MSPM0 C-Series 24-MHz Microcontrollers

Technical Reference Manual

Literature Number: SLAU893B
OCTOBER 2023 – REVISED JULY 2024

Table of Contents

This page intentionally left blank.

Read This First

About This Manual

This manual describes the modules and peripherals of the MSPM0C family of devices. Each description presents the module or peripheral in a general sense. Not all features and functions of all modules or peripherals are present on all devices. In addition, modules or peripherals can differ in their exact implementation on different devices. Pin functions, internal signal connections, and operational parameters differ from device to device. See the device-specific data sheet for these details.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers can be shown with the suffix h or the prefix 0x. For example, the following number is 40 hexadecimal (decimal 64): 40h or 0x40.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties with default reset value below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure can have one of multiple meanings:
    - Not implemented on the device
    - Reserved for future device expansion
    - Reserved for TI testing
    - Reserved configurations of the device that are not supported
  - Writing nondefault values to the Reserved bits could cause unexpected behavior and should be avoided.

Glossary

TI Glossary — This glossary lists and explains terms, acronyms, and definitions.

Related Documentation

Support Resources

TI E2E™ support forums are an engineer's go-to source for fast, verified answers and design help — straight from the experts. Search existing answers or ask your own question to get the quick design help you need.

Linked content is provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

Trademarks

TI E2E™ and EnergyTrace™ are trademarks of Texas Instruments.
All trademarks are the property of their respective owners.

Chapter 1

Architecture

The device architecture includes the bus organization, the platform memory map, and the boot configuration.

1.1 Architecture Overview

MSPM0 C-series MCUs (MSPM0Cxx) combine 32-bit compute performance together with precision analog to enable a wide variety of sensing, interface, control, and housekeeping applications. The device architecture supports both general-purpose and low-power applications through a flexible, easy-to-configure power management and clocking system with fast wake-up from low-power modes.

MSPM0 C-series devices also offer support for 125°C ambient temperature and AEC-Q100 Grade 1 qualification.

This chapter introduces the device architecture, including an overview of the power domains and bus organization, the platform memory map, and the device boot configuration.

1.2 Bus Organization

There are three main power domains on MSPM0Cxx devices:

- PD1 (power domain 1) which contains the CPU subsystem, memory interfaces, and high-speed peripherals
- PD0 (power domain 0) which contains the low-speed low-power peripherals
- The supply voltage (VDD) which powers IOs, analog modules, and limited logic directly from the supply

The PD1 domain is disabled in certain operating modes to minimize power consumption. The PD0 domain supports ultra-low-power performance and is always enabled in operating modes in which the core regulator is operating.

There are four main data buses on MSPM0Cxx MSPM0Hxx devices:

- The AHB bus matrix, which interfaces the CPU to the device memory systems (ROM, SRAM, and flash memory) and the peripheral buses
- The PD1 (power domain 1) CPU-only peripheral bus, clocked from MCLK
- The PD1 (power domain 1) peripheral bus, clocked from MCLK
- The PD0 (power domain 0) peripheral bus, clocked from ULPCLK

The CPU and the DMA controller are the only two bus controllers in the device. Arbitration between the CPU and the DMA for shared peripherals happens between the CPU-only PD1 peripheral bus and the CPU/DMA PD1 peripheral bus. The DMA does not have access to peripherals on the CPU-only PD1 peripheral bus or the CPU bus matrix (the green components in the bus diagram). As such, the CPU can access peripherals on the CPU-only PD1 peripheral bus at the same time that the DMA is processing a transaction on the PD1 or PD0 bus.

Likewise, the CPU can access SRAM or flash memory through the AHB bus matrix at the same time that the DMA is processing a transaction, so long as the DMA is not accessing the same memory that the CPU is attempting to access. Arbitration between the CPU and the DMA for memory systems (SRAM or flash memory) happens at the memory interface itself. All arbitration between the CPU and DMA is done on a round-robin basis.

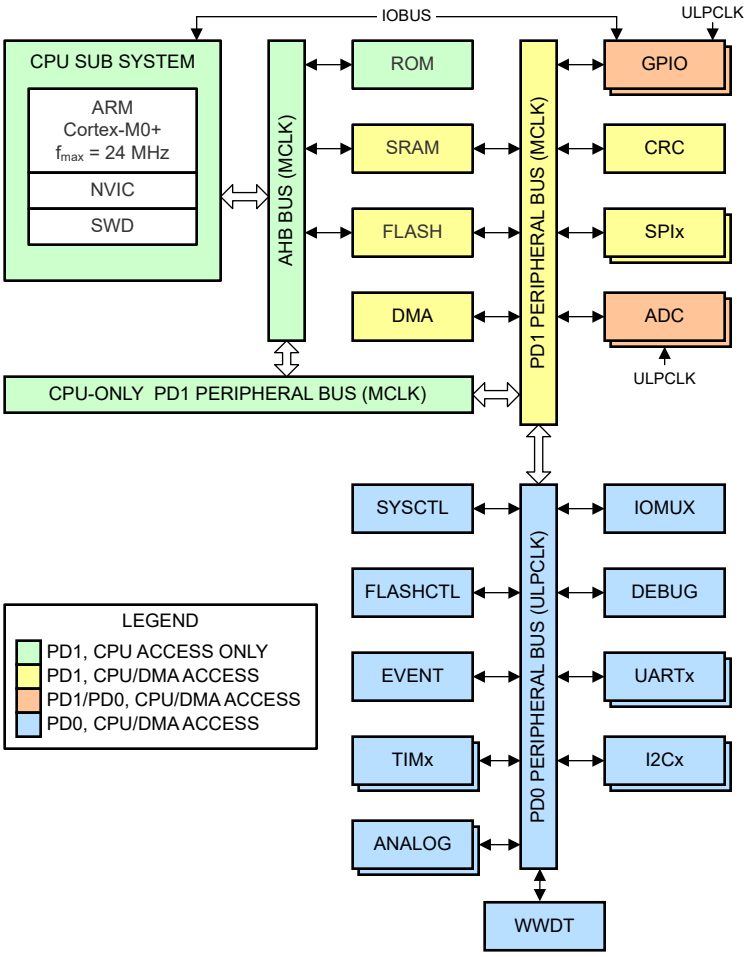


Figure 1-1. MSPM0Cxx Bus Organization

Note

This is a generic diagram of the typical peripherals on an MSPM0Cxx device and their respective bus locations. Not all devices have all peripheral options shown here. To determine the peripherals which are available on a given device, see the device-specific data sheet.

The GPIO and ADC peripherals (the orange components in the bus diagram) have special capabilities to enable both fast register access from the CPU and operation in low power operating modes.

- GPIO peripherals interface to the system through the PD1 peripheral bus.
  - The GPIO DOUT registers (data out) are also available on the PD1 peripheral bus, primarily so that the DMA can be used to load values to the GPIO DOUT registers. Note that not all MSPM0Cxx devices have DMA supporting for GPIO, See the device-specific data sheet to determine if DMA is present for GPIO.
  - While the bus interfaces to the GPIO peripherals are in the PD1 power domain (for best read/write performance), the GPIO logic itself is in the PD0 power domain so that it is available in all operating modes in which the core regulator is active.
- ADC peripherals interface to the system through the PD1 peripheral bus but contain functional logic in the PD0 power domain.
  - ADC peripheral register accesses are processed through the PD1 peripheral bus (for best read/write performance)
  - The the ADC conversion logic is in the PD0 power domain to enable running timer-triggered ADC conversions without CPU interaction in certain low-power modes when PD1 is disabled.

1.3 Platform Memory Map

All MSPM0Cxx devices share a common platform memory map. Peripherals are assigned a fixed address space and have the same address space on all devices within the family. The memory map is compliant with the standard Arm Cortex-M memory regions.

Table 1-1. Top Level Memory Map

Memory Region	Start Address	End Address	Description
Code	0x0000.0000	0x1FFF.FFFF	Flash memory and ROM
SRAM	0x2000.0000	0x3FFF.FFFF	SRAM
Peripheral	0x4000.0000	0x5FFF.FFFF	Global peripheral memory-mapped registers and global nonexecutable data memory
Subsystem	0x6000.0000	0x7FFF.FFFF	Local CPU subsystem memory-mapped registers
System PPB	0xE000.0000	0xE00F.FFFF	Arm private peripheral bus

1.3.1 Code Region

The code region contains the flash memory used to store executable code and data. Accesses to the flash memory from the CPU through the code region are processed through the AHB bus matrix to the flash read interface directly. See Section 5.2.3.1 for the detailed flash memory map.

The code region also contains the read-only memory (ROM) used for the TI device boot code and the bootstrap loader. Note that not all MSPM0Cxx devices have the bootstrap loader. See the device-specific data sheet to determine if the bootstrap loader is present. The ROM is only available during the initial device boot process.

1.3.2 SRAM Region

The SRAM region contains the system memory (SRAM). The SRAM supports zero wait state access at the maximum MCLK frequency (24MHz). Accesses to the SRAM from the CPU are processed through the AHB bus matrix to the SRAM interface directly. See the device-specific data sheet for the amount of SRAM present on a given device.

Certain devices optionally support parity or parity and ECC checking of the SRAM. Refer to the device-specific data sheet to determine if a device supports ECC or parity checked SRAM. For information on how parity and ECC errors are handled by the device, see Section 2.4.9.

Parity Checking

In the case of parity checking (if available), 1 parity bit is provided per 8 data bits. Parity checking is capable of detecting a single bit error in the corresponding 8 bits of data (SED). Writing data to a parity checked address updates the corresponding parity bits based on the new data. Reading data from a parity checked address checks the read data against the corresponding parity bits. Upon a read, if the data does not match the corresponding parity bits, a parity error is generated. For information on how parity errors are handled by the device, see Section 2.4.9

ECC Checking

In the case of ECC checking (if available), 8 ECC bits are provided per 64 data bits. ECC is capable of correcting single bit errors (SEC) and detecting dual bit errors (DED) in the corresponding 64 data bits. Writing data to an ECC checked address updates the corresponding ECC code based on the new data. Reading data from an ECC checked address checks the read data against the corresponding ECC code. If a single bit error is found, it is corrected automatically and a correctable ECC error is generated. If a dual bit error is found, an uncorrectable ECC error is generated.

Aliased Subregions

The physical SRAM on a device is aliased into multiple address subregions in the overall SRAM region, as shown in the following table. The default, parity checked, and unchecked address subregions all map to the same physical SRAM memory. The difference between each aliased subregion is in the type of integrity checks

which are applied to the access. For example, writing data to address 0x2000.0000 (the default subregion) will cause the same data to appear at address 0x2020.0000 (the unchecked subregion).

The default subregion (0x2000.0000) is available on all MSPM0 devices, and when used, provides the highest level of integrity checking available on the device. The parity checked subregion (0x2010.0000) is available on devices which support ECC or parity checking, and accesses are always processed with parity checking. The unchecked subregion (0x2020.0000) is available on all devices, and when using this subregion, no integrity checks are performed. The parity/ECC code subregion (0x2030.0000) is available on devices with ECC or parity checking, and it returns the parity or ECC code which corresponds to the address being read.

Note

Not all the MSPM0C devices support SRAM parity check or ECC. See the device-specific data sheet to determine if SRAM parity check or ECC is present.

Note

To improve robustness, there is no mechanism provided to disable parity checking of the parity-checked SRAM address space. To operate without parity checking, link the application against the unchecked SRAM address space.

Table 1-2. SRAM Region Memory Map

Subregion	Start	End	Description
Default	0x2000.0000	0x200F.FFFF	The highest available integrity check on the device is always applied to accesses in this subregion: • If the device supports ECC, this subregion is ECC checked. • If the device only supports parity (no ECC), this subregion is parity checked and accesses are equivalent to accesses to the parity checked subregion. • If the device does not support ECC or parity, no checks are applied to accesses in this subregion and the region is equivalent to the unchecked subregion.
Parity checked	0x2010.0000	0x201F.FFFF	If the device supports parity, accesses to this subregion are parity checked.
Unchecked	0x2020.0000	0x202F.FFFF	No ECC or parity checks are applied to accesses in this subregion.
Parity/ECC code	0x2030.0000	0x203F.FFFF	If the device supports parity or ECC, the parity or ECC codes may be directly accessed through this subregion: • If the device supports ECC, access to any address within a 64-bit boundary returns the 8 bits corresponding to the ECC code (if the application is linked against the default region) or the parity bits (one per byte) if the application is linked against the parity checked region. • If the device supports parity only, access to any address within a 32-bit boundary returns the 4 parity bits (one per byte) if the application is linked against the default region or parity checked region. • If the device does not support ECC or parity, access to this region always returns zeros.

