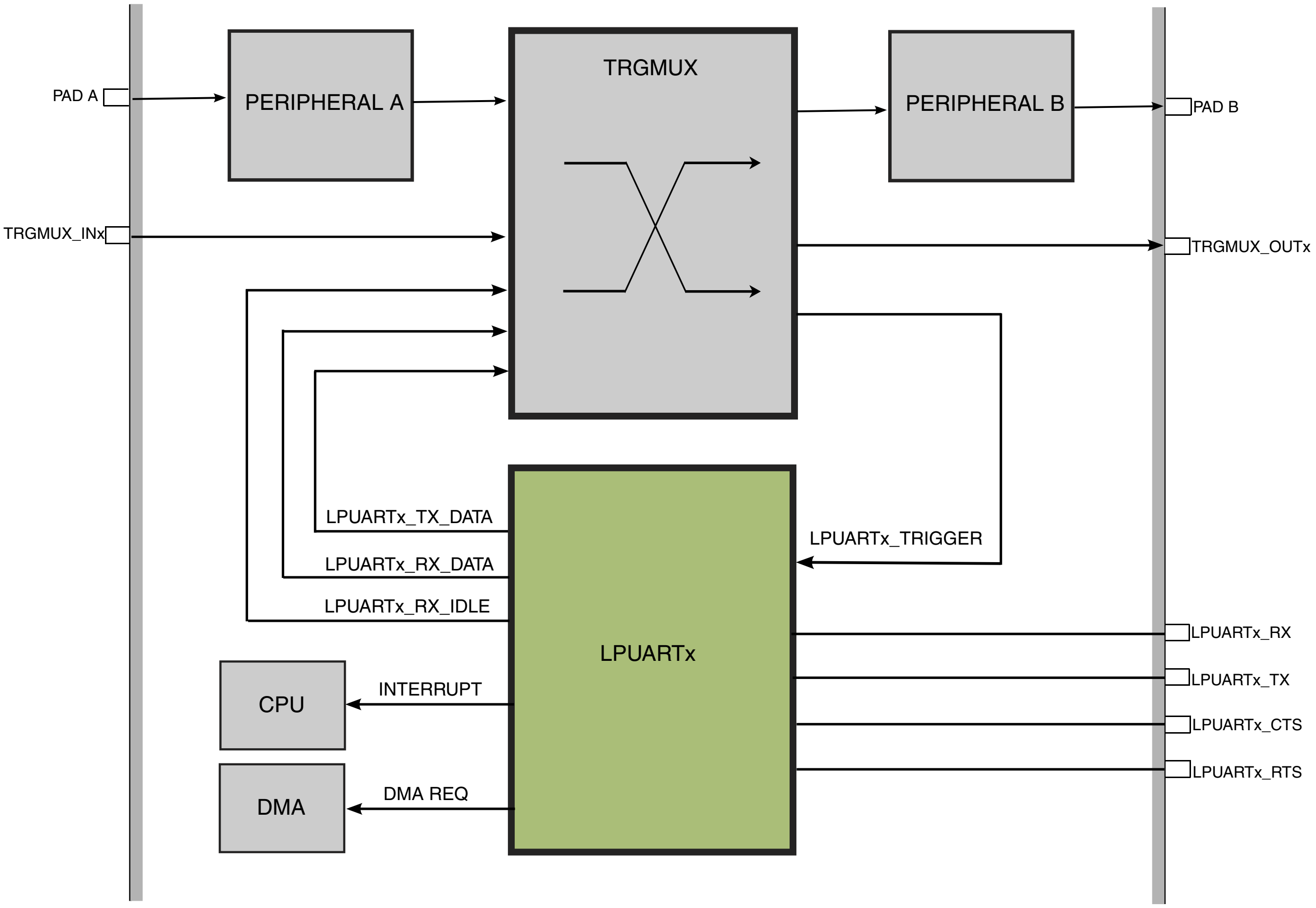


Peripheral Clocking - LPUART, etc.

Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, LPIT, FlexIO, etc.

41.1.3 Inter-connectivity Information

The LPUART inter-connectivity is shown in following diagram.



41.2 Overview

LPUART provides asynchronous, serial communication capabilities with external devices. It supports the non-return-to-zero (NRZ) encoding format and infrared data association (IrDA)-compatible, low-speed serial infrared (SIR) protocol. LPUART can continue operating when the processor is in Low-Power mode, if an appropriate peripheral clock is available.

41.2.1 Block diagram

Figure 41-1 shows the transmitter portion of LPUART.

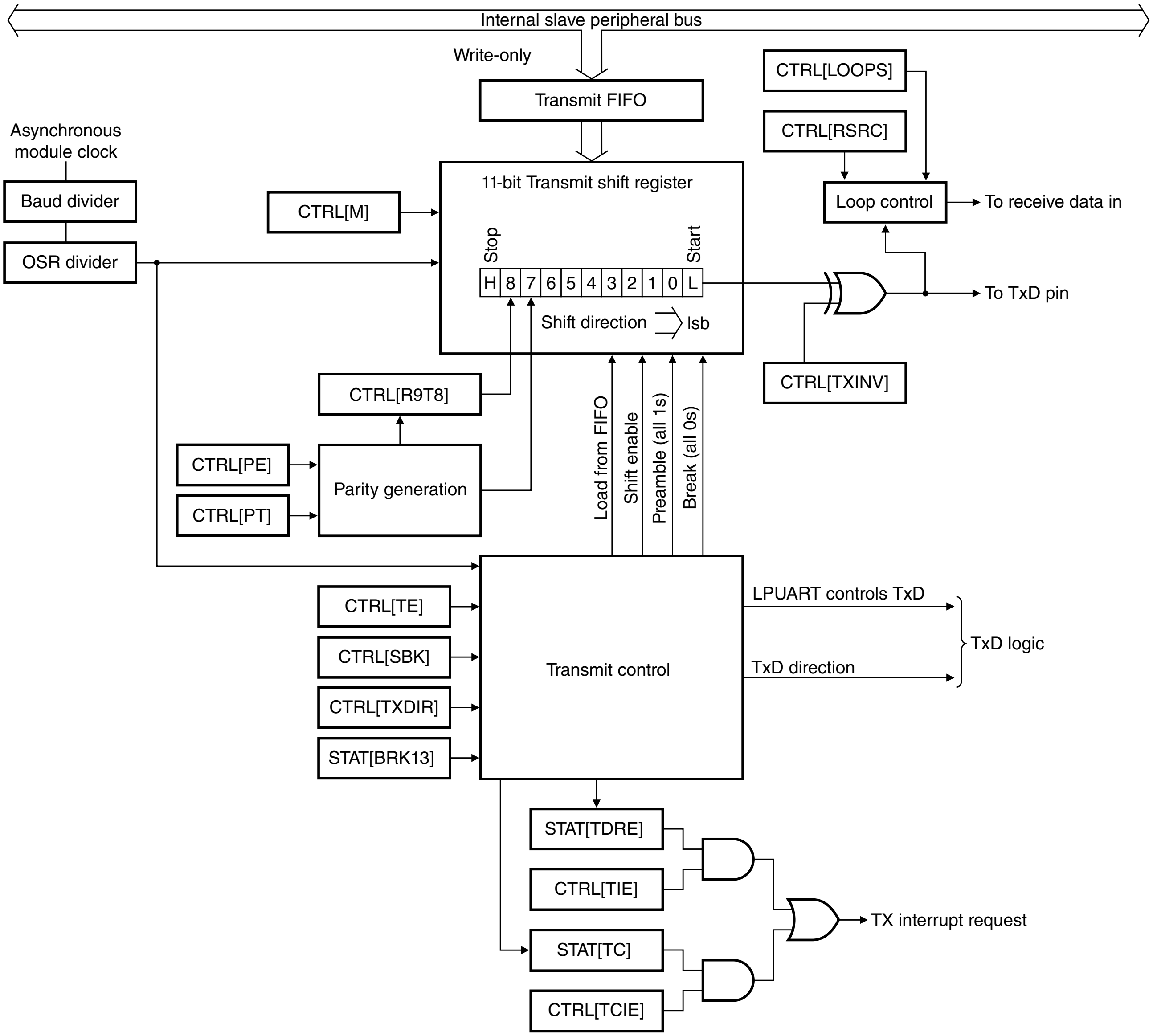


Figure 41-1. Transmitter block diagram

Figure 41-2 shows the receiver portion of LPUART.

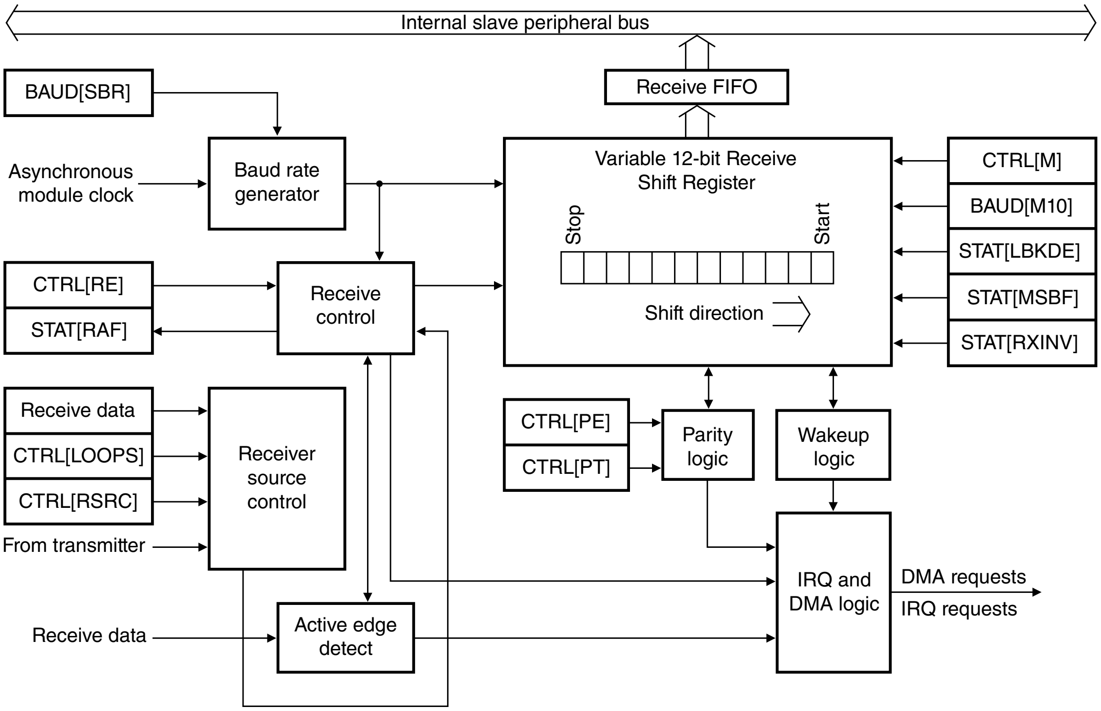


Figure 41-2. Receiver block diagram

41.2.2 Features

- Full-duplex, standard NRZ format
- Programmable baud rates (13-bit modulo divider) with a configurable oversampling ratio (OSR) from 4× to 32×
- Asynchronous operation of transmit and receive baud rates with respect to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency.
  - Operation in Low-Power modes is supported.
- Interrupt, DMA, or polled operations:
  - Transmit data empty and transmission complete
  - Receive data 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 7-bit, 8-bit, 9-bit, or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Support for three receiver wake-up methods:

  - Idle line wake-up
  - Address mark wake-up
  - 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 and 11-bit break character generation
- 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 a programmable pulse width
- Independent FIFO structure for transmit and receive functions:
  - Separate configurable watermarks for receive and transmit requests
  - Option for receiver to assert request after a configurable number of idle characters, if receive FIFO is not empty

41.3 Functional description

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 sections describe all LPUART blocks.

41.3.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and transmitter. The value, ranging from 1 to 8191, written to BAUD[SBR] determines the baud clock divisor for the asynchronous LPUART baud clock. The baud rate clock drives the receiver, while a bit clock, generated from the baud rate clock divided by the OSR, drives the transmitter. Depending on the OSR, the receiver has an acquisition rate of 4 to 32 samples per bit time.

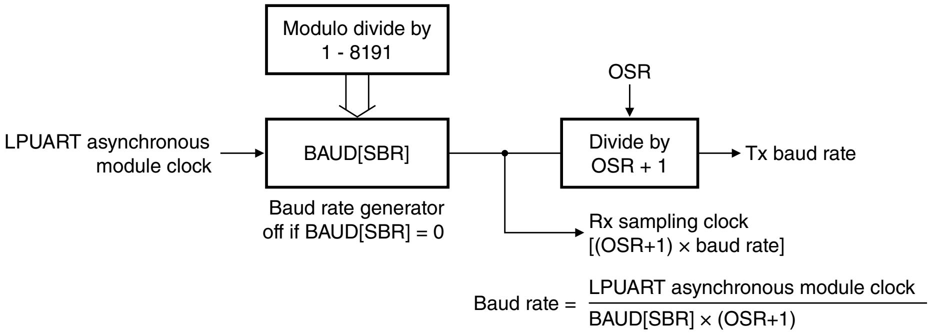

Modulo divide by 1 - 8191
LPUART asynchronous module clock
BAUD[SBR]
Baud rate generator off if BAUD[SBR] = 0
OSR
Divide by OSR + 1
Tx baud rate
Rx sampling clock [(OSR+1) × baud rate]
Baud rate = LPUART asynchronous module clock / BAUD[SBR] × (OSR+1)

Figure 41-3. Baud rate generation

Baud rate generation is subject to these 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 lead to a phase shift.

Baud rate generation is a free-running counter that continues whenever the transmitter or receiver is enabled. The transmitter bit clock continues whenever the transmitter is enabled; each transmitted character aligns to the next edge of the transmit bit clock.

In general, configuring OSR for a higher ratio and/or sampling on both edges of the clock slightly improves LPUART's tolerance to baud rate mismatch between the received data and LPUART configured baud rate. However, the three data samples in each bit (see Data sampling technique) are also closer together, which may impact noise sensitivity.

41.3.2 Transmitter functional description

This section describes the functioning of the LPUART transmitter, as shown in the transmitter portion of Block diagram, as well as specialized functions for sending break and idle characters.

The transmitter output (TX) idle state defaults to logic high; the transmitter output is inverted when you write 1 to CTRL[TXINV], which becomes 0 following reset. You can enable the transmitter by writing 1 to CTRL[TE]. 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 FIFO and programs store data in the transmit FIFO by writing to Data (DATA).

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13-bit long depending on the settings of CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS]. Going forward in this discussion, assume that CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] are 0, selecting the normal 8-bit Data mode, in which 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 transmit FIFO is transferred to the transmit shift register, synchronized with the baud rate clock, and STAT[TDRE] becomes 1 to indicate that another character may be written to the transmit FIFO at Data (DATA).

If no new character is waiting in the transmit FIFO after a stop bit is shifted out of the TX pin, the transmitter sets the transmit complete flag and enters an idle mode, with 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 does not start transmitting another character.

41.3.2.1 Break character length

CTRL[SBK] sends break characters, originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 9-bit to 12-bit times, including the start and stop bits. You can enable a longer break of 13-bit times by writing 1 to STAT[BRK13]. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, the program writes 1 and then writes 0 to CTRL[SBK]. This action queues a break character to be sent as soon as the shifter is available. If CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized with the baud rate clock, an additional break character is queued. When LPUART is the receiving module, it receives a break character as 0s in all data bits and a framing error (STAT[FE] = 1) is detected.

You can also transmit a break character by writing to Data (DATA) with DATA[FRETSC] = 1 and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows DMA to transmit a break character.

When idle line wake-up is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program waits for STAT[TDRE] to become 1 to indicate that the last character of a message has moved to the transmit shifter. Next, write 0 and then write 1 to CTRL[TE]. 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 CTRL[TE] becomes 0, the LPUART transmitter does not release control of the TX pin.

You can also write to Data (DATA) to transmit an idle character, with DATA[FRETSC] and DATA[R9T9] = 1 and the values of all the other fields = 0. This supports transmitting the idle character as part of the normal data stream and also allows DMA to transmit an idle character.

As shown in the following table, STAT[BRK13], CTRL[M], CTRL[M7], BAUD[M10], and BAUD[SBNS] affect the length of the break character.

Table 41-2. Break character length

STAT[BRK13]	CTRL[M]	BAUD[M10]	CTRL[M7]	BAUD[SBNS]	Break character length (in bit times)
0	0	0	0	0	10
0	0	0	0	1	11
0	0	0	1	0	9
0	0	0	1	1	10
0	1	0	—	0	11
0	1	0	—	1	12
0	—	1	—	0	12
0	—	1	—	1	13
1	0	0	0	0	13
1	0	0	0	1	13
1	0	0	1	0	12
1	0	0	1	1	12
1	1	0	—	0	14
1	1	0	—	1	14
1	—	1	—	0	15
1	—	1	—	1	15

41.3.2.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS. If the CTS 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 transmitter FIFO, the character in the transmit shift register is complete. Any characters in the FIFO wait for CTS_B to assert again, and TX remains in the mark state (idle state) until CTS is reasserted. The CTS pin must assert for longer than one bit period to guarantee that a new transmission is started when the transmitter is idle with DTS.

If the CTS operation is disabled, the transmitter ignores the state of CTS.

The transmitter's CTS signal can be enabled even if the same LPUART receiver's RTS signal is disabled.

41.3.2.3 Transceiver driver enable

The transmitter can use RTS as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS for details. If the RTS operation is enabled, when a character is placed into an empty transmit shift register, RTS asserts 1-bit time before the start bit is transmitted. RTS remains asserted for the whole time that the transmit shift register has any characters. RTS deasserts 1-bit time after all characters in the transmit FIFO and shift register are completely sent, including the last stop bit. In other words, when RTS is used as a transceiver enable, RTS asserts 1-bit time before the transmitter starts transmitting and negates 1-bit time after the transmitter goes idle.

Transmitting a break character also asserts RTS, with the same assertion and deassertion timing as having a character in the transmit shift register.

The transmitter's RTS signal asserts only when the transmitter is enabled. However, the transmitter's RTS signal is unaffected by its CTS signal. RTS remains asserted until the transfer is complete, even if the transmitter is disabled mid-way through a data transfer.

41.3.2.4 Transceiver driver enable using RTS

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is three-stated unless LPUART is driving. The transmitter can use the RTS signal to enable the driver of a transceiver. The polarity of RTS can be matched to the polarity of the transceiver's driver enable signal.

The following figure shows the receiver enable signal asserted. This connection can also connect RTS to both DE and RE_B. The transceiver's receiver is disabled when driving. A pullup can pull RX to a nonfloating value during this time. You can refine this option further by operating LPUART in Single-Wire mode, freeing the RX pin for other uses.

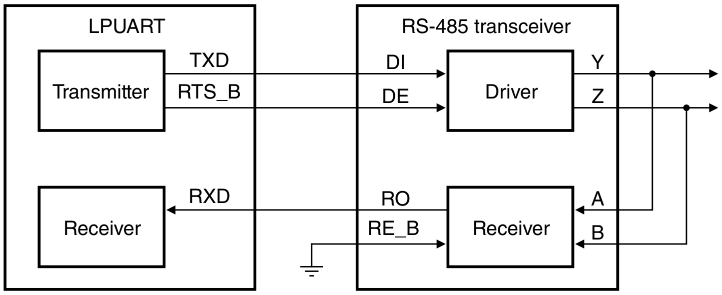


Figure 41-4. Transceiver driver enable using RTS

41.3.3 Receiver functional description

This section discusses the functioning of the LPUART receiver, as shown in the receiver portion of Block diagram. The section also discusses:

- The data sampling technique used to reconstruct receiver data.
- Different variations of the receiver wake-up function.

You can invert the receiver input by writing 1 to STAT[RXINV] and enable the receiver by writing 1 to CTRL[RE]. Character frames consist of a start bit of logic 0, sever to 10 data bits (MSB or LSB first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit, or 10-bit Data mode, see Data modes. Going forward in this discussion, assume that LPUART is configured for a normal 8-bit Data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full (STAT[RDRF] = 0), the data character is transferred to the receive FIFO, resulting in STAT[RDRF] becoming 1. However, if STAT[RDRF] is already 1, indicating that the receive data buffer is already full, STAT[OR] becomes 1 and the new data is lost.

Because the LPUART receiver is separate from the receive FIFO, the receive shift register can receive the next word when the receive FIFO is full, and it is only at the end of the character that the next data is written into the receive FIFO, potentially triggering the overrun flag if the FIFO is full.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the FIFO by reading Data (DATA). See Interrupts for details about flag clearing.

41.3.3.1 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 considering logic level samples at the oversampling rate times the baud rate to search for a falling edge on the 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 ensure that this is a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes they are synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts 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, noise flag (STAT[NF]) becomes 1 when the received character is transferred to the receive FIFO.

When the LPUART receiver is configured to sample on both edges of the baud rate clock (that is, when BAUD[BOTHEDGE] = 1), 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. You must enable sampling on both edges of the clock for oversampling rates of 4× to 7×. This sampling is optional for higher oversampling rates.

The synchronization feature of LPUART synchronizes the internal oversampling counter with a detected falling edge on the receive signal, and to adjust the data sampling window. The falling edge detection needs three consecutive 1s prior to the "1->0" (one to zero) transition. After the initial falling edge detection for the start bit, the circuit continuously monitors the next falling edge, and resets the counter after another falling edge is detected. This is called resynchronization.

When BAUD[RESYNCDIS] is 0, you perform this falling edge detection and resynchronization not only for the start bit but also for the rest of the character reception after the start bit.

When BAUD[RESYNCDIS] is 1, you perform the falling edge detection and resynchronization only for the start bit. The use case for disabling the resynchronization is protocols that require this (for example, LIN 2.1 prohibits resynchronization within a byte).

The following table and figure explain LPUART resynchronization.

Table 41-3. LPUART resynchronization settings

Resynchronization	BAUD[RESYNCDIS] = 0	BAUD[RESYNCDIS] = 1
For the starting bit falling edge	Yes	Yes
For all falling edges after the start bit	Yes	No

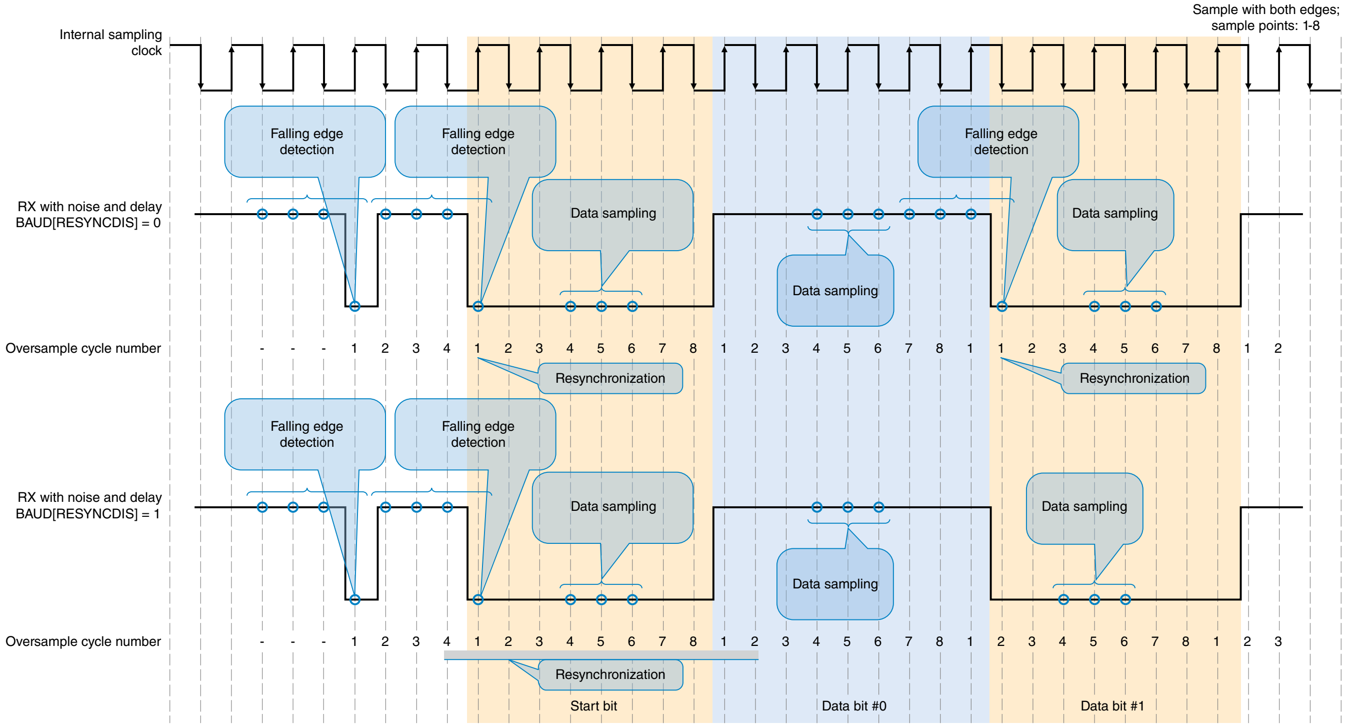


Figure 41-5. LPUART resynchronization diagram

41.3.3.2 Receiver wake-up operation

Receiver wake-up and receiver address matching are hardware mechanisms that allow an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wake-up, 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 1 to CTRL[RWU].

When CTRL[RWU] and STAT[RWUID] are 1, the status fields associated with the receiver, with the exception of STAT[IDLE], are inhibited from becoming 1, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, all receivers automatically force CTRL[RWU] to become 0. This results in all receivers waking 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 ignores all characters that do not meet the address match requirements.

Table 41-4. Receiver wake-up options

CTRL[RWU]	BAUD[MAEN1] | BAUD[MAEN2]	BAUD[MATCFG]	CTRL[WAKE] :STAT[RWUID]	Receiver wake-up
0	0	X	X	Normal operation
1	0	00	00	Receiver wake-up on idle line; STAT[IDLE] = 0
1	0	00	01	Receiver wake-up on idle line; STAT[IDLE] = 1
1	0	00	10	Receiver wake-up on address mark
1	1	11	10	Receiver wake-up on data match
0	1	00	X0	Address mark address match; STAT[IDLE] = 0 for discarded characters
0	1	00	X1	Address mark address match; STAT[IDLE] = 1 for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Match on and match off; STAT[IDLE] = 0 for discarded characters
0	1	10	X1	Match on and match off; STAT[IDLE] = 1 for discarded characters

41.3.3.2.1 Idle line wake-up

When CTRL[WAKE] is 0, you can configure the receiver for an idle line wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a full character time of the idle-line level.

CTRL[M], CTRL[M7], and BAUD[M10] select 7-bit to 10-bit Data mode and BAUD[SBNS] selects a 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 9 to 13 bit times because of the start and stop bits.

When CTRL[RWU] is 1 and STAT[RWUID] is 0, the idle condition that wakes up the receiver does not lead to STAT[IDLE] becoming 1. The receiver wakes up and waits for the first data character of the next message that leads to STAT[RDRF] becoming 1 and generates an interrupt if enabled. When STAT[RWUID] is 1, any idle condition leads to STAT[IDLE] becoming 1 and generates an interrupt if enabled, regardless of whether CTRL[RWU] is 0 or 1.

These are the ways to detect an idle line:

- When CTRL[ILT] is 0, the idle bit counter starts after the start bit so that the stop bit and any logic 1s at the end of a character count to calculate the full character time of idle.
- When CTRL[ILT] is 1, the idle bit counter does not start until after the stop bit time so that the data in the last character of the previous message does not impact the idle detection.

41.3.3.2.2 Address mark wake-up

When CTRL[WAKE] is 1, you can configure the receiver for an address mark wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a logic 1 in the most significant bit of the received character. When parity is enabled, the second most significant bit is used for address mark wake-up.

Address mark wake-up allows messages to contain idle characters, but requires one bit to be reserved for use in address frames. The logic 1 in the most significant bit (or second most significant bit when parity is enabled) of an address frame writes 0 to CTRL[RWU] and writes 1 to STAT[RDRF]. In this case, the character with the address mark bit is received even if the receiver is sleeping during most of this character time.

41.3.3.2.3 Data match wake-up

When CTRL[RWU] and CTRL[WAKE] are 1, and BAUD[MATCFG] equals 11, the receiver is configured for a data match wake-up. In this mode, CTRL[RWU] becomes 0 automatically when the receiver detects a character that matches MATCH[MA1] when BAUD[MAEN1] is 1, or that matches MATCH[MA2] when BAUD[MAEN2] is 1.

41.3.3.2.4 Address match operation

You can enable the address match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 0. In this function, a character that the RX pin receives with a logic 1 in the most significant bit (or the second most significant bit when parity is enabled) is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. The character is only transferred to the receive buffer, and STAT[RDRF] becomes 1 if the comparison matches. All subsequent characters received with a logic 0 in the most significant bit (or the second most significant bit when parity is enabled) are considered to be data associated with the address and are transferred to the receive FIFO. If no marked address match occurs, no transfer is made to the receive FIFO, and all the characters that follow, with logic 0 in the most significant bit (or second most significant bit when parity is enabled), are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

The address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] :

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, a marked address is compared only to the associated Match Address (MATCH) and data is transferred to the receive FIFO only on a match.
- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, a marked address is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

41.3.3.2.5 Idle match operation

You can enable the idle match operation when either BAUD[MAEN1] or BAUD[MAEN2] is 1 and BAUD[MATCFG] is 1. In this function, the first character that the RX pin receives after an idle line condition is considered an address and is compared to the associated MATCH[MA1] or MATCH[MA2]. The character is transferred only to the receive buffer, and STAT[RDRF] becomes 1, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive FIFO until the next idle line condition is detected. If no address match occurs, no transfer is made to the receive FIFO, and all the frames that follow, until the next idle condition, are also discarded. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

An idle match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] :

- If either BAUD[MAEN1] or BAUD[MAEN2] is 1, the first character after an idle line is compared only to the associated Data (DATA) and data is transferred to the receive FIFO only on a match.
- If both BAUD[MAEN1] and BAUD[MAEN2] are 1, the first character after an idle line is compared to both MATCH[MA1] and MATCH[MA2] and data is transferred only on a match with either of these fields.

41.3.3.2.6 Match on, match off operation

The match on, match off operation is enabled when both BAUD[MAEN1] and BAUD[MAEN2] are 1 and BAUD[MATCFG] = 10. In this function, a character that the RX pin receives matches MATCH[MA1] and is transferred to the receive buffer, and STAT[RDRF] becomes 1. All subsequent characters are considered to be data and are also transferred to the receive FIFO, until a character that matches MATCH[MA2] is received. The character that matches MATCH[MA2], along with all subsequent characters, is discarded; and this continues until another character that matches MATCH[MA1] is received. If both BAUD[MAEN1] and BAUD[MAEN2] are 0, the receiver operates normally, and all the received data is transferred to the receive FIFO.

NOTE

The match on, match off operation requires both BAUD[MAEN1] and BAUD[MAEN2] to be 1.

41.3.3.3 Hardware flow control

To support hardware flow control, you can program the receiver to automatically assert and deassert RTS:

- RTS remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS for more information.
- If the receiver RTS functionality is enabled, the receiver automatically deasserts RTS if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1.
- The receiver asserts RTS when STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. There is no impact if STAT[RDRF] is 1 already.
- Even if RTS is deasserted, the receiver continues to receive characters until the receive FIFO is overrun.
- If the receiver RTS functionality is disabled, the receiver's RTS remains deasserted.

41.3.4 Additional LPUART functions

41.3.4.1 Data modes

You can configure the LPUART transmitter and receiver to operate in 7-bit Data mode by writing 1 to CTRL[M7], 9-bit Data mode by writing 1 to CTRL[M], or 10-bit Data mode by writing 1 to BAUD[M10]. In 9-bit Data mode, there exists a ninth data bit and in 10-bit mode, there exists a tenth data bit.

When performing 8-bit writes to the transmit FIFO, the ninth and tenth bits are pushed into the FIFO from CTRL[T8] and CTRL[T9]. For coherent 8-bit writes, you must write to CTRL[T8] and CTRL[T9] before writing to Data (DATA) [7:0]. However, if the values in CTRL[T8] or CTRL[T9] do not need to change, it is not necessary to update CTRL[T8] and CTRL[T9] before every 8-bit write to Data (DATA).

When performing 16-bit or 32-bit writes to the transmit FIFO, all 10 bits are pushed into the transmit FIFO from the write data.

When performing 8-bit reads of the receive FIFO, the ninth and tenth bits are held in CTRL[R8] and CTRL[R9] but you must read them before reading Data (DATA). A 16-bit or 32-bit read of the receive FIFO returns all 10 bits in Data (DATA).

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 the address mark wake-up so that the ninth data bit can serve as the wake-up bit. The 10-bit Data mode is typically used with parity and address mark wake-up so that the ninth data bit can serve as the wake-up bit and the tenth bit can serve as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

41.3.4.2 Idle length

An idle character is one where the start bit, all data bits, and stop bits are in the mark position (idle state, generally logic 1). You can configure CTRL[ILT] to start detecting an idle character from the previous start bit (any data bits and stop bits count for idle character detection) or from the previous stop bit.

You can also use CTRL[IDLECFG] to configure the number of idle characters that must be received before an idle line condition is detected. This field configures the number of idle characters that must be received before STAT[IDLE] becomes 1, STAT[RAF] becomes 0, and DATA[IDLINE] becomes 1 with the next received character.

CTRL[IDLECFG] also affects the idle line wake-up and idle match operations. When either the address match or match on/off operation is enabled, writing 1 to STAT[RWUID] causes any discarded characters to be treated as idle characters.

41.3.4.3 Loop mode

When CTRL[LOOPS] is 1, CTRL[RSRC] selects between Loop mode (CTRL[RSRC] = 0) or Single-Wire mode (CTRL[RSRC] = 1). You, sometimes, use Loop mode 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 LPUART does not use the RX pin.

41.3.4.4 Single-Wire mode

When CTRL[LOOPS] is 1, CTRL[RSRC] selects either Loop mode (CTRL[RSRC] = 0) or Single-Wire mode (CTRL[RSRC] = 1). Single-Wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and TX pin (the RX pin is not used).

In Single-Wire mode, CTRL[TXDIR] controls the direction of serial data on the TX pin. When CTRL[TXDIR] becomes 0, the TX pin is an input to the receiver and the transmitter is temporarily disconnected from the TX pin so that an external device can send serial data to the receiver. When CTRL[TXDIR] = 1, the TX pin is an output that the transmitter drives. The internal loop back connection is disabled, and as a result, the receiver is unable to receive characters that the transmitter sends out.

41.3.5 Peripheral triggers

The connection of the LPUART peripheral triggers with other peripherals is chip-specific.

41.3.5.1 Output triggers

LPUART generates the following output triggers that can be connected to other peripherals on the chip:

- The transmit word trigger asserts at the end of each transmitted word and negates after 1-bit period.
- The receive word trigger asserts at the end of each received word that is written to the receive FIFO, for one oversampling clock period.
- The receive idle trigger asserts when STAT[IDLE] becomes 1, for one oversampling clock period.

41.3.5.2 Input trigger

LPUART supports a peripheral input trigger that you can configure in one of the following ways:

- By enabling the CTS function: You can connect the input trigger instead of the CTS pin input. The input trigger must assert for longer than 1-bit clock period when the transmitter is idle, with data to send, to guarantee a new transmission.
- By making the input trigger modulate the transmit data output (trigger is logically ANDed with the TX output): The input trigger is expected to be a free-running clock (carrier signal) that generates from a timer or PWM source with a frequency that is greater than the bit-clock frequency. The carrier signal must not toggle faster than the maximum supported bit time.
- By connecting the input trigger instead of the RX pin input: The input trigger is expected to be generated from a receive data source, such as an analog comparator or external pin.

41.3.6 Infrared (IR) interface

LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses, transforming them to serial bits, which are then sent to LPUART. The IrDA physical layer specification defines a half-duplex IR communication link for exchanging data. The full standard includes data rates up to 16 Mbit/s. The LPUART IrDA support is limited to SIR mode that supports data rates only between 2.4 kbit/s and 115.2 kbit/s.

LPUART has an infrared transmit encoder and a receive decoder. The infrared decoder converts the received character from the IrDA format to the NRZ format, which the receiver uses. It also has an OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received. LPUART transmits serial bits of data, which the infrared submodule encodes, to transmit a narrow pulse for every zero bit. No pulse is transmitted for every single bit. When receiving data, an IR photo diode (external to

LPUART) detects the IR pulses. The IR receive decoder transforms them to CMOS levels. The infrared receive decoder then stretches the narrow pulses to get back to a serial bit stream that LPUART receives. You can invert the polarity of transmitted pulses and expected receive pulses so that a direct connection can be made to external IrDA transceiver modules that use active-high pulses.

The IR submodule receives its clock sources from LPUART. The submodule selects one of these clocks to generate either 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulses during transmission.

41.3.6.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from the transmit shift register to the TX signal. A narrow pulse is transmitted for a 0 bit and no pulse is transmitted for a 1 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 0 bit when CTRL[TXINV] is 0, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] is 1.

41.3.6.2 Infrared receive decoder

The infrared receive block converts data from the RX signal to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. A narrow low pulse is expected for a 0 bit when STAT[RXINV] is 0, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] is 1. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

41.3.6.3 Start-bit detection

When STAT[RXINV] is 0, 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 of each other.

41.3.6.4 Noise filtering

The decoder ignores any rising edges detected during the first half of the infrared decoder counter, and can leave any pulses less than one oversampling baud clock as undetected. This is regardless of whether the pulse is seen in the first or second half of the count.

41.3.6.5 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as 0, which is sent to the receiver. The decoder counter is also reset.

41.3.6.6 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, the decoder sends a 1 to the receiver.

If the next bit is 0, which arrives late, a low bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, the delay of a 0 is not recorded as noise.

41.3.7 Modes of operation

41.3.7.1 Low-Power modes

LPUART remains functional during low-power modes, provided CTRL[DOZEEN] is 0 and the LPUART functional clock is enabled. LPUART can generate an interrupt or DMA request to cause a wake-up from low-power modes.

You can configure LPUART to be disabled in low-power modes, when CTRL[DOZEEN] is 1. In this case, the transmitter and receiver finish transmitting and receiving the current word.

If LPUART is disabled in low-power modes, it can generate a wake-up via STAT[RXEDGIF] if the receiver detects an active edge.

NOTE

See the chip-specific information for specific low-power modes available on your chip.

41.3.7.2 Debug mode

LPUART remains functional in Debug mode.

41.3.8 Clocking

Table 41-5. Types of clocks

Clock	Description
Functional	Is asynchronous to the bus clock and can remain enabled in Low-Power modes to support transmit and/or receive functions, including low-power wake-up.
Bus	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 LPUART transmit and receive registers, including the FIFOs.

41.3.9 Reset

Table 41-6. Types of resets

Reset	Description
Chip	Enables the logic and registers for the LPUART transmitter and receiver to reset to their default states.
Software	Resets the LPUART logic and registers to their default states, except for Global (GLOBAL). GLOBAL[RST] controls the LPUART software reset.
FIFO	Implements write-only control fields that reset the transmit FIFO (FIFO[TXFLUSH]) and receive FIFO (FIFO[RXFLUSH]). After a FIFO is reset, that FIFO becomes empty.

41.3.10 Interrupts

The LPUART transmitter has two status fields that can optionally generate hardware interrupt requests. If STAT[TDRE] is 1, it indicates that there is room in the transmit FIFO to write another transmit character to Data (DATA). If CTRL[TIE] is 1, a hardware interrupt is requested when STAT[TDRE] is 1.

STAT[TC] indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TX at the inactive level. This field is often used in systems with modems to determine when it is safe to turn off the modem. If CTRL[TCIE] is 1, a hardware interrupt is requested when STAT[TC] is 1. Instead of hardware interrupts, software polling may be used to monitor STAT[TDRE] and STAT[TC] if the corresponding CTRL[TIE] or CTRL[TCIE] field is 0.

When a program detects that STAT[RDRF] is 1, it gets the data from this field by reading Data (DATA). The field becomes 0 by reading Data (DATA).

STAT[IDLE] includes logic that prevents it from becoming 1 repeatedly when the RX line remains idle for an extended period of time. STAT[IDLE] becomes 0 when you write 1 to it, and cannot become 1 again until the receiver has received at least one new character and has 1 as the value of STAT[RDRF].

If the associated error is detected in the received character that caused STAT[RDRF] to become 1, STAT[NF], STAT[FE], and STAT[PF] become 1 at the same time STAT[RDRF] becomes 1. These flags do not become 1 in overrun cases.

If STAT[RDRF] is already 1 when a new character is ready to be transferred from the receive shifter to the receive FIFO, STAT[OR] becomes 1, instead of the data along with any associated STAT[NF], STAT[FE], or STAT[PF] condition getting lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2], then STAT[MA1F] and/or STAT[MA2F] become 1 at the same time that STAT[RDRF] becomes 1.

At any time, an active edge on the RX serial data input pin causes STAT[RXEDGIF] to become 1. STAT[RXEDGIF] becomes 0 when you write 1 to it. This function depends on the receiver being enabled (the value of CTRL[RE] being 1).

41.4 External signals

Table 41-7. External signals

Signal	Description	I/O
TX	Transmit data: This pin is normally an output, but is an input (tristated) in Single-Wire mode whenever the transmitter is disabled or the transmit direction is configured for receive data.	I/O
RX	Receive data	I
CTS	Clear-to-send	I
RTS	Request-to-send	O

41.5 Initialization

This module does not require initialization.

41.6 Register definition

LPUART includes registers to control baud rate, select options, report status, and store transmit and receive data. Access to an address outside the valid memory map generates a bus error.

NOTE

Writing to a read-only (RO) register or reading a write-only (WO) register can cause bus errors. LPUART does not verify whether programmed values in the registers are correct; you must write valid values to them.

41.6.1 LPUART register descriptions

41.6.1.1 LPUART memory map

LPUART0 base address: 4006_A000h

LPUART1 base address: 4006_B000h

LPUART2 base address: 4006_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0401_0003h
4h	Parameter (PARAM)	32	R	0000_0202h
8h	Global (GLOBAL)	32	RW	0000_0000h
Ch	Pin Configuration (PINCFG)	32	RW	0000_0000h
10h	Baud Rate (BAUD)	32	RW	0F00_0004h
14h	Status (STAT)	32	RW	00C0_0000h
18h	Control (CTRL)	32	RW	0000_0000h
1Ch	Data (DATA)	32	RW	0000_1000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
20h	Match Address (MATCH)	32	RW	0000_0000h
24h	MODEM IrDA (MODIR)	32	RW	0000_0000h
28h	FIFO (FIFO)	32	RW	00C0_0011h
2Ch	Watermark (WATER)	32	RW	0000_0000h

41.6.1.2 Version ID (VERID)

41.6.1.2.1 Offset

Register	Offset
VERID	0h

41.6.1.2.2 Function

Indicates the version integrated for this instance on the chip and also specifies the inclusion and exclusion of several optional features.

41.6.1.2.3 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	1	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	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

41.6.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number for the module specification.

Table continues on the next page...

Field	Function
23-16 MINOR	Minor Version Number Indicates the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number Indicates the feature set number. 0000_0000_0000_0001b - Standard feature set 0000_0000_0000_0011b - Standard feature set with MODEM and IrDA support

41.6.1.3 Parameter (PARAM)

41.6.1.3.1 Offset

Register	Offset
PARAM	4h

41.6.1.3.2 Function

Indicates the parameter configuration for this instance on the chip.

41.6.1.3.3 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	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

41.6.1.3.4 Fields

Field	Function
31-16 —	Reserved
15-8	Receive FIFO Size

Table continues on the next page...

Field	Function
RXFIFO	Indicates the number of characters in the receive FIFO, which is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size Indicates the number of characters in the transmit FIFO, which is 2^TXFIFO.

41.6.1.4 Global (GLOBAL)

41.6.1.4.1 Offset

Register	Offset
GLOBAL	8h

41.6.1.4.2 Function

Performs global functions.

41.6.1.4.3 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														RST	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.4.4 Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Specifies whether the module is reset.

Table continues on the next page...

Field	Function
	This field resets all internal logic and registers, except Global (GLOBAL). The reset takes effect immediately and remains asserted until you negate it. There is no minimum delay required before clearing the software reset. 0b - Not reset 1b - Reset
0 —	Reserved

41.6.1.5 Pin Configuration (PINCFG)

41.6.1.5.1 Offset

Register	Offset
PINCFG	Ch

41.6.1.5.2 Function

Enables the selection of input pins.

41.6.1.5.3 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														TRGSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.5.4 Fields

Field	Function
31-2 —	Reserved

Table continues on the next page...

Field	Function
1-0 TRGSEL	Trigger Select Configures the input trigger usage. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Input trigger disabled 01b - Input trigger used instead of the RX pin input 10b - Input trigger used instead of the CTS pin input 11b - Input trigger used to modulate the TX pin output, which (after TXINV configuration) is internally ANDed with the input trigger

41.6.1.6 Baud Rate (BAUD)

41.6.1.6.1 Offset

Register	Offset
BAUD	10h

41.6.1.6.2 Function

Configures the baud rate.