Note

When ECC checking is used on devices which support ECC, writes to the SRAM require two cycles to complete. Read accesses only require a single cycle and do not incur any additional performance penalty.

On devices supporting parity or ECC, it is possible for application software to partition the usage of the physical SRAM into arbitrary zones which are intended to be ECC checked, parity checked, or unchecked. For example, if a device has 32KB of total SRAM memory, and it supports ECC and parity checking, it is possible to configure the application software to link against two subregions, one being ECC checked and the other being parity checked.

Note

It is not recommended to mix and match ECC checked, parity checked, and unchecked accesses to the same memory locations, as this can result in unintended parity/ECC errors. For example, if an SRAM memory location is written to through the parity checked subregion on a device which supports ECC, and then later that location is read through the default (ECC checked) subregion, an ECC error may be generated because the data was written with parity stored to the ECC/parity code memory, as opposed to the correct ECC code being stored.

Note

The SRAM contents may be random at power-up or when existing SHUTDOWN mode. A read of an SRAM location through an ECC/parity checked region which has not yet been written to may result in a ECC/parity error being detected, causing a hard fault in the processor. Ensure that all SRAM locations are initialized first before being read.

If the application is using only the unchecked memory region, then the Parity/ECC code region can be used for additional application SRAM memory.

1.3.3 Peripheral Region

The peripheral region contains the memory-mapped peripherals on the three peripheral buses. The flash memory is also aliased in the peripheral region.

Table 1-3. Peripheral Region Memory Map

Type	Start	End	Description
Peripherals	0x4000.0000	0x40FF.FFFF	Memory-mapped registers of peripherals on the peripheral buses
Aliased flash memory	0x4100.0000	0x41FF.FFFF	See Section 5.2.3.1

1.3.4 Subsystem Region

The subsystem region contains memory-mapped registers which are specific to the CPU subsystem and do not need to be accessed globally. See the CPU subsystem chapter for the memory-mapped registers in the subsystem region.

1.3.5 System PPB Region

The system private peripheral bus (PPB) region contains memory-mapped registers on the Arm private peripheral bus. These registers are tightly coupled to the CPU and are the interface for peripherals such as the memory protection unit (MPU), SysTick timer, and CPU power management and reset functions.

1.4 Boot Configuration

After a BOOTRST, the device executes the start-up boot routines to configure the device for operation before starting the main application. Boot routines are executed from read-only memory (ROM) before the main application is started. There are two boot routines: the boot configuration routine (BCR). The boot configuration routine sets up the device security policies, configures the device for operation, and optionally starts the BSL if it presents. .

After the start-up routines have successfully completed execution, the CPU is reset and the application is started by unconditionally fetching the stack pointer (SP) and reset vector from 0x0000.0000 and 0x0000.0004 of the

flash memory. To enable secure boot, this single point of entry into the application code is enforced by the boot sequence. It is not possible to boot into a different memory location.

1.4.1 Configuration Memory (NONMAIN)

The NONMAIN is a dedicated region of flash memory which stores the configuration data used by the BCR and BSL to boot the device. The region is not used for any other purpose. The BCR has configuration policies which can be left at their default values (as is typical during development and evaluation), or modified for specific purposes (as is typical during production programming) by altering the values programmed into the NONMAIN flash region.

The BCR configuration data structures are both contained within a single flash sector in the NONMAIN flash memory region. To change any parameter in the boot configuration, it is necessary to erase the entire NONMAIN sector and re-program both the BCR configuration structures with the desired settings.

The configuration data in the NONMAIN flash region is not affected by a mass erase command, but it is erased and re-programmed to factory defaults by a factory reset command sent to the BCR via the debug sub system mailbox (DSSM) over SWD.

The address ranges for the NONMAIN data structures are given in the *NONMAIN Registers* section. A detailed breakdown of the NONMAIN region is provided at the end of this section.

1.4.1.1 CRC-Backed Configuration Data

The BCR configuration data for TI factory trim in the NONMAIN memory includes a CRC16 value corresponding to the CRC16 digest of the respective structure. During the device boot process, the BCR will compute the CRC digest of the data structures and compare it with the stored CRC values before the data contained within the structures is trusted for use.

TI Factory Trim Data CRC Fail Handling

Ifthe TI factory trim fails its CRC check during boot, a catastrophic boot error will also result with the following limitations:

- The error cause will be logged in the CFG-AP as a boot diagnostic
- The user application is not started
- No application debug access is enabled
- A pending TI failure analysis flow entry, if enabled, is honored
- The boot process will re-attempt up to 3 times
  - If the 2nd or 3rd attempt pass, the device boots normally
  - If the 3rd attempt does not pass, no further boot attempts are made until the next BOR or POR

1.4.1.2 16-bit Pattern Match for Critical Fields

Critical policies in the BCR configuration memory, such as the SWD security policies, are implemented as 16-bit pattern-match fields in the NONMAIN memory, with the following characteristics:

- An exact pattern match is required to enable lower security states
- Any value in the 16-bit field not matching the exact defined patterns results in a maximally secure state for the respective parameter

This behavior prevents single bit flips from causing the device to enter a lower security state than that which was originally specified.

1.4.2 Boot Configuration Routine (BCR)

The boot configuration routine is the first firmware to run on the device after a BOOTRST. The BCR manages the following at boot time:

- Configuring the debug interface security policy
- Optionally executing a mass erase
- Optionally executing a factory reset
- Configuring the flash memory static write protection policy

1.4.2.1 Serial Wire Debug Related Policies

The serial wire debug related policies configure the functionality which is available through the device's physical debug interface (SWD). By default, MSPM0 devices come from TI in an unrestricted state. This state allows for easy production programming, evaluation, and development. However, this unrestricted state is not recommended for mass production, as it leaves a large attack surface present. To accommodate a variety of needs while keeping the configuration process simple, MSPM0 devices support three generic security levels: no restrictions (Level 0), custom restrictions (Level 1), and fully restricted (Level 2). Table 1-4 shows the three generic security levels, from least restrictive to most restrictive.

There are 2 main uses of the SWD interface for which protection needs to be considered:

- Application debug access, which includes:
  - Full access to the processor, memory map, and peripherals through the AHB-AP
  - Access to the device EnergyTrace+ state information through the ET-AP
  - Access to the device power state controls for debug through the PWR-AP
- Factory reset access, which includes:
  - Ability to send a command through SWD to erase the MAIN memory region and reset the NONMAIN device configuration memory to TI factory defaults (Level 0)

Table 1-4. Generic Security Levels

Level	Scenario	SW-DP Policy	App Debug Policy	Mass Erase Policy	Factory Reset Policy	TI FA Policy
0	No restrictions	EN	EN	DIS	EN	EN
1	Custom restrictions	EN	EN, DIS	DIS	EN, DIS	EN, DIS
2	Fully restricted	DIS	Don't care (access not possible with SW-DP disabled)(1)			

(1) When the SW-DP policy is **SW-DP disabled**, the mass erase and factory reset policies are a don't care from the point of view of the SWD interface.

1.4.2.1.1 SWD Security Level 0

SWD security level 0 is the least restrictive SWD security state. This is the default state of a new device from TI, and it is also the state of a device following a successful factory reset. There are no restrictions on application debug access, mass erase, factory reset, for failure analysis in this state.

When to Use This State

Level 0 is well suited for prototyping and development, as it allows programming of the device memory and debug of the processor and peripherals.

When to Not Use this State

Level 0 should not be used in mass production. An attacker would have full freedom to read the contents of the device memory, manipulate the execution of the device, and possibly change the flash memory contents (depending on the flash memory write protection scheme).

1.4.2.1.2 SWD Security Level 1

SWD security level 1 allows for a customized security configuration. The physical debug port (SW-DP) is left enabled, and each function (application debug, mass erase command, factory reset command, and TI failure analysis) may be individually enabled, disabled, or (in some cases) enabled through password authentication, providing considerable flexibility to tailor the device behavior to specific use-cases.

When to Use This State

Level 1 is well suited for restricted prototyping/development scenarios and for mass production scenarios where the desire is to retain certain SWD functions (such as factory reset and TI failure analysis) while disabling other functions (such as application debug). Common examples of Level 1 customized configurations are given in Table 1-5.

Table 1-5. Examples of Level 1 Configurations

Level 1 Scenario	Configuration			
	App Debug	Mass Erase	Factory Reset	TI FA
This scenario restricts debug access with a user-specified password, but it leaves the factory reset and TI failure analysis available. This configuration allows field debug (with password), and it also allows the device to be brought back to the default "Level 0" state through factory reset.	EN with PW	DIS	EN	EN
This scenario does not allow debug. It does allow factory reset, but only with a user-specified password. This provides a way to open up a device in the field by clearing the MAIN memory contents and bringing the device back to a "Level 0" state if the password is known. Importantly, even if the factory reset password were compromised, it would not be possible for an attacker to read proprietary information in the MAIN flash memory.	DIS	DIS	EN with PW	EN
This scenario does not allow debug and it does not allow TI failure analysis. This prevents TI from performing a factory reset and further FA activities on the device, unless the user executes a factory reset with their user-specified password before returning the devices to TI for FA.	DIS	DIS	EN with PW	DIS

Note

Level 1 is the recommended configuration for most standard production use-cases. For applications which do not require secure boot, TI recommends using Level 1 in production with factory reset left enabled (with password) and TI failure analysis left enabled. In such a configuration, the device may be recovered to a less restrictive state after provisioning either by the user (with password) or by TI (through the failure analysis return flow). In use-cases requiring maximum secure boot assurance, a more restrictive Level 1 or Level 2 may be used for production, with the trade-off that devices may not be recoverable to a less restrictive state once provisioned.

When to Not Use this State

Level 1 should not be used during prototyping if complete access to the device is desired; in such a case, Level 0 should be used instead.

Level 1 should also not be used in a mass production scenario where a maximally restrictive state is desired and no SWD functions are to be enabled; in such a case, Level 2 should be used instead as it directly disables the complete SWD physical interface and minimizes the possibility of misconfiguration.

Note

If a device is configured with application debug and factory reset disabled, the only way for a user to restore debug access to the device is if the user application code provides a mechanism to change the NONMAIN configuration to a less restrictive state. If the NONMAIN is locked through static write protection then the state is not reversible and there is no way for a user to re-gain debug access.

1.4.2.1.3 SWD Security Level 2

SWD security level 2 configures the device in a maximally restrictive state. The physical debug port (SW-DP) is completely disabled, and all of the SWD-accessible functions (application debug, mass erase, factory reset, and TI failure analysis) are not accessible through SWD, regardless of their individual configuration.

When level 2 is selected (SW-DP disabled), the application debug configuration and TI failure analysis configuration fields are don't care fields which do not impact the device configuration.

If the BSL is disabled, then the mass erase and factory reset configuration fields are also don't care fields. However, if the BSL is enabled, then the mass erase and factory reset configuration fields are still used by the BSL to authorize mass erase or factory reset commands originating from the BSL interface.

When to Use This State

Use Level 2 only for mass production when no further access to any SWD functions is required and a maximally secure state is desired for the device.

When to Not Use this State

Do not use Level 2 in the following cases:

- Future application debug or reprogramming through SWD is required
- So that TI can perform failure analysis on the device
- To remove proprietary information from the flash memory by sending a mass erase or factory reset command through SWD

Note

After a device is configured for level 2 (SW-DP disabled), further access to the device through SWD **is not possible**. The only way to bring a device back to a level 0 or level 1 state with SWD access restored is if a mechanism in the user application code is included which can change the NONMAIN configuration to a less restrictive state. If the NONMAIN is locked through static write protection then the level 2 state is not reversible and there is no way to re-gain SWD access.

1.4.2.2 SWD Factory Reset Commands

The BCR provides mass erase and factory reset functionality through commands sent to the device over SWD from a debug probe using the debug subsystem mailbox (DSSM). These commands are not available in SWD security level 2, but they are optionally available in security level 0 and 1. When the device is not configured for SWD security level 2, the factory reset command can be individually configured to be enabled, enabled with a unique 128-bit password, or disabled. By default, both commands are enabled.

The SWD factory reset DSSM commandsuperseded any static write protection policies. For example, if SWD factory reset is configured to be enabled or enabled, the BCR configuration data can be reset even if it is statically write protected.

SWD Factory Reset

A SWD factory reset is an erase of the MAIN flash regions followed by a reset of the NONMAIN flash region to default values. Such an erase is useful for completely resetting the BCR and BSL device boot policies while also erasing the application code and data.