41.6.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAEN1	MAEN2	M10	OSR					TDMAE	0	RDMAE	Reserved	MATCFG		BOTHEDGE	RESYNCDIS
W												0				
Reset	0	0	0	0	1	1	1	1	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	LBKDIE	RXEDGIE	SBNS	SBR												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

41.6.1.6.4 Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 Enables automatic address matching or data matching mode for MATCH[MA1]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
30 MAEN2	Match Address Mode Enable 2 Enables automatic address matching or data matching mode for MATCH[MA2]. If this field is 0, normal operation takes place. 0b - Disable 1b - Enable
29 M10	10-Bit Mode Select Causes the tenth bit to be a part of the serial transmission. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters 1b - Receiver and transmitter use 10-bit data characters
28-24 OSR	Oversampling Ratio Configures the OSR of the receiver. You must change the value of this field only when both the transmitter and receiver are disabled. **NOTE:** BAUD[OSR] results in an OSR of BAUD[OSR] + 1, for example, BAUD[OSR] = 0_0101b results in a final division by 6. 0_0000b - Results in an OSR of 16 0_0001b - Reserved 0_0010b - Reserved 0_0011b - Results in an OSR of 4 (requires BAUD[BOTHEDGE] to be 1) 0_0100b - Results in an OSR of 5 (requires BAUD[BOTHEDGE] to be 1) 0_0101b - Results in an OSR of 6 (requires BAUD[BOTHEDGE] to be 1) 0_0110b - Results in an OSR of 7 (requires BAUD[BOTHEDGE] to be 1) 0_0111b - Results in an OSR of 8 0_1000b - Results in an OSR of 9 0_1001b - Results in an OSR of 10 0_1010b - Results in an OSR of 11 0_1011b - Results in an OSR of 12 0_1100b - Results in an OSR of 13 0_1101b - Results in an OSR of 14 0_1110b - Results in an OSR of 15 0_1111b - Results in an OSR of 16 1_0000b - Results in an OSR of 17 1_0001b - Results in an OSR of 18 1_0010b - Results in an OSR of 19 1_0011b - Results in an OSR of 20 1_0100b - Results in an OSR of 21 1_0101b - Results in an OSR of 22 1_0110b - Results in an OSR of 23 1_0111b - Results in an OSR of 24 1_1000b - Results in an OSR of 25 1_1001b - Results in an OSR of 26 1_1010b - Results in an OSR of 27 1_1011b - Results in an OSR of 28

Table continues on the next page...

Field	Function
	1_1100b - Results in an OSR of 29 1_1101b - Results in an OSR of 30 1_1110b - Results in an OSR of 31 1_1111b - Results in an OSR of 32
23 TDMAE	Transmitter DMA Enable Enables STAT[TDRE] to generate a DMA request. 0b - Disable 1b - Enable
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable Enables STAT[RDRF] to generate a DMA request. 0b - Disable 1b - Enable
20 —	Reserved
19-18 MATCFG	Match Configuration Configures the match addressing mode used. You must change the value of this field only when both the transmitter and receiver are disabled. 00b - Address match wake-up 01b - Idle match wake-up 10b - Match on and match off 11b - Enables RWU on data match and match on or off for the 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 OSR. This field must be 1 for OSRs between x4 and x7 and is optional for higher OSRs. You must change the value of this field only when the receiver is disabled. If this field is 0, the receiver samples input data using the rising edge of the baud rate clock. If this field is 1, the receiver samples input data using the rising and falling edges of the baud rate clock. 0b - Rising edge 1b - Both rising and falling edges
16 RESYNCDIS	Resynchronization Disable Disables resynchronization of the received data word when a data one followed by data zero transition is detected. You must change the value of this field only when the receiver is disabled. 0b - Enable 1b - Disable
15 LBKDIE	LIN Break Detect Interrupt Enable Enables STAT[LBKDIF] to generate hardware interrupt requests. If this field is 0, hardware interrupts from STAT[LBKDIF] (uses polling) are disabled. If this field is 1, hardware interrupts are requested when STAT[LBKDIF] is 1. 0b - Disable 1b - Enable
14	RX Input Active Edge Interrupt Enable

Table continues on the next page...

Field	Function
RXEDGIE	Enables STAT[RXEDGIF] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[RXEDGIF] are disabled. If this field is 1, hardware interrupts are requested when STAT[RXEDGIF] is 1. Changing the value of CTRL[LOOPS] or CTRL[RSRC] when this field (RXEDGIE) is 1 can cause STAT[RXEDGIF] to become 1. 0b - Disable 1b - Enable
13 SBNS	Stop Bit Number Select Determines whether data characters include one or two stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - One stop bit 1b - Two stop bits
12-0 SBR	Baud Rate Modulo Divisor Sets the modulo divide rate for the baud rate generator. • If SBR is 0, baud rate generator is disabled. • If SBR is 1–8191, baud rate = baud clock ÷ ((OSR + 1) × SBR). You must update the 13-bit baud rate setting [SBR12:SBR0] only when both the transmitter and receiver are disabled (both CTRL[RE] and CTRL[TE] are 0).

41.6.1.7 Status (STAT)

41.6.1.7.1 Offset

Register	Offset
STAT	14h

41.6.1.7.2 Function

Provides the module status.

41.6.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	W1C	W1C										W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0				0		0						0	
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.7.4 Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag Indicates whether a LIN break character is detected. This field becomes 1 when the LIN break detect circuitry is enabled and a LIN break character is detected. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not detected 1b - Detected When writing 0b - No effect 1b - Clear the flag
30 RXEDGIF	RX Pin Active Edge Interrupt Flag Indicates whether an active edge on the receive pin has occurred. This field becomes 1 whenever the receiver is enabled and an active edge (falling if STAT[RXINV] is 0; rising if STAT[RXINV] is 1) on the RX pin occurs. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not occurred 1b - Occurred When writing 0b - No effect

Table continues on the next page...

Field	Function
	1b - Clear the flag
29 MSBF	MSB First Specifies the first bit that is transmitted after the start bit. If this field is 0, LSB (bit 0) is the first bit transmitted after the start bit (which means, the first bit received after the start bit is identified as bit 0). If this field is 1, MSB (identified as bit 9, bit 8, bit 7, or bit 6) is the first bit that is transmitted, after the start bit, depending on the settings of CTRL[M], CTRL[PE], and BAUD[M10]. Writing 1 to this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - LSB 1b - MSB
28 RXINV	Receive Data Inversion Specifies whether receive data is inverted. Writing 1 to this field reverses the polarity of the received data input. You must change the value of this field only when the receiver is disabled. **NOTE:** Writing 1 to this field inverts the RX input for all cases: data bits, start and stop bits, break, and idle. 0b - Inverted 1b - Not inverted
27 RWUID	Receive Wake Up Idle Detect Controls, for CTRL[RWU] on idle character detection, whether the idle character that wakes up the receiver writes 1 to STAT[IDLE]. For address match wake-up, this field controls whether STAT[IDLE] = 1 when the address does not match. You must change the value of this field only when the receiver is disabled. If this field is 0, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] does not become 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] does not become 1 when an address does not match. If this field is 1, during the Receive Standby state (CTRL[RWU] = 1), STAT[IDLE] becomes 1 upon detection of an idle character. During address match wake-up, STAT[IDLE] becomes 1 when an address does not match. 0b - STAT[IDLE] does not become 1 1b - STAT[IDLE] becomes 1
26 BRK13	Break Character Generation Length Selects the longer transmitted break character length. The state of this field does not affect the detection of a framing error. You must change the value of this field only when the transmitter is disabled. You can send a break character by writing 1 to CTRL[SBK], or by writing the transmit FIFO when DATA[FRETSC] is 1 and DATA[R9T9] is 0. 0b - 9 to 13 bit times 1b - 12 to 15 bit times
25 LBKDE	LIN Break Detection Enable Enables LIN break detection. If this field is 0, LIN break detect is disabled, and only a normal break character can be detected. If this field is 1, LIN break detect is enabled and the LIN break character is detected at a length of 11 bit times (if CTRL[M] is 0), 12 bit times (if CTRL[M] is 1), or 13 bit times (if BAUD[M10] is 1).

Table continues on the next page...

Field	Function
	This field selects a longer break character detection length. When the field is 1, receive data is not stored in the receive FIFO. **NOTE:** This field enables the LIN break detect circuit and disables writing receive data to FIFO. Therefore, it ignores all characters except a LIN break. 0b - Disable 1b - Enable
24 RAF	Receiver Active Flag Indicates whether the LPUART receiver is idle or active. This field becomes 1 when the receiver detects the beginning of a valid start bit, and the field becomes 0 automatically when the receiver detects an idle line. 0b - Idle, waiting for a start bit 1b - Receiver active (RX pin input not idle)
23 TDRE	Transmit Data Register Empty Flag Indicates whether the transmit FIFO level is greater than, equal to, or less than the watermark. After the transmit FIFO is enabled, this field becomes 1 when the number of datawords in the transmit FIFO is equal to, or less than the number that WATER[TXWATER] indicates. To make the value of this field 0, write to it until the number of words in the transmit FIFO is greater than the number that WATER[TXWATER] indicates. After the transmit FIFO is disabled, this field becomes 1 to indicate that the FIFO level is less than the watermark. To make the value of this field 0 again, write to Data (DATA). This register is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Greater than watermark 1b - Equal to or less than watermark
22 TC	Transmission Complete Flag Indicates whether the transmitter is active. This field becomes 0 when a transmission is in progress or a preamble or break character is loaded; in other words, when the transmitter is active (sending data, a preamble, or a break). The field becomes 1 when the transmit buffer is empty and no data, preamble, or break character is being transmitted; in other words, when the transmission activity is complete. When this happens, the transmit data output signal becomes idle (logic 1). This field becomes 0 after you write to Data (DATA) to transmit new data, queuing a preamble by first writing 0 and then writing 1 to CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active 1b - Transmitter idle
21 RDRF	Receive Data Register Full Flag Indicates whether the receive FIFO level is less than, equal to, or greater than the watermark. This field becomes 1 when the number of datawords in the receive buffer is greater than the number that WATER[RXWATER] indicates and the receive FIFO is enabled. To write 0 to this field, read Data (DATA) until the number of datawords in the receive FIFO is equal to, or less than the number that WATER[RXWATER] indicates. When the receive FIFO is disabled, this field (RDRF) becomes 1 if the receive buffer (Data (DATA)) is full. To make this field 0, read Data (DATA). A character that is in the process of being received does not cause a change in this field until the entire character is received. Even if this field is 1, the character continues to be received until an overrun condition occurs after the entire character is received. 0b - Equal to or less than watermark 1b - Greater than watermark
20	Idle Line Flag

Table continues on the next page...

Field	Function
IDLE	Indicates whether an idle line is detected. This field becomes 1 when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] is 0, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bit time count towards the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. After CTRL[ILT] becomes 1, the receiver does not 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 towards the full character time of logic high needed for the receiver to detect an idle line. For this field to become 0, write 1 to it. After the field becomes 0, you cannot write 1 to it again until after a new character is stored in the receive buffer or a LIN break character writes 1 to STAT[LBKDIF]. This field becomes 1 only once, even if the receive line remains idle for an extended period. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Idle line detected 1b - Idle line not detected When writing 0b - No effect 1b - Clear the flag
19 OR	Receiver Overrun Flag Indicates whether there is receive overrun. This field becomes 1 when you cannot prevent STAT[RDRF] from overflowing with data. The field becomes 1 immediately after the stop bit is completely received for the dataword that overflows the buffer and all the other error fields (STAT[FE], STAT[NF], and STAT[PF]) are prevented from becoming 1. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If STAT[LBKDE] is enabled and a LIN break is detected, this field becomes 1 if STAT[LBKDIF] is not 0 before the next data character is received. When this field is 1, no additional data is stored in the receive FIFO even if sufficient room exists. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No overrun 1b - Receive overrun (new LPUART data is lost) When writing 0b - No effect 1b - Clear the flag
18 NF	Noise Flag Indicates whether noise is detected in the received character of Data (DATA). The advanced sampling technique used in the receiver takes three samples in each of the received bits. If some of these samples disagree with the rest of the samples within any bit time in the frame, then noise is detected for that character. This field becomes 1 whenever the next character to be read from Data (DATA) is received with noise detected within the character. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No noise detected 1b - Noise detected When writing

Table continues on the next page...

Field	Function
	0b - No effect 1b - Clear the flag
17 FE	Framing Error Flag Indicates whether a framing error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) is received with logic 0 detected where a stop bit was expected. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No framing error detected (this does not guarantee that the framing is correct) 1b - Framing error detected When writing 0b - No effect 1b - Clear the flag
16 PF	Parity Error Flag Indicates whether a parity error is detected. This field becomes 1 whenever the next character to be read from Data (DATA) is received when parity is enabled (CTRL[PE] is 1) and the parity bit in the received character does not agree with the expected parity value. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No parity error detected 1b - Parity error detected When writing 0b - No effect 1b - Clear the flag
15 MA1F	Match 1 Flag Indicates whether the received data is equal to MATCH[MA1]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA1]. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - Not equal to MA1 1b - Equal to MA1 When writing 0b - No effect 1b - Clear the flag
14 MA2F	Match 2 Flag Indicates whether the received data is equal to MATCH[MA2]. This field becomes 1 whenever the next character to be read from Data (DATA) matches the value of MATCH[MA2]. **NOTE:** This field behaves differently for register reads and writes. When reading

Table continues on the next page...

Field	Function
	0b - Not equal to MA2 1b - Equal to MA2 When writing 0b - No effect 1b - Clear the flag
13-10 —	Reserved
9-8 —	Reserved
7-2 —	Reserved
1-0 —	Reserved

41.6.1.8 Control (CTRL)

41.6.1.8.1 Offset

Register	Offset
CTRL	18h

41.6.1.8.2 Function

Controls various optional features of the LPUART system.

You must write to the fields of this register only when both the transmitter and receiver are disabled.

41.6.1.8.3 Diagram

Bits	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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1IE	MA2IE	0	0	M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.8.4 Fields

Field	Function
31 R8T9	Receive Bit 8 Transmit Bit 9 Contains R8 and T9 that correspond to different functions. R8 is the ninth data bit received after you configure LPUART for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading Data (DATA). T9 is the tenth data bit transmitted after you configure LPUART for 10-bit data formats. When writing 10-bit data, write T9 before writing to Data (DATA). If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE:** R8 is a read-only bit and T9 is a write-only bit; the value read is different from the value written.
30 R9T8	Receive Bit 9 Transmit Bit 8 Contains R9 and T8 that correspond to different functions. R9 is the tenth data bit received after you configure LPUART for 10-bit data formats. When reading 10-bit data, read R9 before reading Data (DATA). T8 is the ninth data bit transmitted after you configure LPUART for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing to Data (DATA). If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then you need not write to it each time you write to Data (DATA). **NOTE:** R9 is a read-only field and T8 is a write-only field; the value read is different from the value written.
29 TXDIR	TX Pin Direction in Single-Wire Mode Determines the direction of data at the TX pin, in Single-Wire mode, when LPUART is configured for a single-wire half-duplex operation (CTRL[LOOPS] and CTRL[RSRC] are 1). When writing 0 to this field, the transmitter finishes transmitting the current character (if any) before the receiver starts receiving data from the TX pin. 0b - Input 1b - Output
28 TXINV	Transmit Data Inversion Specifies whether transmit data is inverted.

Table continues on the next page...

Field	Function
	Writing 1 to this field reverses the polarity of the transmitted data output. This action inverts the TX output for all cases: data bits, start and stop bits, break, and idle. 0b - Not inverted 1b - Inverted
27 ORIE	Overrun Interrupt Enable Enables STAT[OR] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when OR interrupts are disabled. 0b - Disable 1b - Enable
26 NEIE	Noise Error Interrupt Enable Enables STAT[NF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when NF interrupts are disabled. 0b - Disable 1b - Enable
25 FEIE	Framing Error Interrupt Enable Enables STAT[FE] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when FE interrupts are disabled. 0b - Disable 1b - Enable
24 PEIE	Parity Error Interrupt Enable Enables STAT[PF] to generate hardware interrupt requests. When this field is 1, a hardware interrupt is requested. Use polling when PF interrupts are disabled. 0b - Disable 1b - Enable
23 TIE	Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests if STAT[TDRE] is 1. 0b - Disable 1b - Enable
22 TCIE	Transmission Complete Interrupt Enable Enables STAT[TC] to generate interrupt requests if STAT[TC] is 1. 0b - Disable 1b - Enable
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate hardware interrupt requests if STAT[RDRF] is 1. 0b - Disable 1b - Enable
20 ILIE	Idle Line Interrupt Enable Enables hardware interrupts. This field enables STAT[IDLE] to generate interrupt requests. If this field is 0, hardware interrupts from STAT[IDLE] are disabled and polling is used, and if this field is 1, hardware interrupts are enabled when STAT[IDLE] is 1. 0b - Disable 1b - Enable
19 TE	Transmitter Enable Enables the LPUART transmitter.

Table continues on the next page...

Field	Function
	Using this field, you can also queue an idle preamble by first writing 0 and then writing 1 to this field. After this field becomes 0, the field reads 1 until the transmitter has completed the current character and the TX pin is tristated. You can also queue a single idle character by writing to the transmit FIFO with DATA[FRETSC] and DATA[R9T9] = 1. 0b - Disable 1b - Enable
18 RE	Receiver Enable Enables the LPUART receiver. After you write 0 to this field, this field remains 1 until the receiver finishes receiving the current character (if any). 0b - Disable 1b - Enable
17 RWU	Receiver Wake-Up Control Specifies whether the LPUART receiver in standby is waiting for a wake-up condition. You can write 1 to this field to place the LPUART receiver in a Standby state. The field becomes 0 automatically when an RWU event occurs, that is, in case of an idle event when CTRL[WAKE] is 0 or an address match when CTRL[WAKE] is 1 and STAT[RWUID] is 0. **NOTE:** You must write 1 to this field only when CTRL[WAKE] is 0 (wake-up on idle), if the channel is currently not idle. You can determine this by the value of STAT[RAF]. If the field is 1 to wake up an idle event and the channel is already idle, LPUART, possibly, discards the data. This is because the data must be received or a LIN break is detected after an Idle condition is detected before the IDLE flag is allowed to be reasserted. 0b - Normal receiver operation 1b - LPUART receiver in standby, waiting for a wake-up condition
16 SBK	Send Break Specifies whether queue break character(s) are to be sent. Writing 1 and then 0 to this field queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] is 1, and bit times of logic 0 are queued as long as this field is 1. Depending on the timing when this field is 1 and 0, relative to the character currently being transmitted, a second break character may be queued before you write 0 to this field. If the time taken to write 0 to this field is too long, for example, if the field does not become 0 by the end of the first break character, a second break character is sent. This is compared to queuing a break character through the transmit FIFO that guarantees only one break character is sent. You can also queue a single break character by writing to the transmit FIFO when DATA[FRETSC] is 1 and DATA[R9T9] is 0. 0b - Normal transmitter operation 1b - Queue break character(s) to be sent
15 MA1IE	Match 1 (MA1F) Interrupt Enable Enables the MA1F interrupt. 0b - Disable 1b - Enable
14 MA2IE	Match 2 (MA2F) Interrupt Enable Enables the MA2F interrupt. 0b - Disable 1b - Enable
13	Reserved

Table continues on the next page...

Field	Function
—	
12 —	Reserved
11 M7	7-Bit Mode Select Specifies the data characters that the receiver and transmitter use. You must change the value of this field only after both the transmitter and receiver are disabled. 0b - 8-bit to 10-bit 1b - 7-bit
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before you write 1 to STAT[IDLE]. 000b - 1 001b - 2 010b - 4 011b - 8 100b - 16 101b - 32 110b - 64 111b - 128
7 LOOPS	Loop Mode Select Selects Loop mode. After this field becomes 1, the RX pin is disconnected from LPUART and the transmitter output is internally connected to the receiver input. The transmitter and receiver must be enabled to use the loop function. In Loop mode or Single-Wire mode, the transmitter outputs are internally connected to the receiver input (see CTRL[RSRC]). 0b - Normal operation: RX and TX use separate pins 1b - Loop mode or Single-Wire mode
6 DOZEEN	Doze Mode Enables LPUART in Doze mode. If this field is 1, LPUART remains active when not in Doze mode. 0b - Enable 1b - Disable
5 RSRC	Receiver Source Select Determines the source of the receiver shift register input if CTRL[LOOPS] is 1. This field has no effect unless CTRL[LOOPS] is 1. If this field is 0, internal Loopback mode is selected. LPUART does not use the RX pin. If this field is 1, single-wire LPUART mode is selected where the TX pin is connected to the transmitter output and receiver input. 0b - Internal Loopback mode 1b - Single-wire mode
4 M	9-Bit Or 8-Bit Mode Select Specifies the data characters that the receiver and transmitter use. 0b - 8-bit 1b - 9-bit
3 WAKE	Receiver Wake-Up Method Select

Table continues on the next page...

Field	Function
	Determines which condition wakes up LPUART when CTRL[RWU] = 1 and BAUD[MATCFG] = 0 (this field must be 1 when BAUD[MATCFG] = 11): • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character • An idle condition on the receive pin input signal If this field is 0, CTRL[RWU] is configured for idle line wake-up, and if this field is 1, CTRL[RWU] is configured with address mark wake-up. 0b - Idle 1b - Mark
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or the stop bit. If the count begins after the start bit, 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 you write 1 to this field, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - After the start bit 1b - After the stop bit
1 PE	Parity Enable Enables hardware parity generation and checking. If parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - Disable 1b - Enable
0 PT	Parity Type Selects the type of parity, even or odd, if parity is enabled (CTRL[PE] = 1): • Odd parity means that the total number of logic 1 bits in the data character, including the parity bit, is odd. • Even parity means that the total number of 1s in the data character, including the parity bit, is even. 0b - Even parity 1b - Odd parity

41.6.1.9 Data (DATA)

41.6.1.9.1 Offset

Register	Offset
DATA	1Ch

41.6.1.9.2 Function

Supports 8-bit, 16-bit, or 32-bit writes, each type of write performing a separate function. An 8-bit write to DATA[7:0] pushes {CTRL[R8T9], CTRL[R9T8], DATA[7:0]} the transmit FIFO with TSC clear. A 16-bit or 32-bit write pushes the data written into the FIFO and does not update the value of CTRL[R8T9] or CTRL[R9T8].

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status fields.

NOTE

Reads return the contents of the read-only receive FIFO and writes go to the write-only transmit FIFO, making this register work as a set of two separate registers.

41.6.1.9.3 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	NOISY	PARITYE	FRETSC	RXEMPT	IDLINE	0	R9T9	R8T8	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.9.4 Fields

Field	Function
31-16 —	Reserved
15 NOISY	Noisy Data Received Indicates whether the current received dataword contained in DATA[R9:R0] is received with noise. 0b - Received without noise 1b - Received with noise
14 PARITYE	Parity Error Indicates whether the current received dataword contained in DATA[R9:R0] is received with a parity error. 0b - Received without a parity error 1b - Received with a parity error

Table continues on the next page...

Field	Function
13 FRETSC	Frame Error Transmit Special Character Specifies the way the dataword is received. For reads, this field indicates that the current received dataword contained in DATA[R9:R0] is received with a frame error. For writes, the field indicates that a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 indicates a break character when it is 0 and indicates an idle character when it is 1. The contents of DATA[T8:T0] must be 0. 0b - Received without a frame error on reads or transmits a normal character on writes 1b - Received with a frame error on reads or transmits an idle or break character on writes
12 RXEMPT	Receive Buffer Empty Indicates whether the receive buffer contains valid data. This field becomes 1 when there is no data in the receive buffer. The field does not consider data in the receive shift register. 0b - Valid data 1b - Invalid data and empty
11 IDLINE	Idle Line Indicates whether the receiver line was idle before receiving the character in DATA[9:0]. Unlike STAT[IDLE], you can write 1 to this field for the first character received when the receiver is first enabled. 0b - Not idle 1b - Idle
10 —	Reserved
9 R9T9	Read receive FIFO bit 9 or write transmit FIFO bit 9
8 R8T8	Read receive FIFO bit 8 or write transmit FIFO bit 8
7 R7T7	Read receive FIFO bit 7 or write transmit FIFO bit 7
6 R6T6	Read receive FIFO bit 6 or write transmit FIFO bit 6
5 R5T5	Read receive FIFO bit 5 or write transmit FIFO bit 5
4 R4T4	Read receive FIFO bit 4 or write transmit FIFO bit 4
3 R3T3	Read receive FIFO bit 3 or write transmit FIFO bit 3
2 R2T2	Read receive FIFO bit 2 or write transmit FIFO bit 2
1 R1T1	Read receive FIFO bit 1 or write transmit FIFO bit 1
0 R0T0	Read receive FIFO bit 0 or write transmit FIFO bit 0

41.6.1.10 Match Address (MATCH)

41.6.1.10.1 Offset

Register	Offset
MATCH	20h

41.6.1.10.2 Function

Provides addresses for address matching during the receiver operation.

41.6.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						MA2									
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						MA1									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.10.4 Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] only when the associated Baud Rate (BAUD) field is 0.
15-10 —	Reserved
9-0 MA1	Match Address 1

Field	Function
	Is compared to input data addresses when the most significant bit is 1 and the associated Baud Rate (BAUD) field is 1. If a match occurs, the data that follows is transferred to Data (DATA). If a match fails, the data that follows is discarded. You must write to MATCH[MA1] and MATCH[MA2] fields only when the associated field in Baud Rate (BAUD) is 0.

41.6.1.11 MODEM IrDA (MODIR)

41.6.1.11.1 Offset

Register	Offset
MODIR	24h

41.6.1.11.2 Function

Controls options for setting the MODEM configuration.

41.6.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													IREN	TNP	
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						RTSWATER		0		TXCTSSRC	TXCTSC	RXRTSE	TXRTSPOL	TXRTSE	TXCTSE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.11.4 Fields

Field	Function
31-19 —	Reserved
18	IR Enable

Table continues on the next page...

Field	Function
IREN	Enables IR modulation and demodulation. You must change the value of this field only when both the transmitter and receiver are disabled. 0b - Disable 1b - Enable
17-16 TNP	Transmitter Narrow Pulse Specifies whether LPUART transmits a 1 ÷ OSR, 2 ÷ OSR, 3 ÷ OSR, or 4 ÷ OSR narrow pulse when the IR pulse is enabled. You must change the value of this field only when both the transmitter and receiver are disabled. The IR pulse width must be configured to less than half of the OSR. Common pulse widths are 3 ÷ 16, 1 ÷ 16, 1 ÷ 32, or 1 ÷ 4 of the bit length. You can configure these by selecting the appropriate OSR and pulse width. 00b - 1 ÷ OSR 01b - 2 ÷ OSR 10b - 3 ÷ OSR 11b - 4 ÷ OSR
15-10 —	Reserved
9-8 RTSWATER	Receive RTS Configuration Configures the assertion and negation of the receiver's RTS output. The receiver's RTS output negates when the number of empty words in the receive FIFO is greater or equal to the value of this field. If this field is 0, the RTS pin negates when the receive FIFO is full. For the purpose of receive RTS generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between RTS negation and the external transmitter ceasing transmission. If both receive RTS and address or data matching is enabled, RTS could assert at the end of a character if there exists no match. You must change the value of this field only when the receiver is disabled.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - The CTS pin 1b - An internal connection to the receiver address match result
4 TXCTSC	Transmit CTS Configuration Configures whether the CTS state or input is checked or sampled at the start of each character or only when the transmitter is idle. 0b - Sampled at the start of each character 1b - Sampled when the transmitter is idle
3 RXRTSE	Receiver RTS Enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. You must change the value of this field only when the receiver is disabled. If this field is 0, the receiver has no effect on RTS. If this field is 1, RTS is deasserted if STAT[RDRF] is 1 or a start bit is detected that causes STAT[RDRF] to become 1. RTS is asserted if STAT[RDRF] is 0 and has not detected a start bit that causes STAT[RDRF] to become 1. **NOTE:** Do not write 1 to both MODIR[RXRTSE] and MODIR[TXRTSE]. 0b - Disable

Table continues on the next page...

Field	Function
	1b - Enable
2 TXRTSPOL	Transmitter RTS Polarity Controls the polarity of the transmitter RTS. This field does not affect the polarity of the receiver RTS that remains negated in the active-low state unless MODIR[TXRTSE] is 1. You must change the value of this field only when the transmitter is disabled. 0b - Active low 1b - Active high
1 TXRTSE	Transmitter RTS Enable Controls the operation of RTS before and after a transmission. You must change the value of this field only when the transmitter is disabled. If this field is 0, the transmitter has no effect on RTS, and if this field is 1, a character is placed into an empty transmit shift register. RTS asserts 1-bit time before the start bit is transmitted and deasserts 1-bit time after all characters in the transmitter FIFO and shift register are completely sent, including the last stop bit. 0b - Disable 1b - Enable
0 TXCTSE	Transmitter CTS Enable Enables the operation of the transmitter. You can write 1 to this field irrespective of the states of MODIR[TXRTSE] and MODIR[RXRTSE]. If this field is 1, the transmitter checks the state of the CTS signal each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the TXD signal remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS, when a character is being sent, do not affect its transmission. 0b - Disable 1b - Enable

41.6.1.12 FIFO (FIFO)

41.6.1.12.1 Offset

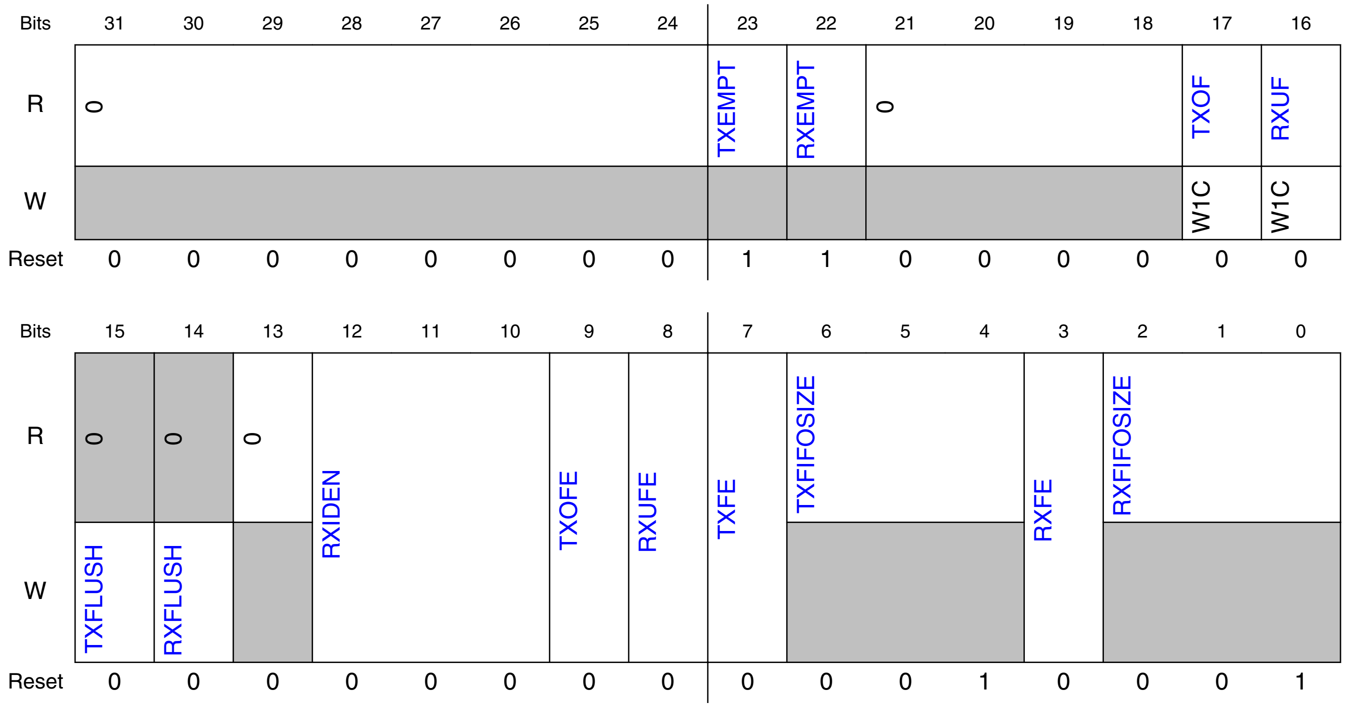
Register	Offset
FIFO	28h

41.6.1.12.2 Function

Provides you the ability to turn on and turn off the FIFO functionality.

This register also provides you the size of the FIFO that has been implemented. You can read this register at any time and must write to it only when CTRL[RE] and CTRL[TE] are 0 and the FIFO is empty.

41.6.1.12.3 Diagram



Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXEMPT	RXEMPT	0				TXOF	RXUF
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	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	0	0	RXIDEN			TXOFE	RXUFE	TXFE	TXFIFOSIZE			RXFE	RXFIFOSIZE		
W	TXFLUSH	RXFLUSH														
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

41.6.1.12.4 Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit FIFO Or Buffer Empty Indicates whether the transmit buffer is empty. This field becomes 1 when there is no data in the transmit FIFO or buffer. The field does not consider data in the transmit shift register. 0b - Not empty 1b - Empty
22 RXEMPT	Receive FIFO Or Buffer Empty Indicates whether the receive buffer is empty. This field becomes 1 when there is no data in the receive FIFO or buffer. The field does not consider data in the receive shift register. 0b - Not empty 1b - Empty
21-18 —	Reserved
17	Transmitter FIFO Overflow Flag

Table continues on the next page...

Field	Function
TXOF	Indicates whether more data has been written to the transmit FIFO than it can hold. If this field is 0, no transmit FIFO overflow has occurred since the last time the field was cleared, and if this field is 1, at least one transmit FIFO overflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[TXOFE]. However, an interrupt is issued to the host only if FIFO[TXOFE] is 1. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No overflow 1b - Overflow When writing 0b - No effect 1b - Clear the flag
16 RXUF	Receiver FIFO Underflow Flag Indicates whether more data has been read from the receive FIFO than was present. If this field is 0, no receive FIFO underflow has occurred since the last time the field was cleared, and if this field is 1, at least one receive FIFO underflow has occurred since the last time the field was cleared. This field becomes 1 regardless of the value of FIFO[RXUFE]. However, an interrupt is issued to the host only if FIFO[RXUFE] is 1. **NOTE:** This field behaves differently for register reads and writes. When reading 0b - No underflow 1b - Underflow When writing 0b - No effect 1b - Clear the flag
15 TXFLUSH	Transmit FIFO Flush Causes all data that is stored in the transmit FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the transmit FIFO or buffer clears out. This does not affect data in the transmit shift register. 0b - No effect 1b - All data flushed out
14 RXFLUSH	Receive FIFO Flush Causes all data that is stored in the receive FIFO to be flushed. If you write 0 to this field, no flush operation occurs, and if you write 1 to this field, all data in the receive FIFO or buffer clears out. This does not affect data in the receive shift register. 0b - No effect 1b - All data flushed out
13 —	Reserved
12-10	Receiver Idle Empty Enable

Table continues on the next page...

Field	Function
RXIDEN	Enables STAT[RDRF] to become 1 when the receiver is idle for a number of idle characters and the FIFO is not empty. 000b - Disable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle 001b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for one character 010b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for two characters 011b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for four characters 100b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for eight characters 101b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 16 characters 110b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 32 characters 111b - Enable STAT[RDRF] to become 1 because of partially filled FIFO when the receiver is idle for 64 characters
9 TXOFE	Transmit FIFO Overflow Interrupt Enable Enables FIFO[TXOF] to generate an interrupt to the host. 0b - Disable 1b - Enable
8 RXUFE	Receive FIFO Underflow Interrupt Enable Enables FIFO[RXUF] to generate an interrupt to the host. 0b - Disable 1b - Enable
7 TXFE	Transmit FIFO Enable Enables the transmit FIFO. If this field is 0, the transmit buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[TXFIFOSIZE]. Both CTRL[TE] and CTRL[RE] must be 0 before you change the value of this field. If this field is 1, the built-in FIFO structure for the transmit buffer is enabled. FIFO[TXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth Indicates the maximum number of transmit datawords (transmit FIFO buffer depth) that can be stored in the transmit buffer. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256
3 RXFE	Receive FIFO Enable Enables the receive FIFO. If this field is 0, the receive buffer operates as a FIFO of depth equal to 1 dataword, regardless of the value in FIFO[RXFIFOSIZE]. Both CTRL[RE] and CTRL[TE] must be 0 before you change the value of this field.

Table continues on the next page...

Field	Function
	If this field is 1, the built-in FIFO structure for the receive buffer is enabled. FIFO[RXFIFOSIZE] indicates the size of the FIFO structure. 0b - Disable 1b - Enable
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth Indicates the maximum number of receive datawords (receive FIFO buffer depth) that can be stored in the receive buffer before an overrun occurs. 000b - 1 001b - 4 010b - 8 011b - 16 100b - 32 101b - 64 110b - 128 111b - 256

41.6.1.13 Watermark (WATER)

41.6.1.13.1 Offset

Register	Offset
WATER	2Ch

41.6.1.13.2 Function

Provides the ability to set a programmable threshold for notification, or sets the programmable thresholds to indicate that transmit data can be written or receive data can be read.

You may read this register at any time but must write to it only when CTRL[TE] is 0.

41.6.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					RXCOUNT			0						RXWATER	
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					TXCOUNT			0						TXWATER	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

41.6.1.13.4 Fields

Field	Function
31-27 —	Reserved
26-24 RXCOUNT	Receive Counter Indicates the number of datawords in the receive FIFO or buffer. If a dataword is being received in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate the room left in the receive FIFO or buffer.
23-18 —	Reserved
17-16 RXWATER	Receive Watermark Generates an interrupt or a DMA request if the number of datawords in the receive FIFO or buffer is greater than the value of this field. For proper operation, the value of this field must be less than the size of the receive FIFO or buffer, as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE].
15-11 —	Reserved
10-8 TXCOUNT	Transmit Counter Indicates the number of datawords in the transmit FIFO or buffer. If a dataword is being transmitted to the transmit shift register, it is not included in the count. This value may be used in conjunction with the value of FIFO[TXFIFOSIZE] to calculate the room left in the transmit FIFO or buffer.
7-2 —	Reserved

Table continues on the next page...

Field	Function
1-0 TXWATER	Transmit Watermark Generates an interrupt or a DMA request when the number of datawords in the transmit FIFO or buffer is equal to or less than the value of this field. For proper operation, the value of this field must be less than the size of the transmit buffer or FIFO, as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

Chapter 42
Serial Communications Interface (SCI) / Universal Asynchronous Receiver/Transmitter (UART)

42.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) allows asynchronous serial communication with peripheral devices and CPUs.

42.1.1 Block diagram

The following figure shows the transmitter portion of this module.

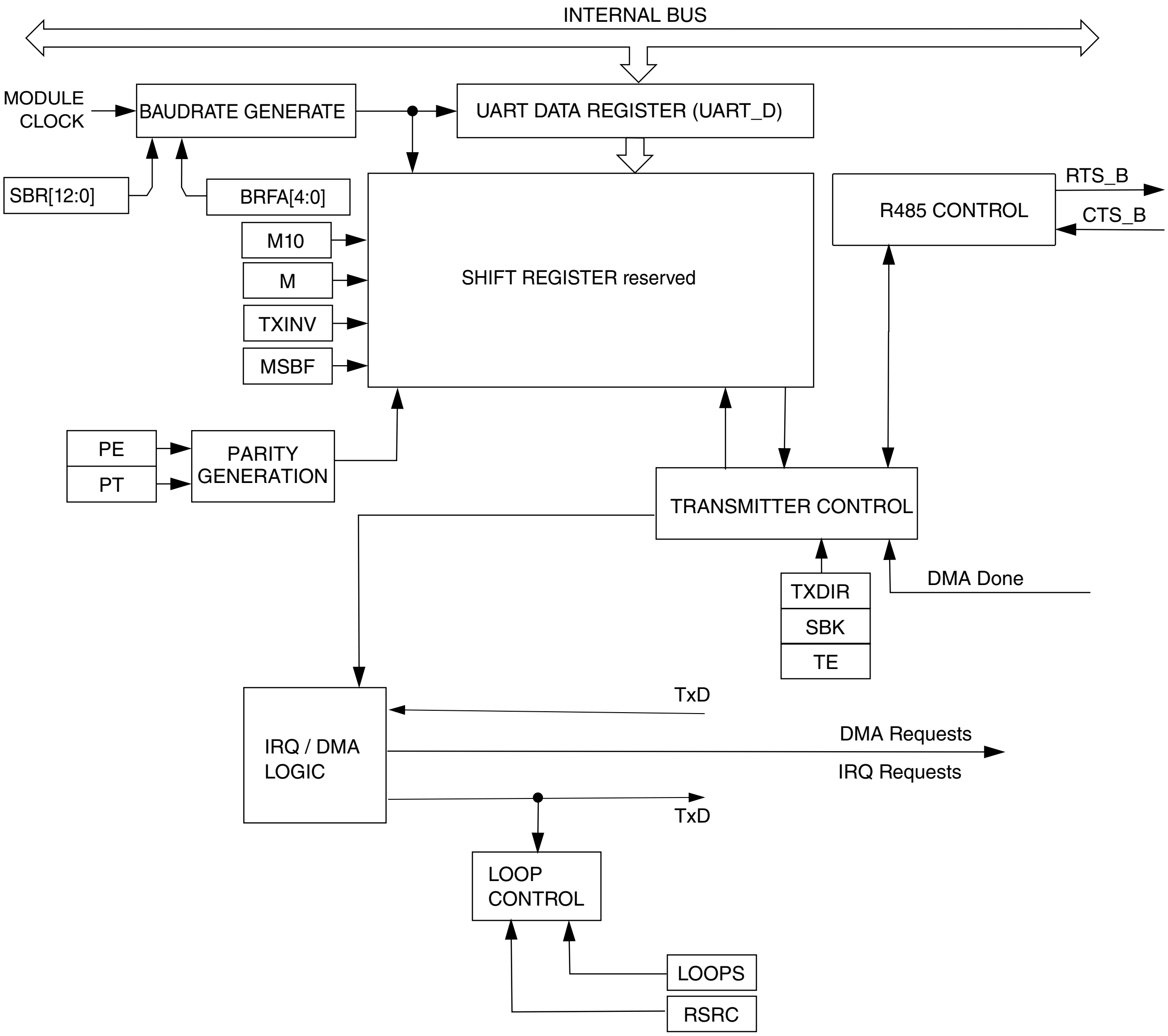


Figure 42-1. UART transmitter block diagram

The following figure shows the receiver portion of this module.