To set the factory reset command mode, configure the BOOTCFG3.FACTORYRESETCMDACCESS fieldfieldin the NONMAIN memory.

1.4.2.3 Flash Memory Protection and Integrity Related Policies

The flash memory protection and integrity policies specify which sectors of flash memory are locked from modification, as well as which sectors are to be checked for integrity during the boot process before the user application is started.

1.4.2.3.1 Locking the Application (MAIN) Flash Memory

MSPM0 MCUs implement a static write protection scheme to lock out user defined sectors in the MAIN flash region from any program or erase operations at runtime. The desired static write protection scheme is configured as a part of the boot security policies in the NONMAIN flash region.

Purpose

Static write protection enables placement of a fixed, user-defined, application in the flash memory that has the following characteristics:

- Once programmed and locked, the application is not modifiable by the application code or ROM bootloader
- If placed at the beginning of the flash memory, the application is the first code that executes when the ROM boot configuration routine transfers execution to the user application

MSPM0 static write protection supports both characteristics, which must be satisfied to implement a secure boot image manager.

Capabilities

Any sector that is configured in the NONMAIN to be write-locked is functionally immutable when the boot configuration routine transfers execution to either the bootstrap loader or the user application code in MAIN flash. Any attempt to program or erase a statically protected sector by the application code or the bootstrap loader results in a hardware flash operation error, and the sector is not modified.

While static write protection prevents any modification by application code or the bootloader, a mass erase or factory reset command sent through the SWD interface is honored. If this behavior is not desired, the factory reset SWD commandcan be protected or disabled(see the SWD policies). To completely remove any means of modifying statically write protected MAIN flash sectors, the factory reset command (or the SW-DP) must be disabled, and the NONMAIN boot configuration memory must also be statically write protected to prevent application code from changing the underling write protection scheme by modifying the contents NONMAIN region. This is discussed in the following section.

1.4.2.3.2 Locking the Configuration (NONMAIN) Flash Memory

MSPM0 MCUs implement a static write protection scheme to lock out the NONMAIN flash region from any program/erase operations at runtime. The write protection scheme is configured as a part of the boot security policies in the NONMAIN flash region.

Purpose

By default, the NONMAIN configuration memory (which contains the user-specified boot security policies and bootstrap loader policies) is not write protected. This enables the NONMAIN to be erased by the user during provisioning and re-programmed with the user-specified policies to use in mass production.

In many cases, it is desirable for the configuration memory to be locked after it has been provisioned. Locking the configuration memory has the benefit of preventing any unauthorized modification of the security policies, bootstrap loader policies, and static write protection policies by either the bootstrap loader or the application code. In most applications, devices in mass production do not require modification of the configuration memory, even when the device firmware is updated.

Capabilities

When configured to be protected, the entire NONMAIN region will be write-locked and will be functionally immutable when the boot configuration routine transfers execution to either the bootstrap loader or the user application code in MAIN flash. Any attempt to program or erase the NONMAIN by the application code or the bootstrap loader will result in a hardware flash operation error, and the sector will not be modified.

While static write protection prevents any modification by application code or the boot loader, a factory reset command sent through the SWD interface would still be honored. If this behavior is not desired, the factory reset SWD command may be protected or disabled altogether (see the SWD policies). To completely remove any means of modifying the NONMAIN configuration memory, the factory reset command and TI FA (or the SW-DP) must be disabled.

Note

When the NONMAIN is statically write protected, and the factory reset command and TI FA (or the SW-DP) are disabled, the NONMAIN is equivalent to immutable read-only memory, and it is no longer possible to change the device configuration by any means. Further, if any MAIN memory region sectors are configured with static protection, these sectors also can not be modified by any means and may be considered as immutable.

1.4.2.3.3 Static Write Protection NONMAIN Fields

Write protection may be enabled on a per-sector basis for the first 32 sectors of the MAIN flash memory. For the remaining sectors of flash memory, if present, write protection may be enabled per 8 sectors. To set a static write protection policy, configure the FLASHSWP0 and FLASHSWP1 fields in the NONMAIN memory.

Note

Mass erase and factory reset commands Factory reset commandsent to the BCR via the debug sub system mailbox (DSSM) will override the specified static write protection policy. If this behavior is not desired, configure the factory reset commandto be enabled with password or disabled.

1.5 NONMAIN_C1103_C1104 Registers

Table 1-6 lists the memory-mapped registers for the NONMAIN_C1103_C1104 registers. All register offset addresses not listed in Table 1-6 should be considered as reserved locations and the register contents should not be modified.

Table 1-6. NONMAIN_C1103_C1104 Registers

Offset	Acronym	Register Name	Section
41C00000h	BCRCONFIGID	Configuration ID of BCR Structure.	Section 1.5.1
41C00004h	BOOTCFG0	Serial wire debug (SWD) lock policy.	Section 1.5.2
41C00008h	BOOTCFG1	Factory reset mode and static write protection for NONMAIN.	Section 1.5.3
41C0000Ch	FLASHSWP0	Programs static write protection of first 32K bytes.	Section 1.5.4
41C00010h	FLASHSWP1	Programs static write protection of first 32K bytes.	Section 1.5.5
41C00014h	RESERVED		

Complex bit access types are encoded to fit into small table cells. Table 1-7 shows the codes that are used for access types in this section.

Table 1-7. NONMAIN_C1103_C1104 Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value

1.5.1 BCRCONFIGID Register (Offset = 41C00000h) [Reset = 00000003h]

BCRCONFIGID is shown in Figure 1-2 and described in Table 1-8.

Return to the Table 1-6.

Configuration ID of BCR Structure.

Figure 1-2. BCRCONFIGID Register

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
CONFIG																															
R/W-3h																															

Table 1-8. BCRCONFIGID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CONFIG	R/W	3h	Configuration ID of the BOOTCFG

1.5.2 BOOTCFG0 Register (Offset = 41C00004h) [Reset = AABBAABBh]

BOOTCFG0 is shown in Figure 1-3 and described in Table 1-9.

Return to the Table 1-6.

Serial wire debug (SWD) lock policy.

Figure 1-3. BOOTCFG0 Register

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
SWDP_MODE																DEBUGACCESS															
R/W-AABBh																W-AABBh															

Table 1-9. BOOTCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SWDP_MODE	R/W	AABBh	The serial wire debug port (SW-DP) access policy. This policy sets whether any communication is allowed with the device via the SWD pins (to any DAP). When disabled, no SWD communication is possible regardless of the configuration of the DEBUGACCESS field. AABBh = Enabled; FFFFh = Disabled (all other values).
15-0	DEBUGACCESS	W	AABBh	The debug access policy for accessing the AHB-AP, ET-AP, and PWR-AP debug access ports. Note that if SWDP_MODE is set to DISABLED, the value of this field is ignored and the debug port will remain fully locked. AABBh = Access to AHB-AP, ET-AP, and PWR-AP via SWD is enabled; FFFFh = Access to AHB-AP, ET-AP, and PWR-AP via SWD is disabled (all other values).

1.5.3 BOOTCFG1 Register (Offset = 41C00008h) [Reset = FFFFAABBh]

BOOTCFG1 is shown in Figure 1-4 and described in Table 1-10.

Return to the Table 1-6.

Factory reset mode and static write protection for NONMAIN.

Figure 1-4. BOOTCFG1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							NONMAINSWP
R-0h							R/W-1h
15	14	13	12	11	10	9	8
factoryResetMode							
R/W-AABBh							
7	6	5	4	3	2	1	0
factoryResetMode							
R/W-AABBh							

Table 1-10. BOOTCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	7FFFh	
16	NONMAINSWP	R/W	1h	Static Write Protection configuration for NONMAIN. 0h = Disabled (all other values). 1h = Enabled;
15-0	factoryResetMode	R/W	AABBh	Static Write Protection configuration for NONMAIN. AABBh = Enabled; FFFFh = Disabled (all other values).

1.5.4 FLASHSWP0 Register (Offset = 41C0000Ch) [Reset = FFFFFFFFh]

FLASHSWP0 is shown in Figure 1-5 and described in Table 1-11.

Return to the Table 1-6.

Programs static write protection of first 32K bytes.

Figure 1-5. FLASHSWP0 Register

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
MAINLOW																															
R/W-FFFFFFFFh																															

Table 1-11. FLASHSWP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MAINLOW	R/W	FFFFFFFFh	1 bit per sector (Setting a bit to 0 disables write, 1 enables write).

1.5.5 FLASHSWP1 Register (Offset = 41C00010h) [Reset = FFFFFFFFh]

FLASHSWP1 is shown in Figure 1-6 and described in Table 1-12.

Return to the Table 1-6.

Programs static write protection of first 32K bytes.

Figure 1-6. FLASHSWP1 Register

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
MAINHIGH																															
R/W-FFFFFFFFh																															

Table 1-12. FLASHSWP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MAINHIGH	R/W	FFFFFFFFh	1 bit per 8 sectors. Bits 3:0, not used as covered with above (Setting a bit to 0 disables write, 1 enables write).

1.6 Factory Constants

All devices include a memory-mapped FACTORY region which provides read-only data describing the capabilities of a device as well as any factory-provided trim information for use by application software.

Key data provided in the FACTORY memory region includes:

- The device unique 96-bit identity
- The total MAIN region flash memory size (in KB)
- The total DATA region flash memory size (in KB), if present
- The flash bank count
- The total SRAM memory size (in KB)
- The temperature sensor calibration value

1.6.1 FACTORYREGION Registers

Table 1-13 lists the memory-mapped registers for the FACTORYREGION registers. All register offset addresses not listed in Table 1-13 should be considered as reserved locations and the register contents should not be modified.

Table 1-13. FACTORYREGION Registers

Offset	Acronym	Register Name	Section
41C40000h	TRACEID	Trace identifier	Go
41C40004h	DEVICEID	Device identifier	Go
41C40008h	USERID	Device variant identifier	Go
41C40018h	SRAMFLASH	The encoding of memory size and flash bank number	Go
41C4003Ch	TEMP_SENSE0	Temperature sensor room temperature calibration code. This is ADC conversion results of temperature sensor output voltage.	Go

Complex bit access types are encoded to fit into small table cells. Table 1-14 shows the codes that are used for access types in this section.

Table 1-14. FACTORYREGION Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value

1.6.1.1 TRACEID Register (Offset = 41C40000h) [Reset = 00000000h]

TRACEID is shown in Figure 1-7 and described in Table 1-15.

Return to the Summary Table.

Unique value per device shipped

Figure 1-7. TRACEID Register

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
DATA																															
R-0h																															

Table 1-15. TRACEID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	

1.6.1.2 DEVICEID Register (Offset = 41C40004h) [Reset = 0BBA102Fh]

DEVICEID is shown in Figure 1-8 and described in Table 1-16.

Return to the Summary Table.

Device identifier (die revision specific)

Figure 1-8. DEVICEID Register

31	30	29	28	27	26	25	24
VERSION				PARTNUM			
R-0h				R-BBA1h			
23	22	21	20	19	18	17	16
PARTNUM							
R-BBA1h							
15	14	13	12	11	10	9	8
PARTNUM				MANUFACTURER			
R-BBA1h				R-17h			
7	6	5	4	3	2	1	0
MANUFACTURER							ALWAYS_1
R-17h							R-1h

Table 1-16. DEVICEID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	VERSION	R	0h	Revision of the device
27-12	PARTNUM	R	BBA1h	Part number of the device
11-1	MANUFACTURER	R	17h	TI's JEDEC bank and company code
0	ALWAYS_1	R	1h	This is always 1

1.6.1.3 USERID Register (Offset = 41C40008h) [Reset = X0000000h]

USERID is shown in Figure 1-9 and described in Table 1-17.

Return to the Summary Table.

The variant feature set

Figure 1-9. USERID Register

31	30	29	28	27	26	25	24
START	MAJORREV			MINORREV			
R-X	R-0h			R-0h			
23	22	21	20	19	18	17	16
VARIANT							
R-0h							
15	14	13	12	11	10	9	8
PART							
R-0h							
7	6	5	4	3	2	1	0
PART							
R-0h							

Table 1-17. USERID Register Field Descriptions

Bit	Field	Type	Reset	Description
31	START	R	X	
30-28	MAJORREV	R	0h	Monotonic increasing value indicating a new revision significant enough that users of the device may have to revise PCB or software design
27-24	MINORREV	R	0h	Monotonic increasing value indicating a new revision that preserves compatibility with lesser MINORREV values. New capability may be introduced such that lesser MINORREV numbers may not be compatible with greater if the new capability is used.
23-16	VARIANT	R	0h	Bit pattern uniquely identifying the variant of a part
15-0	PART	R	0h	Bit pattern that uniquely identifying a part

1.6.1.4 SRAMFLASH Register (Offset = 41C40018h) [Reset = XXXXXXXXh]

SRAMFLASH is shown in Figure 1-10 and described in Table 1-18.