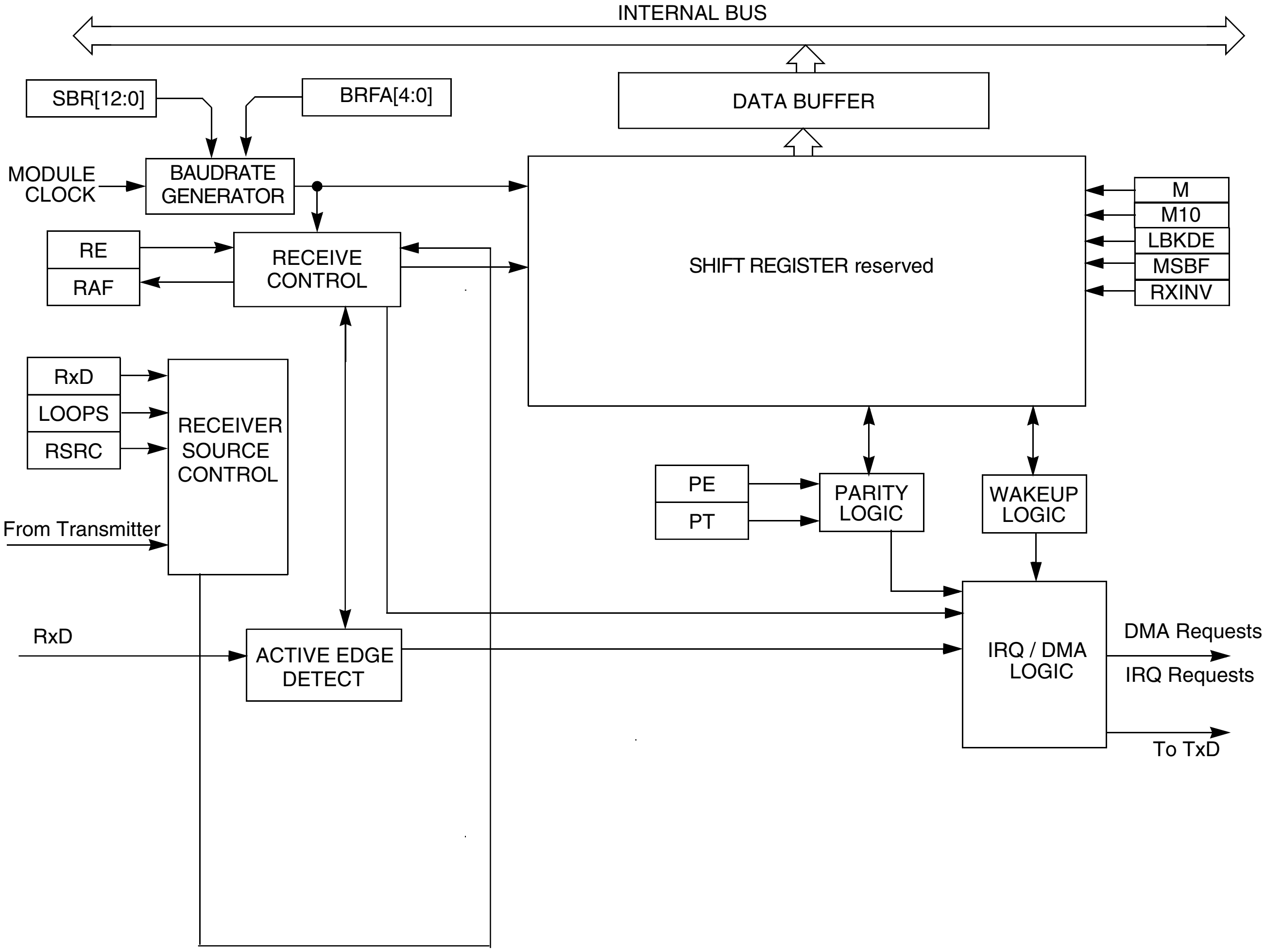


Figure 42-2. UART receiver block diagram

42.1.2 Features

The UART includes the following features:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 13-bit baud rate selection with /32 fractional divide, based on the module clock frequency
- Programmable 8-bit or 9-bit data format
- Programmable 1 or 2 stop bits in a data frame.
- Separately enabled transmitter and receiver

- Programmable transmitter output polarity
- Programmable receive input polarity
- Up to 16-bit break character transmission.
- 11-bit break character detection option
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Address match feature in the receiver to reduce address mark wakeup ISR overhead
- Ability to select MSB or LSB to be first bit on wire
- Interrupt-driven operation with flags
  - Transmission complete
  - Idle receiver input
  - Receiver data buffer overrun
  - Noise error
  - Framing error
  - Parity error
  - Active edge on receive pin
  - LIN break detect
- Receiver framing error detection
- Hardware parity generation and checking
- 1/16 bit-time noise detection
- DMA interface

42.2 Functional description

This section provides a complete functional description of the UART block.

The UART allows full duplex, asynchronous, NRZ serial communication between the CPU and remote devices, including other CPUs. The UART transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the UART, writes the data to be transmitted, and processes received data.

42.2.1 Submodule sections

42.2.1.1 Transmitter

42.2.1.1.1 Transmitter character length

The UART transmitter can accommodate either 8, 9, or 10-bit data characters. The state of the C1[M] and C1[PE] bits and the C4[M10] bit determine the length of data characters. When transmitting 9-bit data, bit C3[T8] is the ninth bit (bit 8).

42.2.1.1.2 Transmission bit order

When S2[MSBF] is set, the UART automatically transmits the MSB of the data word as the first bit after the start bit. Similarly, the LSB of the data word is transmitted immediately preceding the parity bit, or the stop bit if parity is not enabled. All necessary bit ordering is handled automatically by the module. Therefore, the format of the data written to D for transmission is completely independent of the S2[MSBF] setting.

42.2.1.1.3 Character transmission

To transmit data, the MCU writes the data bits to the UART transmit buffer using UART data registers C3[T8] and D. Data in the transmit buffer is then transferred to the transmitter shift register as needed. The transmit shift register then shifts a frame out through the transmit data output signal after it has prefaced it with any required start and stop bits. The UART data registers, C3[T8] and D, provide access to the transmit buffer structure.

The UART also sets a flag, the transmit data register empty flag S1[TDRE], and generates an interrupt or DMA request (C5[TDMAS]) when transmit buffer is empty.. The transmit driver routine may respond to this flag by writing additional datawords to the transmit buffer using C3[T8]/D as space permits.

See Application information for specific programing sequences.

Setting C2[TE] automatically loads the transmit shift register with the following preamble:

Table 42-1. Transmit preamble length

BDH[SBNS]	C1[M]	C4[M10]	C1[PE]	Bits transmitted
0	0	—	—	10
1	0	—	—	11
0	1	0	—	11
1	1	0	—	12
0	1	1	1	12
1	1	1	1	13

After the preamble shifts out, control logic transfers the data from the D register into the transmit shift register. The transmitter automatically transmits the correct start bit and stop bit before and after the dataword. The number of stop bits transmitted after the dataword can be programmed using BDH[SBNS] field.

Hardware supports odd or even parity. When parity is enabled, the bit immediately preceding the stop bit is the parity bit.

When the transmit shift register is not transmitting a frame, the transmit data output signal goes to the idle condition, logic 1. If at any time software clears C2[TE], the transmitter enable signal goes low and the transmit signal goes idle.

If the software clears C2[TE] while a transmission is in progress, the character in the transmit shift register continues to shift out, provided S1[TC] was cleared during the data write sequence. To clear S1[TC], the S1 register must be read followed by a write to D register.

If S1[TC] is cleared during character transmission and C2[TE] is cleared, the transmission enable signal is deasserted at the completion of the current frame. Following this, the transmit data out signal enters the idle state even if there is data pending in the UART transmit data buffer.

42.2.1.1.4 Transmitting break characters

Setting C2[SBK] loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on C1[M], C1[PE], S2[BRK13], BDH[SBNS] and C4[M10]. See the following table.

Table 42-2. Transmit break character length

S2[BRK13]	BDH[SBNS]	C1[M]	C4[M10]	C1[PE]	Bits transmitted
0	0	0	—	—	10
0	1	0	—	—	11
0	0	1	0	—	11
0	1	1	0	—	12
0	0	1	1	1	12
0	1	1	1	1	13
1	0	0	—	—	13
1	0	1	—	—	14
1	1	0	—	—	15
1	1	1	—	—	16

As long as C2[SBK] is set, the transmitter logic continuously loads break characters into the transmit shift register. After the software clears C2[SBK], the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character.

NOTE

When queuing a break character, it will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that, if data is queued in the data buffer to be transmitted, the break character preempts that queued data. The queued data is then transmitted after the break character is complete.

42.2.1.1.5 Idle characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on C1[M], C1[PE], BDH[SBNS] and C4[M10]. The preamble is a synchronizing idle character that begins the first transmission initiated after setting C2[TE].

If C2[TE] is cleared during a transmission, the transmit data output signal becomes idle after completion of the transmission in progress. Clearing and then setting C2[TE] during a transmission queues an idle character to be sent after the dataword currently being transmitted.

Note

When queuing an idle character, the idle character will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that if data is queued in the data buffer to be transmitted, the idle character preempts that queued data. The queued data is then transmitted after the idle character is complete.

If C2[TE] is cleared and the transmission is completed, the UART is not the master of the TXD pin.

42.2.1.2 Receiver

42.2.1.2.1 Receiver character length

The UART receiver can accommodate 8-, 9-, or 10-bit data characters. The states of C1[M], C1[PE], BDH[SBNS] and C4[M10] determine the length of data characters. When receiving 9 or 10-bit data, C3[R8] is the ninth bit (bit 8).

42.2.1.2.2 Receiver bit ordering

When S2[MSBF] is set, the receiver operates such that the first bit received after the start bit is the MSB of the dataword. Similarly, the bit received immediately preceding the parity bit, or the stop bit if parity is not enabled, is treated as the LSB for the dataword. All necessary bit ordering is handled automatically by the module. Therefore, the format of the data read from receive data buffer is completely independent of S2[MSBF].

42.2.1.2.3 Character reception

During UART reception, the receive shift register shifts a frame in from the unsynchronized receiver input signal. After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the UART receive buffer. The receive data buffer is accessible via the D and C3[T8] registers. S1[RDRF] is set if the receive buffer is full. If the C2[RIE] is also set, RDRF generates an RDRF interrupt request. Alternatively, by programming C5[RDMAS], a DMA request can be generated.

42.2.1.2.4 Data sampling

The receiver samples the unsynchronized receiver input signal at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see the following figure) is re-synchronized:

- After every start bit.
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0).

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

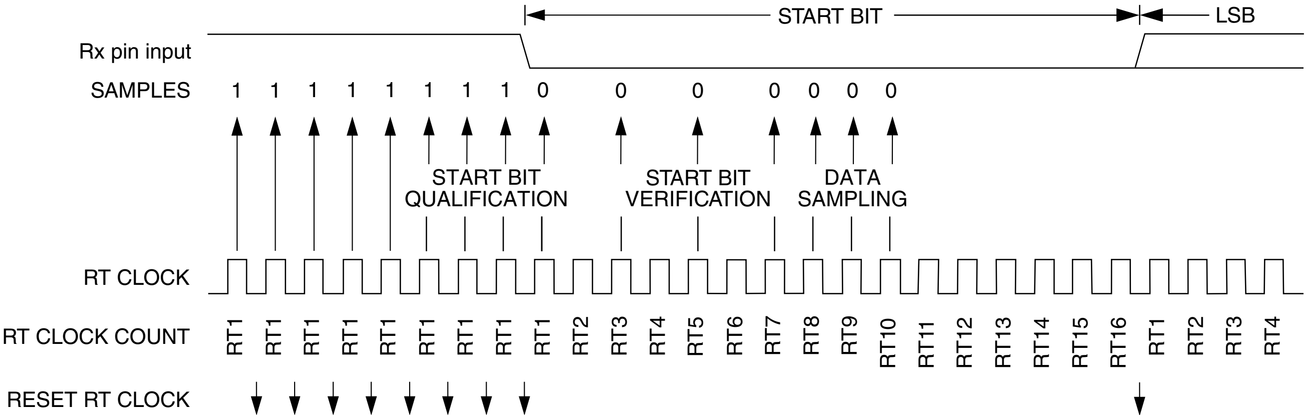


Figure 42-3. Receiver data sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7 . The following table summarizes the results of the start bit verification samples.

Table 42-3. Start bit verification

RT3, RT5, and RT7 samples	Start bit verification	Noise flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. The following table summarizes the results of the data bit samples.

Table 42-4. Data bit recovery

RT8, RT9, and RT10 samples	Data bit determination	Noise flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

Note

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (S1[NF]) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. The following table summarizes the results of the stop bit samples.

Table 42-5. Stop bit recovery

RT8, RT9, and RT10 samples	Framing error flag	Noise flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In the following figure, the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

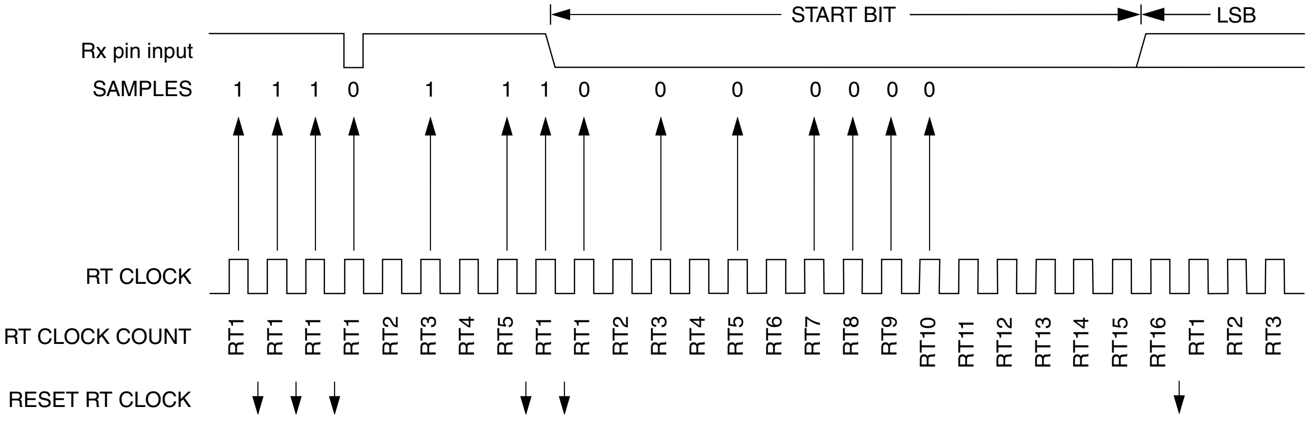

START BIT
LSB
Rx pin input
SAMPLES
1 1 1 0 1 1 1 0 0 0 0 0 0 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3
RESET RT CLOCK

Figure 42-4. Start bit search example 1

In the following figure, verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

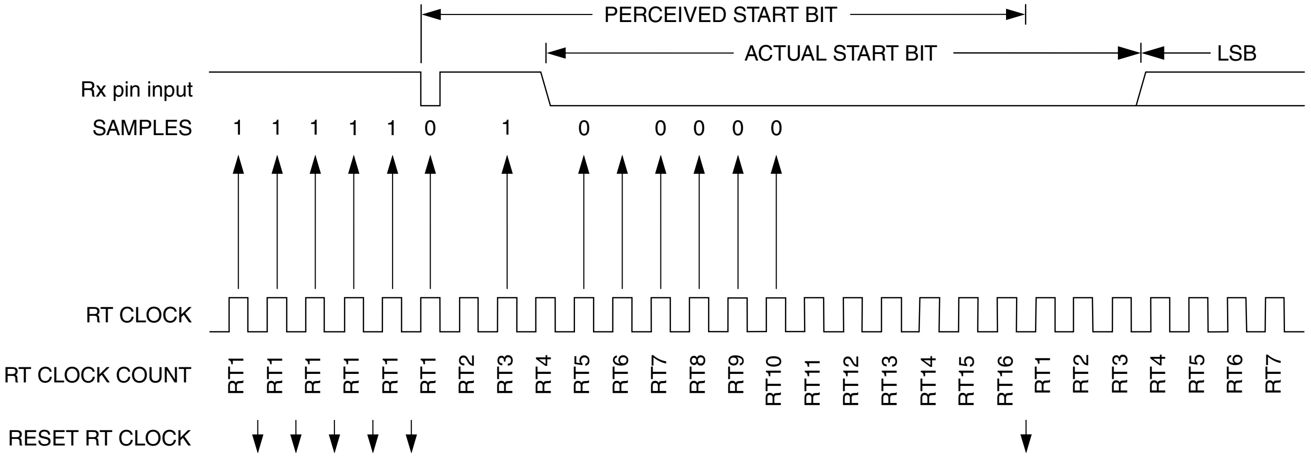

PERCEIVED START BIT
ACTUAL START BIT
LSB
Rx pin input
SAMPLES
1 1 1 1 1 0 1 0 0 0 0 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3 RT4 RT5 RT6 RT7
RESET RT CLOCK

Figure 42-5. Start bit search example 2

In the following figure, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

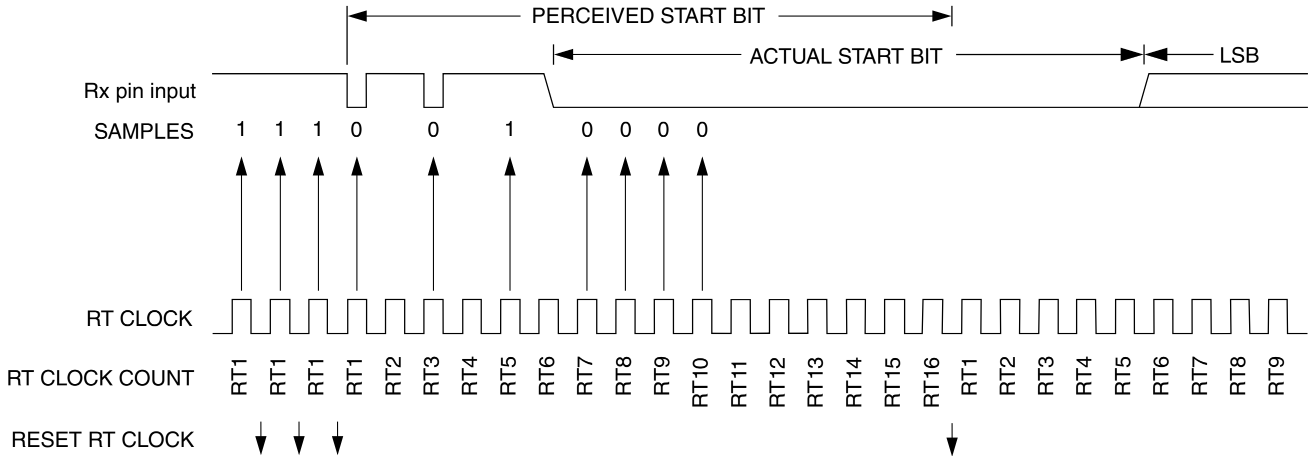

PERCEIVED START BIT
ACTUAL START BIT
LSB
Rx pin input
SAMPLES
1 1 1 0 0 1 0 0 0 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9
RESET RT CLOCK

Figure 42-6. Start bit search example 3

The following figure shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

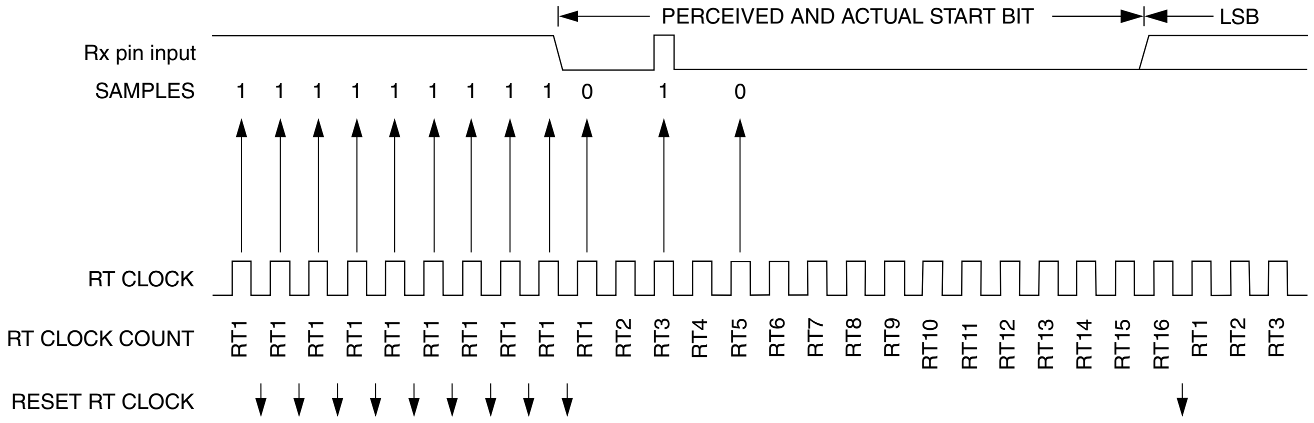

PERCEIVED AND ACTUAL START BIT
LSB
Rx pin input
SAMPLES
1 1 1 1 1 1 1 1 1 0 1 0
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3
RESET RT CLOCK

Figure 42-7. Start bit search example 4

The following figure shows a burst of noise near the beginning of the start bit that resets the RT clock. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

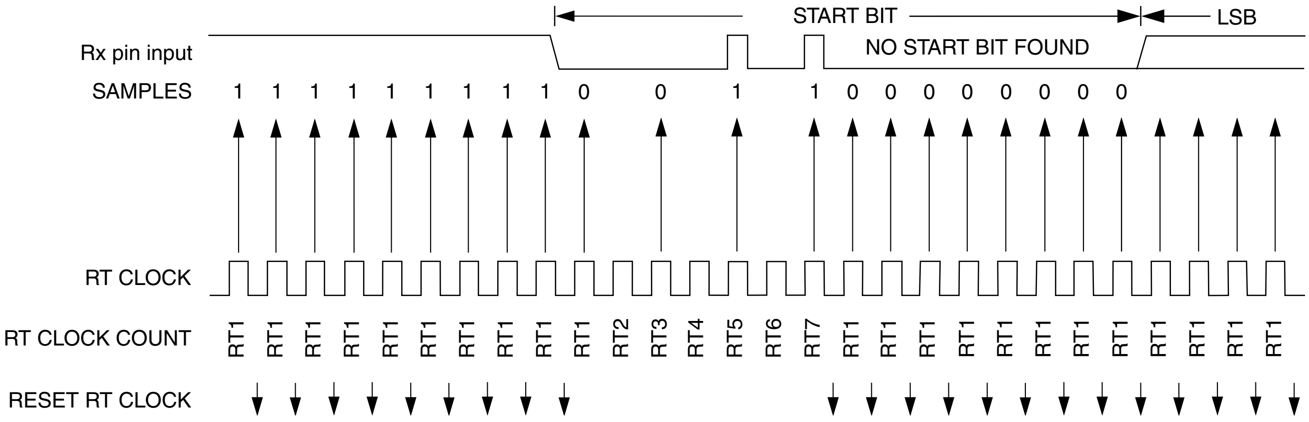


Figure 42-8. Start bit search example 5

In the following figure, a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

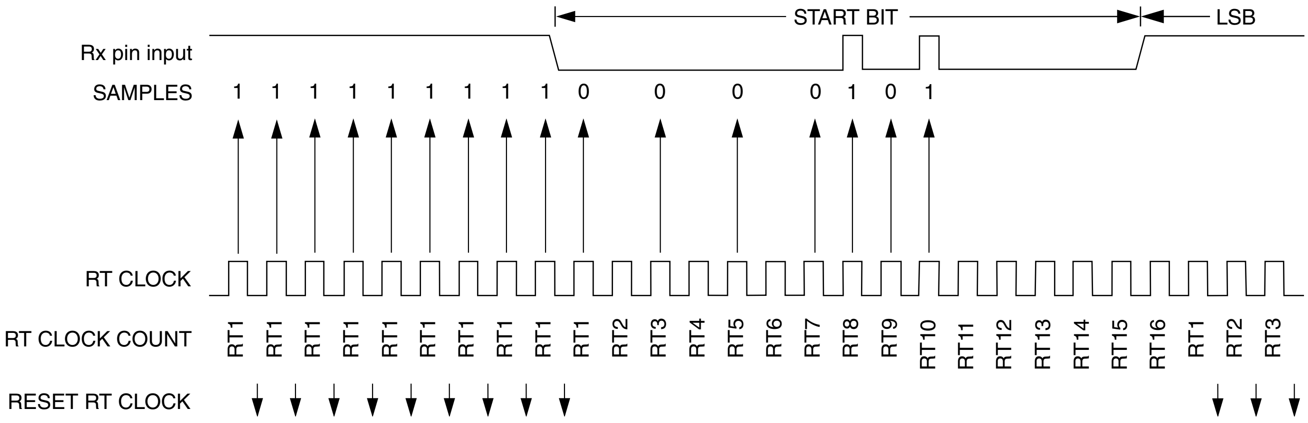

START BIT
LSB
Rx pin input
SAMPLES
1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 1
RT CLOCK
RT CLOCK COUNT
RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 RT1 RT2 RT3
RESET RT CLOCK

Figure 42-9. Start bit search example 6

42.2.1.2.5 Framing errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, S1[FE], if S2[LBKDE] is disabled. When S2[LBKDE] is disabled, a break character also sets the S1[FE] because a break character has no stop bit. S1[FE] is set at the same time that received data is placed in the receive data buffer.

42.2.1.2.6 Receiving break characters

The UART recognizes a break character when a start bit is followed by eight, nine, or ten logic 0 data bits and a logic 0 where the stop bit should be. Receiving a break character has these effects on UART registers:

- Sets the framing error flag, S1[FE].
- Writes an all 0 dataword to the data buffer, which may cause S1[RDRF] to set, depending on the watermark and number of values in the data buffer.
- May set the overrun flag, S1[OR], noise flag, S1[NF], parity error flag, S1[PF], or the receiver active flag, S2[RAF].

The detection threshold for a break character can be adjusted when using an internal oscillator in a LIN system by setting S2[LBKDE]. The UART break character detection threshold depends on C1[M], C1[PE], S2[LBKDE] and C4[M10]. See the following table.

Table 42-6. Receive break character detection threshold

LBKDE	SBNS	M	M10	PE	Threshold (bits)
0	0	0	—	—	10
0	1	0	—	—	11
0	0	1	0	—	11
0	1	1	0	—	12
0	0	1	1	1	12
0	1	1	1	1	13
1	—	0	—	—	11
1	—	1	—	—	12

While S2[LBKDE] is set, it will have these effects on the UART registers:

- Prevents S1[RDRF], S1[FE], S1[NF], and S1[PF] from being set. However, if they are already set, they will remain set.
- Sets the LIN break detect interrupt flag, S2[LBKDIF], if a LIN break character is received.

42.2.1.2.7 Baud rate tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic 0.

As the receiver samples an incoming frame, it resynchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames corrects a misalignment between transmitter bit times and receiver bit times.

42.2.1.2.7.1 Slow data tolerance

The following figure shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

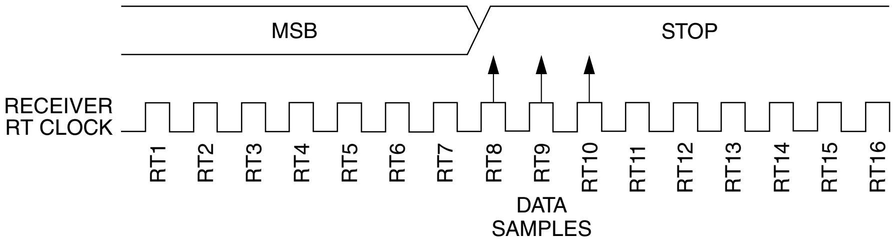


Figure 42-10. Slow data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 42-10, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 147 RT cycles (9 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$((154 - 147) \div 154) \times 100 = 4.54\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 42-10, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 163 RT cycles (10 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$((170 - 163) \div 170) \times 100 = 4.12\%$$

42.2.1.2.7.2 Fast data tolerance

The following figure shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

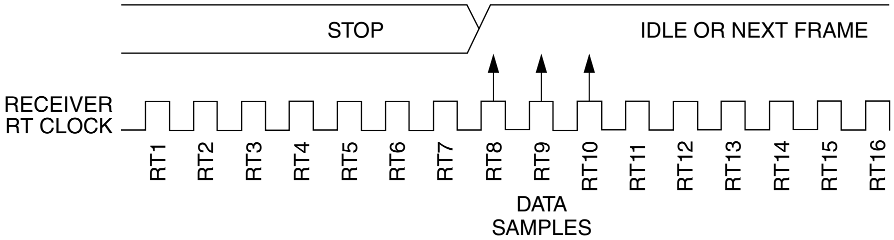


Figure 42-11. Fast data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 42-11, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 160 RT cycles (10 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$((154 - 160) \div 154) \times 100 = 3.90\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 42-11, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 176 RT cycles (11 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$((170 - 176) \div 170) \times 100 = 3.53\%$$

42.2.1.2.8 Receiver wakeup

C1[WAKE] determines how the UART is brought out of the standby state to process an incoming message. C1[WAKE] enables either idle line wakeup or address mark wakeup.

42.2.1.2.8.1 Idle input line wakeup (C1[WAKE] = 0)

In this wakeup method, an idle condition on the unsynchronized receiver input signal clears C2[RWU] and wakes the UART. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] and return to the standby state. C2[RWU] remains set and the receiver remains in standby until another idle character appears on the unsynchronized receiver input signal.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

When C2[RWU] is 1 and S2[RWUID] is 0, the idle character that wakes the receiver does not set S1[IDLE] or the receive data register full flag, S1[RDRF]. The receiver wakes and waits for the first data character of the next message which is stored in the receive data buffer. When S2[RWUID] and C2[RWU] are set and C1[WAKE] is cleared, any idle condition sets S1[IDLE] and generates an interrupt if enabled.

42.2.1.2.8.2 Address mark wakeup (C1[WAKE] = 1)

In this wakeup method, a logic 1 in the bit position immediately preceding the stop bit of a frame clears C2[RWU] and wakes the UART. A logic 1 in the bit position immediately preceeding the stop bit marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] and return to the standby state. C2[RWU] remains set and the receiver remains in standby until another address frame appears on the unsynchronized receiver input signal.

A logic 1 in the bit position immediately preceding the stop bit clears the receiver's C2[RWU] after the stop bit is received and places the received data into the receiver data buffer. Note that if Match Address operation is enabled i.e. C4[MAEN1] or C4[MAEN2] is set, then received frame is transferred to receive buffer only if the comparison matches.

Address mark wakeup allows messages to contain idle characters but requires that the bit position immediately preceding the stop bit be reserved for use in address frames.

If module is in standby mode and nothing triggers to wake the UART, no error flag is set even if an invalid error condition is detected on the receiving data line.

42.2.1.2.8.3 Match address operation

Match address operation is enabled when C4[MAEN1] or C4[MAEN2] is set. In this function, a frame received by the RX pin with a logic 1 in the bit position of the address mark is considered an address and is compared with the associated MA1 or MA2 register. The frame is transferred to the receive buffer, and S1[RDRF] is set, only if the comparison matches. All subsequent frames received with a logic 0 in the bit position of the address mark 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 frames with logic 0 in the bit position of the address mark are also discarded. If both C4[MAEN1] and C4[MAEN2] are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Match address operation functions in the same way for both MA1 and MA2 registers. Note that the position of the address mark is the same as the Parity Bit when parity is enabled for 8 bit and 9 bit data formats.

- If only one of C4[MAEN1] and C4[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 C4[MAEN1] and C4[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

42.2.1.3 Baud rate generation

A 13-bit modulus counter and a 5-bit fractional fine-adjust 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 module clock divisor. The SBR bits are in the UART baud rate registers, BDH and BDL. The baud rate clock is synchronized with the module clock and drives the receiver. The fractional fine-adjust counter adds fractional delays to the baud rate clock to allow fine trimming of the baud rate to match the system baud rate. The transmitter is driven by the baud rate clock divided by 16. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to two sources of error:

- Integer division of the module clock may not give the exact target frequency. This error can be reduced with the fine-adjust counter.
- Synchronization with the module clock can cause phase shift.

The Table 42-7 lists the available baud divisor fine adjust values.

UART baud rate = UART module clock / (16 × (SBR[12:0] + BRFD))

The following table lists some examples of achieving target baud rates with a module clock frequency of 10.2 MHz, with and without fractional fine adjustment.

Table 42-7. Baud rates (example: module clock = 10.2 MHz)

Bits SBR (decimal)	Bits BRFA	BRFD value	Receiver clock (Hz)	Transmitter clock (Hz)	Target Baud rate	Error (%)
17	00000	0	600,000.0	37,500.0	38,400	2.3
16	10011	19/32=0.59375	614,689.3	38,418.08	38,400	0.047
33	00000	0	309,090.9	19,318.2	19,200	0.62
33	00110	6/32=0.1875	307,344.6	19,209.04	19,200	0.047
66	00000	0	154,545.5	9659.1	9600	0.62
133	00000	0	76,691.7	4793.2	4800	0.14
266	00000	0	38,345.9	2396.6	2400	0.14
531	00000	0	19,209.0	1200.6	1200	0.11
1062	00000	0	9604.5	600.3	600	0.05
2125	00000	0	4800.0	300.0	300	0.00
4250	00000	0	2400.0	150.0	150	0.00
5795	00000	0	1760.1	110.0	110	0.00

Table 42-8. Baud rate fine adjust

BRFA	Baud Rate Fractional Divisor (BRFD)
0 0 0 0 0	0/32 = 0
0 0 0 0 1	1/32 = 0.03125
0 0 0 1 0	2/32 = 0.0625
0 0 0 1 1	3/32 = 0.09375
0 0 1 0 0	4/32 = 0.125
0 0 1 0 1	5/32 = 0.15625
0 0 1 1 0	6/32 = 0.1875
0 0 1 1 1	7/32 = 0.21875
0 1 0 0 0	8/32 = 0.25
0 1 0 0 1	9/32 = 0.28125
0 1 0 1 0	10/32 = 0.3125
0 1 0 1 1	11/32 = 0.34375
0 1 1 0 0	12/32 = 0.375
0 1 1 0 1	13/32 = 0.40625
0 1 1 1 0	14/32 = 0.4375
0 1 1 1 1	15/32 = 0.46875
1 0 0 0 0	16/32 = 0.5
1 0 0 0 1	17/32 = 0.53125
1 0 0 1 0	18/32 = 0.5625
1 0 0 1 1	19/32 = 0.59375
1 0 1 0 0	20/32 = 0.625

Table continues on the next page...

Table 42-8. Baud rate fine adjust (continued)

BRFA	Baud Rate Fractional Divisor (BRFD)
1 0 1 0 1	21/32 = 0.65625
1 0 1 1 0	22/32 = 0.6875
1 0 1 1 1	23/32 = 0.71875
1 1 0 0 0	24/32 = 0.75
1 1 0 0 1	25/32 = 0.78125
1 1 0 1 0	26/32 = 0.8125
1 1 0 1 1	27/32 = 0.84375
1 1 1 0 0	28/32 = 0.875
1 1 1 0 1	29/32 = 0.90625
1 1 1 1 0	30/32 = 0.9375
1 1 1 1 1	31/32 = 0.96875

42.2.1.4 Data format

Each data character is contained in a frame that includes a start bit and a stop bit. The rest of the data format depends upon C1[M], C1[PE], S2[MSBF], BDH[SBNS] and C4[M10].

42.2.1.4.1 Eight-bit configuration

Clearing C1[M] configures the UART for 8-bit data characters, that is, eight bits are memory mapped in D. A frame with eight data bits has a total of 10 bits (This becomes 11 bits if BDH[SBNS] = 1). The most significant bit of the eight data bits can be used as an address mark to wake the receiver. If the most significant bit is used in this way, then it serves as an address or data indication, leaving the remaining seven bits as actual data. When C1[PE] is set, the eighth data bit is automatically calculated as the parity bit. See the following table.

Table 42-9. Configuration of 8-bit data format

UART_C1[PE]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	1	8	0	0	1
0	1	7	1[1]	0	1
1	1	7	0	1	1

[1] The address bit identifies the frame as an address character. See Receiver wakeup.

NOTE

In the last column of the above table, the number of stop bits become 2 when BDH[SBNS] is set.

42.2.1.4.2 Nine-bit configuration

When C1[M] is set and C4[M10] is cleared and BDH[SBNS] is cleared, the UART is configured for 9-bit data characters. If C1[PE] is enabled, the ninth bit is either C3[T8/R8] or the internally generated parity bit. This results in a frame consisting of a total of 11 bits. In the event that the ninth data bit is selected to be C3[T8], it will remain unchanged after transmission and can be used repeatedly without rewriting it, unless the value needs to be changed. This feature may be useful when the ninth data bit is being used as an address mark.

When C1[M] and C4[M10] are set and BDH[SBNS] is cleared, the UART is configured for 9-bit data characters, but the frame consists of a total of 12 bits. The 12 bits include the start and stop bits, the 9 data character bits, and a tenth internal data bit. Note that if C4[M10] is set, C1[PE] must also be set. In this case, the tenth bit is the internally generated parity bit. The ninth bit can either be used as an address mark or a ninth data bit.

See the following table.

Table 42-10. Configuration of 9-bit data formats

C1[PE]	UC1[M]	C1[M10]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	0	0	See Eight-bit configuration				
0	0	1	Invalid configuration				
0	1	0	1	9	0	0	1
0	1	0	1	8	1[1]	0	1
0	1	1	Invalid Configuration				
1	0	0	See Eight-bit configuration				
1	0	1	Invalid Configuration				
1	1	0	1	8	0	1	1
1	1	1	1	9	0	1	1
1	1	1	1	8	1[1]	1	1

[1] The address bit identifies the frame as an address character.

NOTE

In the last column of the above table, the number of stop bits become 2 when BDH[SBNS] is set.

Note

Unless in 9-bit mode with M10 set, do not use address mark wakeup with parity enabled.

42.2.1.4.3 Timing examples

Timing examples of these configurations in the NRZ mark/space data format are illustrated in the following figures. The timing examples show all of the configurations in the following sub-sections along with the LSB and MSB first variations. This section explains the data formats available assuming single stop bit mode is selected.

42.2.1.4.3.1 Eight-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

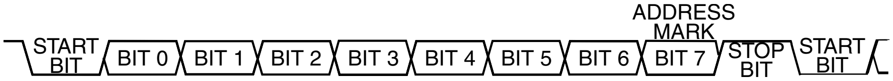


Figure 42-12. Eight bits of data with LSB first

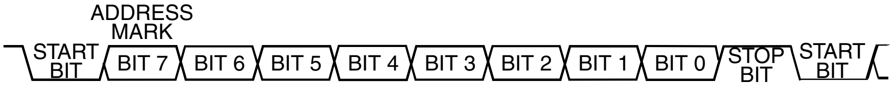


Figure 42-13. Eight bits of data with MSB first

42.2.1.4.3.2 Eight-bit format with parity enabled

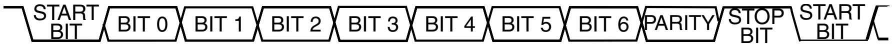


Figure 42-14. Seven bits of data with LSB first and parity

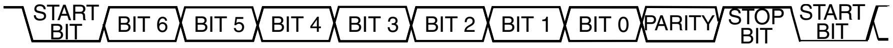


Figure 42-15. Seven bits of data with MSB first and parity

42.2.1.4.3.3 Nine-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

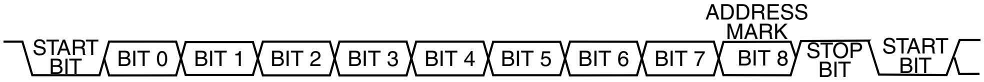


Figure 42-16. Nine bits of data with LSB first

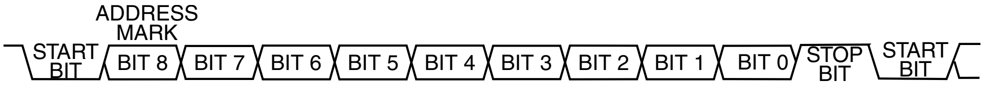


Figure 42-17. Nine bits of data with MSB first

42.2.1.4.3.4 Nine-bit format with parity enabled

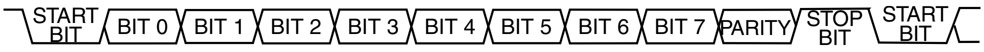


Figure 42-18. Eight bits of data with LSB first and parity

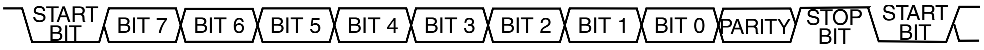


Figure 42-19. Eight bits of data with MSB first and parity

42.2.1.4.3.5 Non-memory mapped tenth bit for parity

The most significant memory-mapped bit can be used for address mark wakeup.

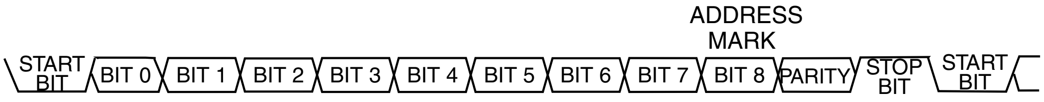


Figure 42-20. Nine bits of data with LSB first and parity

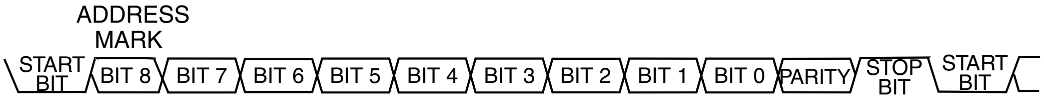


Figure 42-21. Nine bits of data with MSB first and parity

42.2.2 Operation sections

42.2.2.1 Single-wire operation

Normally, the UART uses two pins for transmitting and receiving. In single wire operation, the RXD pin is disconnected from the UART and the UART implements a half-duplex serial connection. The UART uses the TXD pin for both receiving and transmitting, see C3[TXDIR].

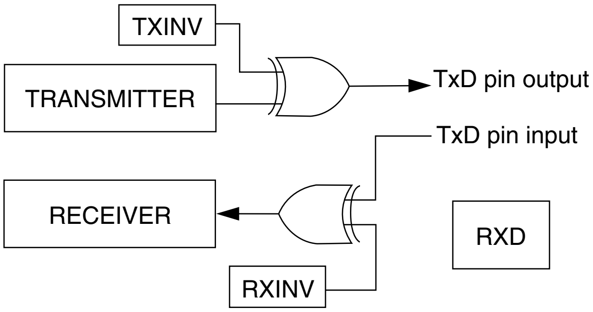


Figure 42-22. Single-wire operation (C1[LOOPS] = 1, C1[RSRC] = 1)

Enable single wire operation by setting C1[LOOPS] and the receiver source field, C1[RSRC]. Setting C1[LOOPS] disables the path from the unsynchronized receiver input signal to the receiver. Setting C1[RSRC] connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1).

42.2.2.2 Loop operation

In loop operation, the transmitter output goes to the receiver input. The unsynchronized receiver input signal is disconnected from the UART.

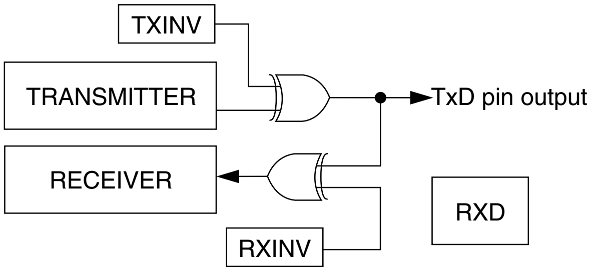


Figure 42-23. Loop operation (C1[LOOPS] = 1, C1[RSRC] = 0)

Enable loop operation by setting C1[LOOPS] and clearing C1[RSRC]. Setting C1[LOOPS] disables the path from the unsynchronized receiver input signal to the receiver. Clearing C1[RSRC] connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1).

42.2.2.3 Overrun operation

The assertion of S1[OR] indicates that a significant event has occurred. The assertion indicates that received data has been lost because there was a lack of room to store it in the data buffer. Therefore, while S1[OR] is set, no further data is stored in the data buffer until S1[OR] is cleared. This ensures that the application will be able to handle the overrun condition.

In most applications, because the total amount of lost data is known, the application will attempt to return the system to a known state. Before S1[OR] is cleared, all received data will be dropped. For this, the software does the following.

1. Remove data from the receive data buffer. This could be done by reading data from the data buffer and processing it.
2. Clear S1[OR].

When LIN break detect (LBKDE) is asserted, S1[OR] has significantly different behavior than in other modes. S1[OR] will be set, regardless of how much space is actually available in the data buffer, if a LIN break character has been detected and the corresponding flag, S2[LBKDIF], is not cleared before the first data character is received after S2[LBKDIF] asserted. This behavior is intended to allow the software sufficient time to read the LIN break character from the data buffer to ensure that a break character was actually detected. The checking of the break character was used on some older implementations and is therefore supported for legacy reasons. Applications that do not require this checking can simply clear S2[LBKDIF] without checking the stored value to ensure it is a break character.

42.2.3 Mode sections

The UART functions in the same way in all the normal modes.

It has the following low power modes:

- Wait mode
- Stop mode

42.2.3.1 Run mode

This is the normal mode of operation.

42.2.3.2 Wait mode

UART operation in the Wait mode depends on the state of the C1[UARTSWAI] field.

- If C1[UARTSWAI] is cleared, and the CPU is in Wait mode, the UART operates normally.
- If C1[UARTSWAI] is set, and the CPU is in Wait mode, the UART clock generation ceases and the UART module enters a power conservation state.

Setting C1[UARTSWAI] does not affect the state of the C2[RE] or C2[TE].

If C1[UARTSWAI] is set, any ongoing transmission or reception stops at the Wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of Wait mode. Bringing the CPU out of Wait mode by reset aborts any ongoing transmission or reception and resets the UART.

42.2.3.3 Stop mode

The UART is inactive during Stop mode for reduced power consumption. The STOP instruction does not affect the UART register states, but the UART module clock is disabled. The UART operation resumes after an external interrupt brings the CPU out of Stop mode. Bringing the CPU out of Stop mode by reset aborts any ongoing transmission or reception and resets the UART.

42.2.4 Clocking

SCI_C1[UARTSWAI] controls the run of UART clock in Wait mode. It can be set to freeze UART clock in wait mode. UART module clock is disabled during stop mode.

42.2.5 Reset

All registers reset to a particular value are indicated in Memory map and registers.

42.2.6 Interrupts

There are several interrupt signals that are sent from the UART. The following table lists the interrupt sources generated by the UART. The local enables for the UART interrupt sources are described in this table. Details regarding the individual operation of each interrupt are contained under various sub-sections of Memory map and registers.

However, RXEDGIF description also outlines additional details regarding the RXEDGIF interrupt because of its complexity of operation. Any of the UART interrupt requests listed in the table can be used to bring the CPU out of Wait mode.

Table 42-11. UART interrupt sources

Interrupt Source	Flag	Local enable	DMA select
Transmitter	TDRE	TIE	TDMAS = 0
Transmitter	TC	TCIE	-
Receiver	IDLE	ILIE	-
Receiver	RDRF	RIE	RDMAS = 0
Receiver	LBKDIF	LBKDIE	LBKDDMAS = 0
Receiver	RXEDGIF	RXEDGIE	-
Receiver	OR	ORIE	-
Receiver	NF	NEIE	-
Receiver	FE	FEIE	-
Receiver	PF	PEIE	-

42.2.6.1 RXEDGIF description

S2[RXEDGIF] is set when an active edge is detected on the RxD pin. Therefore, the active edge can be detected only when in two wire mode. A RXEDGIF interrupt is generated only when S2[RXEDGIF] is set. If RXEDGIE is not enabled before S2[RXEDGIF] is set, an interrupt is not generated.

42.2.6.1.1 RxD edge detect sensitivity

Edge sensitivity can be software programmed to be either falling or rising. The polarity of the edge sensitivity is selected using S2[RXINV]. To detect the falling edge, S2[RXINV] is programmed to 0. To detect the rising edge, S2[RXINV] is programmed to 1.

Synchronizing logic is used prior to detect edges. Prior to detecting an edge, the receive data on RxD input must be at the deasserted logic level. A falling edge is detected when the RxD input signal is seen as a logic 1 (the deasserted level) during one module clock cycle, and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input is seen as a logic 0 during one module clock cycle and then a logic 1 during the next cycle.

42.2.6.1.2 Clearing RXEDGIF interrupt request

Writing a logic 1 to S2[RXEDGIF] immediately clears the RXEDGIF interrupt request even if the RxD input remains asserted. S2[RXEDGIF] remains set if another active edge is detected on RxD while attempting to clear S2[RXEDGIF] by writing a 1 to it.

42.2.6.1.3 Exit from low-power modes

The receive input active edge detect circuit is still active on low power modes (Wait and Stop). An active edge on the receive input brings the CPU out of low power mode if the interrupt is not masked (S2[RXEDGIF] = 1).

42.2.7 DMA

In the transmitter, S1[TDRE] can be configured to assert a DMA transfer request. In the receiver, S1[RDRF],and S2[LBKDIF] can be configured to assert a DMA transfer request. The following table shows the configuration field settings required to configure each flag for DMA operation.

Table 42-12. DMA configuration

Flag	Request enable bit	DMA select bit
TDRE	TIE = 1	TDMAS = 1
RDRF	RIE = 1	RDMAS = 1
LBKDIF	LBKDIE = 1	LBKDDMAS = 1

When a flag is configured for a DMA request, its associated DMA request is asserted when the flag is set. When S1[RDRF] is configured as a DMA request, the clearing mechanism of reading S1, followed by reading D, does not clear the associated flag. The DMA request remains asserted until an indication is received that the DMA transactions are done. When this indication is received, the flag bit and the associated DMA request is cleared. If the DMA operation failed to remove the situation that caused the DMA request, another request is issued.

42.3 External signals

The UART signals are shown in the following table.

Table 42-13. UART signal descriptions

Signal	Description	I/O
RXD	Receive data	I
TXD	Transmit data	I/O
RTS_B	Request to send	O
CTS_B	Clear to send	I

When RTS_B is set 0, data receive is enabled.

When CTS_B is set 0, data send is enabled.

42.3.1 Detailed signal descriptions

The detailed signal descriptions of the UART are shown in the following table.

Table 42-14. UART—Detailed signal descriptions

Signal	I/O	Description	
RXD	I	Receive data. Serial data input to receiver.	
		State meaning	Whether RXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Sampled at a frequency determined by the module clock divided by the baud rate.
TXD	O	Transmit data. Serial data output from transmitter.	
		State meaning	Whether TXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Driven at the beginning or within a bit time according to the bit encoding method along with other configuration settings. Otherwise, transmissions are independent of reception timing.

42.4 Initialization

42.4.1 Initialization sequence

To initiate a UART transmission:

1. Configure the UART.

   a. Select a baud rate. Write this value to the UART baud registers (BDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the BDH has no effect without also writing to BDL.
   b. Write to C1 to configure word length, parity, and other configuration bits (LOOPS, RSRC, M, WAKE, ILT, PE, and PT). Write to C4, MA1, and MA2 to configure.
   c. Enable the transmitter, interrupts, receiver, and wakeup as required, by writing to C2 (TIE, TCIE, RIE, ILIE, TE, RE, RWU, and SBK), S2 (MSBF and BRK13), and C3 (ORIE, NEIE, PEIE, and FEIE). A preamble or idle character is then shifted out of the transmitter shift register.
2. Transmit procedure for each byte.
   a. Monitor S1[TDRE] by reading S1 or responding to the TDRE interrupt.
   b. If the TDRE flag is set, or there is space in the transmit buffer, write the data to be transmitted to (C3[T8]/D). A new transmission will not result until data exists in the transmit buffer.
3. Repeat step 2 for each subsequent transmission.

Note

During normal operation, S1[TDRE] is set when the shift register is loaded with the next data to be transmitted from the transmit buffer. This occurs 9/16ths of a bit time after the start of the stop bit of the previous frame.

To separate messages with preambles with minimum idle line time, use this sequence between messages.

1. Write the last dataword of the first message to C3[T8]/D.
2. Wait for S1[TDRE] to go high, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting C2[TE].
4. Write the first and subsequent datawords of the second message to C3[T8]/D.

42.5 Application information

This section describes the UART application information.

42.5.1 Overrun (OR) flag implications

To be flexible, the overrun flag (OR) operates slight differently depending on the mode of operation. There may be implications that need to be carefully considered. This section clarifies the behavior and the resulting implications. Regardless of mode, if a dataword is received while S1[OR] is set, S1[RDRF] and S1[IDLE] are blocked from asserting. If S1[RDRF] or S1[IDLE] were previously asserted, they will remain asserted until cleared.

42.5.2 Match address registers

The two match address registers allow a second match address function for a broadcast or general call address to the serial bus, as an example.

42.6 Memory map and registers

This section provides a detailed description of all memory and registers.

Accessing reserved addresses within the memory map results in a transfer error. None of the contents of the implemented addresses are modified as a result of that access.

Only byte accesses are supported.

SCI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_D000	UART Baud Rate Registers: High (SCI0_BDH)	8	R/W	00h	42.6.1/1224
4006_D001	UART Baud Rate Registers: Low (SCI0_BDL)	8	R/W	04h	42.6.2/1225
4006_D002	UART Control Register 1 (SCI0_C1)	8	R/W	00h	42.6.3/1226
4006_D003	UART Control Register 2 (SCI0_C2)	8	R/W	00h	42.6.4/1227
4006_D004	UART Status Register 1 (SCI0_S1)	8	R	C0h	42.6.5/1229
4006_D005	UART Status Register 2 (SCI0_S2)	8	R/W	00h	42.6.6/1231
4006_D006	UART Control Register 3 (SCI0_C3)	8	R/W	00h	42.6.7/1233
4006_D007	UART Data Register (SCI0_D)	8	R/W	00h	42.6.8/1234
4006_D008	UART Match Address Registers 1 (SCI0_MA1)	8	R/W	00h	42.6.9/1235
4006_D009	UART Match Address Registers 2 (SCI0_MA2)	8	R/W	00h	42.6.10/ 1236
4006_D00A	UART Control Register 4 (SCI0_C4)	8	R/W	00h	42.6.11/ 1236
4006_D00B	UART Control Register 5 (SCI0_C5)	8	R/W	00h	42.6.12/ 1237

SCI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_E000	UART Baud Rate Registers: High (SCI1_BDH)	8	R/W	00h	42.6.1/1224
4006_E001	UART Baud Rate Registers: Low (SCI1_BDL)	8	R/W	04h	42.6.2/1225
4006_E002	UART Control Register 1 (SCI1_C1)	8	R/W	00h	42.6.3/1226
4006_E003	UART Control Register 2 (SCI1_C2)	8	R/W	00h	42.6.4/1227
4006_E004	UART Status Register 1 (SCI1_S1)	8	R	C0h	42.6.5/1229
4006_E005	UART Status Register 2 (SCI1_S2)	8	R/W	00h	42.6.6/1231
4006_E006	UART Control Register 3 (SCI1_C3)	8	R/W	00h	42.6.7/1233
4006_E007	UART Data Register (SCI1_D)	8	R/W	00h	42.6.8/1234
4006_E008	UART Match Address Registers 1 (SCI1_MA1)	8	R/W	00h	42.6.9/1235
4006_E009	UART Match Address Registers 2 (SCI1_MA2)	8	R/W	00h	42.6.10/ 1236
4006_E00A	UART Control Register 4 (SCI1_C4)	8	R/W	00h	42.6.11/ 1236
4006_E00B	UART Control Register 5 (SCI1_C5)	8	R/W	00h	42.6.12/ 1237

42.6.1 UART Baud Rate Registers: High (SCIx_BDH)

This register, along with the BDL register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting (SBR[12:0]), first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written.

BDL is reset to a nonzero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] is set.

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read Write	LBKDIE	RXEDGIE	SBNS	SBR				
Reset	0	0	0	0	0	0	0	0

SCIx_BDH field descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt or DMA Request Enable Enables the LIN break detect flag, LBKDIF, to generate interrupt requests based on the state of LBKDDMAS. or DMA transfer requests,

Table continues on the next page...

SCIx_BDH field descriptions (continued)

Field	Description
	0 LBKDIF interrupt and DMA transfer requests disabled. 1 LBKDIF interrupt or DMA transfer requests enabled.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable Enables the receive input active edge, RXEDGIF, to generate interrupt requests. 0 Hardware interrupts from RXEDGIF disabled using polling. 1 RXEDGIF interrupt request enabled.
5 SBNS	Stop Bit Number Select SBNS selects the number of stop bits present in a data frame. This field valid for all 8, 9 and 10 bit data formats available. 0 Data frame consists of a single stop bit. 1 Data frame consists of two stop bits.
SBR	UART Baud Rate Bits The baud rate for the UART is determined by the 13 SBR fields. See Baud rate generation for details. **NOTE:** • The baud rate generator is disabled until C2[TE] or C2[RE] is set for the first time after reset.The baud rate generator is disabled when SBR = 0. • Writing to BDH has no effect without writing to BDL, because writing to BDH puts the data in a temporary location until BDL is written.

42.6.2 UART Baud Rate Registers: Low (SCIx_BDL)

This register, along with the BDH register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting, SBR[12:0], first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written. BDL is reset to a nonzero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] is set.

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SBR							
Reset	0	0	0	0	0	1	0	0

SCIx_BDL field descriptions

Field	Description
SBR	UART Baud Rate Bits The baud rate for the UART is determined by the 13 SBR fields. See Baud rate generation for details.

SCIx_BDL field descriptions (continued)

Field	Description
	NOTE: • The baud rate generator is disabled until C2[TE] or C2[RE] is set for the first time after reset.The baud rate generator is disabled when SBR = 0. • Writing to BDH has no effect without writing to BDL, because writing to BDH puts the data in a temporary location until BDL is written.

42.6.3 UART Control Register 1 (SCIx_C1)

This read/write register controls various optional features of the UART system.

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	LOOPS	UARTSWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

SCIx_C1 field descriptions

Field	Description
7 LOOPS	Loop Mode Select When LOOPS is set, the RxD pin is disconnected from the UART 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. 1 Loop mode where transmitter output is internally connected to receiver input. The receiver input is determined by RSRC.
6 UARTSWAI	UART Stops in Wait Mode 0 UART clock continues to run in Wait mode. 1 UART clock freezes while CPU is in Wait 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 Selects internal loop back mode. The receiver input is internally connected to transmitter output. 1 Single wire UART mode where the receiver input is connected to the transmit pin input signal.
4 M	9-bit or 8-bit Mode Select 0 Normal—start + 8 data bits (MSB/LSB first as determined by MSBF) + stop. 1 Use—start + 9 data bits (MSB/LSB first as determined by MSBF) + stop.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the UART: • Address mark in the most significant bit position of a received data character, or • An idle condition on the receive pin input signal. 0 Idle line wakeup. 1 Address mark wakeup.

Table continues on the next page...

SCIx_C1 field descriptions (continued)

Field	Description
2 ILT	Idle Line Type Select 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 UART is programmed with ILT = 1, a logic of 1'b0 is automatically shifted after a received stop bit, therefore resetting the idle count. • In case the UART is programmed for IDLE line wakeup (RWU = 1 and WAKE = 0), ILT has no effect on when the receiver starts counting logic 1s as idle character bits. In idle line wakeup, an idle character is recognized at anytime the receiver sees 10, 11, or 12 1s depending on the M, PE, and C4[M10] fields. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables the parity function. When parity is enabled, parity function inserts a parity bit in the bit position immediately preceding the stop bit. 0 Parity function disabled. 1 Parity function enabled.
0 PT	Parity Type Determines whether the UART generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. 0 Even parity. 1 Odd parity.

42.6.4 UART Control Register 2 (SCIx_C2)

This register can be read or written at any time.

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read Write	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

SCIx_C2 field descriptions

Field	Description
7 TIE	Transmitter Interrupt or DMA Transfer Enable. Enables S1[TDRE] to generate interrupt requests or DMA transfer requests, based on the state of C5[TDMAS].

Table continues on the next page...

SCIx_C2 field descriptions (continued)

Field	Description
	NOTE: If C2[TIE] and C5[TDMAS] are both set, then TCIE must be cleared, and D[D] must not be written unless servicing a DMA request. 0 TDRE interrupt and DMA transfer requests disabled. 1 TDRE interrupt or DMA transfer requests enabled.
6 TCIE	Transmission Complete Interrupt Enable Enables the transmission complete flag, S1[TC], to generate interrupt requests . 0 TC interrupt requests disabled. 1 TC interrupt requests enabled.
5 RIE	Receiver Full Interrupt or DMA Transfer Enable Enables S1[RDRF] to generate interrupt requests or DMA transfer requests, based on the state of C5[RDMAS]. 0 RDRF interrupt and DMA transfer requests disabled. 1 RDRF interrupt or DMA transfer requests enabled.
4 ILIE	Idle Line Interrupt Enable Enables the idle line flag, S1[IDLE], to generate interrupt requests 0 IDLE interrupt requests disabled. 1 IDLE interrupt requests enabled.
3 TE	Transmitter Enable Enables the UART transmitter. TE can be used to queue an idle preamble by clearing and then setting TE. 0 Transmitter off. 1 Transmitter on.
2 RE	Receiver Enable Enables the UART receiver. 0 Receiver off. 1 Receiver on.
1 RWU	Receiver Wakeup Control This field can be set to place the UART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when C1[WAKE] is clear or an address match when C1[WAKE] is set. **NOTE:** RWU must be set only with C1[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by S2[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the UART 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 reasserted. 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	Send Break

Table continues on the next page...

SCIx_C2 field descriptions (continued)

Field	Description
	Toggling SBK sends one break character from the following: See Transmitting break characters for the number of logic 0s for the different configurations. Toggling implies clearing the SBK field before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10, 11, or 12 bits, or 13 or 14 bits). Ensure that C2[TE] is asserted atleast 1 clock before assertion of this bit. • 10, 11, or 12 logic 0s if S2[BRK13] is cleared • 13 or 14 logic 0s if S2[BRK13] is set. 0 Normal transmitter operation. 1 Queue break characters to be sent.

42.6.5 UART Status Register 1 (SCIx_S1)

The S1 register provides inputs to the MCU for generation of UART interrupts or DMA requests. This register can also be polled by the MCU to check the status of its fields. To clear a flag, the status register should be read followed by a read or write to D register, depending on the interrupt flag type. Other instructions can be executed between the two steps as long the handling of I/O is not compromised, but the order of operations is important for flag clearing. When a flag is configured to trigger a DMA request, assertion of the associated DMA done signal from the DMA controller clears the flag.

NOTE

- If the condition that results in the assertion of the flag, interrupt, or DMA request is not resolved prior to clearing the flag, the flag, and interrupt/DMA request, reasserts.

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write								
Reset	1	1	0	0	0	0	0	0

SCIx_S1 field descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag TDRE will set when the transmit data register (D) is empty. To clear TDRE, read S1 when TDRE is set and then write to the UART data register (D). 0 Transmit data buffer is full. 1 Transmit data buffer is empty.

Table continues on the next page...

SCIx_S1 field descriptions (continued)

Field	Description
6 TC	Transmit Complete Flag 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 reading S1 with TC set and then doing one of the following: • Writing to D to transmit new data. • Queuing a preamble by clearing and then setting C2[TE]. • Queuing a break character by writing 1 to SBK in C2. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
5 RDRF	Receive Data Register Full Flag RDRF is set when the receive buffer (D) is full. To clear RDRF, read S1 when RDRF is set and then read D. 0 Receive data buffer is empty. 1 Receive data buffer is full.
4 IDLE	Idle Line Flag After the IDLE flag is cleared, a frame must be received (although not necessarily stored in the data buffer, for example if C2[RWU] is set), or a LIN break character must set the S2[LBKDIF] flag before an idle condition can set the IDLE flag. To clear IDLE, read UART status S1 with IDLE set and then read D. IDLE is set when either of the following appear on the receiver input: • 10 consecutive logic 1s if C1[M] = 0 • 11 consecutive logic 1s if C1[M] = 1 and C4[M10] = 0 • 12 consecutive logic 1s if C1[M] = 1, C4[M10] = 1, and C1[PE] = 1 **NOTE:** When RWU is set and WAKE is cleared, an idle line condition sets the IDLE flag if RWUID is set, else the IDLE flag does not become set. 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared. 1 Receiver input has become idle or the flag has not been cleared since it last asserted.
3 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 UART data registers is not affected. If the OR flag is set, no data is stored in the data buffer even if sufficient room exists. Additionally, while the OR flag is set, the RDRF and IDLE flags are blocked from asserting, that is, transition from an inactive to an active state. To clear OR, read S1 when OR is set and then read D. See functional description for more details regarding the operation of the OR bit.If LBKDE is enabled and a LIN Break is detected, the OR field asserts if S2[LBKDIF] is not cleared before the next data character is received. 0 No overrun has occurred since the last time the flag was cleared. 1 Overrun has occurred or the overrun flag has not been cleared since the last overrun occured.
2 NF	Noise Flag NF is set when the UART detects noise on the receiver input. NF does not become set in the case of an overrun or while the LIN break detect feature is enabled (S2[LBKDE] = 1). To clear NF, read S1 and then read D.

Table continues on the next page...

SCIx_S1 field descriptions (continued)

Field	Description
	0 No noise detected. 1 Noise detected in the received character in D.
1 FE	Framing Error Flag FE is set when a logic 0 is accepted as the stop bit. When BDH[SBNS] is set, then FE will set when a logic 0 is accepted for either of the two stop bits. FE does not set in the case of an overrun or while the LIN break detect feature is enabled (S2[LBKDE] = 1). FE inhibits further data reception until it is cleared. To clear FE, read S1 with FE set and then read D. The last data in the receive buffer represents the data that was received with the frame error enabled. 0 No framing error detected. 1 Framing error.
0 PF	Parity Error Flag PF is set when PE is set and the parity of the received data does not match its parity bit. The PF is not set in the case of an overrun condition. To clear PF, read S1 and then read D., S2[LBKDE] is disabled, 0 No parity error detected. 1 Parity error.

42.6.6 UART Status Register 2 (SCIx_S2)

The S2 register provides inputs to the MCU for generation of UART interrupts or DMA requests. Also, this register can be polled by the MCU to check the status of these bits. This register can be read or written at any time, with the exception of the MSBF and RXINV bits, which should be changed by the user only between transmit and receive packets.

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF
Write	w1c	w1c						▨
Reset	0	0	0	0	0	0	0	0

SCIx_S2 field descriptions

Field	Description
7 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when LBKDE is set and a LIN break character is detected on the receiver input. The LIN break characters are 11 consecutive logic 0s if C1[M] = 0 or 12 consecutive logic 0s if C1[M] = 1. LBKDIF is set after receiving the last LIN break character. LBKDIF is cleared by writing a 1 to it. 0 No LIN break character detected. 1 LIN break character detected.

Table continues on the next page...

SCIx_S2 field descriptions (continued)

Field	Description
6 RXEDGIF	RxD Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge occurs on the RxD pin. The active edge is falling if RXINV = 0, and rising if RXINV=1. RXEDGIF is cleared by writing a 1 to it. See for additional details. RXEDGIF description **NOTE:** The active edge is detected only in two wire mode and on receiving data coming from the RxD pin. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
5 MSBF	Most Significant Bit First Setting this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. 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 (bit8, bit7 or bit6) is the first bit that is transmitted following the start bit, depending on the setting of C1[M] and C1[PE]. Further, the first bit received after the start bit is identified as bit8, bit7, or bit6, depending on the setting of C1[M] and C1[PE].
4 RXINV	Receive Data Inversion Setting this field reverses the polarity of the received data input. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. **NOTE:** Setting RXINV inverts the RxD input for data bits, start and stop bits, break, and idle. 0 Receive data is not inverted. 1 Receive data is inverted.
3 RWUID	Receive Wakeup Idle Detect When RWU is set and WAKE is cleared, this field controls whether the idle character that wakes the receiver sets S1[IDLE]. 0 S1[IDLE] is not set upon detection of an idle character. 1 S1[IDLE] is set upon detection of an idle character.
2 BRK13	Break Transmit Character Length Determines whether the transmit break character is 10, 11, or 12 bits long, or 13 or 14 bits long. See for the length of the break character for the different configurations. The detection of a framing error is not affected by this field. Transmitting break characters 0 Break character is 10, 11, or 12 bits long. 1 Break character is 13 or 14 bits long.
1 LBKDE	LIN Break Detection Enable Enables the LIN Break detection feature. While LBKDE is set, S1[RDRF], S1[NF], S1[FE], and S1[PF] are prevented from setting. When LBKDE is set, see . Overrun operation 0 Break character detection is disabled. 1 Break character is detected at length of 11 bit times if C1[M] = 0 or 12 bits time if C1[M] = 1.
0 RAF	Receiver Active Flag

Table continues on the next page...

SCIx_S2 field descriptions (continued)

Field	Description
	RAF is set when the UART receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 UART receiver idle/inactive waiting for a start bit. 1 UART receiver active, RxD input not idle.

42.6.7 UART Control Register 3 (SCIx_C3)

Writing R8 does not have any effect. TXDIR and TXINV can be changed only between transmit and receive packets.

Address: Base address + 6h offset

Bit	7	6	5	4	3	2	1	0
Read	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
Write								
Reset	0	0	0	0	0	0	0	0

SCIx_C3 field descriptions

Field	Description
7 R8	Received Bit 8 R8 is the ninth data bit received when the UART is configured for 9-bit data format, that is, if C1[M] = 1 or C4[M10] = 1. The R8 value corresponds to the current data value in the UARTx_D register. To read the 9th bit, read the value of UARTx_C3[R8], then read the UARTx_D register.
6 T8	Transmit Bit 8 T8 is the ninth data bit transmitted when the UART is configured for 9-bit data format, that is, if C1[M] = 1 or C4[M10] = 1. **NOTE:** If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten. The same value is transmitted until T8 is rewritten. To correctly transmit the 9th bit, write UARTx_C3[T8] to the desired value, then write the UARTx_D register with the remaining data.
5 TXDIR	Transmitter Pin Data Direction in Single-Wire mode Determines whether the TXD pin is used as an input or output in the single-wire mode of operation. This field is relevant only to the single wire mode. 0 TXD pin is an input in single wire mode. 1 TXD pin is an output in single wire mode.
4 TXINV	Transmit Data Inversion. Setting this field reverses the polarity of the transmitted data output. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity.

Table continues on the next page...

SCIx_C3 field descriptions (continued)

Field	Description
	NOTE: Setting TXINV inverts all transmitted values, including idle, break, start, and stop bits. In loop mode, if TXINV is set, the receiver gets the transmit inversion bit when RXINV is disabled. 0 Transmit data is not inverted. 1 Transmit data is inverted.
3 ORIE	Overrun Error Interrupt Enable Enables the overrun error flag, S1[OR], to generate interrupt requests. 0 OR interrupts are disabled. 1 OR interrupt requests are enabled.
2 NEIE	Noise Error Interrupt Enable Enables the noise flag, S1[NF], to generate interrupt requests. 0 NF interrupt requests are disabled. 1 NF interrupt requests are enabled.
1 FEIE	Framing Error Interrupt Enable Enables the framing error flag, S1[FE], to generate interrupt requests. 0 FE interrupt requests are disabled. 1 FE interrupt requests are enabled.
0 PEIE	Parity Error Interrupt Enable Enables the parity error flag, S1[PF], to generate interrupt requests. 0 PF interrupt requests are disabled. 1 PF interrupt requests are enabled.

42.6.8 UART Data Register (SCIx_D)

This register is actually two separate registers. Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

NOTE

- In 8-bit or 9-bit data format, only UART data register (D) needs to be accessed to clear the S1[RDRF] bit . The C3 register needs to be read, prior to the D register, only if the ninth bit of data needs to be captured.
- In the normal 8-bit mode (M bit cleared) if the parity is enabled, you get seven data bits and one parity bit. That one parity bit is loaded into the D register. So, for the data

bits, mask off the parity bit from the value you read out of this register.

- When transmitting in 9-bit data format and using 8-bit write instructions, write first to transmit bit 8 in UART control register 3 (C3[T8]), then D. A write to C3[T8] stores the data in a temporary register. If D register is written first, and then the new data on data bus is stored in D, the temporary value written by the last write to C3[T8] gets stored in the C3[T8] register.

Address: Base address + 7h offset

Bit	7	6	5	4	3	2	1	0
Read Write	RT							
Reset	0	0	0	0	0	0	0	0

SCIx_D field descriptions

Field	Description
RT	Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

42.6.9 UART Match Address Registers 1 (SCIx_MA1)

The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] field is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. These registers can be read and written at anytime.

Address: Base address + 8h offset

Bit	7	6	5	4	3	2	1	0
Read Write	MA							
Reset	0	0	0	0	0	0	0	0

SCIx_MA1 field descriptions

Field	Description
MA	Match Address

42.6.10 UART Match Address Registers 2 (SCIx_MA2)

These registers can be read and written at anytime. The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] field is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded.

Address: Base address + 9h offset

Bit	7	6	5	4	3	2	1	0
Read Write	MA							
Reset	0	0	0	0	0	0	0	0

SCIx_MA2 field descriptions

Field	Description
MA	Match Address

42.6.11 UART Control Register 4 (SCIx_C4)

Address: Base address + Ah offset

Bit	7	6	5	4	3	2	1	0
Read Write	MAEN1	MAEN2	M10	BRFA				
Reset	0	0	0	0	0	0	0	0

SCIx_C4 field descriptions

Field	Description
7 MAEN1	Match Address Mode Enable 1 See Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN2 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA1 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer.
6 MAEN2	Match Address Mode Enable 2 See Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN1 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA2 register. If no match occurs, the data is discarded. If a match occurs, data is transferred to the data buffer.
5 M10	10-bit Mode select

Table continues on the next page...

SCIx_C4 field descriptions (continued)

Field	Description
	Causes a tenth, non-memory mapped bit to be part of the serial transmission. This tenth bit is generated and interpreted as a parity bit. The M10 field does not affect the LIN send or detect break behavior. If M10 is set, then both C1[M] and C1[PE] must also be set. See Data format for more information. 0 The parity bit is the ninth bit in the serial transmission. 1 The parity bit is the tenth bit in the serial transmission.
BRFA	Baud Rate Fine Adjust This bit field is used to add more timing resolution to the average baud frequency, in increments of 1/32. See Baud rate generation for more information.

42.6.12 UART Control Register 5 (SCIx_C5)

Address: Base address + Bh offset

Bit	7	6	5	4	3	2	1	0
Read	TDMAS	0	RDMAS	0	LBKDDMAS		0	
Write		▨		▨		▨	▨	▨
Reset	0	0	0	0	0	0	0	0

SCIx_C5 field descriptions

Field	Description
7 TDMAS	Transmitter DMA Select Configures the transmit data register empty flag, S1[TDRE], to generate interrupt or DMA requests if C2[TIE] is set. **NOTE:** • If C2[TIE] is cleared, TDRE DMA and TDRE interrupt request signals are not asserted when the TDRE flag is set, regardless of the state of TDMAS. • If C2[TIE] and TDMAS are both set, then C2[TCIE] must be cleared, and D must not be written unless a DMA request is being serviced. 0 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE interrupt request signal is asserted to request interrupt service. 1 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE DMA request signal is asserted to request a DMA transfer.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 RDMAS	Receiver Full DMA Select Configures the receiver data register full flag, S1[RDRF], to generate interrupt or DMA requests if C2[RIE] is set. **NOTE:** If C2[RIE] is cleared, and S1[RDRF] is set, the RDRF DMA and RDFR interrupt request signals are not asserted, regardless of the state of RDMAS.

Table continues on the next page...

SCIx_C5 field descriptions (continued)

Field	Description
	0 If C2[RIE] and S1[RDRF] are set, the RDFR interrupt request signal is asserted to request an interrupt service. 1 If C2[RIE] and S1[RDRF] are set, the RDRF DMA request signal is asserted to request a DMA transfer.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LBKDDMAS	LIN Break Detect DMA Select Bit Configures the LIN break detect flag, S2[LBKDIF], to generate interrupt or DMA requests if BDH[LBKDIE] is set. **NOTE:** If BDH[LBKDIE] is cleared, and S2[LBKDIF] is set, the LBKDIF DMA and LBKDIF interrupt signals are not asserted, regardless of the state of LBKDDMAS. 0 If BDH[LBKDIE] and S2[LBKDIF] are set, the LBKDIF interrupt signal is asserted to request an interrupt service. 1 If BDH[LBKDIE] and S2[LBKDIF] are set, the LBKDIF DMA request signal is asserted to request a DMA transfer.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Chapter 43
Flexible I/O (FlexIO)

43.1 Chip-specific Information for this Module

43.1.1 Instantiation Information

Table 43-1. FlexIO Configuration

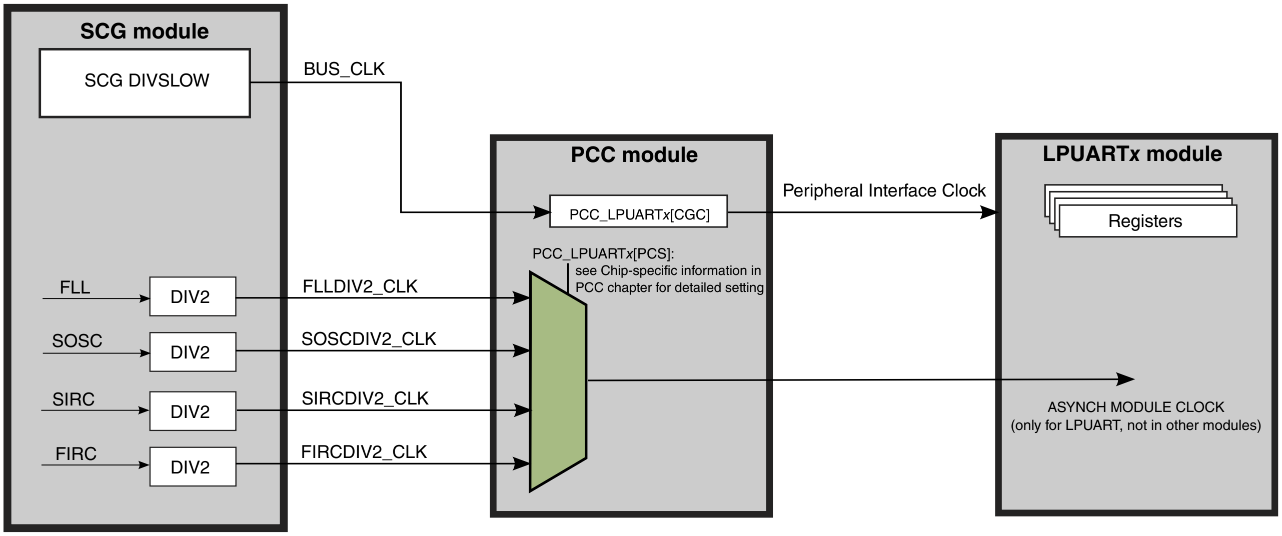
	Timers	Shifters	Pins
Number	4	4	8

43.1.2 FlexIO Clocking Information

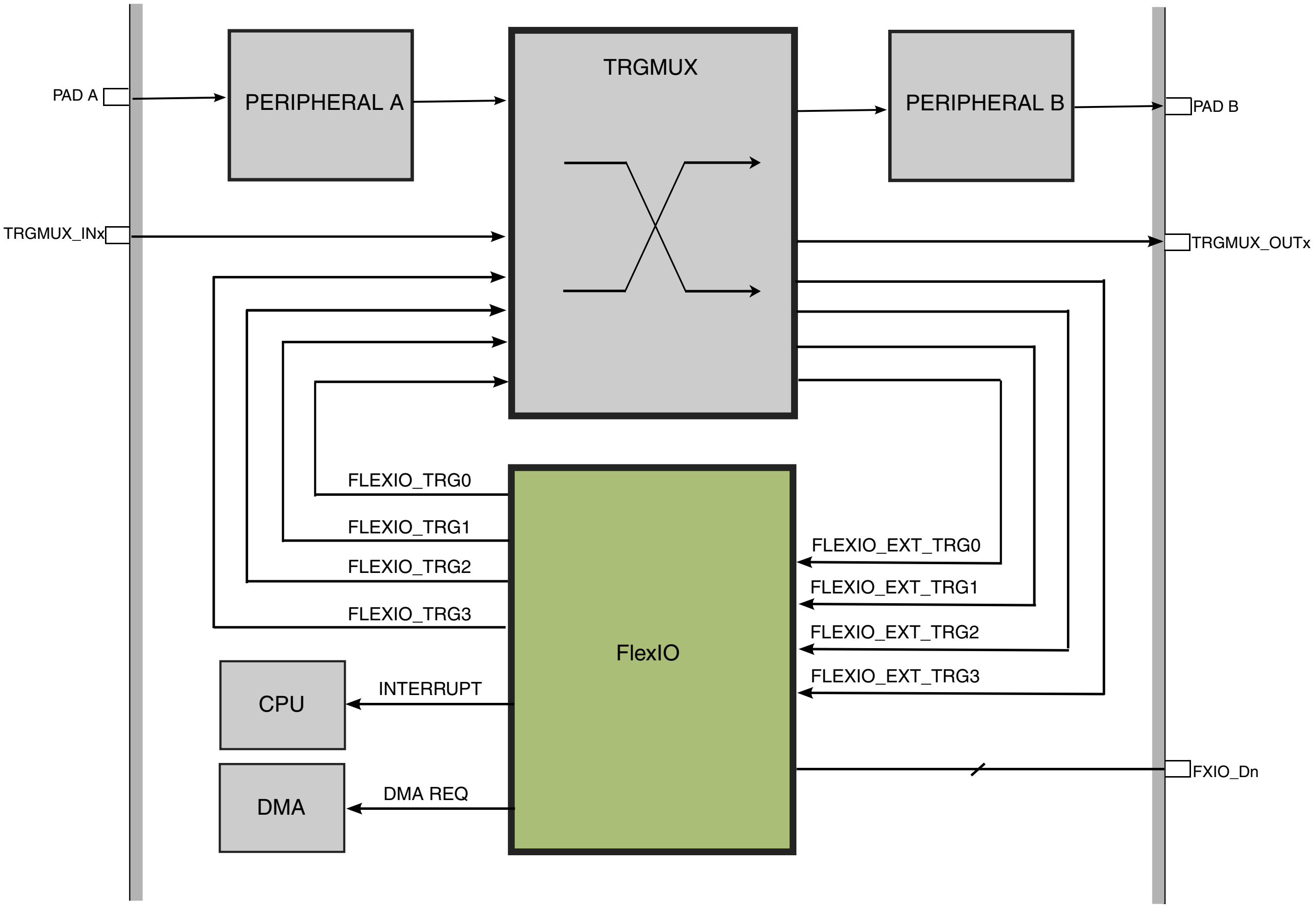
The FlexIO blocks are clocked from a single FlexIO clock that can be selected from OSCCLK, SCGIRCLK, SCGFIRCLK, or SCGFCLK. The selected source is controlled by the PCC_FLEXIO register in the PCC module. You have to select a clock for FlexIO and enable the clock gate before accessing any of the FlexIO registers.

Peripheral Clocking - LPUART, etc.

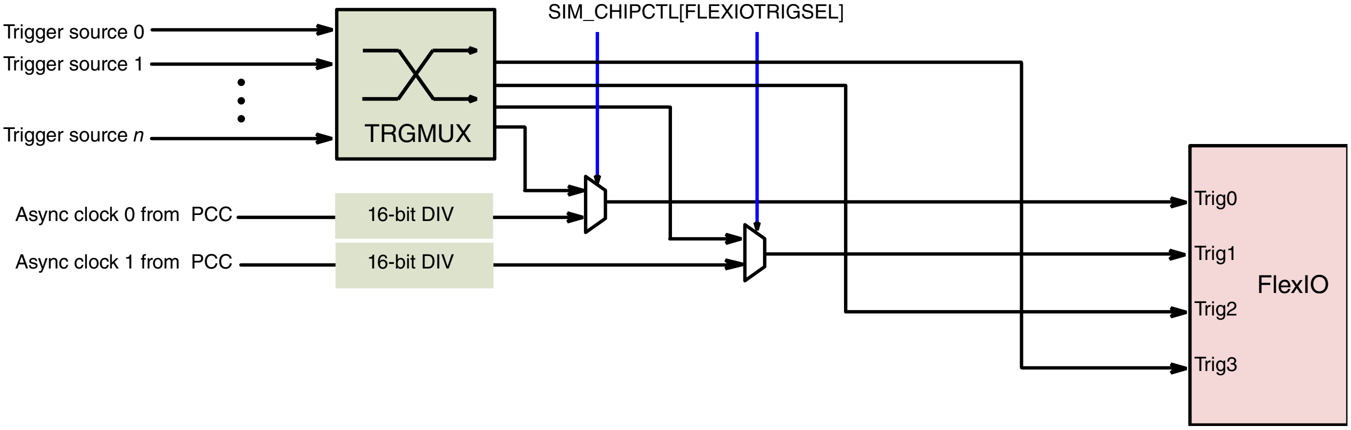
Note: this example figure also applies similarly to the clocking for LPSPI, LPI2C, LPIT, FlexIO, etc.



43.1.3 Inter-connectivity Information



FlexIO has a selectable trigger input source controlled by FlexIO_TIMCTLn[TRGSEL] (4-bit field) to use for starting the counter and/or reloading the counter. The trigger signal is from the FlexIO module itself which is called internal triggers, or from other modules which is called external triggers. The external triggers selection is controlled by the TRGMUX_FLEXIO register in the TRGMUX module. For this device, the external triggers can be selected from any of the TRGMUX trigger sources. FlexIO trigger inputs can come from TRGMUX or two independent divided asynchronous clocks.



43.2 Overview

FLEXIO is a highly configurable module that provides:

- Emulation of various serial communication protocols.
- Flexible 16-bit timers with support for various trigger, reset, enable, and disable conditions.

43.2.1 Block diagram

The following diagram provides a high-level overview of the FLEXIO timer and shifter configuration.

FLEXIO uses shifters, timers, and external triggers to shift data into or out of FLEXIO. As shown in the block diagram, timers control the timing of this data shift. You can configure the timers to use generic timer functions, external triggers, or various other conditions to determine the control.

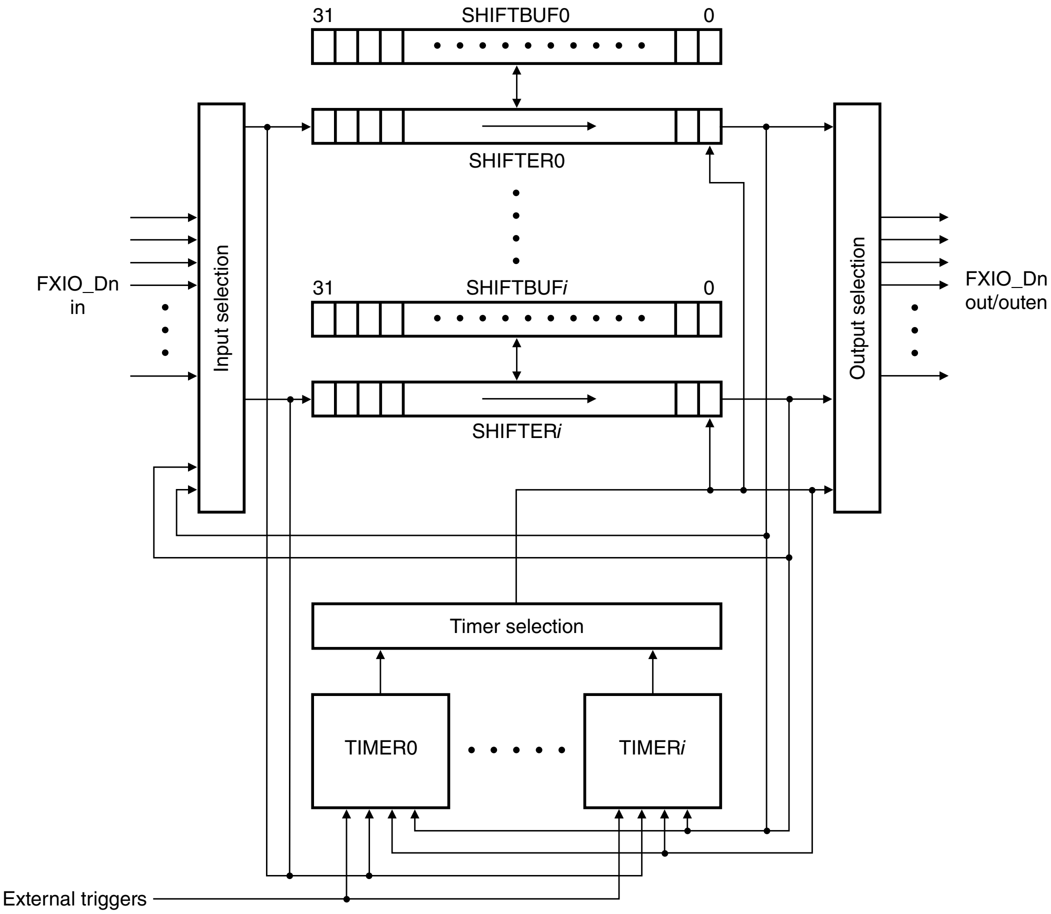


Figure 43-1. Block diagram

43.2.2 Features

- 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 and stop bit generation
  - Interrupt, DMA, or polled transmit and receive operation
- Highly flexible 16-bit timers with support for various internal or external triggers, reset, enable, and disable conditions:
  - Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during Stop mode
- Support for a wide range of protocols, including but not limited to:
  - UART
  - I2C
  - SPI

- I2S
- PWM or waveform generation

43.3 Functional description

43.3.1 Shifter operation

Shifters are responsible for buffering and shifting data into or out of FLEXIO. The timer assigned to the shifter controls the timing of shift, load, and store events via SHIFTCTL*n*[TIMSEL]. Shifters are designed to support either DMA, interrupt, or polled operations. The following figure provides a detailed view of the shifter microarchitecture.

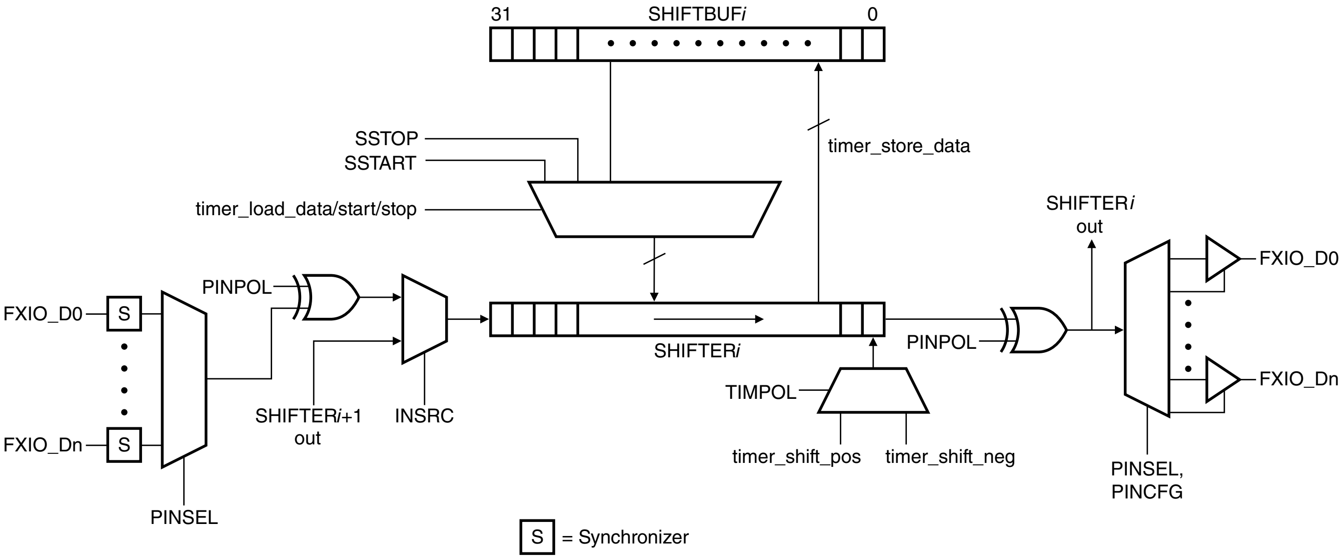


Figure 43-2. Shifter microarchitecture

43.3.1.1 Transmit mode

In Transmit mode (SHIFTCTL*n*[SMOD] = 010b), the shifter loads data from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) and shifts data out when the assigned timer signals a load event. An optional start and stop bit can be automatically loaded before or after SHIFTBUF register data by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] in the shifter and timer.