Return to the Summary Table.

The encoding of memory size and flash bank number

Figure 1-10. SRAMFLASH Register

31	30	29	28	27	26	25	24
DATAFLASH_SZ						SRAM_SZ	
R-X						R-X	
23	22	21	20	19	18	17	16
SRAM_SZ							
R-X							
15	14	13	12	11	10	9	8
RESERVED		MAINNUMBANKS		MAINFLASH_SZ			
R-0h		R-X		R-X			
7	6	5	4	3	2	1	0
MAINFLASH_SZ							
R-X							

Table 1-18. SRAMFLASH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	DATAFLASH_SZ	R	X	The encoding of the field is that the value of the field is an integer to be interpreted as number of KB. For example, if the value of the field id 4, then it is 4KB, if the value is 32, then 32KB, and so on.
25-16	SRAM_SZ	R	X	The encoding of the field is that the value of the field is an integer to be interpreted as number of KB. For example, if the value of the field id 4, then it is 4KB, if the value is 32, then 32KB, and so on.
15-14	RESERVED	R	0h	
13-12	MAINNUMBANKS	R	X	The encoding of the field is that the value of the field is an integer to be interpreted as number of banks. 0 is one bank, 1 is two banks, 2 is three banks and 3 is 4 banks. 0h = 0 1h = 1 2h = 2 3h = 3
11-0	MAINFLASH_SZ	R	X	The encoding of the field is that the value of the field is an integer to be interpreted as number of KB. For example, if the value of the field id 4, then it is 4KB, if the value is 32, then 32KB, and so on.

1.6.1.5 TEMP_SENSE0 Register (Offset = 41C4003Ch) [Reset = 00000000h]

TEMP_SENSE0 is shown in Figure 1-11 and described in Table 1-19.

Return to the Summary Table.

Temperature sensor room temperature calibration code. This is ADC conversion results of temperature sensor output voltage.

Figure 1-11. TEMP_SENSE0 Register

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
DATA																															
R-0h																															

Table 1-19. TEMP_SENSE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	

Chapter 2

PMCU

The power management and clock unit (PMCU) is a unified system module which provides all power management, clock configuration, and reset control functionality for the device. All power management unit (PMU) and clock module (CKM) policies for device operation are configured through memory-mapped registers in the system controller (SYSCTL).

2.1 PMCU Overview

The power management and clock unit (PMCU) provides all power, clocking, reset, and system control services for the device. The PMCU contains three submodules to provide this functionality: the power management unit (**PMU**), the clock module (**CKM**), and the system controller (**SYSCTL**).

The PMU is an analog submodule that generates the internal regulated supplies for the device and supervises the condition of the external supply. The PMU also contains voltage and current reference circuits used by the on-chip regulators and analog peripherals.

The CKM is an analog submodule that provides clock sources (internal oscillators) and presents these clock sources to SYSCTL. SYSCTL distributes these clock sources to the CPU, buses, and peripherals on the device.

The SYSCTL is a digital submodule that provides the control logic for all functions in the PMCU. In addition, SYSCTL contains the memory-mapped registers used by software to configure power management and clocks, assess the status of the device, and control resets. SYSCTL also provides 4 bytes of general-purpose memory that is retained in SHUTDOWN mode and can be used to store status information in SHUTDOWN mode when SRAM and register contents are lost.

Figure 2-1 shows the interfaces between the PMCU and the device supplies, clocks, and signals. Configuration of the PMCU by software is always done through memory-mapped registers in the SYSCTL submodule.

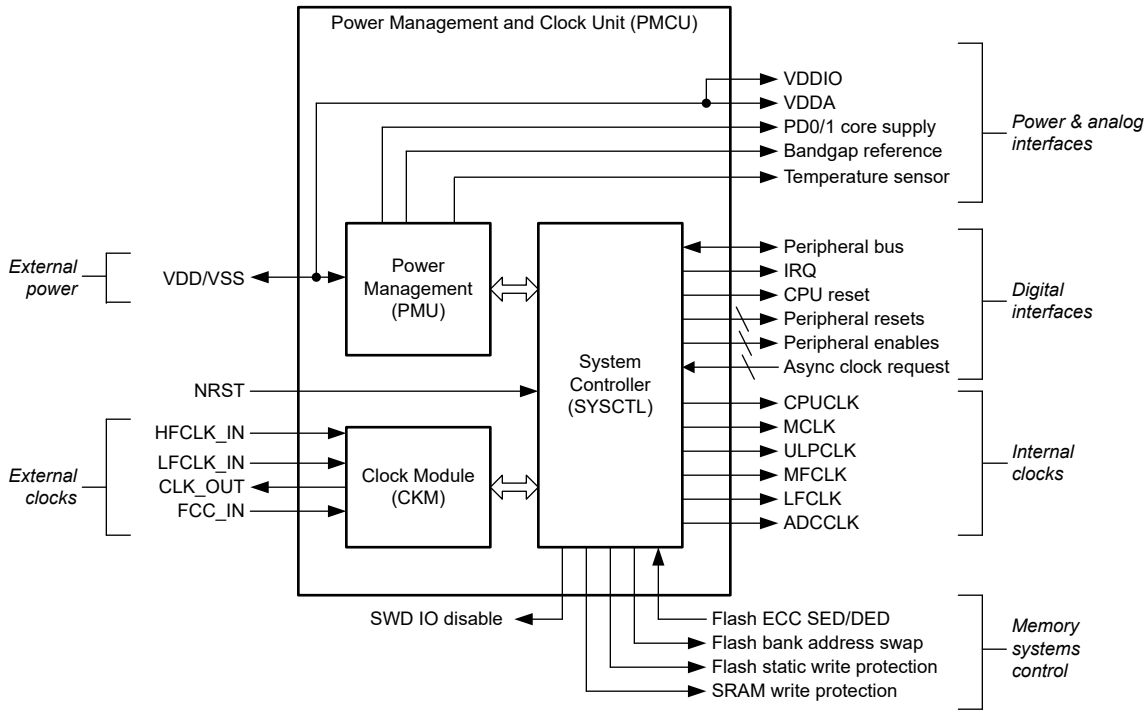


Figure 2-1. PMCU Top-Level Diagram

Note

Not all devices have all of the PMCU features shown in Figure 2-1. See the device-specific data sheet to understand the features present on a given device.

Using this Guide

The PMU, CKM, and SYSCTL sections of this chapter describe the functionality provided by each submodule in detail.

The quick start section describes overall system level operation of the PMCU and how to configure the PMCU for different application scenarios.

2.1.1 Power Domains

Two core power domains are provided on the device: PD1 and PD0. PD1 is always powered in RUN and SLEEP modes, but is disabled in all other modes. PD0 is always powered in RUN, SLEEP, STOP and STANDBY modes. PD1 and PD0 are both disabled in SHUTDOWN mode.

- The PD1 domain includes the CPU subsystem, the SRAM memory, PD1 peripherals, and the PD1 peripheral bus, which runs from MCLK (including the DMA) with a maximum frequency of 24MHz. While PD1 is disabled in STOP and STANDBY mode, the CPU registers, SRAM, and peripheral MMR configuration registers are maintained in retention such that they are available to resume operation immediately when STOP or STANDBY modes are exited.
- The PD0 domain includes the PD0 peripherals and PD0 bus segment which runs from ULPCLK with a max frequency of 24MHz in RUN and SLEEP mode, 4MHz in STOP mode, and 32kHz in STANDBY mode. The PD0 domain is powered in all modes except SHUTDOWN and can be thought of as an "always-on" domain.

The device-specific data sheet describes which peripherals on a device are in PD1 and which are in PD0.

The device also has a single external supply (VDD) domain that provides power to the IO and analog peripherals.

2.1.2 Operating Modes

Five operating modes (power modes) are provided to allow for optimization of the device power consumption based on application requirements. In order of decreasing power, the modes are: RUN, SLEEP, STOP, STANDBY. The following figure shows the interaction between the modes.

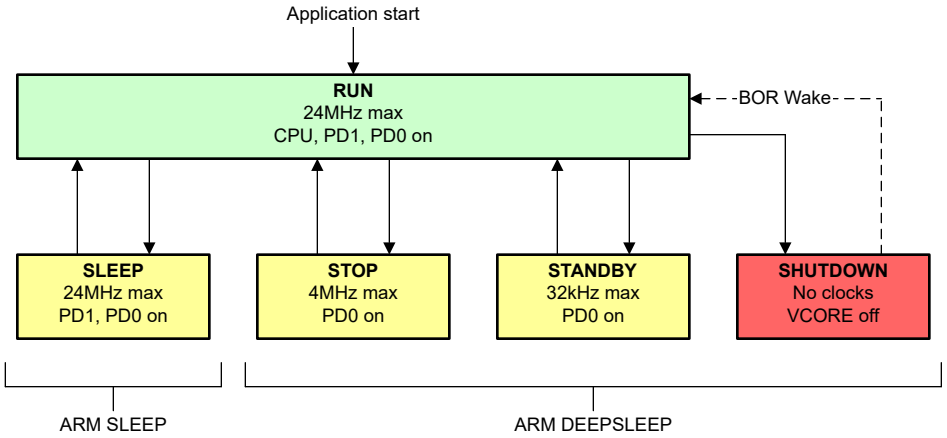


Figure 2-2. MSPM0Cxx Operating Modes

Section 2.1.2.6 indicates what functions are available in each operating mode of the device. See the operating mode selection section for information on how to configure the device for a particular operating mode.

Operating Mode Concept

MSPM0 MCUs implement a policy-based power and clock management scheme. Policies can be configured through application software for how the clocking is to managed in each operating mode to obtain the best balance of power and performance for a given application.

After the operating policy for each mode is configured, application software can enter and exit the various operating modes through simple register commands, and SYSCTL automatically manages all the necessary PMU states, oscillator and clock enable and disables, and the SYSOSC frequency according to the software-defined policies and software-selected mode.

A variety of hardware-triggered low-power mode suspension mechanisms also exist to enable on-demand access to a fast clock when requested by supported peripherals, as well as functions such as DMA and ADC triggering from low-power modes.

The policy-driven operating mode scheme together with the asynchronous low-power mode suspension mechanisms enable application software to select the operating mode and corresponding policy that provide the lowest possible power consumption for background activities, with transient foreground activities either bringing up the DMA, bringing up a fast clock, or bringing the device to RUN (in the case of an IRQ) for burst handling.

2.1.2.1 RUN Mode

In RUN mode, the CPU is active executing code and any peripheral can be enabled.

There are three RUN mode policy options: RUN0, RUN1, and RUN2.

- **RUN0**: The MCLK and the CPUCLK run from a fast clock source (SYSOSC).
- **RUN1**: The MCLK and the CPUCLK run from LFCLK (at 32kHz) to reduce active power, but SYSOSC is left enabled to service analog modules such as an ADC, .
- **RUN2**: The MCLK and the CPUCLK run from LFCLK (at 32kHz), and SYSOSC is completely disabled to save power. This is the lowest power state with the CPU running.

2.1.2.2 SLEEP Mode

In SLEEP mode, the CPU is disabled (clock gated) but otherwise the device configuration is the same as RUN. There are three SLEEP mode policy options: SLEEP0, SLEEP1, and SLEEP2. The SLEEPx policy is determined by the current RUNx policy when SLEEP mode is entered.

- **SLEEP0**: Identical to RUN0, with the CPU disabled.
- **SLEEP1**: Identical to RUN1, with the CPU disabled.
- **SLEEP2**: Identical to RUN2, with the CPU disabled.

2.1.2.3 STOP Mode

In STOP mode, the CPU, SRAM, and PD1 peripherals are disabled and in retention (if applicable). PD0 peripherals are available with a max ULPCLK frequency of 4MHz. SYSOSC can run at higher frequencies to support ADC operation, but ULPCLK will be automatically limited to the 4MHz SYSOSC output by SYSCTL. High speed oscillator HFCLK_IN is automatically disabled.

DMA is available to be triggered. A DMA trigger wakes the PD1 power domain to make the SRAM and DMA available for processing the DMA transfer, and the DMA transfer is processed at the current MCLK and ULPCLK rate. After the transfer completes, the SRAM is returned to retention and PD1 is disabled automatically.

STOP mode is the lowest power mode that supports ADC operation.

There are three policy options for STOP mode: STOP0 and STOP2.