NOTE

If a stop bit is enabled, the shifter immediately loads a stop bit when it is initially configured for Transmit mode.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data has either been loaded from the SHIFTBUF register into the shifter or when the shifter is initially configured for Transmit mode. To clear the flag, write 1 or write new data to SHIFTBUF. In Transmit mode, write any value to the SHIFTBUF register to clear the corresponding shifter status flag, which is cleared regardless of what is writing to the register (DMA or interrupt), or the state of the DMA or interrupt enables. See the functional description of SHIFTSTAT[SSF] for information on how the flag is set and cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). Clear the flag by writing 1.

43.3.1.2 Receive mode

When the assigned timer signals a store event in Receive mode (SHIFTCTL*n*[SMOD] = 001b), the shifter shifts and stores data in Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). You can check for a start and stop bit before or after the shifter data is sampled by configuring either SHIFTCFG[SSTART] and TIMCFG[TSTART], or SHIFTCFG[SSTOP] and TIMCFG[TSTOP] in the shifter and timer.

The shifter status flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests are set when data is stored in the SHIFTBUF register from the shifter. To clear the flag, write 1 to or read the data from SHIFTBUF. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is in Receive mode. The flag is cleared regardless of what is reading the register (DMA or interrupt) or the state of the DMA or interrupt enables. See the functional description of SHIFTSTAT[SSF] for information on how the flag is set or cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set either when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start or stop bit check. Write 1 to clear the flag.

43.3.1.3 Match Store mode

In Match Store mode (SHIFTCTL*n*[SMOD] = 100b), the shifter shifts data in, checks for a match result, and stores matched data in Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) when the assigned timer signals a store event. By configuring either

SHIFTCFG[SSTART], TIMCFG[TSTART], and SHIFTCFG[SSTOP], or TIMCFG[TSTOP] in the shifter and timer, you can check for a start and stop bit before or after the shifter data is sampled. You can compare up to 16 bits of data 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 are set when a match occurs and the matched data is stored in the SHIFTBUF register from the shifter. To clear the flag, read the matched data from the SHIFTBUF register or write 1 to the flag. Any read of the SHIFTBUF register clears the corresponding shifter status flag when the shifter is configured in Match Store mode. The flag is cleared regardless of what is reading the register (DMA or interrupt) or the state of the DMA or interrupt enables. See the functional description for SHIFTSTAT[SSF] to know how the flag is set or cleared for each mode.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun), or when a mismatch occurs on a start or stop bit check. Write 1 to clear the flag.

43.3.1.4 Match Continuous mode

In Match Continuous mode (SHIFTCTL*n*[SMOD] = 101b), the shifter shifts data in and continuously checks for a match result whenever a shift event is signaled by the assigned timer. You can compare up to 16 bits of data 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 are set when a match occurs. The flag clears automatically as soon as no match exists between the shifter data and Shifter Buffer (SHIFTBUF0 - SHIFTBUF3).

You cannot clear the flag by reading the SHIFTBUF register.

The shifter error flag (SHIFTERR[SEF]) and any enabled interrupts are set when a match occurs. To clear the flag, write 1 or perform a read from the SHIFTBUF register.

43.3.2 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. The counters can perform generic timer functions such as generating a clock, select output, or a PWM waveform. You can configure these timers to perform any of the following functions:

- 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.
- Disable on a trigger or pin condition or on a timer compare.

Timers can optionally include a start condition and a stop condition.

Although each timer operates independently, you can configure a timer to enable or disable at the same time as the previous timer (for example, timer 1 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 as a timer output, shifter status flag, pin input, or an external trigger input. The trigger configuration is separate from pin configuration; you can perform it to configure input, output data, or output enable. See the chip-specific FLEXIO information for information on external trigger connections.

You must configure Timer Configuration (TIMCFG0 - TIMCFG3) before writing 1 to TIMCTL*n*[TIMOD].

43.3.2.1 Timer 8-bit Baud Counter mode

In 8-bit Baud Counter mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the baud rate of the shift clock and the upper 8 bits are used to configure the number of shift clock edges in the transfer. When the lower 8 bits decrement to zero, the timer output is toggled and the lower 8 bits reload from the compare register. The upper 8 bits decrement when the lower 8 bits become zero and decrement.

NOTE

A timer reset event in 8-bit Baud Counter mode only resets the lower 8-bit counter. The upper 8-bit counter is not affected and can decrement if the timer reset is configured to update the state of the timer output, which toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8 bits equal zero and decrement. The timer status flag is set on a timer compare event.

43.3.2.2 Timer 8-bit High PWM mode

In 8-bit High PWM mode, the 16-bit counter is divided into two 8-bit counters. The lower 8 bits are used to configure the timer output high period and the upper 8 bits are used to configure the timer output low period. The lower 8 bits decrement when the output is high. When the lower 8 bits become zero and decrement, the timer output is cleared and the lower 8 bits are reloaded from the compare register. The upper 8 bits decrement when the output is low. When the upper 8 bits become zero and decrement, the timer output is set and the upper 8 bits are reloaded from the compare register.

A timer compare event occurs when the upper 8 bits become zero and decrement. The timer status flag is set on a timer compare event.

43.3.2.3 Timer 16-bit Counter mode

In 16-bit Counter mode, you can use the 16-bit counter to configure either the baud rate of the shift clock (for example, TIMDEC[1:0] ≠ 10 or 11) or the number of shift clock edges in the transfer (for example, TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer compare event.

43.3.2.4 Timer enable and start functions

The following events occur when you configure TIMCTL*n*[TIMOD] for the desired mode and the condition configured by the timer enable (TIMCFG*n*[TIMENA]) is detected.

- The timer counter loads the current value of the compare register and starts decrementing, as configured by TIMCFG*n*[TIMDEC].
- The timer output may update to its initial state depending on the configuration of TIMCFG*n*[TIMOUT]. Shifters that are controlled by this timer do not see this as a rising edge on the timer shift clock.
- Transmit shifters controlled by this timer either output their start bit value or load the shift register from the shift buffer and output the first bit, as configured by SHIFTCFG*n*[SSTART].

If the timer start bit is enabled, the timer counter reloads 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 TIMCFG*n*[TIMOUT] = 1), a shifter that is configured to shift on the falling edge and load on the first shift does not load correctly.

43.3.2.5 Timer decrement and reset functions

The timer generates the timer output and timer shift clock depending on the fields, TIMCTL*n*[TIMOD] and TIMCFG*n*[TIMDEC]. The shifter clock is either equal to the timer output (when TIMCFG*n*[TIMDEC] ≠ 10 or 11) or equal to the decrement clock (when TIMCFG*n*[TIMDEC] = 10 or 11). If you configure TIMCFG*n*[TIMDEC] to decrement from a pin or trigger, the timer decrements on both rising and falling edges.

If you configure the timer to reset as determined by TIMCFG*n*[TIMRST], then the timer counter loads the current value of the compare register again. You can configure the timer output and timer shift clock to update on timer reset, as configured by TIMCFG*n*[TIMOUT]. If the time output toggles as a result of the timer reset, this can result in a timer shift clock edge. In 8-bit Baud Counter mode, this also decrements the upper 8 bits of the counter.

In general, when the timer counter decrements to zero, a timer compare event is triggered. The timer compare event causes:

- The timer counter to load the contents of the timer compare register.
- The timer output to toggle.
- Any configured transmit shift registers to load.
- Any configured receive shift registers to store.

Depending on the timer mode, the timer status flag may also be set.

43.3.2.6 Timer disable and stop functions

When the timer is configured to add a stop bit on each compare, the following additional events occur:

- Transmit shifters controlled by this timer output their stop bit value (if configured by SHIFTCFG*n*[SSTOP]).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SHIFTCFG*n*[SSTOP].
- The timer counter reloads the current value of the compare register on the first rising edge of the shifter clock after the compare.

If you configure the timer to insert a stop bit on each compare, you must configure the transmit shifters to load on the first shift.

When the condition configured by timer disable (TIMCFG*n*[TIMDIS]) is detected, the following events occur:

- Timer counter reloads the current value of the compare register and starts decrementing as configured by TIMCFG*n*[TIMDEC].

- Timer output clears. Shifters that are controlled by this timer do not see this as a falling edge on the timer shift clock, but can generate a shift event if the timer shift clock otherwise generates one.
- Transmit shifters controlled by this timer output their stop bit value (if configured by SHIFTCFG*n*[SSTOP]).
- Receive shifters controlled by this timer store the contents of the shift register in their shift buffer, as configured by SHIFTCFG*n*[SSTOP].

If the timer stop bit is enabled, the timer counter continues decrementing until the next rising edge of the shift clock is detected, at which point it finishes decrementing. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit. The timer output does not generate shift events during the stop bit.

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). When the timer is in the stop state condition, receive shift registers with stop bit enabled store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge. 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.

43.3.3 Pin operation

The pin configuration for each timer and shifter can be set to use any FLEXIO pin with either polarity. You can configure each timer and shifter 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 because the output enable assertion causes logic zero to be output on the pin) or to control the enable on the bidirectional output. You can configure any timer or shifter to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

43.3.3.1 Pin synchronization

When you configure a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized with the FLEXIO clock before a timer or shifter could use the signal. This introduces a small latency of 0.5–1.5 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

> FLEXIO pins are also connected internally. Configuring a FLEXIO shifter or timer to output data on an unused pin makes an internal connection that allows other shifters and timers to use this pin as an input. This allows a shifter output to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized with the FLEXIO clock and therefore incurs a one-cycle latency.

When using a pin input as a timer trigger, timer clock, or shifter data input, the following synchronization delays occur:

- 0.5–1.5 FLEXIO clock cycles for an external pin
- One FLEXIO clock cycle for an internally driven pin

See Application information for timing considerations such as output valid time and input setup time for specific applications (SPI controller, SPI target, I2C controller, I2S controller, and I2S target).

43.3.4 Low-power modes

FLEXIO remains functional during low-power modes, if CTRL[DOZEN] is 0 and the FLEXIO functional clock remains enabled.

43.3.5 Debug mode

FLEXIO remains functional in Debug mode, provided the value of CTRL[DBGE] is 1.

43.3.6 Clocking

Table 43-2. FLEXIO clocks

Clock	Description
Functional clock	Is asynchronous to the bus clock and can remain enabled in low-power modes. You must enable the FLEXIO functional clock before accessing any of the FLEXIO registers. Provided the FLEXIO functional clock is at least two times faster than the bus clock, you can configure CTRL[FASTACC] to support fast register accesses.
Bus clock	Is used only for bus accesses to the control and configuration registers.

43.3.7 Reset

Table 43-3. FLEXIO reset types

Reset	Description
Chip reset	Resets the FLEXIO logic and registers to their default states on chip reset.
Software reset	Resets, using CTRL[SWRST], all logic and registers to their default states, except for the Control register.

43.3.8 Interrupts and DMA requests

The following table shows the status flags that generate the FLEXIO interrupt and DMA requests.

Table 43-4. FLEXIO interrupts and DMA requests

Flag	Description	Interrupt	DMA request	Low-power wake-up
SHIFTSTAT[SSF]	Shifter status flag	Y	Y	Y
SHIFTERR[SEF]	Shifter error flag	Y	N	Y
TIMSTAT[TSF]	Timer status flag	Y	N	Y

43.3.9 Peripheral triggers

The connection between FLEXIO peripheral triggers and other peripherals is device-specific.

43.3.9.1 Output triggers

Each FLEXIO timer generates an output trigger equal to the timer output. The output trigger is not affected by the timer pin polarity configuration.

43.3.9.2 Input trigger

FLEXIO supports multiple external trigger inputs that can be used to trigger one or more FLEXIO timers. The external triggers are synchronized to the FLEXIO functional clock and must assert for at least two cycles of the FLEXIO functional clock to be sampled correctly.

43.4 External signals

Table 43-5. External signals

Signal	Description	Direction
FXIO_D*n* (*n* = 0...7)	Bidirectional FLEXIO shifter and timer pin	Input or output

43.5 Initialization

Perform the following procedure to initialize FLEXIO registers:

1. Enable FLEXIO by writing 1 to CTRL[FLEXEN].
2. Configure shift registers for the given application. It is recommended to write to Shifter Configuration (SHIFTCFG0 - SHIFTCFG3) before writing to the corresponding register, Shifter Control (SHIFTCTL0 - SHIFTCTL3).
3. Configure timer registers for the given application. It is recommended to write to Timer Compare (TIMCMP0 - TIMCMP3) and Timer Configuration (TIMCFG0 - TIMCFG3) before writing to the corresponding register, Timer Control (TIMCTL0 - TIMCTL3).
4. Enable interrupts and/or DMA requests, as appropriate, for the given application.
5. Write transmit data to initiate a transfer (depending on the given application).

43.6 Application information

This section provides examples for a variety of FLEXIO module applications. See FLEXIO register descriptions for more information.

43.6.1 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 are supported using the DMA controller. The timer status flag is 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, you must modify SHIFTCFG*n*[SSTART] and SHIFTCFG*n*[SSTOP] to transmit the required state, and the data to transmit must equal FFh or 00h. 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). When performing byte writes to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) (or Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) for transmitting MSB first), the rest of the register remains unaltered. This allows an address mark bit or additional stop bit to remain undisturbed.

NOTE

FLEXIO does not support automatic insertion of parity bits.

Table 43-6. UART transmit configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0. You can configure the PINPOL field to invert output data, or support open-drain by writing 1h to the PINPOL and PINCFG fields.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0000_2222h	Configure start bit, stop bit, enable on trigger asserted and disable on compare. You can support CTS by configuring the TIMENA field as 3h.
TIMCTL*n*	01C0_0001h	Configure the dual 8-bit counter using the shifter 0 status flag as an inverted internal trigger source. To support CTS, configure the PINSEL (for pin 1) and PINPOL fields as 1h.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the shifter status flag to indicate when data can be written using an interrupt or a DMA request. Write to SHIFTBUFBBS[7:0] instead to support MSB first transfer.

The following table shows an alternative configuration that supports slower baud rates. This configuration requires two timers.

Table 43-7. UART transmit configuration for slow baud rate

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data

Table continues on the next page...

Table 43-7. UART transmit configuration for slow baud rate (continued)

Register	Value	Configuration
		on pin 0. Invert output data by writing 1 to the PINPOL field. Support open-drain by configuring the PINPOL and PINCFG fields as 1h.
TIMCMP*n*	0000_000Fh	Configure for 8-bit transfer, and configure TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG*n*	0030_2622h	Configure start bit, stop bit, enable on trigger rising edge, decrement on trigger and disable on compare.
TIMCTL*n*	0740_0003h	Configure the 16-bit counter using the timer 1 output as an internal trigger source.
TIMCMP(*n* + 1)	0000_0001h	Configure baud rate of divide by 4 of the FLEXIO clock, and configure TIMCMP[15:0] as (baud rate divider ÷ 2) - 1.
TIMCFG(*n* + 1)	0000_1200h	Configure enable on trigger asserted and disable on timer 0 disable. You can configure the TIMEN field as 3h to support CTS.
TIMCTL(*n* + 1)	01C0_0003h	Configure the 16-bit counter using the shifter 0 status flag as an inverted internal trigger source. You can support CTS by configuring the PINSEL (for pin 1) and PINPOL fields as 1h.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer. Use the shifter status flag to indicate when data can be written using an interrupt or a DMA request. Write to SHIFTBUFBBS[7:0] instead to support MSB first transfer.

43.6.2 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 are supported using the DMA controller. The timer status flag is used to indicate when the stop bit of each word is received.

FLEXIO does not support triple voting of the received data, which is sampled only once in the middle of each bit. You can use a timer to implement a glitch filter on the incoming data and a different timer to detect an idle line of programmable length. Break characters cause the error flag to set, and the shifter buffer register returns 00h.

NOTE

FLEXIO does not support automatic verification of parity bits.

Table 43-8. UART receiver configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using timer 0 on the negative edge of clock with input data on pin 0. You can invert input data by writing 1 to the PINPOL field.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0204_2422h	Configure start bit, stop bit, enable on pin positive edge and disable on compare. Enable resynchronization to received data with TIMOUT = 2h and TIMRST = 4h.
TIMCTL*n*	0000_0081h	Configure the dual 8-bit counter using the inverted pin 0 input.
SHIFTBUF*n*	Data to receive	You can read received data from SHIFTBUFBYS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from SHIFTBUFBIS[7:0] instead to support MSB first transfer.

The UART receiver with RTS configuration uses a second timer to generate the RTS output. RTS asserts when the start bit is detected and negates when the data is read from the shifter buffer register. If no start bit is detected when the RTS is asserted, the received data is ignored.

Table 43-9. UART receiver with RTS configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Configure start bit of 0 and stop bit of 1.
SHIFTCTL*n*	0080_0001h	Configure receive using timer 0 on the negative edge of clock with input data on pin 0. Invert input data by writing 1 to the PINPOL field.
TIMCMP*n*	0000_0F01h	Configure 8-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0204_2522h	Configure start bit, stop bit, enable on pin positive edge with trigger asserted and disable on compare. Enable

Table continues on the next page...

Table 43-9. UART receiver with RTS configuration (continued)

Register	Value	Configuration
		resynchronization to received data with TIMOUT = 2h and TIMRST = 4h.
TIMCTL*n*	02C0_0081h	Configure dual 8-bit counter using the inverted pin 0 input. Trigger is internal using the inverted pin 1 input.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0030_6100h	Enable on timer *n* enable and disable on the trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(*n* + 1)	0143_0003h	Configure 16-bit counter and output on pin 1. Trigger is internal using the shifter 0 flag.
SHIFTBUF*n*	Data to receive	You can read received data using SHIFTBUFBYS[7:0]. Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from SHIFTBUFBIS[7:0] instead to support MSB first transfer.

43.6.3 SPI controller

SPI Controller mode can be supported using two timers, two shifters, and four pins. Using the DMA controller, either CPHA = 0 or CPHA = 1 and transfers can be supported. For CPHA = 1, the chip 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 one clock cycle between the target chip select negating and before the next transfer. To initiate each transfer, either the core or DMA writes to the transmit buffer.

NOTE

Because of synchronization delays, the setup time for the serial input data is 1.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 43-10. SPI controller (CPHA = 0) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using timer 0 on the negative edge of clock with output data on pin 0.

Table continues on the next page...

Table 43-10. SPI controller (CPHA = 0) configuration (continued)

Register	Value	Configuration
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0000_0101h	Configure receive using timer 0 on the positive edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare; initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using the pin 2 output (shift clock), with shifter 0 flag as the inverted trigger. Write 1 to the PINPOL field to invert the output shift clock.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0000_1100h	Enable when timer 0 is enabled and disable when timer 0 is disabled.
TIMCTL(*n* + 1)	0003_0383h	Configure 16-bit counter (never compare) using the inverted pin 3 output as target select.
SHIFTBUF*n*	Data to transmit	You can write transmit data to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

Table 43-11. SPI controller (CPHA = 1) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0021h	Start bit loads data on first shift.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the positive edge of clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.

Table continues on the next page...

Table 43-11. SPI controller (CPHA = 1) configuration (continued)

Register	Value	Configuration
SHIFTCTL(*n* + 1)	0080_0101h	Configure receive using timer 0 on the negative edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits x 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0100_2222h	Configure start bit, stop bit, enable on trigger high and disable on compare; initial clock state is logic 0.
TIMCTL*n*	01C3_0201h	Configure dual 8-bit counter using pin 2 output (shift clock), with the shifter 0 flag as the inverted trigger. Write 1 to the PINPOL field to invert the output shift clock, and set the TIMDIS field as 3 to keep target select asserted for as long as there is data in the transmit buffer.
TIMCMP(*n* + 1)	0000_FFFFh	Never compare.
TIMCFG(*n* + 1)	0000_1100h	Enable when timer 0 is enabled and disable when timer 0 is disabled.
TIMCTL(*n* + 1)	0003_0383h	Configure 16-bit counter (never compare) using inverted pin 3 output (as target select).
SHIFTBUF*n*	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. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

43.6.4 SPI target

SPI Target 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.

You must write the transmit data to the transmit buffer register before the external target select asserts; otherwise, the shifter error flag is set.

NOTE

Because of synchronization delays, the output valid time for the serial output data is 2.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

Table 43-12. SPI target (CPHA = 0) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0083_0002h	Configure transmit using timer 0 on the falling edge of shift clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0000_0101h	Configure receive using timer 0 on the rising edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG*n*	0120_0600h	Configure enable on trigger rising edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using pin 2 input (shift clock), with pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

Table 43-13. SPI target (CPHA = 1) configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0001h	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on rising edge of shift clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0080_0101h	Configure receive using timer 0 on falling edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_003Fh	Configure 32-bit transfer. Set TIMCMP[15:0] as (number of bits × 2) - 1).
TIMCFG*n*	0120_6602h	Configure start bit, enable on trigger rising edge, disable on trigger falling edge. Initial clock state is logic 0 and decrements on pin input.
TIMCTL*n*	06C0_0203h	Configure 16-bit counter using pin 2 input (shift clock), with pin 3 input (target select) as the inverted trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3) instead to support MSB first transfer.

43.6.5 I2C controller

I2C Controller mode can be supported using two timers, two shifters, and two pins. One timer is used to generate the SCL output and another one is used to control the shifters. The two shifters that are used to transmit and receive for every word, when receiving the transmitter, must transmit FFh to 3-state the output. FLEXIO inserts a stop bit after every word to generate and verify the ACK or 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 sets on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (start to repeated start or stop condition), so you must program the compare register with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin is equal to output. However, this increases both the clock high and clock low periods by at least one FLEXIO clock cycle each. The second timer uses the SCL input pin to control the transmit and receive shift registers. This enforces an SDA data hold time by an extra two FLEXIO clock cycles.

Both the transmit and receive shifters must be serviced for each word in the transfer. The transmit shifter must transmit FFh when receiving, and the receive shifter returns the data present on the SDA pin. The transmit shifter loads one additional word on the last falling edge of the SCL pin. When generating a stop condition or a repeated start condition, this word must be 00h and FFh, respectively. During the last word of a controller-receiver transfer, you must set the transmit SHIFTCFG*n*[SSTOP] field to generate a NACK.

The receive shift register asserts an error interrupt if a NACK is detected, but you are responsible for generating the stop or repeated start condition. If a NACK is detected during controller-transmit, the interrupt routine must immediately write 00h (when generating a stop condition) or FFh (when generating a repeated start condition) to the transmit shifter register. You must wait for the next rising edge on SCL before disabling both timers. The transmit shifter must be disabled after the setup delay for a repeated start or stop condition.

NOTE

Because of synchronization delays, the data valid time for the transmit output is two FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

To guarantee SDA hold time, the I2C controller data valid is delayed by two cycles because the clock output is passed through a synchronizer before clocking the transmit or receive shifter. Because the SCL output is synchronous with FLEXIO clock, the synchronization delay is one cycle, and then an additional cycle is involved to generate the output.

Table 43-14. I2C controller configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0032h	Start bit enabled (logic 0) and stop bit enabled (logic 1).

Table continues on the next page...

Table 43-14. I2C controller configuration (continued)

Register	Value	Configuration
SHIFTCTL*n*	0101_0082h	Configure transmit using timer 1 on the rising edge of clock with inverted output enable (open-drain output) on pin 0.
SHIFTCFG(*n* + 1)	0000_0020h	Start bit disabled and stop bit enabled (logic 0) for ACK or NACK detection.
SHIFTCTL(*n* + 1)	0180_0001h	Configure receive using timer 1 on the falling edge of clock with input data on pin 0.
TIMCMP*n*	0000_2501h	Configure 2 word transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of words × 18) + 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0102_2222h	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.
TIMCTL*n*	01C1_0101h	Configure dual 8-bit counter using pin 1 output enable (SCL open-drain), with the shifter 0 flag as the inverted trigger.
TIMCMP(*n* + 1)	0000_000Fh	Configure 8-bit transfer. Set TIMCMP[15:0] as (number of bits x 2) - 1.
TIMCFG(*n* + 1)	0020_1112h	Enable when timer 0 is enabled; disable when timer 0 is disabled. Enable start bit and stop bit at the end of each word and decrement on pin input.
TIMCTL(*n* + 1)	01C0_0183h	Configure 16-bit counter using inverted pin 1 input (SCL).
SHIFTBUF*n*	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.

43.6.6 I2S controller

I2S Controller 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 another 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 are supported using the DMA controller and the shifter error flag sets on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FLEXIO clock frequency. The initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure that the frame sync is generated one clock cycle before the first output data.

NOTE

Because of synchronization delays, the setup time for the receiver input is 1.5 FLEXIO clock cycles. This means that the maximum baud rate is divide by 4 of the FLEXIO clock frequency.

Table 43-15. I2S controller configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0001h	Load transmit data on first shift and stop bit disabled.
SHIFTCTL*n*	0003_0002h	Configure transmit using timer 0 on the rising edge of clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0080_0101h	Configure receive using timer 0 on the falling edge of clock with input data on pin 1.
TIMCMP*n*	0000_3F01h	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:8] as (number of bits × 2) - 1, and set TIMCMP[7:0] as (baud rate divider ÷ 2) - 1.
TIMCFG*n*	0000_0202h	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTL*n*	01C3_0281h	Configure dual 8-bit counter using inverted pin 2 output (bit clock), with shifter 0 flag as the inverted trigger. Write 0 to the PINPOL field to invert the polarity of the output shift clock.
TIMCMP(*n* + 1)	0000_007Fh	Configure 32-bit transfer with baud rate of divide by 4 of the FLEXIO clock. Set TIMCMP[15:0] as (number of bits × baud rate divider) ÷ 1.
TIMCFG(*n* + 1)	0000_0100h	Enable when timer 0 is enabled and never disable.
TIMCTL(*n* + 1)	0003_0383h	Configure 16-bit counter using inverted pin 3 output (as frame sync). Write 0 to the PINPOL field to invert the polarity of the output frame sync.

Table continues on the next page...

Table 43-15. I2S controller configuration (continued)

Register	Value	Configuration
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) instead to support LSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be read using interrupt or DMA request. Read from Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) instead to support LSB first transfer.

43.6.7 I2S target

I2S Target mode can be supported using three 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 is set.

NOTE

Because of synchronization delays, the output valid time for the serial output data is 2.5 FLEXIO clock cycles. This means the maximum baud rate is divide by 6 of the FLEXIO clock frequency.

The output valid time of I2S target is maximum 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization, plus one cycle to output the data.

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until the rising edge of bit clock (when the frame sync is normally sampled). Timer 0 detects the rising edge of bit clock with timer 2 output asserted and asserts output for length of frame. Timer 1 detects the falling edge of bit clock with timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 43-16. I2S target configuration

Register	Value	Configuration
SHIFTCFG*n*	0000_0000h	Start and stop bit disabled.
SHIFTCTL*n*	0103_0002h	Configure transmit using timer 1 on the rising edge of shift clock with output data on pin 0.
SHIFTCFG(*n* + 1)	0000_0000h	Start and stop bit disabled.
SHIFTCTL(*n* + 1)	0180_0101h	Configure receive using timer 1 on the falling edge of shift clock with input data on pin 1.
TIMCMP*n*	0000_007Fh	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] as (number of bits × 4) - 1.
TIMCFG*n*	0020_2500h	Configure enable on pin rising edge (inverted bit clock) with trigger high (timer 2) and disable on compare. Initial clock state is logic 1 and decrements on pin input (bit clock).
TIMCTL*n*	0B40_0203h	Configure 16-bit counter using pin 2 input (bit clock), with timer 2 output as the trigger.
TIMCMP(*n* + 1)	0000_003Fh	Configure 32-bit transfers. Set TIMCMP[15:0] as (number of bits × 2) - 1.
TIMCFG(*n* + 1)	0020_2500h	Configure enable on pin (bit clock) rising edge with trigger (timer 0) high and disable on compare. Initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTL(*n* + 1)	0340_0283h	Configure 16-bit counter using inverted pin 2 input (bit clock), with timer 0 output as the trigger.
TIMCMP(*n* + 2)	0000_0000h	Compare on zero (first edge).
TIMCFG(*n* + 2)	0020_6400h	Configure enable on inverted pin (frame sync) rising edge and disable on trigger falling edge (bit clock). Initial clock state is logic 1 and decrement on inverted pin input (frame sync).
TIMCTL(*n* + 2)	04C0_0383h	Configure 16-bit counter using inverted pin 3 input (frame sync), with pin 2 inverted input (bit clock) as the trigger.
SHIFTBUF*n*	Data to transmit	Transmit data can be written to Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be written using interrupt or DMA request. Write to the SHIFTBUF register instead to support LSB first transfer.
SHIFTBUF(*n* + 1)	Data to receive	Received data can be read from Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3). Use the shifter status flag to indicate when data can be read

Table 43-16. I2S target configuration

Register	Value	Configuration
		using interrupt or DMA request. Read from the SHIFTBUF register instead to support LSB first transfer.

43.7 Memory map and registers

43.7.1 FLEXIO register descriptions

NOTE

Invalid register accesses, which include reading a write-only register, writing to a read-only register, or accessing an invalid address, result in a bus error.

43.7.1.1 FLEXIO memory map

FLEXIO base address: 4005_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID (VERID)	32	R	0101_0000h
4h	Parameter (PARAM)	32	R	0408_0404h
8h	FLEXIO Control (CTRL)	32	RW	0000_0000h
Ch	Pin State (PIN)	32	R	0000_0000h
10h	Shifter Status (SHIFTSTAT)	32	RW	0000_0000h
14h	Shifter Error (SHIFTERR)	32	RW	0000_0000h
18h	Timer Status Flag (TIMSTAT)	32	RW	0000_0000h
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable (TIMIEN)	32	RW	0000_0000h
30h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	0000_0000h
80h - 8Ch	Shifter Control (SHIFTCTL0 - SHIFTCTL3)	32	RW	0000_0000h
100h - 10Ch	Shifter Configuration (SHIFTCFG0 - SHIFTCFG3)	32	RW	0000_0000h
200h - 20Ch	Shifter Buffer (SHIFTBUF0 - SHIFTBUF3)	32	RW	0000_0000h
280h - 28Ch	Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
300h - 30Ch	Shifter Buffer Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS3)	32	RW	0000_0000h
380h - 38Ch	Shifter Buffer Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS3)	32	RW	0000_0000h
400h - 40Ch	Timer Control (TIMCTL0 - TIMCTL3)	32	RW	0000_0000h
480h - 48Ch	Timer Configuration (TIMCFG0 - TIMCFG3)	32	RW	0000_0000h
500h - 50Ch	Timer Compare (TIMCMP0 - TIMCMP3)	32	RW	0000_0000h

43.7.1.2 Version ID (VERID)

43.7.1.2.1 Offset

Register	Offset
VERID	0h

43.7.1.2.2 Function

Indicates the version of FLEXIO.

43.7.1.2.3 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	0

43.7.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number Indicates the major version number of the module specification.
23-16	Minor Version Number

Table continues on the next page...

Field	Function
MINOR	Indicates the minor version number of the module specification.
15-0 FEATURE	Feature Specification Number Indicates the feature set number. 0000_0000_0000_0000b - Standard features implemented 0000_0000_0000_0001b - State, logic, and parallel modes supported

43.7.1.3 Parameter (PARAM)

43.7.1.3.1 Offset

Register	Offset
PARAM	4h

43.7.1.3.2 Function

Contains the number of shifters, timers, pins, and triggers.

43.7.1.3.3 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	0	0	1	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	0	1	0	0	0	0	0	0	0	1	0	0

43.7.1.3.4 Fields

Field	Function
31-24 TRIGGER	Trigger Number Indicates the number of external triggers implemented.
23-16 PIN	Pin Number Indicates the number of pins implemented.

Table continues on the next page...

Field	Function
15-8 TIMER	Timer Number Indicates the number of timers implemented.
7-0 SHIFTER	Shifter Number Indicates the number of shifters implemented.

43.7.1.4 FLEXIO Control (CTRL)

43.7.1.4.1 Offset

Register	Offset
CTRL	8h

43.7.1.4.2 Function

Controls various aspects of the FLEXIO operation.

43.7.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DOZEN	DBGE	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													FASTACC	SWRST	FLEXEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

43.7.1.4.4 Fields

Field	Function
31 DOZEN	Doze Enable Disables FLEXIO operation in Doze modes. 0b - Enable

Table continues on the next page...

Field	Function
	1b - Disable
30 DBGE	Debug Enable Enables the FLEXIO operation in Debug mode. 0b - Disable 1b - Enable
29-3 —	Reserved
2 FASTACC	Fast Access Configures fast or normal register accesses to FLEXIO registers, but requires the FLEXIO functional clock to be at least two times faster than the frequency of the bus clock. 0b - Normal 1b - Fast
1 SWRST	Software Reset Specifies whether software reset is enabled. The software reset does not affect this register but it affects all other logic in FLEXIO. All other register accesses are ignored until this field is cleared. The field remains 1 until software clears it and the reset has cleared in the FLEXIO clock domain. If you write 1 to this field, all FLEXIO registers except the Control register are reset. 0b - Disabled 1b - Enabled
0 FLEXEN	FLEXIO Enable Enables FLEXIO. 0b - Disable 1b - Enable

43.7.1.5 Pin State (PIN)

43.7.1.5.1 Offset

Register	Offset
PIN	Ch

43.7.1.5.2 Function

Indicates the status of the pin data input.

43.7.1.5.3 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								PDI							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

43.7.1.5.4 Fields

Field	Function
31-8 —	Reserved
7-0 PDI	Pin Data Input Indicates the input data on each of the FLEXIO pins.

43.7.1.6 Shifter Status (SHIFTSTAT)

43.7.1.6.1 Offset

Register	Offset
SHIFTSTAT	10h

43.7.1.6.2 Function

Contains shifter status flags.

43.7.1.6.3 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

43.7.1.6.4 Fields

Field	Function
31-4 —	Reserved
3-0 SSF	Shifter Status Flag Indicates the shifter status. This flag is updated in one of the following cases: • If SHIFTCTL*n*[SMOD] = 001b (Receive mode), the status flag is set when SHIFTBUF is loaded with data from the shifter (SHIFTBUF is full). The status flag is cleared when you read Shifter Buffer (SHIFTBUF0 - SHIFTBUF3). • If SHIFTCTL*n*[SMOD] = 010b (Transmit mode), the status flag is set when SHIFTBUF data is transferred to the shifter (SHIFTBUF is empty) or when SHIFTCTL*n*[SMOD] is initially configured as 010b (Transmit mode). The status flag is cleared when you write to the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 100b (Match Store mode), the status flag is set when a match occurs between SHIFTBUF and the shifter. The status flag is cleared when you read the SHIFTBUF register. • If SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the status flag returns the current match result between SHIFTBUF and the shifter. You cannot clear the status flag by reading the SHIFTBUF register. You can clear this status flag by writing a logic one to the flag for all modes except Match Continuous mode. **NOTE:** This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

43.7.1.7 Shifter Error (SHIFTERR)

43.7.1.7.1 Offset

Register	Offset
SHIFTERR	14h

43.7.1.7.2 Function

Reports shifter errors.

43.7.1.7.3 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

43.7.1.7.4 Fields

Field	Function
31-4 —	Reserved
3-0 SEF	Shifter Error Flag Indicates shifter error flag status. This flag is set when one of the following events occurs: • If SHIFTCTL*n*[SMOD] = 001b (Receive mode), it indicates that either the shifter is ready to store new data into SHIFTBUF before the previous data is read from SHIFTBUF (SHIFTBUF overrun), or the received start or stop bit does not match the expected value. • If SHIFTCTL*n*[SMOD] = 010b (Transmit mode), it indicates that the shifter is ready to load new data from SHIFTBUF before new data is written into SHIFTBUF (SHIFTBUF underrun). • If SHIFTCTL*n*[SMOD] = 100b (Match Store mode), it indicates the occurrence of a match event before the previous match data is read from SHIFTBUF (SHIFTBUF overrun). • If SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the error flag is set when a match occurs between SHIFTBUF and the shifter. For SHIFTCTL*n*[SMOD] = 101b (Match Continuous mode), the flag can also be cleared when you read Shifter Buffer (SHIFTBUF0 - SHIFTBUF3).

Field	Function
	NOTE: This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

43.7.1.8 Timer Status Flag (TIMSTAT)

43.7.1.8.1 Offset

Register	Offset
TIMSTAT	18h

43.7.1.8.2 Function

Reports timer status.

43.7.1.8.3 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

43.7.1.8.4 Fields

Field	Function
31-4 —	Reserved

Table continues on the next page...

Field	Function
3-0 TSF	Timer Status Flag Indicates timer status. This flag is set depending on Timer mode: • In 8-bit baud counter mode, this flag is set when the upper 8-bit counter equals zero and decrements. • In 8-bit high PWM mode, this flag is set when the upper 8-bit counter equals zero and decrements. • In 16-bit counter mode, this flag is set when the 16-bit counter equals zero and decrements. **NOTE:** This field behaves differently for register reads and writes. When reading 0000b - Clear 0001b - Set When writing 0000b - No effect 0001b - Clear the flag

43.7.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

43.7.1.9.1 Offset

Register	Offset
SHIFTSIEN	20h

43.7.1.9.2 Function

Enables shifter status interrupts.

43.7.1.9.3 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												SSIE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

43.7.1.9.4 Fields

Field	Function
31-4 —	Reserved
3-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when the corresponding SHIFTSTAT[SSF] flag is set. If you write 0 to this field, SHIFTSTAT[SSF] is disabled; and if you write 1 to this field, SHIFTSTAT[SSF] is enabled. 0b - Disable 1b - Enable

43.7.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

43.7.1.10.1 Offset

Register	Offset
SHIFTEIEN	24h

43.7.1.10.2 Function

Enables shifter error interrupts.

43.7.1.10.3 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												SEIE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

43.7.1.10.4 Fields

Field	Function
31-4	Reserved

Table continues on the next page...

Field	Function
—	
3-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when the corresponding SHIFTERR[SEF] flag is set. If you write 0 to this field, SHIFTERR[SEF] is disabled; and if you write 1 to this field, SHIFTERR[SEF] is enabled. 0b - Disable 1b - Enable

43.7.1.11 Timer Interrupt Enable (TIMIEN)

43.7.1.11.1 Offset

Register	Offset
TIMIEN	28h

43.7.1.11.2 Function

Enables timer status interrupts.

43.7.1.11.3 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												TEIE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

43.7.1.11.4 Fields

Field	Function
31-4 —	Reserved
3-0	Timer Status Interrupt Enable

Field	Function
TEIE	Enables interrupt generation when the corresponding TIMSTAT[TSF] flag is set. If you write 0 to this field, TIMSTAT[TSF] is disabled; and if you write 1 to this field, TIMSTAT[TSF] is enabled. 0b - Disable 1b - Enable

43.7.1.12 Shifter Status DMA Enable (SHIFTSDEN)

43.7.1.12.1 Offset

Register	Offset
SHIFTSDEN	30h

43.7.1.12.2 Function

Enables shifter DMA requests.

43.7.1.12.3 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

43.7.1.12.4 Fields

Field	Function
31-4 —	Reserved
3-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when the corresponding SHIFTSTAT[SSF] flag is set. If you write 0 to this field, SHIFTSTAT[SSF] is disabled; and if you write 1 to this field, SHIFTSTAT[SSF] is enabled.

Field	Function
	0b - Disable 1b - Enable

43.7.1.13 Shifter Control (SHIFTCTL0 - SHIFTCTL3)

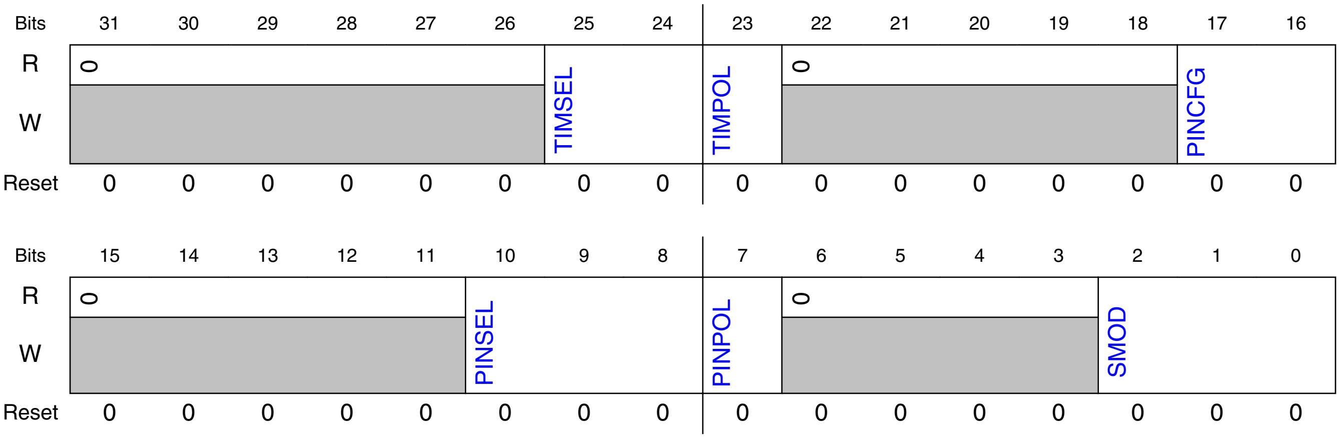
43.7.1.13.1 Offset

Register	Offset
SHIFTCTL0	80h
SHIFTCTL1	84h
SHIFTCTL2	88h
SHIFTCTL3	8Ch

43.7.1.13.2 Function

Provides shifter controls.

43.7.1.13.3 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

43.7.1.13.4 Fields

Field	Function
31-26 —	Reserved

Table continues on the next page...

Field	Function
25-24 TIMSEL	Timer Select Selects which timer is used for controlling the logic or shift register and generating the shift clock. TIMSEL = i selects TIMERi.
23 TIMPOL	Timer Polarity Determines whether the shift occurs on the positive edge or negative edge of the shift clock. 0b - Positive edge 1b - Negative edge
22-18 —	Reserved
17-16 PINCFG	Shifter Pin Configuration Specifies shifter pin configuration. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. **NOTE:** When initially configuring PINCFG as 11b, FLEXIO may briefly drive the pin low. To avoid this, you can configure PINCFG as 10b along with the rest of the Control register and then perform a subsequent write to set PINCFG as 11b. Likewise, when changing the value of PINCFG from 11b to 00b, you must perform an initial write to set PINCFG as 10b and then perform a subsequent write to update the rest of the Control register with the value of PINCFG as 00b. 00b - Shifter pin output disabled 01b - Shifter pin open-drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-11 —	Reserved
10-8 PINSEL	Shifter Pin Select Selects the pin that is used by the shifter input or output. PINSEL = i selects the FXIO_Di pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register.
7 PINPOL	Shifter Pin Polarity Specifies the shifter pin polarity. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to this register. 0b - Active high 1b - Active low
6-3 —	Reserved
2-0 SMOD	Shifter Mode Configures the mode of the shifter. 000b - Disable 001b - Receive mode; capture the current shifter content into 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 - Reserved 111b - Reserved

43.7.1.14 Shifter Configuration (SHIFTCFG0 - SHIFTCFG3)

43.7.1.14.1 Offset

Register	Offset
SHIFTCFG0	100h
SHIFTCFG1	104h
SHIFTCFG2	108h
SHIFTCFG3	10Ch

43.7.1.14.2 Function

Provides fields for shifter configuration.

43.7.1.14.3 Diagram

Bits	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

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			0	0		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

43.7.1.14.4 Fields

Field	Function
31-19 —	Reserved
18-16 —	Reserved
15-13 —	Reserved
12 —	Reserved
11-10	Reserved

Table continues on the next page...