- **STOP0**: The SYSOSC is left running at the current frequency when entering STOP mode. ULPCLK is always limited to 4MHz automatically by hardware, but SYSOSC is not disturbed to support consistent operation of analog peripherals.
  - NOTE: If STOP0 is entered from RUN1 (SYSOSC enabled but MCLK sourced from LFCLK), SYSOSC remains enabled as in RUN1, and ULPCLK remains at 32kHz as in RUN1.
  - NOTE: If STOP0 is entered from RUN2 (SYSOSC was disabled and MCLK was sourced from LFCLK), SYSOSC remains disabled as in RUN2, and ULPCLK remains at 32kHz as in RUN2.
- **STOP2**: The SYSOSC is disabled and the ULPCLK is sourced from LFCLK at 32kHz. This is the lowest power state in STOP mode.

2.1.2.4 STANDBY Mode

In STANDBY mode, the CPU, SRAM, and PD1 peripherals are disabled and in retention. PD0 peripherals, with the exception of the ADC and OPA, are available with a maximum ULPCLK frequency of 32kHz. . High-speed oscillators HFCLK_IN and SYSOSC are disabled.

DMA is available to be triggered. A DMA trigger wakes the PD1 power domain to make the SRAM and DMA available for processing the DMA transfer, and the DMA transfer is processed at the current MCLK and ULPCLK rate (32kHz). After the transfer completes, the SRAM is returned to retention and PD1 is disabled automatically.

ADC operation is not supported in STANDBY mode.

There are 2 policy options for STANDBY mode: STANDBY0 and STANDBY1.

- **STANDBY0**: All PD0 peripherals receive the ULPCLK and LFCLK.
- **STANDBY1**: Only a few general purpose timers receive ULPCLK or LFCLK, see the device-specific data sheet to determine them. A general timer interrupt or ADC trigger in STANDBY1 always triggers an asynchronous fast clock request to wake the system. Other PD0 peripherals (such as UART, I2C, GPIO) can also wake the system upon an external event through an asynchronous fast clock request, but they are not actively clocked in STANDBY1.

2.1.2.5 SHUTDOWN Mode

In SHUTDOWN mode, no clocks are available. The core regulator is completely disabled and all SRAM and register contents are lost, with the exception of the 4 bytes of general-purpose memory in SYSCTL that can be used to store state information. The BOR and bandgap circuit are disabled.

The device can wake through a wake-up capable IO, a debug connection or NRST. .

SHUTDOWN mode has the lowest current consumption of any operating mode. Exiting SHUTDOWN mode triggers a BOR.

2.1.2.6 Supported Functionality by Operating Mode

Supported functionality in each operating mode is given in *Supported Functionality by Operating Mode* table in the detailed description section of the device-specific data sheet.

Functional key:

- **EN**: The function is enabled in the specified mode.
- **DIS**: The function is disabled (either clock or power gated) in the specified mode, but the function's configuration is retained.
- **OPT**: The function is optional in the specified mode, and remains enabled if configured to be enabled.
- **NS**: The function is not automatically disabled in the specified mode, but its use is not supported.
- **OFF**: The function is fully powered off in the specified mode, and no configuration information is retained.

2.1.2.7 Suspended Low-Power Mode Operation

Some peripherals can be configured to temporarily suspend STOP or STANDBY mode operation to handle a temporary activity or process an event. There are two ways in which STOP or STANDBY mode can be suspended:

- An asynchronous fast clock request
- A DMA trigger

Suspended STOP or STANDBY for an Asynchronous Fast Clock Request

An asynchronous fast clock request temporarily suspends any active low-power mode and runs the MCLK and ULPCLK tree at 24MHz/32MHz, sourced from SYSOSC. Asynchronous fast clock requests are also functional in RUN and SLEEP mode if MCLK is sourced from either LFCLK at 32kHz or SYSOSC at a frequency lower than 24MHz/32MHz. While asynchronous fast clock requests suspend the low-power mode and change clock tree configuration to support 24MHz/32MHz operation, these requests do not enable the PD1 power domain if the device was in STOP or STANDBY mode. This functionality enables use cases such as:

- and a few general purpose timers wakeup from STANDBY1
- On-demand UART, I^2C, or SPI communication
- Timer-triggered ADC sampling from STANDBY mode

Suspended STOP or STANDBY for a DMA Trigger

If a DMA trigger is asserted in STOP or STANDBY mode, the low-power mode is temporarily suspended and the PD1 power domain (including the SRAM and flash memory) is enabled to process the DMA request. Unlike the

asynchronous fast clock request, DMA transfers do not change the clock tree configuration. A DMA request in STOP or STANDBY mode is processed at the current ULPCLK rate.

2.2 Power Management (PMU)

The power management unit (PMU) generates the regulated core supplies for the device and provides supervision of the external supply. It also contains a bandgap voltage reference used by the PMU and other analog peripherals.

Key PMU features include:

- Support for device operation across a wide supply range (1.62V to 3.6V)
- Low-dropout linear voltage regulator to generate the internal core logic supply, with multiple operating modes for reducing device current in low-power modes (managed automatically by SYSCTL)
- Power-on reset (POR) supply monitor
- Brownout reset (BOR) supply monitor with four configurable threshold voltages
- Bandgap voltage reference supporting the BOR, core regulator, and analog peripherals
- Temperature sensor with connection to the ADC

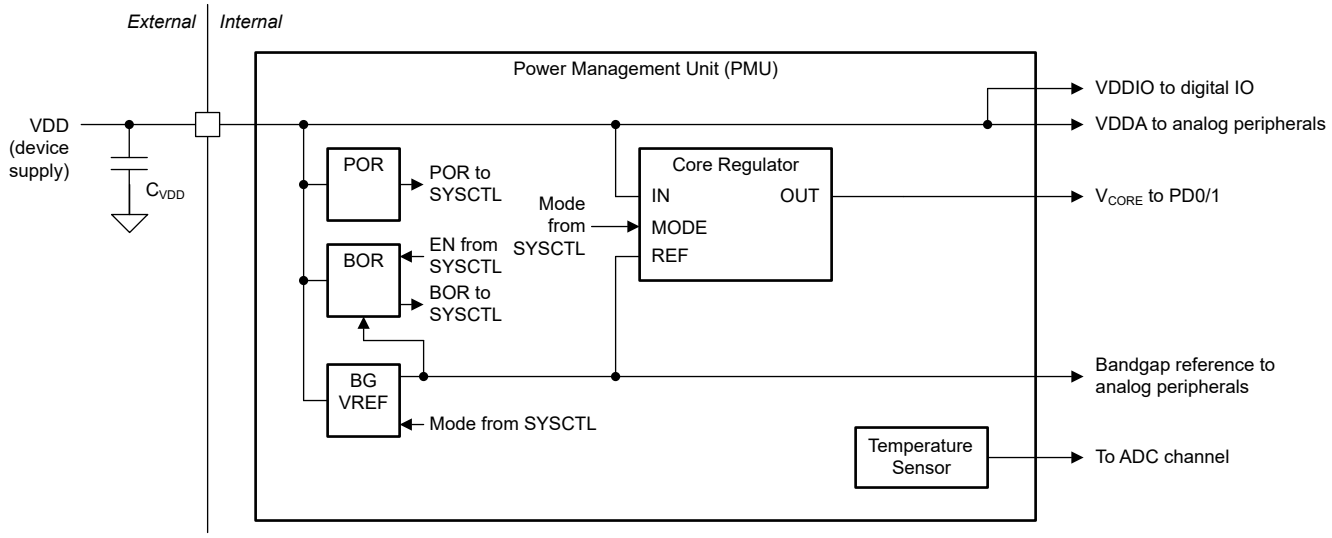


Figure 2-3. MSPM0Cxx PMU Block Diagram

2.2.1 Power Supply

Power is supplied to the device through the VDD and VSS connections. The device supports operation with a supply voltage of 1.62V to 3.6V and will start with a 1.62V supply. A decoupling capacitor (C_{VDD}) must be placed across all VDD and VSS supply pairs. See the device-specific data sheet for the correct value and tolerance of C_{VDD}. Products with 64 pins or less typically have a single VDD/VSS power pair.

VDD is used directly to provide the IO supply (VDDIO) and the analog supply (VDDA). VDDIO and VDDA are internally connected to VDD so that additional power supply pins are not required.

2.2.2 Core Regulator

In all other power modes (RUN, SLEEP, STOP, and STANDBY) the drive strength of the regulator is configured automatically to support the max load current of each mode. This reduces the quiescent current of the regulator when using low-power modes, improving low power performance. SYSCTL automatically configures the regulator for best power consumption based on the power mode which is currently active.

2.2.3 Supply Supervisors

The PMU provides two supply supervisor circuits:

- A power-on reset (POR) circuit to indicate that the external supply has reached sufficient voltage to start the on-chip bandgap reference and BOR circuit
- A user-programmable brownout reset (BOR) circuit which ensures that the external supply is maintained at a sufficient voltage to support correct operation of the device

2.2.3.1 Power-on Reset (POR) Supervisor

The power-on reset (POR) supervisor monitors the external supply (VDD) and asserts or de-asserts a POR violation to SYSCTL. During cold power-up, the device is held in a POR state until VDD passes the POR+ threshold. When VDD has passed POR+, the POR state is released and the bandgap reference and BOR monitor circuit are started. If VDD drops below the POR- level, then a POR- violation is asserted and the device is again held in a POR reset state.

The POR supervisor does not indicate that VDD has reached a level high enough to support correct operation of the device. Rather, it is the first step in the boot process and is used to determine if the supply voltage is sufficient to power up the bandgap reference and BOR circuit, which are then used to determine if the supply has reached a level sufficient to for the device to run correctly.

The POR supervisor is active in all power modes including SHUTDOWN and cannot be disabled.

2.2.3.2 Brownout Reset (BOR) Supervisor

The brownout reset (BOR) supervisor monitors the external supply (VDD) and asserts or de-asserts a BOR violation to SYSCTL. The primary responsibility of the BOR circuit is to make sure that the external supply is maintained high enough to enable correct operation of internal circuits, including the core regulator. The BOR threshold reference is derived from the internal bandgap circuit. The threshold is programmable and is always higher than the POR threshold. During cold start, after VDD passes the POR+ threshold, the bandgap reference and BOR circuit are started. The device is then held in a BOR state until VDD passes the BOR0+ threshold. When VDD passes BOR0+, the BOR supervisor releases the device to continue the boot process, and the PMU is started.

There are four selectable BOR threshold levels (BOR0-BOR3). During startup, the BOR threshold is always BOR0 (the lowest value) to make sure that the device starts at the specified VDD minimum (1.62V). After boot, software can optionally re-configure the BOR circuit to use a different (higher) threshold level (BOR1-BOR3).

When the BOR threshold is BOR0, a BOR0- violation always generates a BOR- violation signal to SYSCTL, generating a BOR level reset. When the BOR threshold is re-configured to BOR1, BOR2, or BOR3, the BOR circuit generates a SYSCTL interrupt rather than asserting the BOR- violation. This can be used to give the application an indication that the supply has dropped below a certain level without causing a reset.

To change the BOR level from the default (BOR0), first select the desired value in the LEVEL field of the BORTHRESHOLD register in SYSCTL. Then, activate the threshold set in the LEVEL field by setting the GO bit in the BORCLRCMD register. The change can be validated by testing the BORCURTHRESOLD field in the SYSSTATUS register, which returns a value corresponding to the currently active BOR threshold. The BOR threshold change takes approximately 15µs to complete, during which time the BOR circuit is blind to changes in the supply.

If the BOR is in interrupt mode (a threshold level of BOR1-BOR3), and the supply drops below the corresponding BORx- level, an interrupt is generated and the BOR circuit automatically switches the BOR threshold level to BOR0 to make sure that a BOR- violation is asserted if VDD drops below BOR0-. Application software can set the BOR level back to the level specified in the LEVEL field of the BORTHRESHOLD register by setting the GO bit again in the BORCLRCMD register.

The BOR supervisor is active in RUN, SLEEP, STOP, and STANDBY modes but is disabled automatically in SHUTDOWN mode.

2.2.3.3 POR and BOR Behavior During Supply Changes

When the supply voltage (VDD) drops below POR-, the entire device state is cleared. Small variations in VDD that do not pass below the BOR0- threshold do not cause a BOR- violation, and the device continues to run.

Behavior for BORx thresholds other than BOR0 (for example, BOR1-BOR3) is the same as is shown for BOR0, except that the BOR circuit is configured to generate an interrupt rather than immediately triggering a BOR reset.