Field	Function
—	
9 —	Reserved
8 INSRC	Input Source Selects the input source for the shifter. Configuring this field as 1 is not supported for the last shifter. 0b - Pin 1b - Shifter n+1 output
7 —	Reserved
6 —	Reserved
5-4 SSTOP	Shifter Stop Allows automatic stop bit insertion, if the selected timer has also enabled a stop bit, when SHIFTCTL*n*[SMOD] is 10b (Transmit mode). If SHIFTCTL*n*[SMOD] is 1b or 100b (Receive mode or Match Store mode), this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. 00b - Stop bit disabled for Transmitter, Receiver, and Match Store modes 01b - Stop bit disabled for Transmitter, Receiver, and Match Store modes; when timer is in stop condition, Receiver and Match Store modes store receive data on the configured shift edge 10b - Transmitter mode outputs stop bit value 0 in Match Store mode; if stop bit is not 0, Receiver and Match Store modes set error flag (when timer is in stop condition, these modes also store receive data on the configured shift edge) 11b - Transmitter mode outputs stop bit value 1 in Match Store mode; if stop bit is not 1, Receiver and Match Store modes set error flag (when timer is in stop condition, these modes also store receive data on the configured shift edge)
3-2 —	Reserved
1-0 SSTART	Shifter Start Allows automatic start bit insertion, if the selected timer has also enabled a start bit, when SHIFTCTL*n*[SMOD] is 10b (Transmit mode). If SHIFTCTL*n*[SMOD] = 1b (Receive mode) or 100b (Match Store mode), this field allows automatic start bit checking if the selected timer has also enabled a start bit. 00b - Start bit disabled for Transmitter, Receiver, and Match Store modes; Transmitter mode loads data on enable 01b - Start bit disabled for Transmitter, Receiver, and Match Store modes; Transmitter mode loads data on first shift 10b - Transmitter mode outputs start bit value 0 before loading data on first shift; if start bit is not 0, Receiver and Match Store modes set error flag 11b - Transmitter mode outputs start bit value 1 before loading data on first shift; if start bit is not 1, Receiver and Match Store modes set error flag

43.7.1.15 Shifter Buffer (SHIFTBUF0 - SHIFTBUF3)

43.7.1.15.1 Offset

Register	Offset
SHIFTBUF0	200h
SHIFTBUF1	204h
SHIFTBUF2	208h
SHIFTBUF3	20Ch

43.7.1.15.2 Function

Contains shift buffer data.

43.7.1.15.3 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

43.7.1.15.4 Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Contains the data to be matched with the shifter contents and is used for various other functions, depending on the setting of SHIFTCTL*n*[SMOD] : • If SHIFTCTL0[SMOD] is 1b (Receive mode), shifter data is transferred into SHIFTBUF at the expiration of the timer. You must read this register only when the corresponding SHIFTSTAT[SSF] flag is set, indicating that new shifter data is available. • If SHIFTCTL0[SMOD] is 10b (Transmit mode), SHIFTBUF data is transferred into the shifter before the timer begins. • If SHIFTCTL0[SMOD] is 100b (Match Store mode), SHIFTBUF[31:16] contains the data to be matched with the shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1 = mask, 0 = no mask). The match is checked when the timer expires. Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. You must read this register only when the corresponding shifter status flag is set, indicating that new shifter data is available. • If SHIFTCTL0[SMOD] is 101b (Match Continuous mode), SHIFTBUF[31:16] contains the data to be matched with the shifter contents, and SHIFTBUF[15:0] can be used to mask the match result (1 = mask, 0 = no mask).

43.7.1.16 Shifter Buffer Bit Swapped (SHIFTBUFBIS0 - SHIFTBUFBIS3)

43.7.1.16.1 Offset

Register	Offset
SHIFTBUFBIS0	280h
SHIFTBUFBIS1	284h
SHIFTBUFBIS2	288h
SHIFTBUFBIS3	28Ch

43.7.1.16.2 Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is bit-swapped.

43.7.1.16.3 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

43.7.1.16.4 Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register are bit-swapped. Reads return SHIFTBUF[0:31].

43.7.1.17 Shifter Buffer Byte Swapped (SHIFTBUFBYS0 - SHIFTBUFBYS3)

43.7.1.17.1 Offset

Register	Offset
SHIFTBUFBYS0	300h
SHIFTBUFBYS1	304h
SHIFTBUFBYS2	308h
SHIFTBUFBYS3	30Ch

43.7.1.17.2 Function

Contains Shifter Buffer (SHIFTBUF0 - SHIFTBUF3) content, but it is byte-swapped.

43.7.1.17.3 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

43.7.1.17.4 Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but reads or writes to this register are byte-swapped. Reads return {SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24]}.

43.7.1.18 Shifter Buffer Bit Byte Swapped (SHIFTBUFBBS0 - SHIFTBUFBBS3)

43.7.1.18.1 Offset

Register	Offset
SHIFTBUFBBS0	380h
SHIFTBUFBBS1	384h
SHIFTBUFBBS2	388h
SHIFTBUFBBS3	38Ch

43.7.1.18.2 Function

Contains the register data for Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), but it is bit-swapped within each byte.

43.7.1.18.3 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

43.7.1.18.4 Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Acts as an alias to Shifter Buffer (SHIFTBUF0 - SHIFTBUF3), except that reads or writes to this register are bit-swapped within each byte. Reads return {SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7]}.

43.7.1.19 Timer Control (TIMCTL0 - TIMCTL3)

43.7.1.19.1 Offset

Register	Offset
TIMCTL0	400h
TIMCTL1	404h
TIMCTL2	408h
TIMCTL3	40Ch

43.7.1.19.2 Function

Controls various settings for timer *n*.

43.7.1.19.3 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

43.7.1.19.4 Fields

Field	Function
31-28 —	Reserved
27-24 TRGSEL	Trigger Select Selects the trigger. The valid values for TRGSEL depend on the configuration of Parameter (PARAM) : • If TRGSRC = 1, the valid values for *n* depend on the settings of PARAM[PIN], PARAM[TIMER], and PARAM[SHIFTER]. • If TRGSRC = 0, the valid values for *n* depend on PARAM[TRIGGER]. See the chip-specific FLEXIO information for external trigger selection. **NOTE:** For a pin, *n* = 0 to 7, for a shifter, *n* = 0 to 3, and for a timer, *n* = 0 to 3. If TRGSRC = 0, configure the trigger selection as *n* = external trigger *n* input.

Table continues on the next page...

Field	Function
	If TRGSRC = 1, you can configure the internal trigger to select an input pin as 2×*n* = pin *n* input. If TRGSRC = 1, you can configure the internal trigger to select a shifter or timer signal as: • 4×*n* + 1 = shifter *n* status flag • 4×*n* + 3 = timer *n* trigger output Following are the values for expanded internal trigger selection (TRGSRC = 1): • 0000 = Pin 0 • 0001 = Shifter 0 flag • 0010 = Pin 1 • 0011 = Timer 0 trigger • 0100 = Pin 2 • 0101 = Shifter 1 flag • 0110 = Pin 3 • 0111 = Timer 1 trigger • ... • This continues up to pin 7, shifter 3, and timer 3.
23 TRGPOL	Trigger Polarity Specifies whether the trigger is active high or active low. 0b - Active high 1b - Active low
22 TRGSRC	Trigger Source Specifies whether the selected trigger source is external or internal. 0b - External 1b - Internal
21-18 —	Reserved
17-16 PINCFG	Timer Pin Configuration Configures the direction of the timer pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. **NOTE:** When you initially configure PINCFG as 11b, FLEXIO may briefly drive the pin low. To avoid this, configure PINCFG as 10b along with the rest of the Control register and then perform a subsequent write to set the value of PINCFG as 11b. Likewise, when changing the value of PINCFG from 11b to 00b, you must perform an initial write to set PINCFG as 10b, and then perform a subsequent write to update the rest of the Control register with PINCFG as 00b. 00b - Timer pin output disabled 01b - Timer pin open-drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output
15-11 —	Reserved
10-8 PINSEL	Timer Pin Select Selects the pin that is used by the timer input or output. PINSEL = i selects the FXIO_Di pin. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register.
7 PINPOL	Timer Pin Polarity Specifies the timer pin polarity. For pins configured as an output (PINCFG = 11b), this field takes effect when you write to the register. 0b - Active high 1b - Active low

Table continues on the next page...

Field	Function
6-2 —	Reserved
1-0 TIMOD	Timer Mode Specifies the timer mode: • In 8-bit baud counter mode, the lower 8 bits of the counter and compare register are used to configure the baud rate of the timer shift clock. The upper 8 bits are used to configure the shifter bit count. • In 8-bit PWM high mode, the lower 8 bits of the counter and compare register are used to configure the high period of the timer shift clock. 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 mode 10b - Dual 8-bit counters PWM high mode 11b - Single 16-bit counter mode

43.7.1.20 Timer Configuration (TIMCFG0 - TIMCFG3)

43.7.1.20.1 Offset

Register	Offset
TIMCFG0	480h
TIMCFG1	484h
TIMCFG2	488h
TIMCFG3	48Ch

43.7.1.20.2 Function

Controls various aspects of timer configuration.

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 (timer 0, for example).

NOTE

The pin and trigger level and edges specified in this register refer to the signal state after being modified by the settings of TIMCTL*n*[PINPOL] and TIMCTL*n*[TRGPOL]. For example, "trigger low" means that a trigger is actually at logic level 1 if TIMCTL*n*[TRGPOL] is 1 (active low).

43.7.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						TIMOUT		0		TIMDEC		0	TIMRST		
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	TIMDIS			0	TIMENA			0		TSTOP		0		TSTART	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

43.7.1.20.4 Fields

Field	Function
31-26 —	Reserved
25-24 TIMOUT	Timer Output Configures the initial state of the timer output and whether it is affected by the timer reset. 00b - Logic one when enabled; not affected by timer reset 01b - Logic zero when enabled; not affected by timer reset 10b - Logic one when enabled and on timer reset 11b - Logic zero when enabled and on timer reset
23-22 —	Reserved
21-20 TIMDEC	Timer Decrement Configures the source of the timer decrement and that of the shift clock. 00b - Decrement counter on FLEXIO clock; shift clock equals timer output 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 —	Reserved
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 only resets the lower 8 bits that configure the baud rate. In all other modes, the timer reset resets full 16 bits of the counter. 000b - Never reset timer 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

Table continues on the next page...

Field	Function
	111b - Timer reset on trigger rising or falling edge
15 —	Reserved
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 (upper 8 bits match and decrement) 011b - Timer disabled on timer compare (upper 8 bits match and decrement) 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 —	Reserved
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 —	Reserved
5-4 TSTOP	Timer Stop Specifies whether the stop bit is enabled. 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 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 - Disabled 01b - Enabled on timer compare 10b - Enabled on timer disable 11b - Enabled on timer compare and timer disable
3-2 —	Reserved
1 TSTART	Timer Start Specifies whether the start bit is enabled. If it is enabled, configured shifters output the contents of the start bit when the timer is enabled. The timer counter reloads from the compare register on the first rising edge of the shift clock. 0b - Disabled 1b - Enabled
0 —	Reserved

43.7.1.21 Timer Compare (TIMCMP0 - TIMCMP3)

43.7.1.21.1 Offset

Register	Offset
TIMCMP0	500h
TIMCMP1	504h
TIMCMP2	508h
TIMCMP3	50Ch

43.7.1.21.2 Function

Contains the timer compare value.

43.7.1.21.3 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 W	CMP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

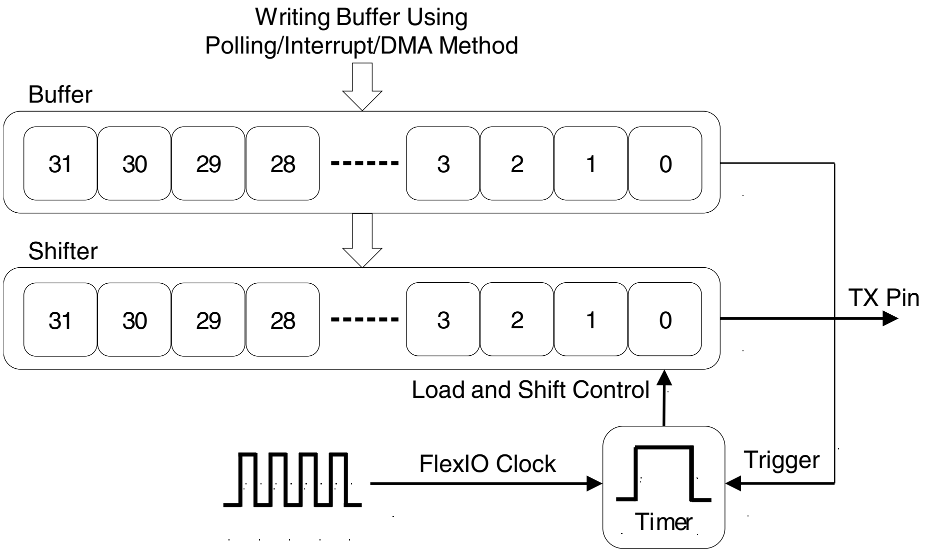
43.7.1.21.4 Fields

Field	Function
31-16 —	Reserved
15-0 CMP	Timer Compare Value Loads into the timer counter when the timer is first enabled, when the timer is reset, and when the timer decrements down to zero. In 8-bit baud counter mode, the lower 8 bits configure the baud rate divider as (CMP[7:0] + 1) × 2. The upper 8 bits configure the number of bits in each word as (CMP[15:8] + 1) ÷ 2. In 8-bit PWM high 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).

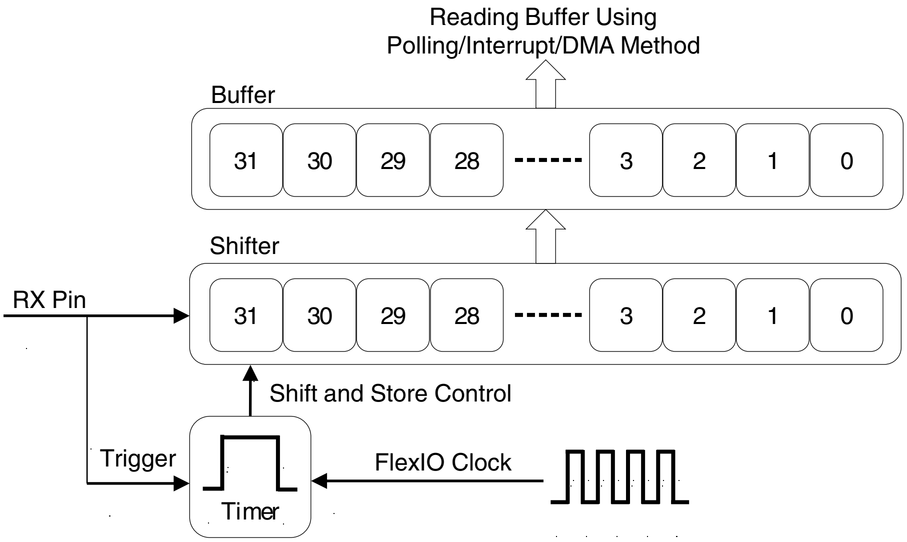
Field	Function
	In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) as (CMP[15:0] + 1) × 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word as (CMP[15:0] + 1) ÷ 2.

43.8 Usage Guide

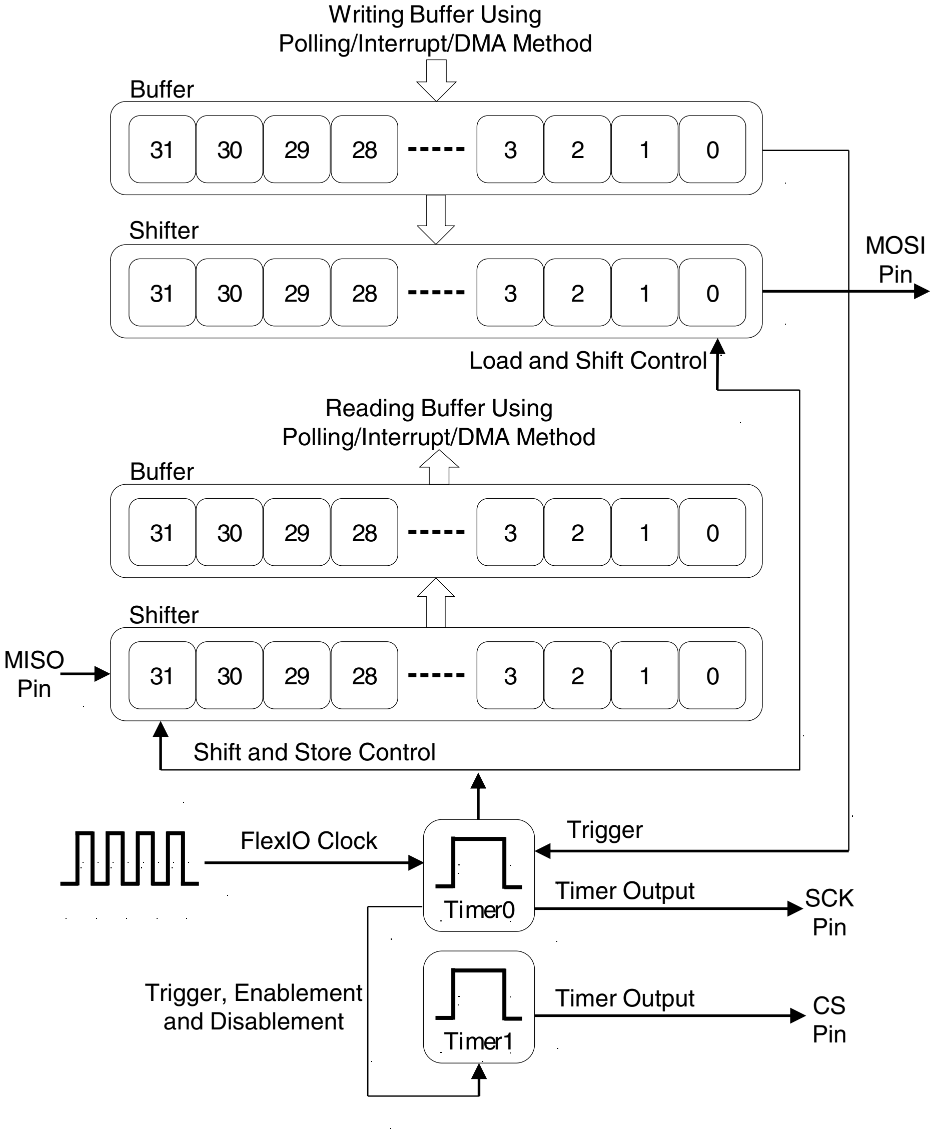
UART Transmit



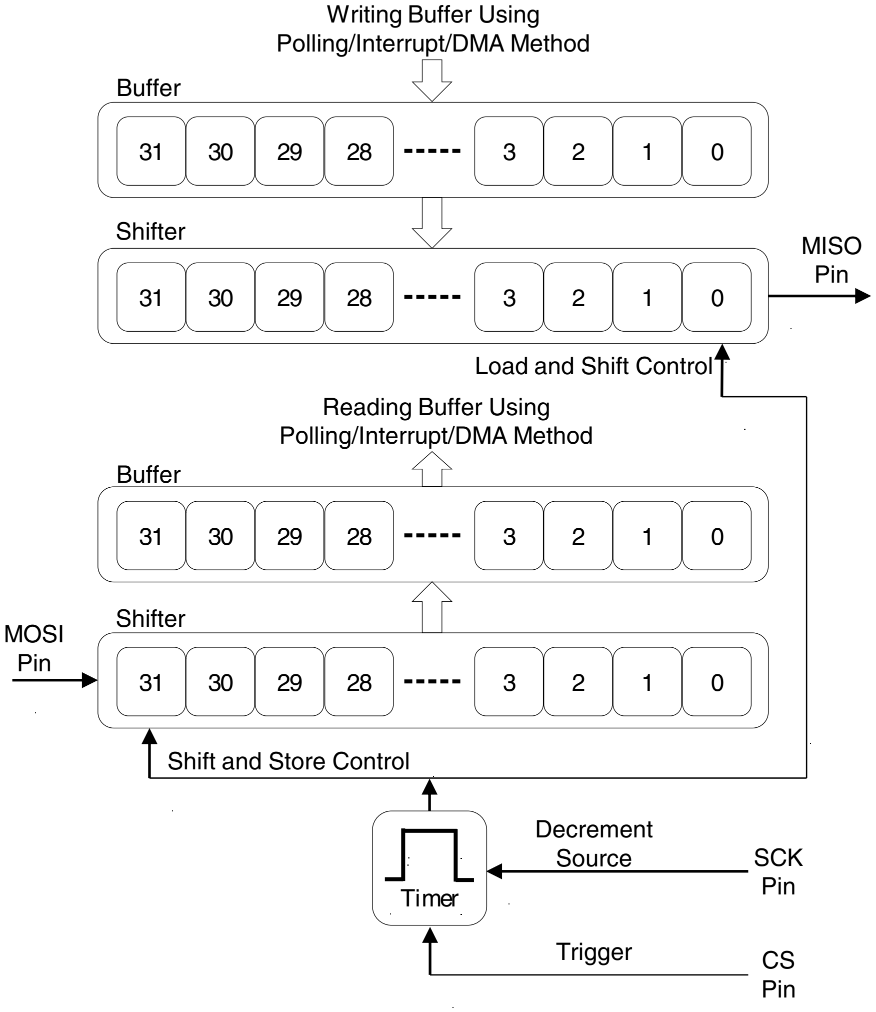
UART Receive



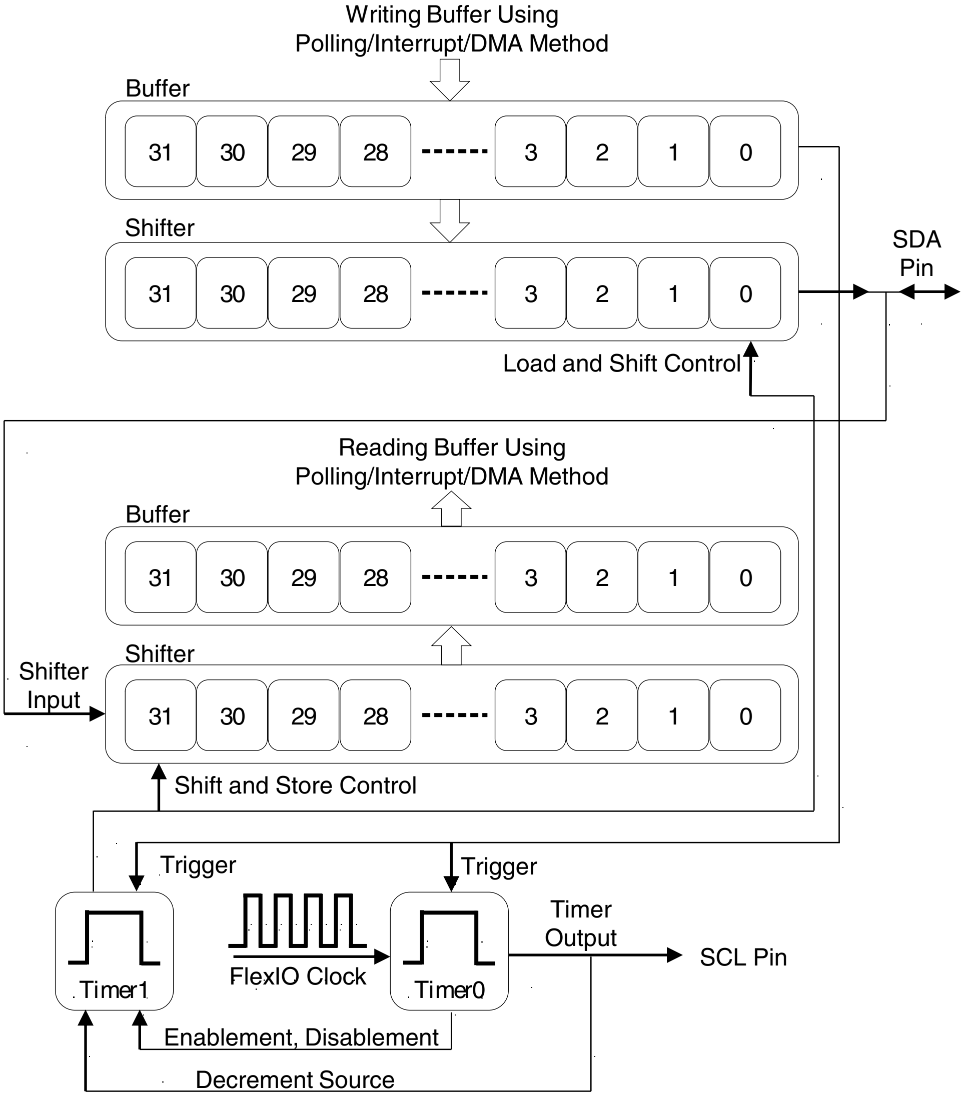
SPI Master


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
31
30
29
28
3
2
1
0
MOSI
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
MISO
Pin
31
30
29
28
3
2
1
0
Shift and Store Control
FlexIO Clock
Trigger
Timer0
Timer Output
SCK
Pin
Trigger, Enablement
and Disablement
Timer1
Timer Output
CS
Pin

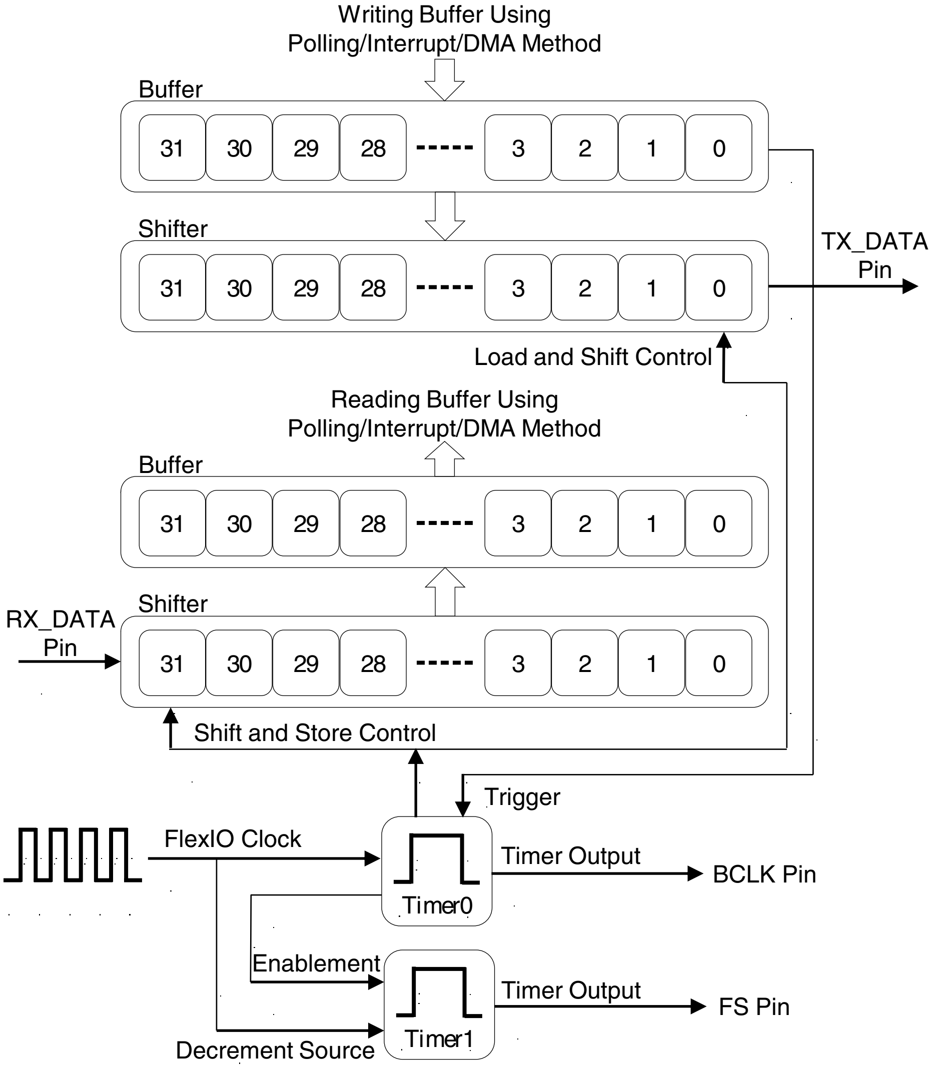
SPI Slave


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
31
30
29
28
3
2
1
0
MISO
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
MOSI
Pin
Shifter
31
30
29
28
3
2
1
0
Shift and Store Control
Timer
Decrement
Source
SCK
Pin
Trigger
CS
Pin

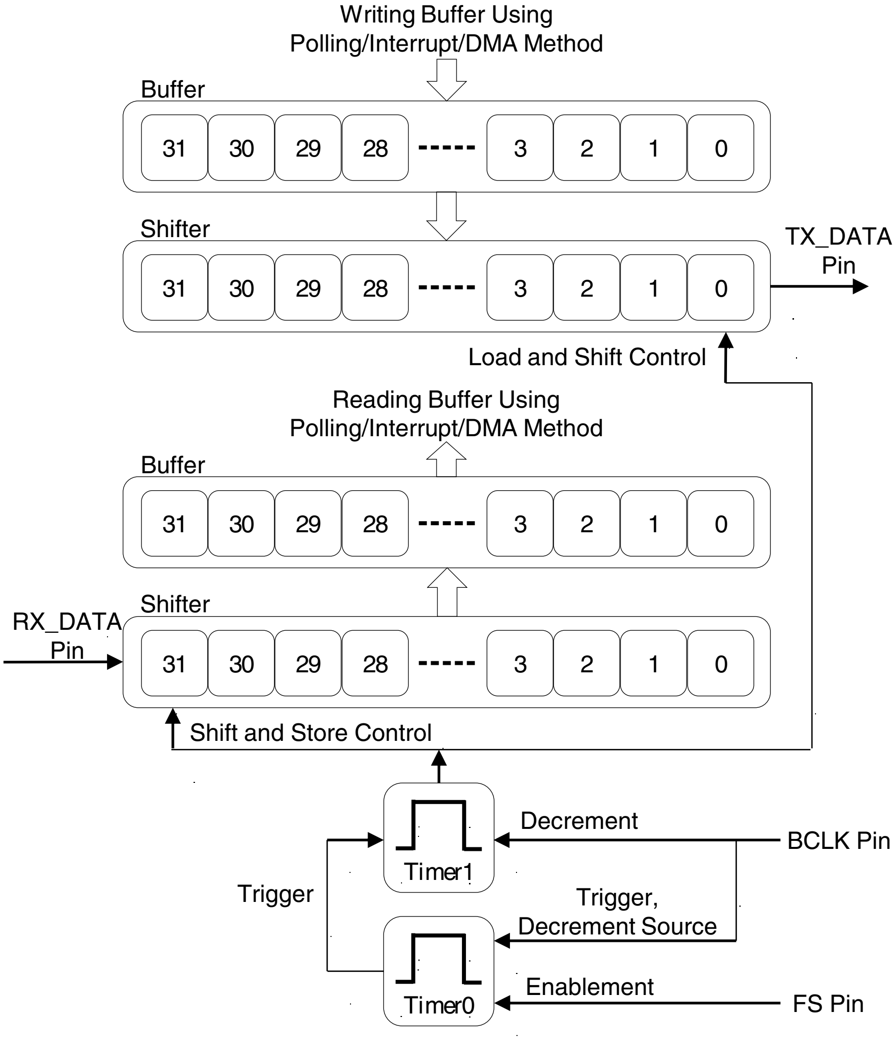
I2C Master


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
31
30
29
28
3
2
1
0
SDA
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
Shifter
Input
31
30
29
28
3
2
1
0
Shift and Store Control
Trigger
Trigger
Timer
Output
SCL Pin
Timer1
FlexIO Clock
Timer0
Enablement, Disablement
Decrement Source

I2S Master


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
Shifter
31
30
29
28
3
2
1
0
TX_DATA
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28
3
2
1
0
RX_DATA
Pin
Shifter
31
30
29
28
3
2
1
0
Shift and Store Control
Trigger
FlexIO Clock
Timer Output
BCLK Pin
Timer0
Enablement
Timer Output
FS Pin
Timer1
Decrement Source

I2S Slave


Writing Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28

3
2
1
0
Shifter
31
30
29
28

3
2
1
0
TX_DATA
Pin
Load and Shift Control
Reading Buffer Using
Polling/Interrupt/DMA Method
Buffer
31
30
29
28

3
2
1
0
Shifter
RX_DATA
Pin
31
30
29
28

3
2
1
0
Shift and Store Control
Decrement
BCLK Pin
Timer1
Trigger
Trigger,
Decrement Source
Timer0
Enablement
FS Pin

Chapter 44
Touch Sensing Input (TSI)

44.1 Chip-specific information for this module

44.1.1 Instantiation Information

Number of TSI module	2
Number of input channels	up to 6 Tx and 6 Rx channels for mutual-cap mode
	up to 25 touch channels for self-cap mode, any one or ones of them can be configured as shield channel
Support for low-power mode	one selectable pin is active

44.1.1.1 TSI module functionality in MCU operation modes

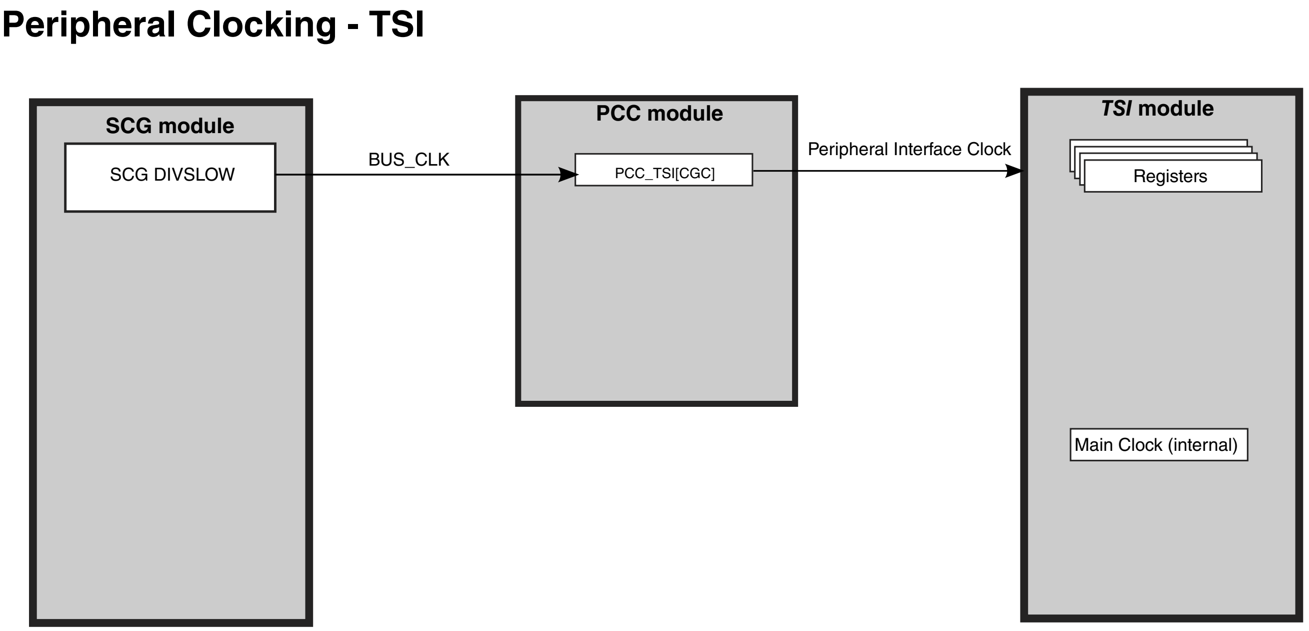
In the Stop or VLPS mode, only one TSI channel can be enabled as the wakeup source.

Table 44-1. TSI module functionality in MCU operation modes

MCU operation mode	TSI clock sources	TSI operation mode when GENCS[TSIEN] is 1	Functional electrode pins	Required GENCS[STPE] state
Run	BUS_CLK	Active mode	All	Do not care
Wait	BUS_CLK	Active mode	All	Do not care
Stop	Asynch operation	Stop mode	only 1	1
VLPR	BUS_CLK	Active mode	All	Do not care
VLPW	BUS_CLK	Active mode	All	Do not care
VLPS	Asynch operation	Stop mode	only 1	1

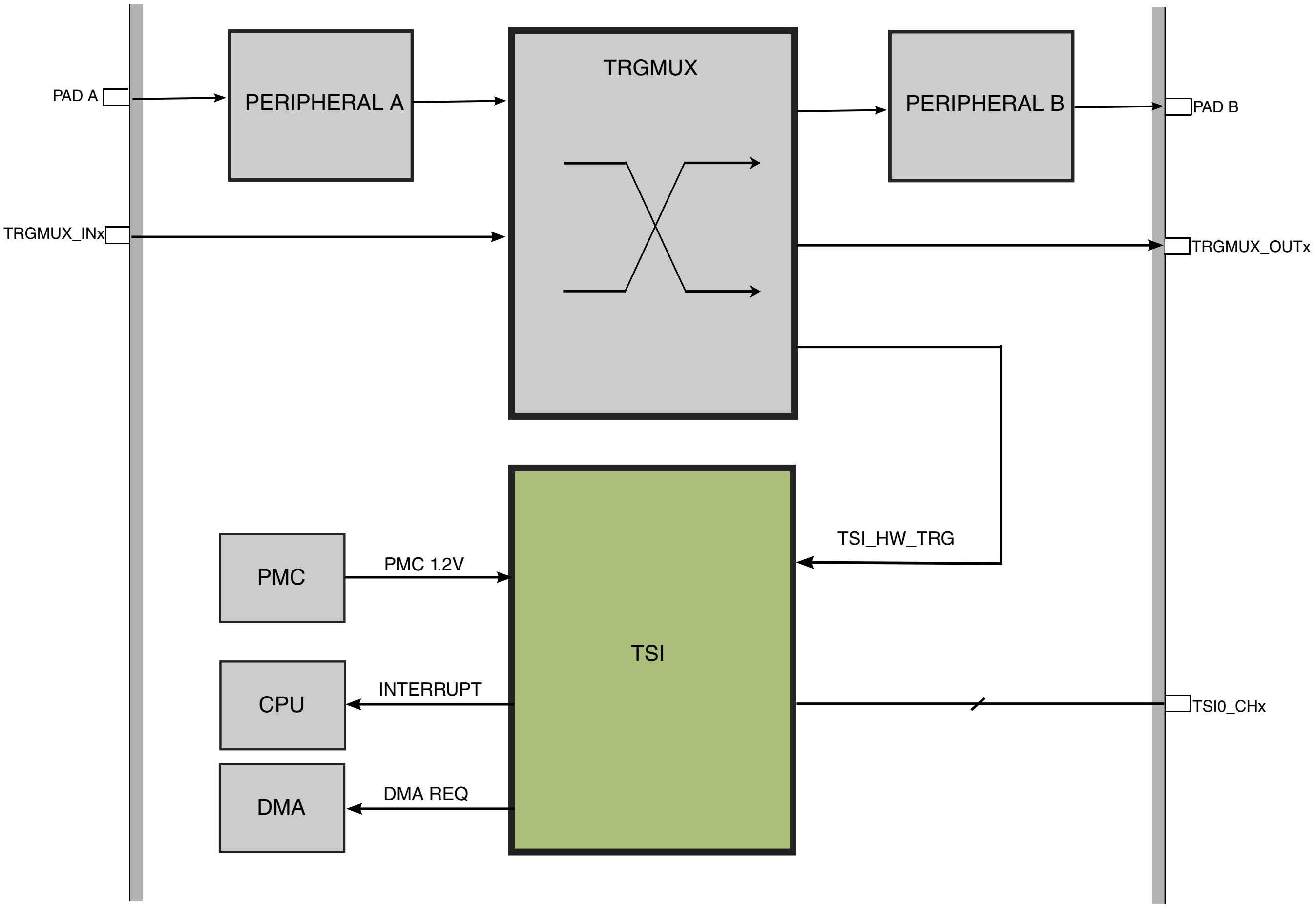
44.1.2 TSI Clocking Information

This following figure shows the TSI clocks.



44.1.3 Inter-connectivity Information

The TSI inter-connectivity is shown in the following diagram.



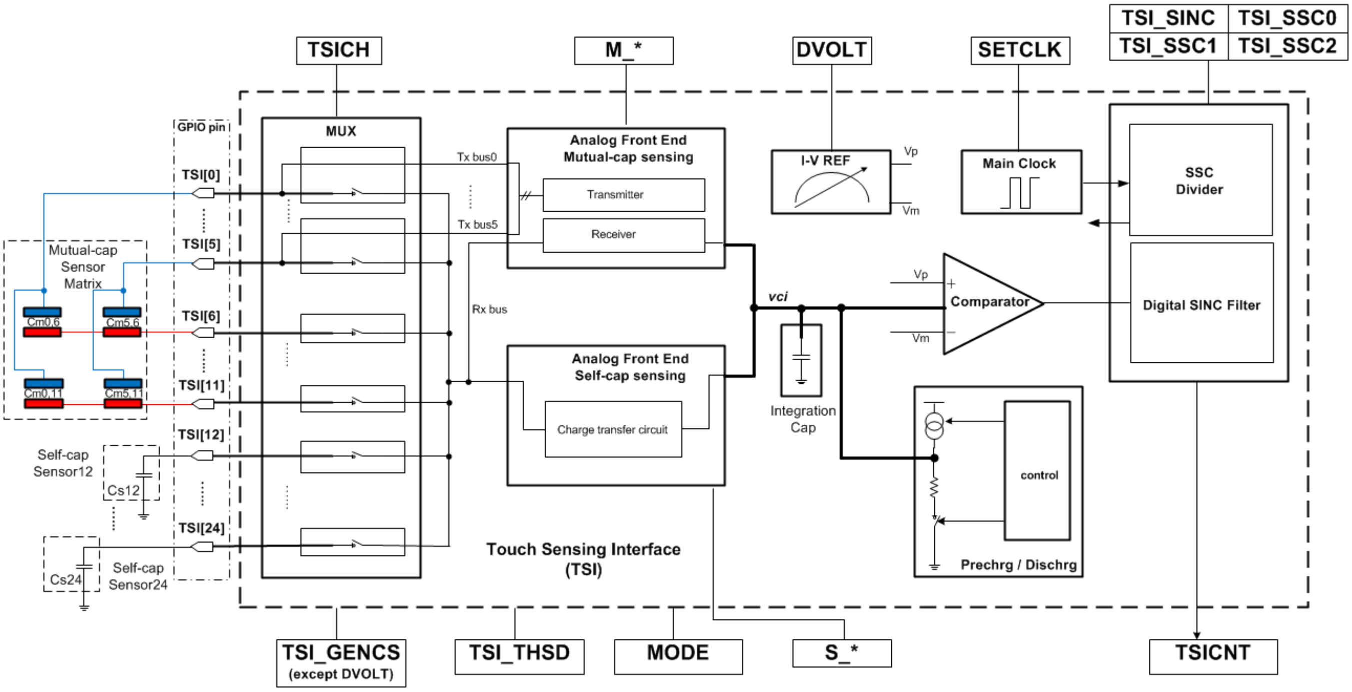
44.2 Overview

TSI provides touch sensing detection on capacitive touch sensors. The external capacitive touch sensor is typically formed on PCB and the sensor electrodes are connected to TSI input channels through the I/O pins in the device.

TSI operates in switching integration mode to achieve low-power, high-sensitivity and advanced EMC robustness. It supports both self-cap and mutual-cap sensors. In self-cap mode, TSI requires only one pin for each touch sensor. In mutual-cap mode, sensing is done using capacitive touch matrix in various Tx-Rx configurations. TSI requires one pin per Tx line and one pin per Rx line.

TSI fully supports NXP touch library which provides a solid capacitive measurement module to the implementation of touch keyboard, rotaries and sliders.

44.2.1 Block diagram



S_* stands for all registers whose name starts from S_. It controls self-cap sensing.
M_* stands for all registers whose name starts from M_. It controls mutual-cap sensing.

Figure 44-1. TSI block diagram

44.2.2 Features

TSI features are as follows:

- Advanced EMC robustness
- Supports both self-cap sensor and mutual-cap sensor
- One pin per electrode – no external components
- Adjustable touch sensing resolution and sensitivity for sensing a variety of overlay materials and thicknesses
- Low power consumption
- Capability to wake up MCU from low power modes for low power application
- Supports DMA data transfer
- Fully supports NXP touch library. See NXP touch library
- Each TSI channel configurable to function as the scan channel or shield channel

For electrode design recommendations, see AN3863: Designing Touch Sensing Electrodes

44.3 Functional description

44.3.1 Touch sensor

Self-cap touch sensor

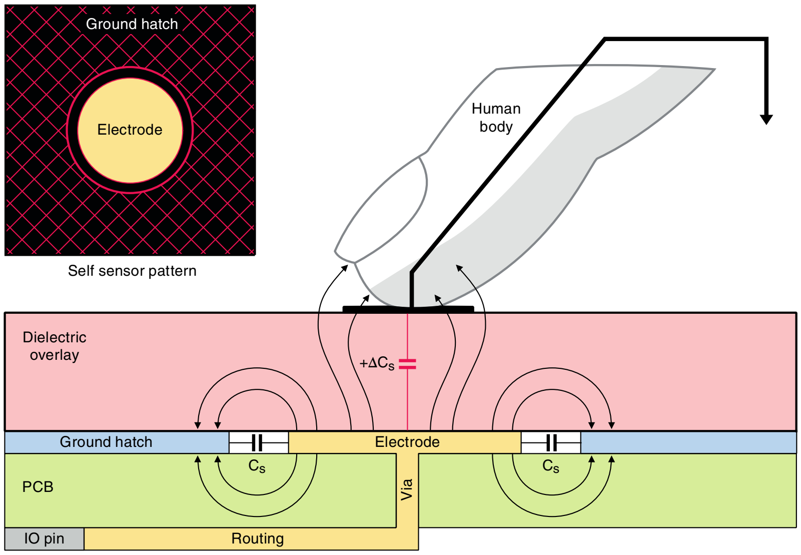


Figure 44-2. Self-cap touch sensor structure and electric field

Sensor structure:

- Cs: Intrinsic self capacitance. 10pF ~ 50pF as usual.
- ΔCs: Touch generated self capacitance. 0.3pF ~ 2pF as usual.
- Sensitivity of sensor: ΔCs/Cs. 1% ~ 10% as usual.