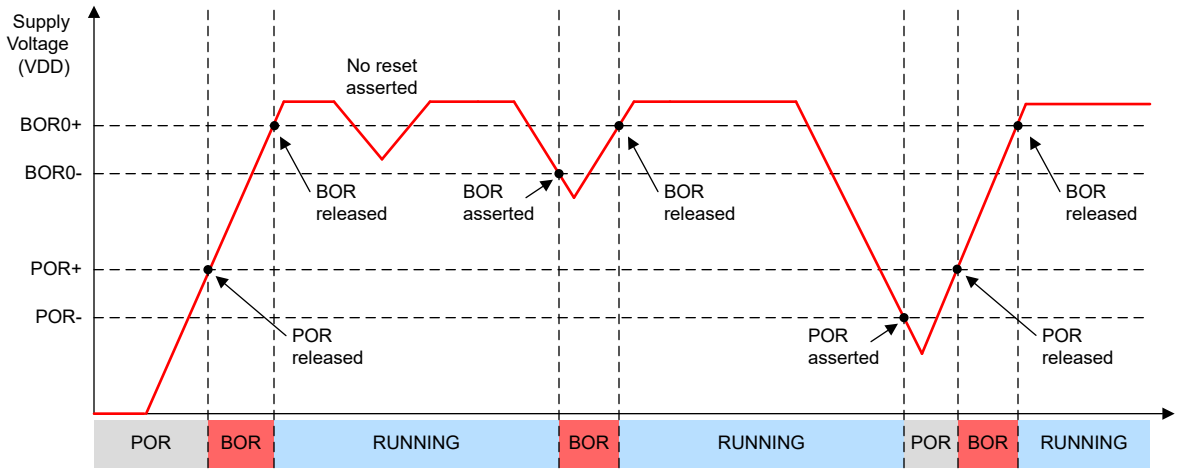


Figure 2-4. POR/BOR vs. Supply Voltage (VDD)

2.2.4 Bandgap Reference

The PMU provides a temperature and supply voltage stable bandgap voltage reference that is used by the device for internal functions, including:

- Deriving the brownout reset circuit thresholds
- Setting the output voltage for the core regulator
- Deriving the on-chip VREF levels for on-chip analog peripherals

The bandgap reference is enabled in RUN, SLEEP, STOP modes. This reference operates in a sampled mode in STANDBY to reduce power consumption and is disabled in SHUTDOWN mode. SYSCTL manages the bandgap state automatically; no user configuration is required.

2.2.5 Temperature Sensor

The PMU provides a basic temperature sensor which can be used to approximate the temperature of the device. The temperature sensor is connected internally to the ADC, and the ADC must be used to perform a temperature measurement. See the device-specific data sheet to determine the correct internal ADC channel to use when measuring the temperature sensor.

The temperature sensor outputs a voltage which has a linear relationship with temperature. The temperature coefficient (TS_c) is the slope of the temperature-voltage relationship (given in mV/C), and is given in the specifications section of the device-specific data sheet.

A unit-specific single-point trim value (TEMP_SENSE0.DATA) is provided in the factory constants memory of each device. This value indicates the temperature sensor output voltage at the factory trim temperature (TS_{TRIM}), in ADC result code format. The ADC result code in TEMP_SENSE0.DATA is based upon 12-bit sampling mode together with the 1.4V internal voltage reference. The TS_{TRIM} temperature is also given in the specifications section of the device-specific data sheet.

The approximate temperature of the device can be computed through the use of the following parameters:

- TS_c, taken from the device data sheet
- TEMP_SENSE0.DATA, taken from the unit-specific factory constants memory
- TS_{TRIM}, taken from the device data sheet
- V_{SAMPLE} (voltage sample of the temperature sensor at time of interest, taken with the ADC)

The temperature is computed through the linear relationship given in Equation 1, where V_{SAMPLE} is the current temperature sensor voltage, and V_{TRIM} is the factory calibrated temperature sensor voltage at the TS_{TRIM} temperature (derived from TEMP_SENSE0.DATA):

$$\mathbf{T_{SAMPLE}} = (1 / TS_c) * (V_{SAMPLE} - V_{TRIM}) + TS_{TRIM} \quad (1)$$

The ADC_{CODE} raw result can be converted to a voltage equivalent (V_{SAMPLE}) as shown in the relationship in Equation 2, where RES is the ADC resolution in bits, and VREF is the ADC reference voltage.

$$\mathbf{V_{SAMPLE}} = (VREF / 2^{RES}) * (ADC_{CODE} - 0.5) \quad (2)$$

Example

To illustrate the process of converting an ADC sample of the temperature sensor into an approximate device temperature, an example is given below.

Example parameters:

- TS_c = -2.04mV/C
- TEMP_SENSE0.DATA = 1857 (ADC result code based on 12-bit mode and a 1.4V reference)
- TS_{TRIM} = 30C
- ADC_{CODE} = 1677 (ADC result code based on 12-bit mode and a 1.4V reference)

First, the current temperature sensor sample voltage is calculated using Equation 3:

$$\mathbf{V_{SAMPLE}} = (1.4V / 4096) * (1677 – 0.5) = \mathbf{0.5730V} \quad (3)$$

Then, the single-point calibration voltage is calculated using the same means:

$$\mathbf{V_{TRIM}} = (1.4V / 4096) * (1857 – 0.5) = \mathbf{0.6345V} \quad (4)$$

Then, the temperature is approximated using Equation 5:

$$\mathbf{T_{SAMPLE}} = (1 / -0.002044) * (0.5730V – 0.6345) + 30°C = \mathbf{60°C} \quad (5)$$

Note

The temperature sensor is not available in STANDBY and SHUTDOWN operating modes.

2.2.6 Peripheral Power Enable Control

All peripherals on a device, with the exception of infrastructure peripherals such as SYSCTL itself and the IOMUX, contain a power enable control register (PWREN) with a KEY and ENABLE field. Before any other peripheral registers are configured by software, the peripheral itself must be enabled by writing the ENABLE bit together with the appropriate KEY value to the peripheral's PWREN register.

When a peripheral ENABLE bit is cleared, the peripheral can be considered to be inactive and the remaining peripheral-specific registers are isolated from the peripheral bus and thus are not be accessible for read/write operations.

Note

After setting the ENABLE | KEY bits in the PWREN register to enable a peripheral, wait at least 4 ULPCLK clock cycles before accessing the rest of the peripheral's memory-mapped registers. The 4 cycles allow for the bus isolation signals at the peripheral's bus interface to update.

2.2.6.1 Automatic Peripheral Disable in Low Power Modes

Peripherals in power domain 1 (PD1) will be forced to a disabled state by SYSCTL upon entry into a STOP or STANDBY low-power mode. As such, these peripherals will not be available for use in STOP or STANDBY.

Most PD1 peripherals will retain their configuration settings after being automatically disabled, such that re-configuration is not required upon exit from STOP or STANDBY mode. See the peripheral-specific chapter in this guide for details on which peripheral registers are retained through STOP and STANDBY mode for PD1 peripherals.

If a PD1 peripheral was multiplexed to an IO pin (through the IOMUX) in an output configuration, the last valid output state (logic 0 or logic 1) from the peripheral to the IO is latched upon entry to STOP or STANDBY mode. This prevents external circuits from being disturbed by SYSCTL disabling a peripheral during low-power operation. Upon exit from STOP or STANDBY mode, the IO is again connected to the peripheral as the peripheral becomes re-enabled.

2.3 Clock Module (CKM)

The clock module contains the oscillators, the clock monitors, and the clock selection and control logic. A frequency clock counter is also provided for checking and/or calibrating the frequency of high-speed clocks against either the LFCLK_IN or a reference period/pulse provided on an IO pin.

2.3.1 Oscillators

Several oscillators are provided for generating low to high frequency clocks for use by the system. The CKM contains all the oscillators in the device and uses them to generate the system clocks. See the device specific data sheet for whether external oscillators are available.

Internal Oscillators

- LFOSC: low frequency oscillator (32kHz typical frequency)
- SYSOSC: system oscillator (24MHz)

2.3.1.1 Internal Low-Frequency Oscillator (LFOSC)

The low-frequency oscillator (LFOSC) is an on-chip low power oscillator which is factory trimmed to a frequency of 32.768kHz.

LFOSC can provide higher accuracy when used over a reduced temperature range. See the device-specific data sheet for details.

The LFOSC is active by default after a BOOTRST, sourcing the LFCLK. The LFOSC startup monitor sets the LFOSCGOOD bit in the CLKSTATUS register when LFOSC is ready.

2.3.1.2 Internal System Oscillator (SYSOSC)

The system oscillator (SYSOSC) is an on-chip, accurate 24MHz frequency .

Key features of the SYSOSC include:

- High accuracy when using optional frequency correction loop (FCL) and reference resistor
  - The frequency correction loop may support correction via an external resistor (ROSC) or an internal resistor, depending on the device capabilities. Refer to the device-specific data sheet to determine if a device supports the FCL with an internal or external resistor, or both
- Fast start-up time from a disabled state
- Capable of switching from base frequency to low frequency, or low frequency to base frequency
  - Phase-aligned transition to minimize disturbance to peripherals
  - Fast settling to specified accuracy
  -
- A secondary output with a constant 4MHz frequency for use by MFCLK
  - When f_{SYSOSC} = 24MHz, the 4MHz output is derived from SYSOSC divided digitally by 6. SYSCTL manages the digital divider on this output to ensure a constant 4MHz output regardless of the selected SYSOSC frequency.

The SYSOSC is active at base frequency (24MHz) by default after a brownout reset, sourcing MCLK.

2.3.1.2.1 SYSOSC Frequency Correction Loop

The SYSOSC frequency accuracy can be improved through the use of the SYSOSC frequency correction loop (FCL) feature. The FCL circuit uses either an internal resistor or external resistor (populated between the ROSC pin and VSS), to stabilize the SYSOSC frequency by providing a precise reference current for the SYSOSC. The overall frequency accuracy which is achievable depends on the operating temperature range together with the tolerance and temperature drift of the selected reference resistor.

Note

Not all devices support operation with an internal resistor and an external resistor for FCL. Some devices support external mode, some devices support internal mode, and some devices support both modes. See the device-specific data sheet to determine the features of a given device.

Note

The power consumption of SYSOSC is marginally higher with the FCL enabled due to the reference current that flows through ROSC. Settling time from startup to target accuracy can also be longer. See the device-specific data sheet for startup times.

2.3.1.2.1.1 SYSOSC FCL in Internal Resistor Mode

This section describes the procedure for selecting the internal resistor mode through the device SYSCTL registers.

For devices which support the internal resistor FCL mode, the device-specific data sheet includes specifications for overall SYSOSC frequency accuracy across various temperature ranges. If the overall accuracy values in the device-specific data sheet meet the application and cost requirements, then there is no need to use the external resistor mode and the ROSC pin can be used for standard functions.

Enabling FCL with Internal Resistor

To increase the SYSOC accuracy with FCL, follow this procedure:

1. Enable FCL mode by setting the SETUSEFCL bit in the SYSOSCFCLCTL register.
   a. If the device supports both internal resistor and external resistor FCL modes, do not set the SETUSEEXRES bit in the SYSOSCFCLCTL register when setting the SETUSEFCL bit.
2. When the FCL mode is enabled, software cannot disable the mode. A BOOTRST is required before a change to the FCL mode.

2.3.1.2.2 Disabling SYSOSC

SYSOSC can be disabled in STOP mode by setting the DISABLESTOP bit in the SYSOSCCFG register. Doing so forces the MCLK to use LFCLK in STOP mode (this is the STOP2 policy). This provides the lowest possible power consumption in STOP mode, as the system runs at 32kHz and SYSOSC consumes no current. When exiting STOP mode to RUN mode, SYSCTL will automatically re-enable SYSOSC and switch MCLK back to SYSOSC.

SYSOSC can be disabled manually by setting the DISABLE bit in the SYSOSCCFG register. When SYSOSCCFG.DISABLE is set, the system will run from LFCLK in all power modes.

Note

The DISABLE and DISABLESTOP bits in SYSOSCCFG are mutually exclusive, and must not be set at the same time.

SYSOSC is always disabled automatically in STANDBY and SHUTDOWN modes.

2.3.1.3 LFCLK_IN (Digital Clock)

It is possible to bypass the LFXT circuit and bring in a 32.768kHz typical frequency digital clock into the device to use as the LFCLK source instead of LFOSC or LFXT. To configure LFCLK to use a digital clock input instead of LFXT or LFOSC, first configure the IOMUX to enable the LFCLK_IN function on the appropriate pin. When IOMUX is configured correctly and the external clock source is outputting a 32kHz clock to LFCLK_IN, set the SETUSEEXLF bit in the EXLFCTL register in SYSCTL.