Intrinsic performance depends on: electrode pattern design, thickness/dielectric of overlay and PCB routing.

Mutual-cap touch sensor

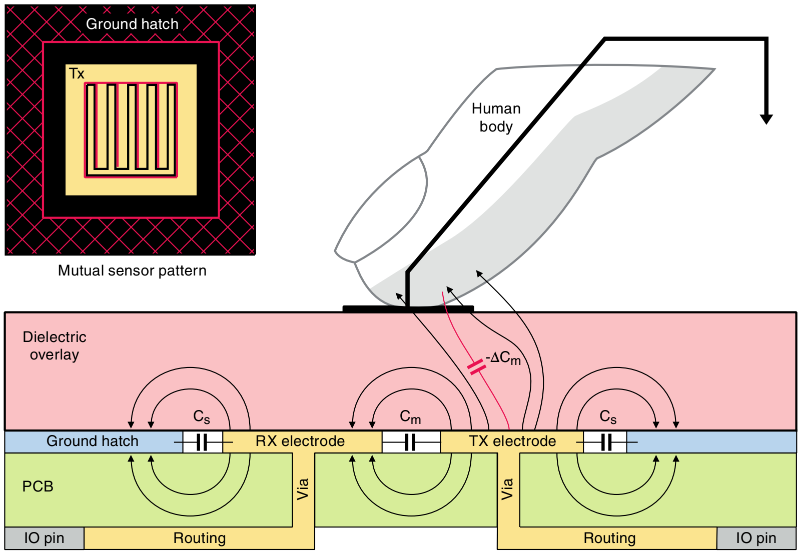


Figure 44-3. Mutual-cap touch sensor structure and electric field

Sensor structure:

- Cm: Intrinsic mutual cap. 2pF ~ 10pF as usual.
- ΔCm: Touch reduced mutual cap. 0.3pF ~ 2pF as usual.
- Cs: Parasitic self cap. 10pF ~ 50pF as usual.
- Sensitivity of sensor: ΔCm/Cm. 1% ~ 20% as usual.

Intrinsic performance depends on: electrode pattern design, thickness/dielectric of overlay and PCB routing.

44.3.2 Brief timing and operation

TSI works by switching integration, no matter under self-cap mode or mutual-cap mode. The difference of sensing modes is on analog processing.

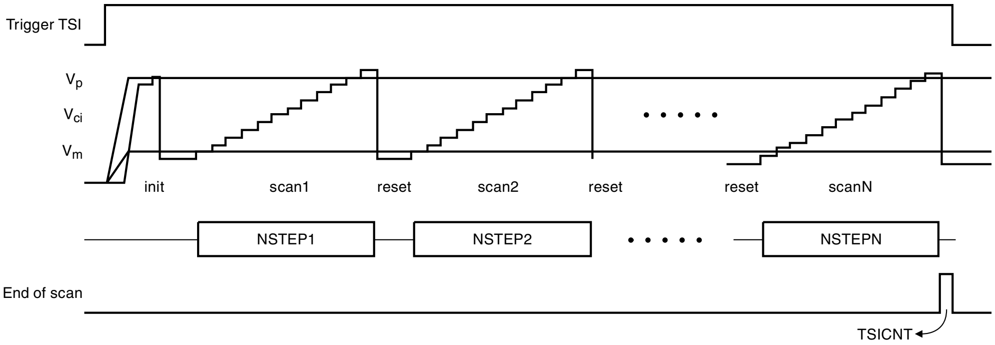


Figure 44-4. Brief timing of TSI

Formula

TSICNT = NSTEP x DECIMATION^{ORDER}

SCANTIME = TNSTEP x DECIMATION x ORDER

where:

DECIMATION: the times of scan, defined by IP configuration DECIMATION<4:0>.

ORDER: the order of sum up, defined by IP configuration ORDER.

NSTEP: the analog integration steps, decided by IP configurations and sensor.

TNSTEP: the scan time of getting each NSTEP.

SCANTIME: the total scan time of getting each TSICNT.

Operation

- TSI needs very short initialization time for each trigger, then starts to scan touch sensor.
- During scanning, analog front end senses self-cap/mutual-cap value and generates voltage steps on integration capacitor. The step voltage depends on touch sensor and IP configuration.
- Once the step voltage (VCI) reach threshold Vp of comparator, the integration cap and analog front end will be reset. The voltage VCI is discharged to Vm for next scanning.
- For each TSI trigger, there are many scan times which is set by registers. The step numbers of each scan are summed up together as final counts for software to use.
- The counts relate with touch sensor capacitance (self-cap/mutual-cap) through formulas and it can be used to sense touch event.

44.3.3 Self-cap sensing mode

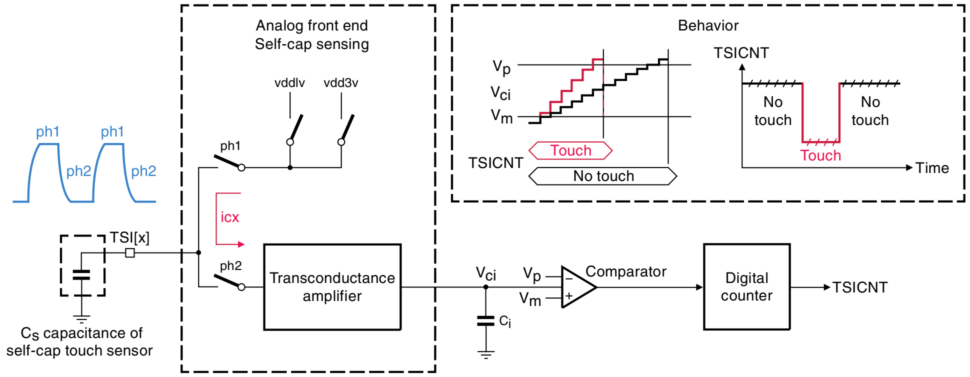


Figure 44-5. Self-cap sensing mode

Charge transfer operates through non-overlapping clock ph1/ph2 and trans-conductance amplifier. Charge accumulates in integration capacitor Ci which creates step voltage Vci.

The basic formula is given by

$$NSTEP = \frac{Ci \times (vp - vm)}{vdd3v \times Cs \times S\_XIN \times S\_XCH}$$

$$TNSTEP = \frac{Ci \times (vp - vm)}{vdd3v \times Cs \times S\_XIN \times S\_XCH} \times \frac{1}{F_{SW}}$$

where

Ci: is integration capacitance. Typical 90pF.

Vp, Vm: dual reference voltage which can be configured by TSI_GENCS[DVOLT].

Vdd3v: is analog power supply voltage. Typical 3.3V.

S_XIN, S_XCH: is parameter of analog front end which can be configured by TSI_MODE[S_XIN], TSI_MODE[S_XCH].

Fsw: is the switching frequency which is controlled by SSC (Spread Spectrum Clocking) block.

Cs: is the self-capacitance of touch sensor.

DVOLT, S_XIN, S_XCH can be used to adjust the sensing resolution.

If the touch sensor intrinsic sensitivity is limited due to parasitic, sensitivity boost feature can be activated by setting S_SEN. The formula is given by

$$NSTEP = \frac{Ci \times (vp - vm)}{vdd3v \times (Cs - S\_CTRIM * (S\_XDN / S\_XCH)) \times S\_XIN \times S\_XCH}$$

Where

S_CTRIM: is internal trim capacitance which can be configured by TSI_MODE[S_CTRIM].

S_XDN: is parameter of analog front end which can be configured by TSI_MODE[S_XDN].

S_CTRIM, S_XDN, S_XCH can be used to adjust the sensitivity. The intrinsic sensitivity of sensor is given by ΔCs/Cs. With this option, sensitivity can be improved to ΔCs/(Cs-S_CTRIM*(S_XDN/S_XCH)).

If touch sensor encounters strong low frequency noise, noise cancellation can be activated by setting S_NOISE. The formula is given by

$$NSTEP = \frac{2 \times Ci \times (vp - vm)}{(vdd3v - vddlv) \times Cx \times S\_XIN \times S\_XCH}$$

Where

Vddlv: is internal power supply voltage. Typical 1.2V.

During noise cancellation mode, vdd3v and vddlv are dual sample voltages. Analog front end samples twice which includes charging phase (sampling vdd3v) and discharging phase (sampling vddlv). At the end of each second phase, low frequency noise will be subtracted. In a long integration period, the noise induced error can be cancelled.

Example

In one typical case, Ci=90 pF, Cx=25 pF, vdd3v=3.3 V, DVOLT=1 V, S_XIN=1/8, S_XCH=1/8; Dec=8, Order=2.

Then,

NSTEP=69; TSICNT=4416; SCANTIME = 1.117 ms.

NOTE

Do not set S_SEN and S_NOISE at the same time.

44.3.4 Mutual-cap sensing mode

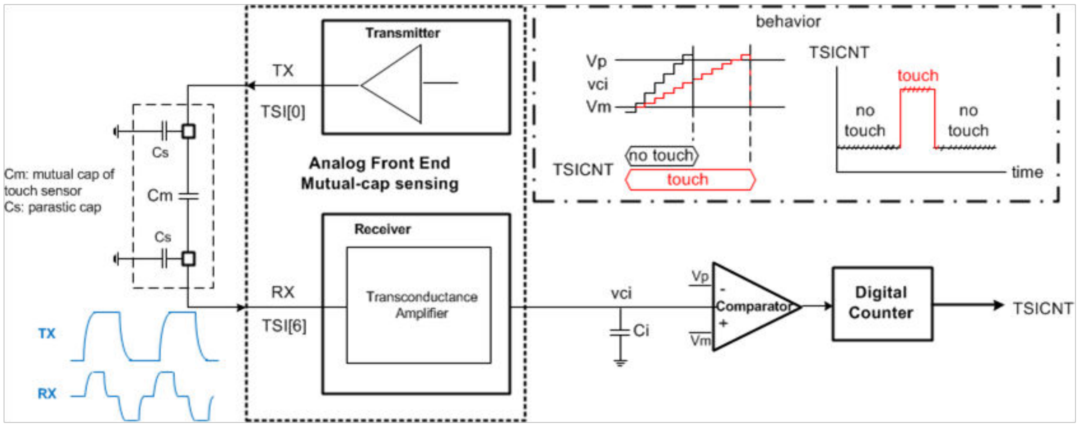


Figure 44-6. Mutual-cap sensing mode

Mutual-cap sensing includes transmitter and receiver. Under clocking, transmitter outputs pulses which couple through mutual cap then reach receiver. Receiver amplifies the signal and converts to charge current on integration cap Ci which creates step voltage Vci.

The formula is given by:

$$\mathrm{NSTEP} = \frac{\mathrm{Ci}\times(\mathrm{Vp}-\mathrm{Vm})\times\mathrm{Rs}}{\Delta\mathrm{V}} \times \frac{M\_\mathrm{PMIRRORL}}{M\_\mathrm{PMIRRORR}} \times \frac{1}{t3},$$

$$\mathrm{TNSTEP} = \frac{\mathrm{Ci}\times(\mathrm{Vp}-\mathrm{Vm})\times\mathrm{Rs}}{\Delta\mathrm{V}} \times \frac{M\_\mathrm{PMIRRORL}}{M\_\mathrm{PMIRRORR}} \times \frac{\mathrm{Tsw}}{t3},$$

where

Ci: is integration capacitance. Typical 90pF.

Vp, Vm: dual reference voltage which can be configured by TSI_GENCS[DVOLT].

Fsw: is the switching frequency which is controlled by SSC (Spread Spectrum Clocking) block.

Tsw: is the switching period, and Tsw =1/Fsw.

t3: is the SSC output low period.

Rs: is parameter of analog front end which can be configured by TSI_MUL0[M_SEN_RES].

M: is a parameter decided by TSI_MUL1[M_PMIRRORR] and TSI_MUL1[M_PMIRRORL].

ΔV: is signal voltage received. It is decided by

$$\Delta V = VDD5V \times \frac{Cm}{Cm + Cs}$$

which can be tens to hundreds of volts.

Cm, Cs: are the mutual capacitance and parasitic capacitance of sensor.

If the touch sensor intrinsic sensitivity is limited due to parasitic, sensitivity boost feature can be activated by setting TSI_MUL1[M_SEN_BOOST]. The basis average charge current will be subtracted by boost current which enlarge the signal current.

Example

In one typical case, ΔV=100 mV, Rs=10k, Vp-Vm=1 V, Ci=90 pF, M_PMIRRORL=8, M_PMIRRORR= M_NMIRROR= 2, Tsw=1 µs, t3=0.25 µs.

NSTEP=144, TNSTEP=144 µs.

Dec=8, Order=2, TSICNT=144 × 64 = 9216, SCANTIME=144 µs × 8 × 2 = 2304 µs.

NOTE

Keep M_PMIRRORR and M_NMIRROR the same.

44.3.5 Water shield

Shield electrode can reduce mis-trigger risk induced by water drop. A parasitic mutual cap is created between shield electrode (channel 12) and sensing electrode. In PH1, Cx is charged and Cm,shield is cleared. In PH2, Cm,shield shares some charge in Cx during transfer. So it induces TSI count increasing, which is an opposite trend comparing with normal touch – count decreasing.

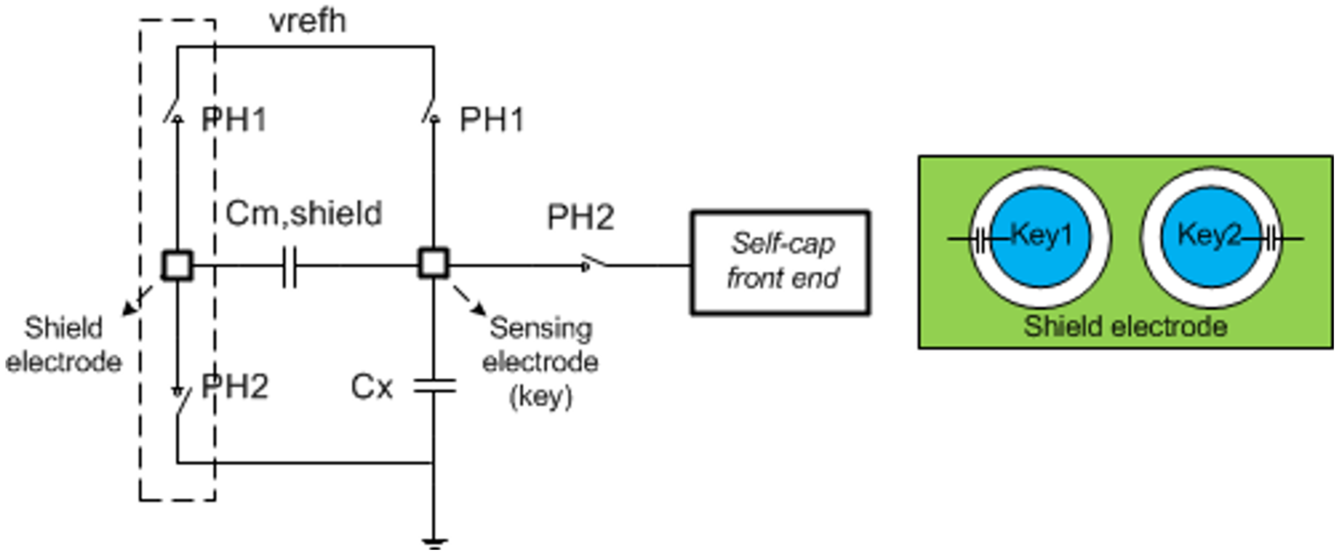


Figure 44-7. Self-cap touch key with water shield function

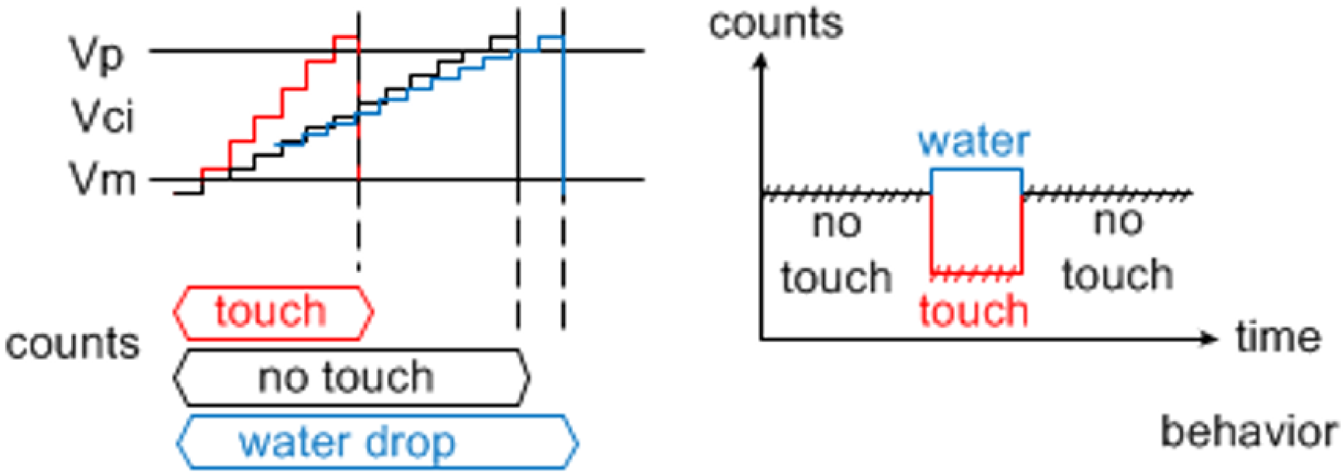


Figure 44-8. Self-cap mode count values of touch, no touch and water drop

Mutual-cap mode does not need the shield electrode. When there is a water drop between RX and TX, a parasitic cap is made between drive electrode and receive electrode. This enlarges the collected charge and reduces the count number. While the panel is touched, there is less coupling between drive electrode and receive electrode. This increases the count number. Therefore, water drops do not send out a mis-trigger in mutual-cap mode.

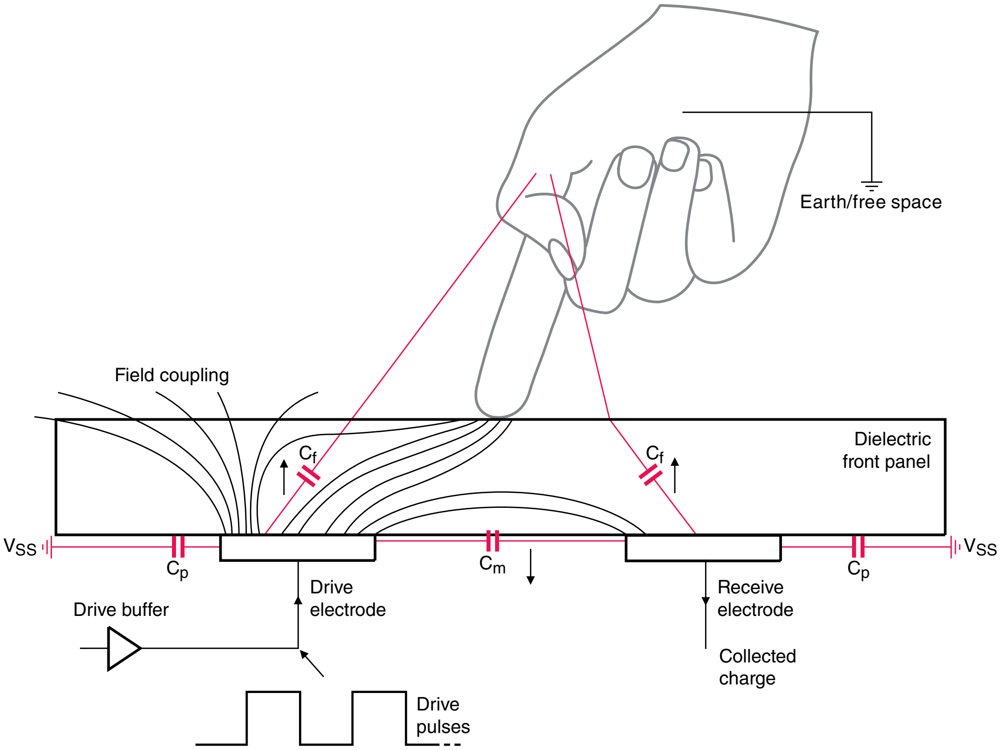


Figure 44-9. Mutual-cap touch key structure

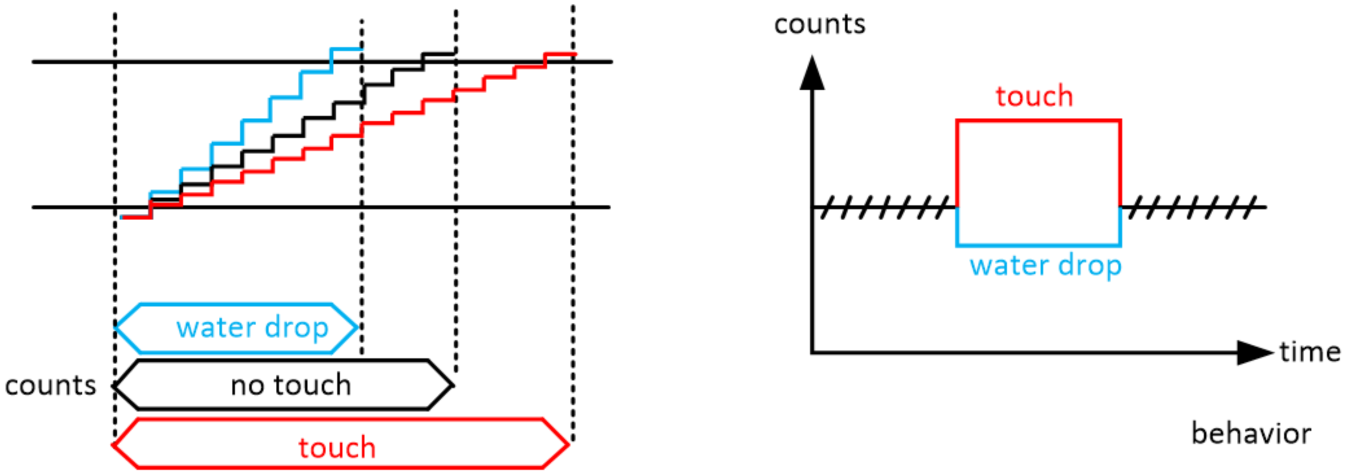


Figure 44-10. Mutual-cap mode count values of touch, no touch and water drop

44.3.6 Shield function

There are 25 TSI channels. In TSI self-cap mode , any one or ones of them can be configured as shield electrode. The more shied channels used, the bigger the shield driver strength there will be.

In all the 25 channels, channel 4, channel 12, channel 21 and channel 24 are designed stronger. If fewer shield pins are needed, these channels mentioned above are recommended.

See SHIELD[SHIELD_SELn] for detailed configuration.

44.3.7 Software and hardware trigger

TSI allows a software or hardware trigger to start a scan. When a software trigger is applied (i.e. TSI_GENCS[STM] bit is cleared), the TSI_DATA[SWTS] bit must be written to 1 to start the scan electrode channel that is identified by TSI_CONFIG. When a hardware trigger is applied (i.e. TSI_GENCS[STM] bit is set), TSI does not start scanning until the hardware trigger arrives. The hardware trigger is different depending on the MCU configuration. Generally, it can be an event that RTC overflows. See chip configuration section for details.

44.3.8 Scan times

TSI provides multi-scan function. The number of scans is indicated by TSI_SINC[DECIMATION] that allow the scan number from 1 to 32. When TSI_SINC[DECIMATION] is set to 0 (only once), the single scan is engaged. The 16-bit counter accumulates all scan results until the scan times reaches TSI_SINC[DECIMATION], 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.

44.3.9 Reference voltage

Reference voltage is used to setup ramp up threshold. It decides TSICNT and SCANTIME. TSI offers dual reference voltages for both comparators. 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 reference voltages are configurable upon the setting of TSI_GENCS[DVOLT]. The following table shows the all the delta voltage configurations.

Table 44-2. Delta voltage configuration

DVOLT	V_p (V)	V_m (V)	ΔV (V)
00	0.3	1.3	1.0

Table continues on the next page...

Table 44-2. Delta voltage configuration (continued)

DVOLT	V_p (V)	V_m (V)	ΔV (V)
01	0.3	1.6	1.3
10	0.3	1.9	1.6
11	0.3	2.3	2.0

44.3.10 End of scan

When a scan starts, TSI_GENCS[SCNIP] bit is set to indicate the scan is in progress. When the scan completes, the TSI_GENCS[EOSF] bit is set to indicate the scan is finished. Before clearing the TSI_GENCS[EOSF] bit, the value in TSI_DATA[TSICNT] must be read. If TSI_GENCS[TSIIEN] and TSI_GENCS[ESOR] are set, and TSI_DATA[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 fetches TSI conversion result from TSI_DATA register, stores it to other memory space, and refreshes the TSI scan channel index (TSI_DATA[TSICH]) for next loop. When DMA transfer is done, TSI_GENCS[EOSF] is cleared automatically.

44.3.11 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, 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.

44.3.12 Modes of operation

TSI operates in the following modes.

Table 44-3. TSI Operation modes

Mode	Description
Stop and low power stop	TSI 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 a scan completes, either end-of-scan or out-of-range interrupt can be selected to bring MCU out of low power modes.
Wait	TSI is fully functional in both modes. When a scan completes, TSI submits an interrupt request to CPU if the interrupt is enabled.
Run	

44.3.13 Clocking

44.3.13.1 Clock setting

Both of self-cap front end and mutual-cap front end are driven by switching clock with frequency Fsw. It comes from SSC clock with flatten emission energy. In addition, the frequency of switching clock can be configured by TSI_SSC0, TSI_SSC1 and TSI_SSC2 (Refer to chapter Spread spectrum clocking for details). The clock source of SSC is from Main Clock block in TSI. The frequency of main clock can be configured by SETCLK<1:0>.

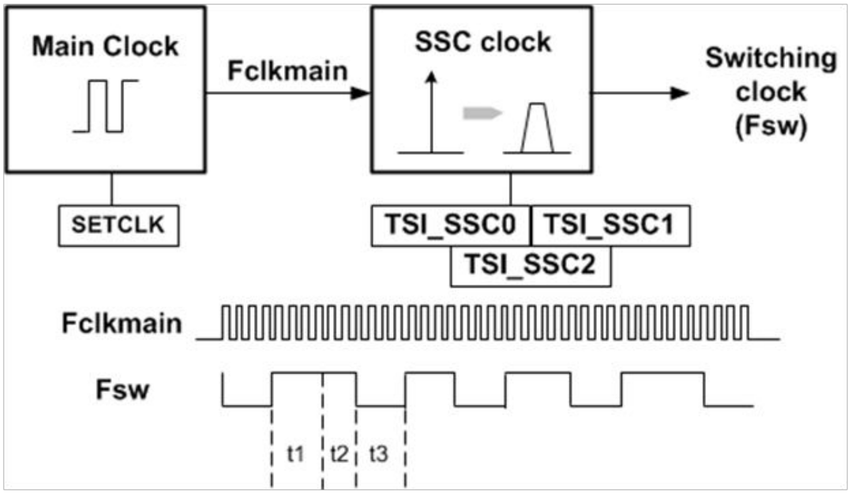


Figure 44-11. TSI clock

Example

- To use no SSC switching clock with frequency of 1MHz.
    - Set SETCLK<1:0> to ‘01’ to get Fclkmain = 16.65MHz.
    - Set SSC_MODE<1:0> to ‘10’ to disable SSC function.
    - Set SSC_PRESCALE_NUM<7:0> to ‘0000-0111’ to get division 16. When SSC mode is disabled, the frequency formula is Fclkmain/[(SSC_PRESCALE_NUM +1) × 2].
    - Keep other registers in TSI_SSC0, TSI_SSC1 and TSI_SSC2 as default value.
    - Then, Fsw = 16.65MHz/16 = 1.04MHz. Fsw is square wave pulse.
- To use PRBS mode SSC switching clock with central frequency of 1MHz.
    - Set SETCLK<1:0> to ‘01’ to get Fclkmain = 16.65MHz. Then the period of mainclock is Tclkmain.
    - Set SSC_MODE<1:0> to ‘00’ to enable PRBS SSC mode.
    - Set BASE_NOCHARGE_NUM<3:0> to ‘0100’ to set t1 = 5*Tclkmain*(SSC_PRESCALE_NUM + 1).
    - Set CHARGE_NUM<3:0> to ‘0110’ to set t3 = 7*Tclkmain*(SSC_PRESCALE_NUM + 1).
    - Set PRBS_OUTSEL<3:0> to ‘0110’ to set t2 range from 1* Tclkmain to 6* Tclkmain. The average t2 is 3.5* Tclkmain*(SSC_PRESCALE_NUM + 1).
    - Keep other registers in TSI_SSC0, TSI_SSC1 and TSI_SSC2 as default value.
    - Then, Fsw = 16.65MHz/[(5+3.5+7)*(0 + 1)] = 1.074MHz. Fsw is spectrum spread pulse.

44.3.13.2 Spread spectrum clocking

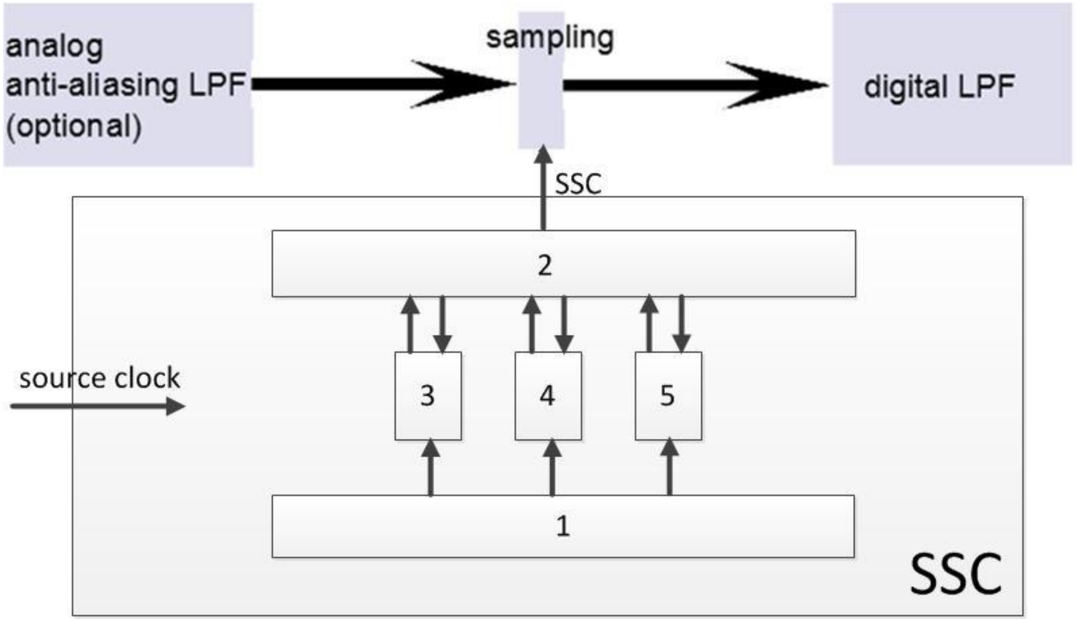


Figure 44-12. Spread spectrum clocking

In Capacitance touch sense systems, the baseband signal is narrow band and it is nearing the DC, while the noise is a wideband noise. For lower cost, the cut-off frequency of the anti-aliasing low pass filter at analog front end is not low enough comparing with the sampling frequency, so when sampling, the noises that frequency is nearing sampling frequency will be overlapped to baseband. For solving this problem in Capacitance touch sense systems, a low cost SSC can be involved. The SSC's center frequency and frequency span range should be flexible enough for handling various frequency noises.

With this SSC, the noises that frequency is nearing sampling frequency can be spanned to a wider frequency range instead of a single peak frequency noise, and only parts of the noise is overlapped into baseband (because baseband is a narrow band), so SNR can be promoted.

This SSC is composed of 5 components, they work together and generate the configurable center frequency and configurable span frequency range, they are:

1. Configurable registers, generating all configurable settings for component 3/4/5 using TSI_SSC0/ TSI_SSC1/ TSI_SSC2 registers;

2. State machine engine, controlling and monitoring the component 3/4/5;

3. A configurable counter for generating “1”, the max value of the counter is controlled by TSI_SSC0[BASE_NOCHARGE_NUM];

4. A configurable up-down counter or a PRBS method for generating “1”; If using up-down counter, the counter value is limited by TSI_SSC2[MOVE_NOCHARGE_MIN] and TSI_SSC2[MOVE_NOCHARGE_MAX]; If using PRBS method, the length of the "1" is controlled by the output of the PRBS method;

5. A configurable up-down counter for “0”, the max value of the counter is controlled by TSI_SSC0[CHARGE_NUM].

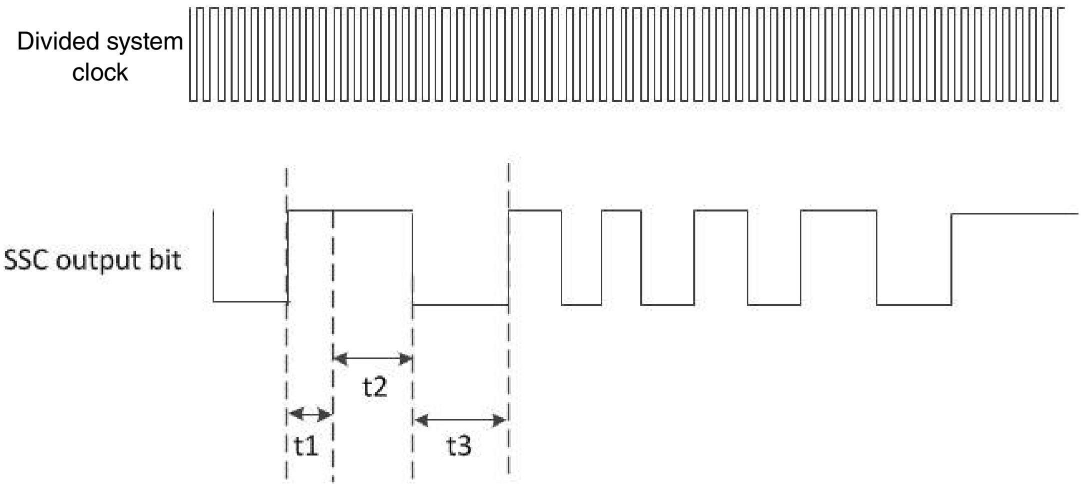


Figure 44-13. Spread spectrum clocking timing

The upper figure is presenting the timing of input system clock and the SSC output bit.

t1: controlled by component 3 and TSI_SSC0[BASE_NOCHARGE_NUM];

t2: controlled by component 4, and TSI_SSC2[MOVE_NOCHARGE_MIN] / TSI_SSC2[MOVE_NOCHARGE_MAX] if using up-down counter, or by PRBS output if using PRBS method;

t3: controlled by component 5 and TSI_SSC0[CHARGE_NUM];

So the average frequency of the SSC output bit will be:

$$frequency_{SSC} = \frac{frequency_{system}}{(t_1+t_2+t_3)*(SSC\_PRESCALE\_NUM+1)}$$

.

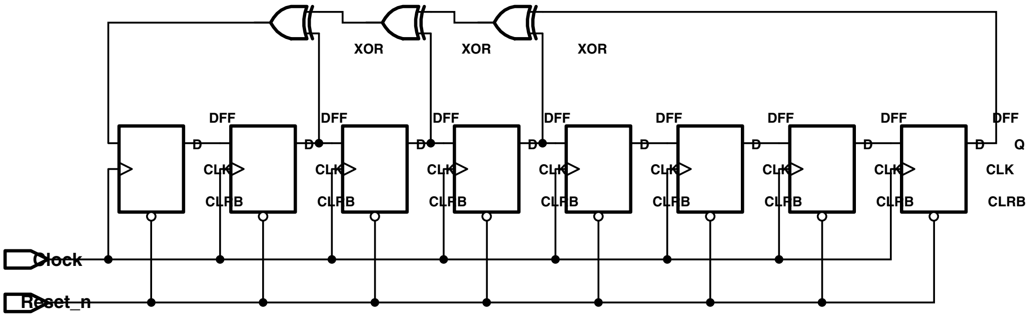


Figure 44-14. LFSR (Linear Feedback Shift Registers)

For using PRBS method generating the SSC output bit, LFSR circuit is involved for this implementation.

For the example in the figure, its eigenpolynomial is:

$$f(x) = 1 + x^2 + x^3 + x^4 + x^8 \quad .$$

44.3.14 Interrupts

If enabled, TSI scans 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 is 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.

The out-of-range interrupt does not occur:

- When the electrode capacitance does not vary in low power mode.
- When in noise detection mode.
- When the counter value reaches 0xFFFF and is treated as an extreme case.

44.3.15 DMA

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 transfer, 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.

44.4 External signals

TSI contains up to 25 external pins for touch sensing. The following tables list all of the external pins TSI uses for self-cap sensing or mutual-cap sensing.

Table 44-4. TSI signal description (self-cap sensing)

Name	Port	Direction	Function	Reset state
TSI[24:0]	TSI	I/O	TSI sensing pins or GPIO pins.	I/O

Table 44-5. TSI signal description (mutual-cap sensing)

Name	Port	Direction	Function	Reset state
TSI[5:0]	TSI	I/O	TSI Tx pins or GPIO pins.	I/O
TSI[11:6]	TSI	I/O	TSI Rx pins or GPIO pins.	I/O
TSI[24:12]	TSI	I/O	GPIO pins.	I/O

44.4.1 TSI[24:0]

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 parasitic capacity on board.

44.5 Initialization

To enable TSI:

1. Specify all the registers required to operate TSI, for example channel number, scan mode, and SCC mode, and etc.
2. Write 1 to the GENCS[TSIEN] bit.
3. Send a software or hardware trigger to TSI. See Software and hardware trigger for more information.
4. Read the scan counter value after the DATA[EOSF] bit becomes 1.

To disable TSI:

1. Write 0 to the GENCS[TSIEN] bit.

44.6 Register descriptions

This section describes the memory map and control/status registers for TSI.

44.6.1 TSI register descriptions

44.6.1.1 TSI memory map

TSI0 base address: 4004_5000h

TSI1 base address: 4004_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TSI general control and status register (GENCS)	32	RW	0000_0000h
4h	TSI DATA register (DATA)	32	RW	0000_0000h
8h	TSI threshold register (TSHD)	32	RW	0000_0000h
Ch	TSI MODE register (MODE)	32	RW	003C_0060h
10h	TSI MUTUAL-CAP Register 0 (MUL0)	32	RW	603F_6300h
14h	TSI MUTUAL-CAP register 1 (MUL1)	32	RW	0005_007Eh
18h	TSI SINC filter register (SINC)	32	RW	0007_0001h
1Ch	TSI SSC register 0 (SSC0)	32	RW	6032_0000h
20h	TSI SSC register 1 (SSC1)	32	RW	0060_0040h
24h	TSI SSC register 2 (SSC2)	32	RW	1008_0101h
28h	TSI shield register (SHIELD)	32	RW	0000_0000h

44.6.1.2 TSI general control and status register (GENCS)

44.6.1.2.1 Offset

Register	Offset
GENCS	0h

44.6.1.2.2 Function

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.

44.6.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OUTRGF	0		ESOR	0							DVOLT		0		
W	W1C															
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	TSI_ANA_TEST			RUN_CTRL	CLKSOC_SEL	0			TSIEN	TSIIEN	STPE	STM	SCNIP	EOSF	0	EOSDMEO
W														W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.6.1.2.4 Fields

Field	Function
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. It can be read once the CPU wakes. Write "1" , when this flag is set, to clear it.
30-29 —	Reserved
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. 0b - Out-of-range interrupt is allowed. 1b - End-of-scan interrupt is allowed.
27-21 —	Reserved
20-19 DVOLT	DVOLT select comparator Vm, Vp. From DIP. 00b - Vm=0.3V; Vp=1.3V; dvolt=1.0V. 01b - Vm=0.3V; Vp=1.6V; dvolt=1.3V. 10b - Vm=0.3V; Vp=1.9V; dvolt=1.6V. 11b - Vm=0.3V; Vp=2.3V; dvolt=2.0V.
18-16 —	Reserved
15-13 TSI_ANA_TEST	TSI_ANA_TEST These bits can only be accessed when in test mode .

Table continues on the next page...

Field	Function
12 RUN_CTRL	RUN_CTRL This bit can only be accessed when in test mode .
11 CLKSOC_SEL	CLKSOC_SEL This bit can only be accessed when in test mode .
10-8 —	Reserved
7 TSIEN	Touch Sensing Input Module Enable This bit enables TSI module. 0b - TSI module disabled. 1b - 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. 0b - TSI interrupt is disabled. 1b - TSI interrupt is enabled.
5 STPE	TSI STOP Enable This bit enables TSI module function in low power modes (stop, VLPS etc). 0b - TSI is disabled when MCU goes into low power mode. 1b - 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. 0b - Software trigger scan. 1b - 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. 0b - No scan in progress. 1b - 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. 0b - Scan not complete. 1b - Scan complete.
1 —	Reserved
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. 0b - 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. 1b - 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.

44.6.1.3 TSI DATA register (DATA)

44.6.1.3.1 Offset

Register	Offset
DATA	4h

44.6.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TSICH					0			DMAEN		0					
W										SWTS						
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	TSICNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.6.1.3.3 Fields

Field	Function
31-27 TSICH	TSICH These bits specify current channel to be measured for self-cap mode. 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. 0_0000b - For self-cap mode: Channel 0. 0_0001b - For self-cap mode: Channel 1. 0_0010b - For self-cap mode: Channel 2. 0_0011b - For self-cap mode: Channel 3. 0_0100b - For self-cap mode: Channel 4. 0_0101b - For self-cap mode: Channel 5. 0_0110b - For self-cap mode: Channel 6. 0_0111b - For self-cap mode: Channel 7. 0_1000b - For self-cap mode: Channel 8. 0_1001b - For self-cap mode: Channel 9. 0_1010b - For self-cap mode: Channel 10. 0_1011b - For self-cap mode: Channel 11. 0_1100b - For self-cap mode: Channel 12. 0_1101b - For self-cap mode: Channel 13. 0_1110b - For self-cap mode: Channel 14. 0_1111b - For self-cap mode: Channel 15. 1_0000b - For self-cap mode: Channel 16. 1_0001b - For self-cap mode: Channel 17.

Table continues on the next page...

Field	Function
	1_0010b - For self-cap mode: Channel 18. 1_0011b - For self-cap mode: Channel 19. 1_0100b - For self-cap mode: Channel 20. 1_0101b - For self-cap mode: Channel 21. 1_0110b - For self-cap mode: Channel 22. 1_0111b - For self-cap mode: Channel 23. 1_1000b - For self-cap mode: Channel 24.
26-24 —	Reserved
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. 0b - Interrupt is selected when the interrupt enable bit is set and the corresponding TSI events assert. 1b - 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 trigger start control. When the STM bit is cleared, write "1" to this bit will start a scan. Read value is always 0. 0b - No effect. 1b - Start a scan.
21-16 —	Reserved
15-0 TSICNT	TSI Conversion Counter Value These read-only bits record the accumulated scan counter value ticked by the reference oscillator.

44.6.1.4 TSI threshold register (TSHD)

44.6.1.4.1 Offset

Register	Offset
TSHD	8h

44.6.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	THRESH															
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	THRESL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.6.1.4.3 Fields

Field	Function
31-16 THRESH	TSI Wakeup Channel High-threshold This half-word specifies the high threshold of the wakeup channel.
15-0 THRESL	TSI Wakeup Channel Low-threshold This half-word specifies the low threshold of the wakeup channel.

44.6.1.5 TSI MODE register (MODE)

44.6.1.5.1 Offset

Register	Offset
MODE	Ch

44.6.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	S_XDN			0		Reserved			S_SEN	S_CTRIM			S_XIN	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	S_XCH			0					SETCLK		0			MODE	S_NOISE
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

44.6.1.5.3 Fields

Field	Function
31 —	Reserved
30-28 S_XDN	S_XDN When TSI_MODE[S_SEN]=1, adjust sensitivity. 000b - 1/16. 001b - 1/8. 010b - 1/4. 011b - 1/2. 100b - NA. 101b - NA. 110b - NA. 111b - NA.
27-26 —	Reserved
25-23 —	Reserved
22 S_SEN	S_SEN Sensitivity boost mode of self-cap. 0b - Sensitivity boost off. 1b - Sensitivity boost on.
21-19 S_CTRIM	Capacitor trim setting When TSI_MODE[S_SEN]=1, adjust sensitivity. 000b - Ctrim=2.5p. 001b - Ctrim=5.0p. 010b - Ctrim=7.5p. 011b - Ctrim=10p. 100b - Ctrim=12.5p. 101b - Ctrim=15p.