LFCLK_IN is compatible with digital square wave CMOS clock inputs and should have a typical duty cycle of 50%.

It is possible to check for a valid clock signal on LFCLK_IN by enabling the LFCLK monitor before setting SETUSEEXLF in the EXLFCTL register. By default, the LFCLK monitor will check LFCLK_IN if the LFXT was not started.

After LFCLK_IN is selected as the LFCLK source, it is not possible to change back to LFOSC or LFXT without going through a BOOTRST.

Note

If MCLK is to be sourced from LFCLK with LFCLK sourced from LFCLK_IN, first configure LFCLK to use LFCLK_IN and then configure MCLK to use LFCLK. Do not switch MCLK to LFCLK with LFCLK running from LFOSC, and then later switch LFCLK to LFCLK_IN.

Note

LFCLK_IN and LFXT are mutually exclusive and must not be enabled at the same time. Do not set the SETUSEEXLF bit in the EXLFCTL register if the SETUSELFXT bit or the STARTLFXT bit is set in the LFXTCTL register.

2.3.1.4 HFCLK_IN (Digital clock)

To configure HFCLK to use a digital clock input, first configure the IOMUX to enable the HFCLK_IN function on the appropriate pin. When IOMUX is configured correctly and the clock source is outputting a clock to HFCLK_IN, set the USEEXTHFCLK bit in the HSCLKEN register in SYSCTL.

Note

SYSOSC must be enabled at base frequency when the HFCLK_IN is enabled.

To source MCLK from HFCLK_IN after selecting HFCLK_IN as the HFCLK source, first set the HSCLKSEL bit in the HSCLKCFG register to select HFCLK as the high-speed clock source. Then, set the USEHSCLK bit in the MCLKCFG register to select the high-speed clock source as the MCLK source. Once USEHSCLK is set, HSCLKCFG must not change and the HFCLK_IN must not be disabled until the MCLK source is switched back to SYSOSC by clearing USEHSCLK and verifying that the HSCLKMUX bit in CLKSTATUS was cleared by hardware.

HFCLK_IN is compatible with digital square wave CMOS clock inputs and should have a typical duty cycle of 50%.

2.3.2 Clocks

The CKM takes oscillator outputs and generates a variety of functional clocks for use by the device.

Clocks

- System Clocks
  - MCLK: Main system clock for PD1 peripherals and PD1 bus
  - CPUCLK: CPU clock, derived from MCLK
  - ULPCLK: Main system clock for PD0 peripherals and PD0 bus, derived from MCLK
  - MFCLK: Fixed 4MHz clock, synchronized to MCLK/ULPCLK
  - LFCLK: Fixed 32kHz clock, synchronized to MCLK/ULPCLK

- Peripheral Specific Clocks
  - ADCCLK: ADC sampling period clock
- External Clocks
  - CLK_OUT: External clock output with divider for pushing out a clock to external circuits

All clocks are disabled in SHUTDOWN mode.

In addition to the configurable clocks listed above, several direct clock connections are made to analog peripherals (see the Section 2.3.2.8 section).

2.3.2.1 MCLK (Main Clock) Tree

The MCLK is the main system clock and the root point of synchronization for all synchronized clocks (MCLK, CPUCLK, ULPCLK, MFCLK, and LFCLK). It is typically the highest speed clock in the system and supports operation up to 24MHz across the full temperature range of the device. The MCLK tree is the root source for the CPUCLK (in RUN mode), the PD1 high speed peripheral bus clock (in RUN and SLEEP modes), and the ULPCLK low power bus clock (in RUN, SLEEP, STOP, and STANDBY modes). In addition, the 4MHz MFCLK and 32kHz LFCLK outputs are synchronized to MCLK.

The MCLK output to PD1 peripherals is enabled in RUN and SLEEP modes, and disabled in all other power modes. While the MCLK output to PD1 is disabled in STOP and STANDBY modes, the MCLK tree is still running to source ULPCLK and to provide synchronization for MFCLK and LFCLK.

The MCLK source is selected with a glitch free clock mux and can be changed dynamically at runtime by user software. It can also be changed automatically by hardware when entering STOP and STANDBY modes or during an asynchronous fast clock request.

The available sources for MCLK include:

- **SYSOSC** at 24MHz
- **LFCLK** at 32kHz for applications where the entire system, including the CPU, runs at 32kHz with low peak operating current

Using MCLK in RUN and SLEEP Mode

After boot, MCLK is sourced from SYSOSC by default. The decision of which oscillator to use to source MCLK is important because MCLK sets both the CPUCLK frequency and the bus clock frequency for PD1 peripherals. As a result, the accuracy and the clock speed of the oscillator selected for MCLK must be appropriate not only for the operation of the CPU but also for the operation of the PD1 peripherals that use the bus clock as their functional clock.

The clock source and frequency selection decisions made for MCLK also affect ULPCLK in RUN and SLEEP modes. See the ULPCLK section for more information on how MCLK and ULPCLK are related in RUN and SLEEP mode.

Using MCLK in STOP and STANDBY Mode

In STOP and STANDBY modes, the MCLK output to PD1 peripherals is disabled, but the ULPCLK, which is the bus clock for PD0 peripherals, is still active in STOP and optionally active in STANDBY. See the ULPCLK section for more information on how the MCLK source and ULPCLK are related in STOP and STANDBY mode.

MCLK Source Selection

Application software can change the MCLK source from SYSOSC to LFCLK in all modes by setting MCLKCFG.USELFCLK, giving the low peak current consumption with the CPU and PD1 peripherals operational. The following table gives the proper register bit configurations for selecting different clocks for MCLK in RUN and SLEEP modes.

Table 2-1. MSPM0Cxx MCLK Source Selection in RUN and SLEEP Mode

Desired Source	MCLKCFG.USELFCLK
SYSOSC	0

Table 2-1. MSPM0Cxx MCLK Source Selection in RUN and SLEEP Mode (continued)

Desired Source	MCLKCFG.USELFCLK
LFCLK	1

To switch MCLK from SYSOSC to LFCLK in RUN mode:

1. Verify that MCLK is sourced from SYSOSC (CLKSTATUS.CURMCLKSEL is cleared)
2. If SYSOSC is not running at base frequency, and SYSOSC is to be left enabled when switching MCLK to LFCLK, set SYSOSC to base frequency before proceeding
3. Set MCLKCFG.USELFCLK to switch MCLK to LFCLK and leave SYSOSC enabled, or set SYSOSCCFG.DISABLE to switch MCLK to LFCLK and disable SYSOSC

To switch MCLK from LFCLK to SYSOSC in RUN mode:

1. Verify that MCLK is sourced from LFCLK (CLKSTATUS.CURMCLKSEL is set)
2. Clear MCLKCFG.USELFCLK or SYSOSCCFG.DISABLE, whichever was set to switch MCLK to LFCLK

2.3.2.2 CPUCLK (Processor Clock)

The processor clock (CPU clock) is always derived directly from MCLK and is active in RUN mode at the MCLK frequency. In all other power modes, CPUCLK is disabled.

2.3.2.3 ULPCLK (Low-Power Clock)

The ULPCLK is the bus clock for peripherals in the PD0 power domain. It supports operation up to 24MHz and is derived directly from the MCLK tree . The ULPCLK frequency is dependent on the MCLK configuration and the selected power mode.

ULPCLK Behavior in RUN and SLEEP Modes

The PD0 power domain has a frequency limit of 24MHz in RUN and SLEEP modes.

ULPCLK Behavior in STOP and STANDBY Modes

- In STOP mode, the MCLK tree (and by extension, the ULPCLK) can run from SYSOSC with a 4MHz rate (if SYSOSCCFG.DISABLESTOP=0x0) or from LFCLK at 32kHz (if SYSOSCCFG.DISABLESTOP=0x1). When SYSOSC is used (SYSOSCCFG.DISABLESTOP=0x0), SYSCTL ensures that ULPCLK is always 4MHz even if SYSOSC is running at a higher frequency (due to user configuration or due to an asynchronous request from a peripheral).
- In STANDBY mode, the MCLK tree (and by extension, the ULPCLK) either run from LFCLK (STANDBY0) or are disabled (STANDBY1) to conserve power. In STANDBY1, only a few timer peripherals receive ULPCLK.

Table 2-2. MSPM0Cxx ULPCLK by Operating Mode

Selected Power Mode	Configuration	Register Settings	ULPCLK Frequency
RUN or SLEEP (24MHz maximum)	MCLK source is SYSOSC (RUN0, SLEEP0)	MCLKCFG.USELFCLK=0x0	ULPCLK is sourced from MCLK according to the MCLK configuration with $f_{ULPCLK} = f_{MCLK}$
	MCLK source is LFCLK (RUN1/2, SLEEP1/2)	MCLKCFG.USELFCLK=0x1 or SYSOSCCFG.DISABLE=0x1	ULPCLK is sourced from LFCLK with $f_{ULPCLK} = f_{LFCLK} = 32kHz$
STOP (4MHz maximum)	STOP with SYSOSC enabled (STOP0)	SYSOSCCFG.DISABLESTOP = 0x0	ULPCLK is sourced from SYSOSC with $f_{ULPCLK} = 4MHz$
	STOP with SYSOSC disabled (STOP2)	SYSOSCCFG.DISABLESTOP = 0x1	ULPCLK is sourced from LFCLK with $f_{ULPCLK} = f_{LFCLK} = 32kHz$
STANDBY (32kHz maximum)	STANDBY with ULPCLK and LFCLK enabled (STANDBY0)	MCLKCFG.STOPCLKSTBY= 0x0	ULPCLK is sourced from LFCLK with $f_{ULPCLK} = f_{LFCLK} = 32kHz$
	STANDBY with ULPCLK and LFCLK disabled (STANDBY1)	MCLKCFG.STOPCLKSTBY= 0x1	ULPCLK is disabled to all peripherals except one timer, which receive $f_{ULPCLK}=f_{LFCLK} = 32kHz$

Table 2-2. MSPM0Cxx ULPCLK by Operating Mode (continued)

Selected Power Mode	Configuration	Register Settings	ULPCLK Frequency
SHUTDOWN (Off)	-	-	ULPCLK is off

2.3.2.4 MFCLK (Middle Frequency Clock)

The MFCLK provides a continuous 4MHz clock to a variety of peripherals on the device. The MFCLK 4MHz rate is always derived from the SYSOSC. As the SYSOSC frequency is not fixed (it can be configured for 32, 24, 16, or 4MHz), SYSCTL automatically applies a divider to SYSOSC to keep MFCLK at a constant 4MHz rate regardless of the current SYSOSC frequency. MFCLK can be used by peripherals such as timers and serial interfaces that require a constant clock source in RUN, SLEEP, and STOP power modes.

After a SYSRST, MFCLK is initially disabled. MFCLK can be enabled in software by setting USEMFTICK in the MCLKCFG register in SYSCTL. MFCLK is active in RUN, SLEEP, and STOP power modes only, and SYSOSC must be enabled for MFCLK to operate.

All MFCLK edges are synchronized to the main system clocks (MCLK and ULPCLK), meaning that the registers of peripherals clocked by MFCLK can be read or written to at any time without any special handling.

Peripherals can select MFCLK as their functional clock source through their respective CLKSEL mux. Not all peripherals support running from MFCLK.

Using MFCLK in STOP Mode

When using MFCLK in STOP mode, SYSOSC can be configured to automatically switch to 4MHz (low frequency) when entering STOP mode and automatically switch back to the previously selected frequency when exiting STOP mode to RUN mode (gear shift mode). As MFCLK is a 4MHz clock source, running SYSOSC at 4MHz in STOP mode reduces power consumption when in STOP mode. .

Requirements for Using MFCLK

1. When using MFCLK, the MDIV (MCLK divider) must be disabled (set to /1). Disable MDIV by setting MDIV in the MCLKCFG register to 0x0. SYSCTL hardware does not allow MFCLK to run when MCLKCFG.MDIV != 0.
2. When using MFCLK, do not enable MFCLK when MCLK is sourced from the low-frequency clock (LFCLK). Application software must enable MFCLK by setting the USEMFTICK bit before switching the MCLK source from SYSOSC to LFCLK. Software can switch MCLK to LFCLK after USEMFTICK is set. In this case, MFCLK halts when MCLK is sourced from LFCLK, and it resumes when MCLK is switched back to SYSOSC.
3. When MFCLK is enabled by setting USEMFTICK in the MCLKCFG register, it is considered by the hardware as a static policy. Do not clear USEMFTICK.

When MFCLK is configured to be enabled, it is only active when SYSOSC is active and MCLK is not sourced from LFCLK. When MCLK is sourced from LFCLK, MFCLK is stopped by hardware automatically. Note that if the device is in STANDBY, MCLK is always sourced from LFCLK and MFCLK is always disabled by hardware.