Table continues on the next page...

Field	Function
	110b - Ctrim=17.5p. 111b - Ctrim=20p.
18 S_XIN	S_XIN Input current multiple. 0b - 1/8. 1b - 1/4.
17-15 —	Reserved
14-12 S_XCH	S_XCH Charge/Discharge current multiple. 000b - 1/16. 001b - 1/8. 010b - 1/4. 011b - 1/2. 100b - NA. 101b - NA. 110b - NA. 111b - NA.
11-7 —	Reserved
6-5 SETCLK	SETCLK Set main clock frequency. 00b - 20.72MHz. 01b - 16.65MHz. 10b - 13.87MHz. 11b - 11.91MHz.
4-2 —	Reserved
1 MODE	MODE Select sensing mod. 0b - self-cap mode. 1b - mutual-cap mode.
0 S_NOISE	S_NOISE Noise cancellation mode of self-cap. 0b - noise cancellation off. 1b - noise cancellation on.

44.6.1.6 TSI MUTUAL-CAP Register 0 (MUL0)

44.6.1.6.1 Offset

Register	Offset
MUL0	10h

44.6.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M_PRE_CURRENT			0				Reserved	0		M_TX_USED					
W																
Reset	0	1	1	0	0	0	0	0	0	0	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M_PRE_RES			0	M_SEN_RES				0	M_SEL_TX			0	M_SEL_RX		
W																
Reset	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0

44.6.1.6.3 Fields

Field	Function
31-29 M_PRE_CURRENT	M_PRE_CURRENT Choose the current used in Vref generator, default 4uA. 000b - 1uA. 001b - 2uA. 010b - 3uA. 011b - 4uA. 100b - 5uA. 101b - 6uA. 110b - 7uA. 111b - 8uA.
28-25 —	Reserved
24 —	Reserved
23-22 —	Reserved
21-16 M_TX_USED	M_TX_USED Indicates which channel used for mutual cap TX. Value 1 means used for mutual cap; value 0 means used for GPIO.
15-13 M_PRE_RES	M_PRE_RES choose the resistor used in pre-charge, default 4k. 000b - 1k. 001b - 2k. 010b - 3k.

Table continues on the next page...

Field	Function
	011b - 4k. 100b - 5k. 101b - 6k. 110b - 7k. 111b - 8k.
12 —	Reserved
11-8 M_SEN_RES	M_SEN_RES Choose the resistor used in the I_sense generator, default 10k. 0000b - 2.5k. 0001b - 5k. 0010b - 7.5k. 0011b - 10k. 0100b - 12.5k. 0101b - 15k. 0110b - 17.5k. 0111b - 20k. 1000b - 22.5k. 1001b - 25k. 1010b - 27.5k. 1011b - 30k. 1100b - 32.5k. 1101b - 35k. 1110b - 37.5k. 1111b - 40k.
7 —	Reserved
6-4 M_SEL_TX	M_SEL_TX TX channel selection when TSI_MODE[MODE] = '1'. 000b - select channel 0 as tx0. 001b - select channel 1 as tx1. 010b - select channel 2 as tx2. 011b - select channel 3 as tx3. 100b - select channel 4 as tx4. 101b - select channel 5 as tx5. 110b - NA. 111b - NA.
3 —	Reserved
2-0 M_SEL_RX	M_SEL_RX RX channel selection when TSI_MODE[MODE] = '1'. 000b - select channel 6 as rx6. 001b - select channel 7 as rx7. 010b - select channel 8 as rx8. 011b - select channel 9 as rx9. 100b - select channel 10 as rx10. 101b - select channel 11 as rx11. 110b - NA. 111b - NA.

44.6.1.7 TSI MUTUAL-CAP register 1 (MUL1)

44.6.1.7.1 Offset

Register	Offset
MUL1	14h

44.6.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								M_SEN_BOOST					M_MODE	0	M_VPRE_CHOOSE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M_TRIM2								M_PMIRRORL			M_PMIRRORR		M_NMIRROR		M_NMIR_CTRL
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0

44.6.1.7.3 Fields

Field	Function
31-24 —	Reserved
23-19 M_SEN_BOOST	M_SEN_BOOST Choose the sensitivity boost current, default 0. 0_0000b - 0u. 0_0001b - 2u. 0_0010b - 4u. 0_0011b - 6u. 0_0100b - 8u.

Table continues on the next page...

Field	Function
	0_0101b - 10u. 0_0110b - 12u. 0_0111b - 14u 0_1000b - 16u. 0_1001b - 18u. 0_1010b - 20u. 0_1011b - 22u. 0_1100b - 24u. 0_1101b - 26u. 0_1110b - 28u. 0_1111b - 30u. 1_0000b - 32u. 1_0001b - 34u. 1_0010b - 36u. 1_0011b - 38u. 1_0100b - 40u. 1_0101b - 42u. 1_0110b - 44u. 1_0111b - 46u. 1_1000b - 48u. 1_1001b - 50u. 1_1010b - 52u. 1_1011b - 54u. 1_1100b - 56u. 1_1101b - 58u. 1_1110b - 60u. 1_1111b - 62u.
18 M_MODE	M_MODE TX drive mode control, default 0V~5V. 0b - -5V~+5V. 1b - 0V~+5V.
17 —	Reserved
16 M_VPRE_CHOOSE	M_VPRE_CHOOSE Digital control signal for pre-voltage choose. 0b - Internal 1.2V voltage. 1b - 1.2V PMC output.
15-8 M_TRIM2	M_TRIM2 M_TRIM2[7:0] is for trim use. For M_TRIM2[0], • value 0: choose Vref as source of Vp/Vm/Vmid; • value 1: choose Vpre in mutual AFE as source of Vp/Vm/Vmid. When this bit is set to 1, it will choose Vp/Vm/Vmid from a resistor divider from VDD5V to ground. Then it could help reduce variation on the power VDD5V. For M_TRIM2[6], • value 0: choose Vp-0.1V as Vmid; • value 1: choose Vp-0.4V as Vmid.
7-5 M_PMIRRORL	M_PMIRRORL PMOS current mirror on the left side, default m=16. 000b - m=4. 001b - m=8. 010b - m=12. 011b - m=16.

Table continues on the next page...

Field	Function
	100b - m=20. 101b - m=24. 110b - m=28. 111b - m=32.
4-3 M_PMIRRORR	M_PMIRRORR PMOS current mirror on the right side, default m=4. 00b - m=1. 01b - m=2. 10b - m=3. 11b - m=4.
2-1 M_NMIRROR	M_NMIRROR NMOS current mirror, default m=4. 00b - m=1. 01b - m=2. 10b - m=3. 11b - m=4.
0 M_NMIR_CTRL	M_NMIR_CTRL NMOS mirror control signal, default enable. 0b - Enable NMOS mirror. 1b - Disable NMOS mirror.

44.6.1.8 TSI SINC filter register (SINC)

44.6.1.8.1 Offset

Register	Offset
SINC	18h

44.6.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CUTOFF				0		ORDER	DECIMATION				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SWITCH_ENABLE	SINC_OVERFLOW_FLAG	SINC_VALID	SSC_CONTROL_OUT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

44.6.1.8.3 Fields

Field	Function
31-28 —	Reserved
27-24 CUTOFF	CUTOFF The value of shifting out lower bits of counter, equal to divide the result by div, default div=0. 0000b - div=1. 0001b - div=2. 0010b - div=4. 0011b - div=8. 0100b - div=16. 0101b - div=32. 0110b - div=64. 0111b - div=128. 1000b - NC. 1001b - NC. 1010b - NC. 1011b - NC. 1100b - NC. 1101b - NC 1110b - NC. 1111b - NC.
23-22	Reserved

Table continues on the next page...

Field	Function
—	
21 ORDER	ORDER Select the order of SINC filter, the SINC filter is a digital decimation filter for filtering out the low frequency noise from EMC (Electro Magnetic Compatibility). 0b - Using 1 order SINC filter. 1b - Using 2 order SINC filter.
20-16 DECIMATION	DECIMATION Choose the decimation value of the SINC filter. 0_0000b - The TSI_DATA[TSICNT] bits is the counter value of 1 scan period. 0_0001b - The TSI_DATA[TSICNT] bits is the counter value of 2 scan periods. 0_0010b - The TSI_DATA[TSICNT] bits is the counter value of 3 scan periods. 0_0011b - The TSI_DATA[TSICNT] bits is the counter value of 4 scan periods. 0_0100b - The TSI_DATA[TSICNT] bits is the counter value of 5 scan periods. 0_0101b - The TSI_DATA[TSICNT] bits is the counter value of 6 scan periods. 0_0110b - The TSI_DATA[TSICNT] bits is the counter value of 7 scan periods. 0_0111b - The TSI_DATA[TSICNT] bits is the counter value of 8 scan periods. 0_1000b - The TSI_DATA[TSICNT] bits is the counter value of 9 scan periods. 0_1001b - The TSI_DATA[TSICNT] bits is the counter value of 10 scan periods. 0_1010b - The TSI_DATA[TSICNT] bits is the counter value of 11 scan periods. 0_1011b - The TSI_DATA[TSICNT] bits is the counter value of 12 scan periods. 0_1100b - The TSI_DATA[TSICNT] bits is the counter value of 13 scan periods. 0_1101b - The TSI_DATA[TSICNT] bits is the counter value of 14 scan periods. 0_1110b - The TSI_DATA[TSICNT] bits is the counter value of 15 scan periods. 0_1111b - The TSI_DATA[TSICNT] bits is the counter value of 16 scan periods. 1_0000b - The TSI_DATA[TSICNT] bits is the counter value of 17 scan periods. 1_0001b - The TSI_DATA[TSICNT] bits is the counter value of 18 scan periods. 1_0010b - The TSI_DATA[TSICNT] bits is the counter value of 19 scan periods. 1_0011b - The TSI_DATA[TSICNT] bits is the counter value of 20 scan periods. 1_0100b - The TSI_DATA[TSICNT] bits is the counter value of 21 scan periods. 1_0101b - The TSI_DATA[TSICNT] bits is the counter value of 22 scan periods. 1_0110b - The TSI_DATA[TSICNT] bits is the counter value of 23 scan periods. 1_0111b - The TSI_DATA[TSICNT] bits is the counter value of 24 scan periods. 1_1000b - The TSI_DATA[TSICNT] bits is the counter value of 25 scan periods. 1_1001b - The TSI_DATA[TSICNT] bits is the counter value of 26 scan periods. 1_1010b - The TSI_DATA[TSICNT] bits is the counter value of 27 scan periods. 1_1011b - The TSI_DATA[TSICNT] bits is the counter value of 28 scan periods. 1_1100b - The TSI_DATA[TSICNT] bits is the counter value of 29 scan periods. 1_1101b - The TSI_DATA[TSICNT] bits is the counter value of 30 scan periods. 1_1110b - The TSI_DATA[TSICNT] bits is the counter value of 31 scan periods. 1_1111b - The TSI_DATA[TSICNT] bits is the counter value of 32 scan periods.
15-4 —	Reserved
3 SWITCH_ENABLE	SWITCH_ENABLE Indicating the state of SSC (spread spectrum clocking), for digital testing. SSC function is used for spreading frequency of sampling clock, reducing EMC (Electro Magnetic Compatibility). 0b - SSC function is disabled. 1b - SSC function is enabled.
2 SINC_OVERFLOW_FLAG	SINC_OVERFLOW_FLAG Indicating whether the counter result in TSI_DATA[TSICNT] has an overflow occurrence in the last scan process. Note: this bit has no default value, please force it to 0 or deposit it if necessary. 0b - The counter result has no overflow occurrence in the last scan process. 1b - The counter result has an overflow occurrence in the last scan process.

Table continues on the next page...

Field	Function
1 SINC_VALID	SINC_VALID Indicating the state of SINC filter, for digital testing. 0b - SINC filter is disabled. 1b - SINC filter is enabled.
0 SSC_CONTROL_OUT	SSC_CONTROL_OUT Indicating the state of SSC output value, for digital testing. 0b - SSC output value is 0. 1b - SSC output value is 1.

44.6.1.9 TSI SSC register 0 (SSC0)

44.6.1.9.1 Offset

Register	Offset
SSC0	1Ch

44.6.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRBS_OUTSEL				0	SSC_MODE		SSC_CONTROL_REVERSE	CHARGE_NUM				BASE_NOCHARGE_NUM			
W																
Reset	0	1	1	0	0	0	0	0	0	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	0								SSC_PRESCALE_NUM							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.6.1.9.3 Fields

Field	Function
31-28 PRBS_OUTSEL	PRBS_OUTSEL When SSC0[SSC_MODE] = 2'b00, choosing the length of the PRBS (Pseudo-RandomBinarySequence) method. 0000b - NC. 0001b - NC. 0010b - The length of the PRBS is 2. 0011b - The length of the PRBS is 3. 0100b - The length of the PRBS is 4. 0101b - The length of the PRBS is 5. 0110b - The length of the PRBS is 6. 0111b - The length of the PRBS is 7. 1000b - The length of the PRBS is 8. 1001b - The length of the PRBS is 9. 1010b - The length of the PRBS is 10. 1011b - The length of the PRBS is 11. 1100b - The length of the PRBS is 12. 1101b - The length of the PRBS is 13. 1110b - The length of the PRBS is 14. 1111b - The length of the PRBS is 15.
27 —	Reserved
26-25 SSC_MODE	SSC_MODE Choosing the SSC mode. 00b - Using PRBS method generating SSC output bit. 01b - Using up-down counter generating SSC output bit. 10b - SSC function is disabled. 11b - NC.
24 SSC_CONTRO L_REVERSE	SSC_CONTROL_REVERSE Reversing the SSC output bit's polarity or not. 0b - Keep the polarity of the SSC output bit. 1b - Reverse the polarity of the SSC output bit.
23-20 CHARGE_NUM	CHARGE_NUM Choosing the period of the SSC output bit 0's period, when using up-down counter mode. 0000b - The SSC output bit 0's period will be 1 clock cycle of system clock. 0001b - The SSC output bit 0's period will be 2 clock cycles of system clock. 0010b - The SSC output bit 0's period will be 3 clock cycles of system clock. 0011b - The SSC output bit 0's period will be 4 clock cycles of system clock. 0100b - The SSC output bit 0's period will be 5 clock cycles of system clock. 0101b - The SSC output bit 0's period will be 6 clock cycles of system clock. 0110b - The SSC output bit 0's period will be 7 clock cycles of system clock. 0111b - The SSC output bit 0's period will be 8 clock cycles of system clock. 1000b - The SSC output bit 0's period will be 9 clock cycles of system clock. 1001b - The SSC output bit 0's period will be 10 clock cycles of system clock. 1010b - The SSC output bit 0's period will be 11 clock cycles of system clock. 1011b - The SSC output bit 0's period will be 12 clock cycles of system clock. 1100b - The SSC output bit 0's period will be 13 clock cycles of system clock. 1101b - The SSC output bit 0's period will be 14 clock cycles of system clock. 1110b - The SSC output bit 0's period will be 15 clock cycles of system clock. 1111b - The SSC output bit 0's period will be 16 clock cycles of system clock.
19-16	BASE_NOCHARGE_NUM

Table continues on the next page...

Field	Function
BASE_NOCHARGE_NUM	Choosing the basic period of the SSC output bit 1's period, when using up-down counter mode. Together with the TSI_SSC2[MOVE_ NOCHARGE_MAX] and TSI_SSC2[MOVE_ NOCHARGE_MIN], they are determining the SSC output 1's period. 0000b - The SSC output bit 1's basic period will be 1 clock cycle of system clock. 0001b - The SSC output bit 1's basic period will be 2 clock cycles of system clock. 0010b - The SSC output bit 1's basic period will be 3 clock cycles of system clock. 0011b - The SSC output bit 1's basic period will be 4 clock cycles of system clock. 0100b - The SSC output bit 1's basic period will be 5 clock cycles of system clock. 0101b - The SSC output bit 1's basic period will be 6 clock cycles of system clock. 0110b - The SSC output bit 1's basic period will be 7 clock cycles of system clock. 0111b - The SSC output bit 1's basic period will be 8 clock cycles of system clock. 1000b - The SSC output bit 1's basic period will be 9 clock cycles of system clock. 1001b - The SSC output bit 1's basic period will be 10 clock cycles of system clock. 1010b - The SSC output bit 1's basic period will be 11 clock cycles of system clock. 1011b - The SSC output bit 1's basic period will be 12 clock cycles of system clock. 1100b - The SSC output bit 1's basic period will be 13 clock cycles of system clock. 1101b - The SSC output bit 1's basic period will be 14 clock cycles of system clock. 1110b - The SSC output bit 1's basic period will be 15 clock cycles of system clock. 1111b - The SSC output bit 1's basic period will be 16 clock cycles of system clock.
15-8 —	Reserved
7-0 SSC_PRESCALE_NUM	SSC_PRESCALE_NUM Selecting the divider ratio for the clock used for generating the SSC output bit. The clock frequency is main_clock/(SSC_PRESCALE_NUM + 1) before going into SSC logic. The average SSC output frequency is determined by SSC_PRESCALE_NUM and detailed SSC configuration. 0000_0000b - div1 0000_0001b - div2 0000_0011b - div4 0000_0111b - div8 0000_1111b - div16 0001_1111b - div32 0011_1111b - div64 0111_1111b - div128 1111_1111b - div256

44.6.1.10 TSI SSC register 1 (SSC1)

44.6.1.10.1 Offset

Register	Offset
SSC1	20h

44.6.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRBS_WEIGHT_HI								PRBS_WEIGHT_LO							
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRBS_SEED_HI								PRBS_SEED_LO							
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

44.6.1.10.3 Fields

Field	Function
31-24 PRBS_WEIGHT_HI	PRBS_WEIGHT_HI Together with the TSI_SSC1[PRBS_WEIGHT_LO], choosing the PRBS's feeding back nodes, when using PRBS method generating SSC output bit. The nodes whose value corresponding with "1" will be feed back and connected to the input of the XOR.
23-16 PRBS_WEIGHT_LO	PRBS_WEIGHT_LO Together with the TSI_SSC1[PRBS_WEIGHT_HI], choosing the PRBS's feeding back nodes, when using PRBS method generating SSC output bit. The nodes whose value corresponding with "1" will be feed back and connected to the input of the XOR.
15-8 PRBS_SEED_HI	PRBS_SEED_HI Together with the TSI_SSC1[PRBS_SEED_LO], choosing the initial value of the PRBS method, when using PRBS method generating SSC output bit.
7-0 PRBS_SEED_LO	PRBS_SEED_LO Together with the TSI_SSC1[PRBS_SEED_HI], choosing the initial value of the PRBS method, when using PRBS method generating SSC output bit.

44.6.1.11 TSI SSC register 2 (SSC2)

44.6.1.11.1 Offset

Register	Offset
SSC2	24h

44.6.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MOVE_NOCHARGE_MIN				0						MOVE_NOCHARGE_MAX					
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					MOVE_STEPS_NUM			0			MOVE_REPEAT_NUM				
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

44.6.1.11.3 Fields

Field	Function
31-28 MOVE_NOCHARGE_MIN	MOVE_NOCHARGE_MIN Choosing the min period of the SSC output bit 1's period, when using up-down counter mode. Together with the TSI_SSC0[BASE_ NOCHARGE_NUM] and TSI_SSC2[MOVE_ NOCHARGE_MAX], they are determining the SSC output 1's period. 0000b - The SSC output bit 1's min period will be (1 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycle of divided system clock. 0001b - The SSC output bit 1's min period will be (2 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 0010b - The SSC output bit 1's min period will be (3 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 0011b - The SSC output bit 1's min period will be (4 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 0100b - The SSC output bit 1's min period will be (5 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 0101b - The SSC output bit 1's min period will be (6 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 0110b - The SSC output bit 1's min period will be (7 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 0111b - The SSC output bit 1's min period will be (8 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock.

Table continues on the next page...

Field	Function
	1000b - The SSC output bit 1's min period will be (9 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1001b - The SSC output bit 1's min period will be (10 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1010b - The SSC output bit 1's min period will be (11 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1011b - The SSC output bit 1's min period will be (12 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1100b - The SSC output bit 1's min period will be (13 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1101b - The SSC output bit 1's min period will be (14 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1110b - The SSC output bit 1's min period will be (15 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock. 1111b - The SSC output bit 1's min period will be (16 + TSI_SSC0[BASE_ NOCHARGE_NUM]) clock cycles of divided system clock.
27-22 —	Reserved
21-16 MOVE_NOCHARGE_MAX	MOVE_NOCHARGE_MAX Similar with TSI_SSC2[MOVE_NOCHARGE_MAX], it is choosing the max period of the SSC output bit 1's period, when using up-down counter mode. Together with the TSI_SSC0[BASE_ NOCHARGE_NUM] and TSI_SSC2[MOVE_ NOCHARGE_MIN], they are determining the SSC output 1's period.
15-11 —	Reserved
10-8 MOVE_STEPS_NUM	MOVE_STEPS_NUM Choosing the steps for the counters of TSI_SSC0[BASE_NOCHARGE_NUM]/ TSI_SSC2[MOVE_NOCHARGE_MAX]/ TSI_SSC2[MOVE_CHARGE_MIN], when using up-down counter mode. 000b - The added value for up-down counter is 0. 001b - The added value for up-down counter is 1. 010b - The added value for up-down counter is 2. 011b - The added value for up-down counter is 3. 100b - The added value for up-down counter is 4. 101b - The added value for up-down counter is 5. 110b - The added value for up-down counter is 6. 111b - The added value for up-down counter is 7.
7-5 —	Reserved
4-0 MOVE_REPEAT_NUM	MOVE_REPEAT_NUM Choosing the repeat times for the same setting of TSI_SSC0[BASE_NOCHARGE_NUM]/ TSI_SSC2[MOVE_NOCHARGE_MAX]/ TSI_SSC2[MOVE_CHARGE_MIN], when using up-down counter mode. Only when this repeat times is reached, these settings can be changed to the next values. 0_0000b - The up_down counter will be updated for every sample-charge cycle. 0_0001b - The up_down counter will be updated for every 2 sample-charge cycles. 0_0010b - The up_down counter will be updated for every 3 sample-charge cycles. 0_0011b - The up_down counter will be updated for every 4 sample-charge cycles. 0_0100b - The up_down counter will be updated for every 5 sample-charge cycles. 0_0101b - The up_down counter will be updated for every 6 sample-charge cycles. 0_0110b - The up_down counter will be updated for every 7 sample-charge cycles.

44.6.1.12 TSI shield register (SHIELD)

44.6.1.12.1 Offset

Register	Offset
SHIELD	28h

44.6.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							SHIELD_SEL24	SHIELD_SEL23	SHIELD_SEL22	SHIELD_SEL21	SHIELD_SEL20	SHIELD_SEL19	SHIELD_SEL18	SHIELD_SEL17	SHIELD_SEL16
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	SHIELD_SEL15	SHIELD_SEL14	SHIELD_SEL13	SHIELD_SEL12	SHIELD_SEL11	SHIELD_SEL10	SHIELD_SEL9	SHIELD_SEL8	SHIELD_SEL7	SHIELD_SEL6	SHIELD_SEL5	SHIELD_SEL4	SHIELD_SEL3	SHIELD_SEL2	SHIELD_SEL1	SHIELD_SEL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.6.1.12.3 Fields

Field	Function
31-25 —	Reserved
24-0 SHIELD_SELn	Selects shield channels. Configures which channels are shield channels. Each bit corresponds to each TSI channel. For example, bit 0 corresponds to TSI channel 0. 0b - The channel is not configured as shield channel. 1b - The channel is configured as shield channel.

44.7 Usage Guide

44.7.1 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, read the TSI status register to determine the exact interrupt source.

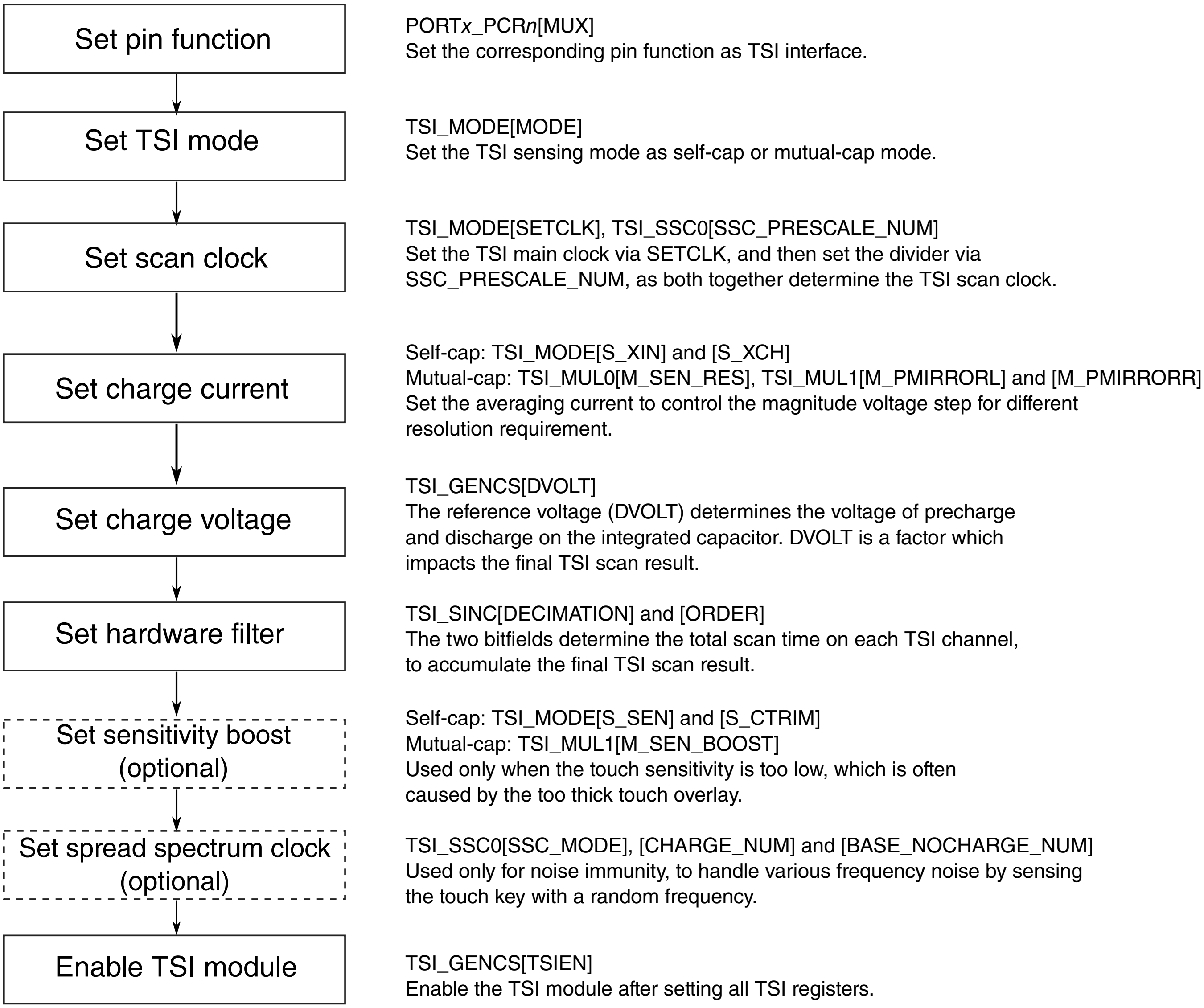
44.7.2 How to use the TSI module

There are several steps as below.

- Initiate the TSI module by configuring registers
- Start TSI scan by hardware or software trigger
- Read the TSI result once TSI scan done (end-of-scan)
- Process the TSI result raw data to determine whether a touch event occurs

44.7.2.1 Initialization sequence

The following figure shows the flowchart of TSI initialization.



PORT*x*_PCR*n*[MUX]
Set the corresponding pin function as TSI interface.

TSI_MODE[MODE]
Set the TSI sensing mode as self-cap or mutual-cap mode.

TSI_MODE[SETCLK], TSI_SSC0[SSC_PRESCALE_NUM]
Set the TSI main clock via SETCLK, and then set the divider via SSC_PRESCALE_NUM, as both together determine the TSI scan clock.

Self-cap: TSI_MODE[S_XIN] and [S_XCH]
Mutual-cap: TSI_MUL0[M_SEN_RES], TSI_MUL1[M_PMIRRORL] and [M_PMIRRORR]
Set the averaging current to control the magnitude voltage step for different resolution requirement.

TSI_GENCS[DVOLT]
The reference voltage (DVOLT) determines the voltage of precharge and discharge on the integrated capacitor. DVOLT is a factor which impacts the final TSI scan result.

TSI_SINC[DECIMATION] and [ORDER]
The two bitfields determine the total scan time on each TSI channel, to accumulate the final TSI scan result.

Self-cap: TSI_MODE[S_SEN] and [S_CTRIM]
Mutual-cap: TSI_MUL1[M_SEN_BOOST]
Used only when the touch sensitivity is too low, which is often caused by the too thick touch overlay.

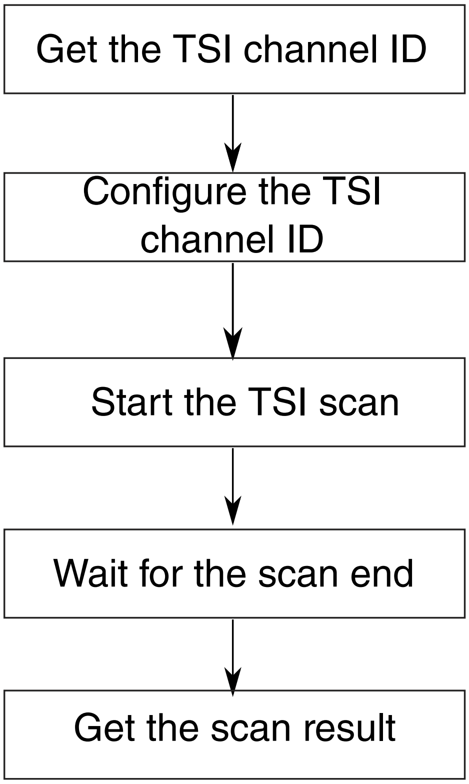
TSI_SSC0[SSC_MODE], [CHARGE_NUM] and [BASE_NOCHARGE_NUM]
Used only for noise immunity, to handle various frequency noise by sensing the touch key with a random frequency.

TSI_GENCS[TSIEN]
Enable the TSI module after setting all TSI registers.

Figure 44-15. TSI initialization sequence

44.7.2.2 TSI scan example

In the self-cap mode, one touch key is connected to one TSI channel, which is measured at each TSI scan round. The following figure shows the software flowchart of TSI scan example, in self-cap mode.



Get the TSI channel ID corresponding to the touch key
Set up a mapping table to define the connection of
TSI channel and touch key.

Self-cap: TSI_DATA[TSICH]
Configure the TSI channel ID to be scanned.

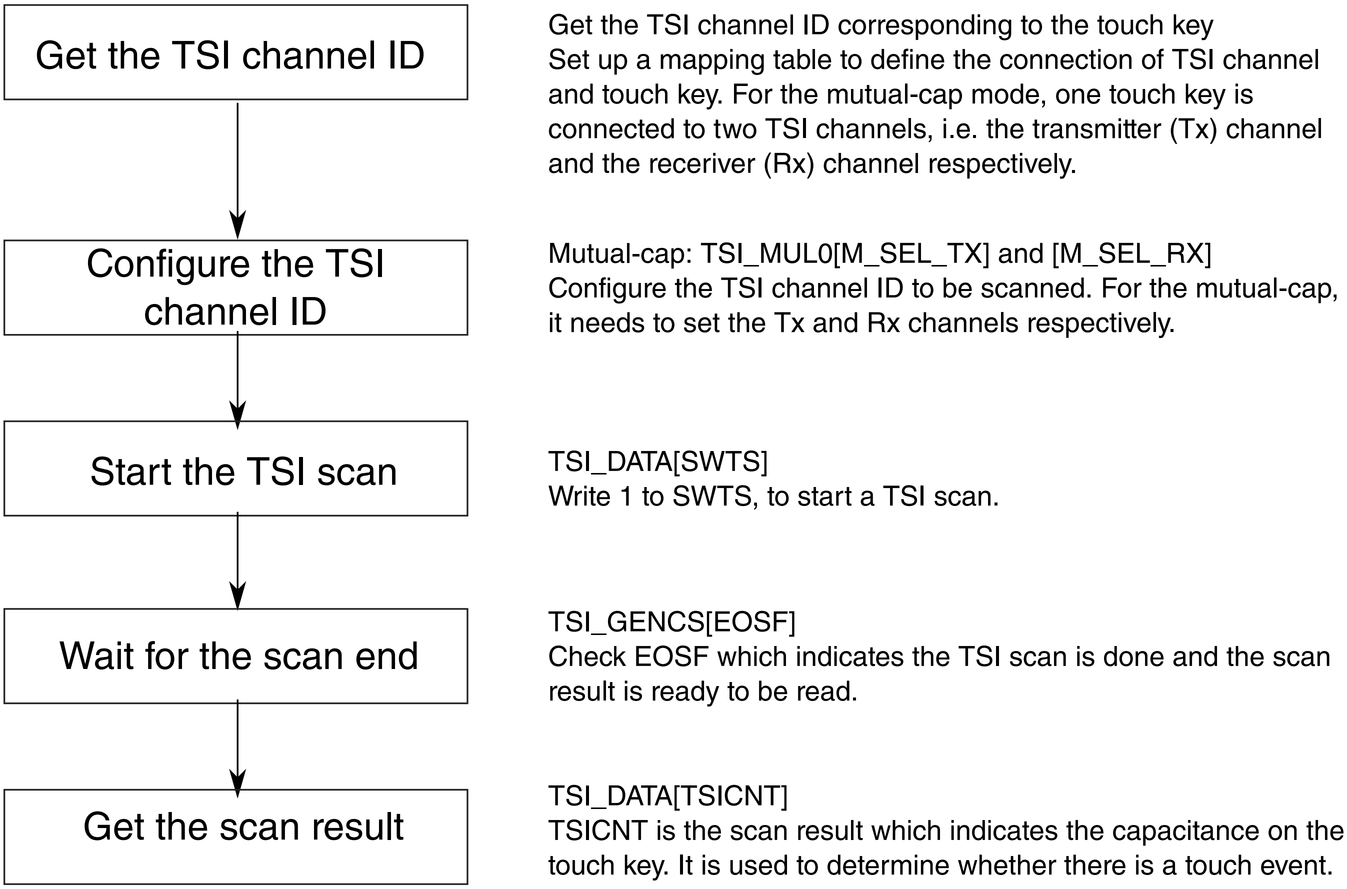
Self-cap: TSI_DATA[SWTS]
Write 1 to SWTS, to start a TSI scan.

TSI_GENCS[EOSF]
Check EOSF which indicates the TSI scan is done
and the scan result is ready to be read.

TSI_DATA[TSICNT]
TSICNT is the scan result which indicates the capacitance on the
touch key. It is used to determine whether there is a touch event.

Figure 44-16. TSI scan example for self-cap mode

In mutual-cap mode, one touch key is connected to two TSI channels, i.e. the transmitter and the receiver channel respectively. The figure below shows the software flowchart of TSI scan example, in mutual-cap mode.



Get the TSI channel ID corresponding to the touch key
Set up a mapping table to define the connection of TSI channel and touch key. For the mutual-cap mode, one touch key is connected to two TSI channels, i.e. the transmitter (Tx) channel and the receriver (Rx) channel respectively.

Mutual-cap: TSI_MUL0[M_SEL_TX] and [M_SEL_RX]
Configure the TSI channel ID to be scanned. For the mutual-cap, it needs to set the Tx and Rx channels respectively.

TSI_DATA[SWTS]
Write 1 to SWTS, to start a TSI scan.

TSI_GENCS[EOSF]
Check EOSF which indicates the TSI scan is done and the scan result is ready to be read.

TSI_DATA[TSICNT]
TSICNT is the scan result which indicates the capacitance on the touch key. It is used to determine whether there is a touch event.

Figure 44-17. TSI scan example for mutual-cap mode

44.7.2.3 Process TSI scan result to detect a touch event

When the touch key is touched by finger, the TSI scan result (TSI_DATA[TSICNT]) changes a lot. By comparing the changed value, the touch event can be determined. The following figure shows an example of detecting a touch event by TSI scan result.

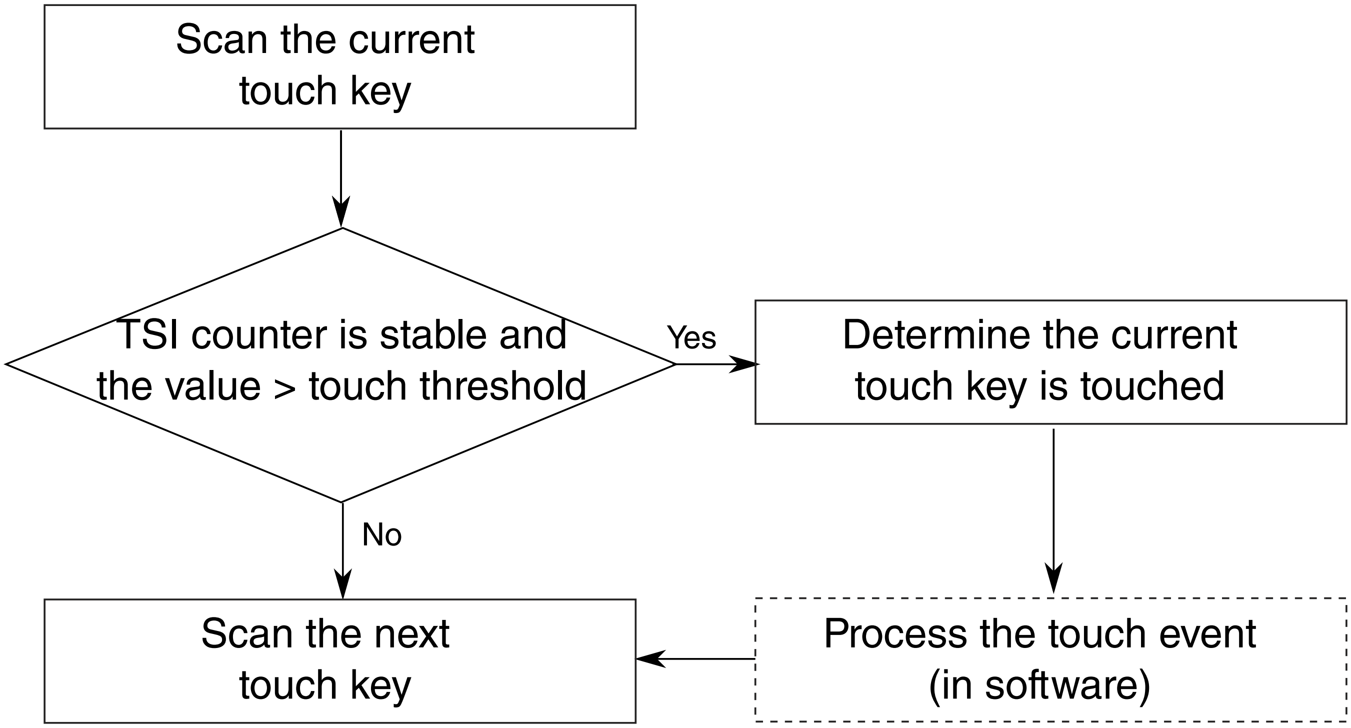

Scan the current touch key
TSI counter is stable and the value > touch threshold
Yes
Determine the current touch key is touched
No
Scan the next touch key
Process the touch event (in software)

Figure 44-18. TSI scan result process

NOTE

For touch electrode hardware design guideline, see AN3863: Designing Touch Sensing Electrodes.

Appendix A
Revision History

The following table provides a revision history for this document.

Table A-1. Revision history

Document ID	Release date	Description
KE1XZP100M96SF0RM v.2	01/2024	Initial public release.

Appendix B
Change Summary for This Revision

Change summary
Chapter 1: About This Manual changes No substantial change
Chapter 2: Introduction changes No substantial change
Chapter 3: Core Overview changes No substantial change
Chapter 4: Interrupts changes No substantial change
Chapter 5: SIM changes No substantial change
Chapter 6: MCM changes No substantial change
Chapter 7: AXBS-Lite changes No substantial change
Chapter 8: AIPS-Lite changes No substantial change
Chapter 9: TRGMUX changes No substantial change
Chapter 10: DMAMUX changes • Updated "Block diagram". • Updated the figure "DMAMUX triggered channels". • Added a short introduction in "Modes of operation".
Chapter 11: eDMA changes • Updated "Interrupts".
Chapter 12: Memory and memory map changes No substantial change
Chapter 13: FMC/FAU changes No substantial change
Chapter 14: FTFE changes No substantial change

Table continues on the next page...

Change summary
Chapter 15: Clock Distribution changes • Added a note in "FTM Clocking Information".
Chapter 16: SCG changes • Minor update in the section "Information of SCG on this device". • Reserved bit 1 in LPFLLCSR. • Updated TRIMSRC: bitfield value 0b11 to Reserved.
Chapter 17: PCC changes No substantial change
Chapter 18: Reset and Boot changes No substantial change
Chapter 19: Kinetis Flashloader changes No substantial change
Chapter 20: RCM changes No substantial change
Chapter 21: Power Management changes No substantial change
Chapter 22: SMC changes No substantial change
Chapter 23: PMC changes • In PMC register, updated the description of REGSC[CLKBIASDIS] and REGSC[BIASEN] bitfields..
Chapter 24: Integrity Functions Overview changes No substantial change
Chapter 25: EWM changes No substantial change
Chapter 26: WDOG changes • In "Functionality in Debug and Low-Power modes", added: Don't set CS[INT] in Stop mode. Otherwise, WDOG reset after a delay of 128 bus clocks (lose bus clock) will not occur, but backup reset will take effect. • In WDOG Register, added a note for CS[CLK] bit field and updated its bit field values. • Updated "Features".
Chapter 27: CRC changes • Updated "Calculating a 32-bit CRC". • Added "Clocking" and "Interrupts". • Renamed section title "CRC initialization and reinitialization" to "Initialization". • Updated the description of CTRL[WAS]. • Updated "Clocking". • Removed the entries for CRC16_DECT_R from the tables "CTRL programming for 16-bit CRC" and "Expected read data fields for 16-bit CRC".
Chapter 28: Debug changes No substantial change
Chapter 29: MTB changes No substantial change
Chapter 30: PORT changes No substantial change

Table continues on the next page...

Change summary
Chapter 31: GPIO changes No substantial change
Chapter 32: ADC changes No substantial changes
Chapter 33: CMP changes No substantial changes
Chapter 34: FTM changes • Added a note in "Instantiation Information". • Added a note in "FTM Clocking Information". • Updated figures in "Initialization Trigger".
Chapter 35: LPIT changes No substantial changes
Chapter 36: PWT changes No substantial changes
Chapter 37: LPTMR changes • Updated "Application information" for chip-specific note pertaining to 32.768 kHz clock source. • Updated "Application information" for Prescaler point (replaced 00h with 0Eh) pertaining to 32.768 kHz clock source.
Chapter 38: RTC changes No substantial changes
Chapter 39: LPSPI changes • Updated CFGR1[MATCFG] description and notes in Data Match 0 (DMR0) and Data Match 1 (DMR1).
Chapter 40: LPI2C changes • Updated "Error conditions". • Updated "Features". • Updated "Address matching". • Updated "Timing parameters". • Updated "Controller operations". • Updated "Pin configuration". • Updated "Block diagram". • Updated MCFGR0[HRPOL] description.
Chapter 41: LPUART changes • Updated "Clocking". • Added "Reset". • Added "Initialization". • Added functions in the following registers: Global (GLOBAL), Pin Configuration (PINCFG), Baud Rate (BAUD), Status (STAT). • Updated BAUD[SBR]. • Replaced the instance of "IrDA" with "infrared data association (IrDA)" in Overview section. • Updated the following sections: Receiver functional description, Data sampling technique, Receiver wake-up operation. • Replaced section title "Interrupts and status fields" with "Interrupts". • Replaced reserved field access type "RW" with "ROZ" in bit 12 of Control (CTRL) register. • Updated "Baud rate generation".
Chapter 42: SCI/UART changes No substantial changes
Chapter 43: FlexIO changes No substantial changes

Table continues on the next page...

Change summary
Chapter 44: TSI changes • Minor update in "Instantiation Information". • Minor update in "Shield function".

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, μVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

Kinetis — is a trademark of NXP B.V.

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 01/2024
Document identifier: KE1XZP100M96SF0RM