Asynchronous fast clock requests, if configured, temporarily enable SYSOSC to handle specific peripheral events and activity. If MFCLK is configured to be enabled (USEMFTICK is set), then MFTICK runs when a peripheral asserts an asynchronous fast clock request.

2.3.2.5 LFCLK (Low-Frequency Clock)

LFCLK provides a continuous 32kHz clock to a variety of peripherals on the device. After a BOOTRST, LFCLK is sourced by the internal 32kHz oscillator (LFOSC).

LFCLK is active in RUN, SLEEP, STOP, and STANDBY power modes. It is possible to disable both ULPCLK and LFCLK together to most peripherals in STANDBY mode to achieve the lowest possible STANDBY mode power consumption (STANDBY1). To do so, set the STOPCLKSTBY bit in the MCLKCFG register in SYSCTL before entering STANDBY. In this state, a few times are the only clocked peripherals.

LFCLK is a synchronized clock. All LFCLK edges are synchronized to the main system clocks (MCLK and ULPCLK), meaning that the registers of peripherals clocked by LFCLK can be read or written to at any time without any special handling.

Note

When MCLK/ULPCLK are not sourced by LFCLK (for example, when they are sourced by SYSOSC) there is a 5 ULPCLK cycle synchronization delay between the low frequency clock source's clock edge and the corresponding LFCLK edge as seen by peripherals running from LFCLK. When the MCLK/ULPCLK frequency is constant, this delay is constant and it does not add jitter to LFCLK. If the MCLK/ULPCLK frequency changes, the synchronization delay changes proportionally and this results in a small single-cycle LFCLK jitter at the MCLK/ULPCLK frequency transition point. This jitter changes the duty cycle of one LFCLK period, but there is no accumulation of error (there is never a change in the number of LFCLK periods, ensuring an accurate LFCLK time base for peripherals).

Peripherals can select LFCLK as their functional clock source through their respective CLKSEL mux. Not all peripherals support running from LFCLK. It is possible to run the main clock (MCLK) from LFCK, in which case the entire device runs at the LFCLK rate (32kHz).

2.3.2.6 ADCCLK (ADC Sample Period Clock)

ADCCLK is used by the ADC module to set the ADC sampling period. The ADCCLK for a given ADC is provided from the CKM to the ADC, but the ADCCLK clock selection is done within each ADC peripheral's configuration registers. See the ADC chapter for information on configuring the ADCCLK. ADCCLK can be selected as SYSOSC or HFCLK_IN.

2.3.2.7 External Clock Output (CLK_OUT)

A clock output unit is provided for sending digital clock signals from the device to external circuits or to the frequency clock counter. This feature is useful for clocking external circuitry such as an external ADC that does not have a clock source. The clock output unit has a flexible set of sources to select and includes a programmable divider.

Available clock sources for CLK_OUT:

- SYSOSC
- ULPCLK
- LFCLK

The selected clock source can be divided by 1 (no divide), 2, 4, 6, 8, 10, 12, 14, or 16 before being output to the pin or to the frequency clock counter.

To use the clock output unit:

1. Configure IOMUX to select the CLK_OUT function on the device pin with CLK_OUT.
2. Select the desired clock source in the EXCLKSRC field of the GENCLKCFG register.
3. Set the desired clock divider, if necessary, in the EXCLKDIVVAL field of the GENCLKCFG register, and enable the divider by setting the EXCLKDIVEN bit.
4. Enable the external clock output by setting the EXCLKEN bit in the GENCLKEN register.

Note

When the CLK_OUT source is selected as ULPCLK or MFPCLK, the clock divider must be enabled (EXCLKDIVEN must be set).

Note

When clearing the EXCLKEN bit to disable CLK_OUT, allow the clock source to run for 10 clock cycles to stabilize the EXCLKSRC mux.

Note

When disabling a clock source which is selected for CLK_OUT, it is recommended to disable the CLK_OUT function before disabling the clock source if it is important that CLK_OUT be logic low (0) when the clock source is disabled. If CLK_OUT is left enabled and the source for CLK_OUT is disabled, it is possible that CLK_OUT may stop in a logic high (1) state.

2.3.2.8 Direct Clock Connections for Infrastructure

Several direct clock connections are made in the device to support specific analog functionality:

- SYSOSC to ADCs

Direct Connections to ADCs

In addition to receiving ADCCLK to set the sampling window, the ADC modules also receive the direct output of SYSOSC. The SYSOSC direct output to the ADCs is used by the charge pump logic in the ADC modules. SYSOSC can be configured at any frequency to support this function. The ADC supports requesting SYSOSC automatically before a conversion, so there is not a requirement for application software to ensure that SYSOSC is running before triggering an ADC conversion.

2.3.3 Clock Tree

The following figure shows the top level clocking tree for MSPM0Lxx family devices. This diagram shows the mapping between oscillators (sources) and clocks (destinations), as well as the SYSCTL register bit fields for the selection muxes. Note that not all devices have all clock system features shown in this figure.

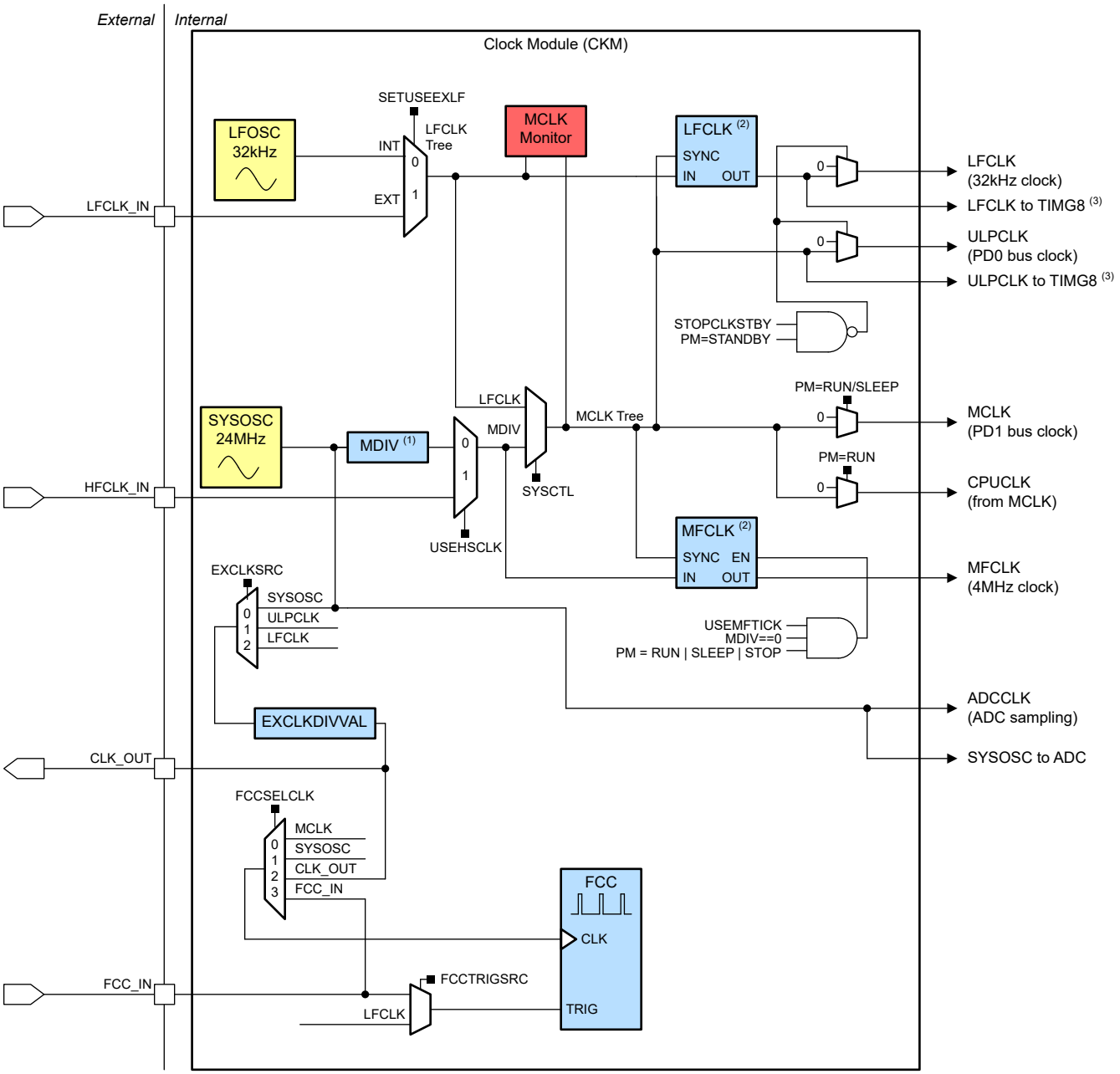


Figure 2-5. MSPM0Cxx Top Level Clock Tree

1. LFCLK and MFCLK are fixed-frequency 32kHz and 4MHz clocks, respectively, that can be selected by certain peripherals for ensuring a constant clock rate even when MCLK or ULPCLK changes source or rate. LFCK and MFCLK are always synchronized to each other and to MCLK and ULPCLK.
2. TIMG8 (general purpose timers) receive an ungated LFCLK and ULPCLK, enabling them to continue operating even in STANDBY1 when STOPCLKSTBY is asserted to gate the LFCLK and ULPCLK to all other peripherals to save additional power in STANDY mode.

2.3.3.1 Peripheral Clock Source Selection

Most peripherals on the device contain an input clock selection mux which is used to select, and optionally divide down, the functional clock for the peripheral. Figure 2-6 shows the superset peripheral clock selection mux and optional clock divider. Note that not every peripheral has every clock source shown in Figure 2-6. For example, accelerators such as CRC, and DMA run off of the bus clock. There is no option to select MFCLK or LFCLK

for these peripherals. To determine the available clock sources for a peripheral, see the chapter for the specific peripheral and review the clock input selections.

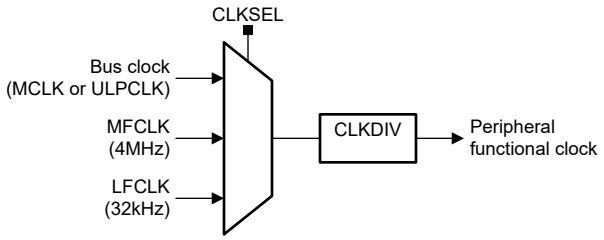


Figure 2-6. Peripheral Clock Selection Mux and Divider

Exceptions

There are also several peripherals that have a unique clock selection scheme and do not use the standard peripheral clock mux shown above. Typically this is due to a requirement for a peripheral to have a clock source that is asynchronous to the rest of the system. Cases where this occurs include:

- An analog-to-digital converter (ADC), where ADCCLK has a special selection mux to take asynchronous clock sources directly (the ADC sampling clock is selected with a special selection in the ADC control registers)

2.3.4 Clock Monitors

Several hardware clock monitors are provided to ensure that the CKM is functioning properly. Clock faults are processed through SYSCTL and result in either a brownout reset (in the event of a fatal fault) or a SYSCTL interrupt.

2.3.4.1 MCLK Monitor

A digital clock monitor can be used with MCLK. The MCLK monitor asserts an MCLK fault if there is no MCLK activity for a period of 1-12 LFCLK cycles. An MCLK fault is always considered fatal to the system and generates a BOOTRST.

The MCLK monitor can be enabled after the LFCLK is configured and running. To enable the MCLK monitor, set the MCLKDEADCHK bit in the MCLKCFG register in SYSCTL. When enabled, the MCLK monitor runs in all operating modes except for STANDBY1 and SHUTDOWN.

2.3.4.2 Startup Monitors

Clock startup monitors are provided for application software to check that the LFOSC is alive before it isselected by software to be used to source a clock in the system. When a clock source has started successfully and is ready, a GOOD indication is given in the CLKSTATUS register in SYSCTL and an interrupt is generated. The startup monitors only provide a status indication when a related clock system configuration change is made. When an initial GOOD indication is given, the clock is not continuously monitored by the startup monitor. Continuous monitoring is provided for MCLK.

2.3.4.2.1 LFOSC Startup Monitor

The LFOSC is started automatically after a BOOTRST. The LFOSC takes some time to start. A startup monitor is provided to indicate to the application software when LFOSC startup has completed, at which time the LFCLK is available for use by peripherals. When LFSOSC startup has completed, the LFOSC startup monitor asserts the LFOSCGOOD bit in the CLKSTATUS register in SYSCTL and the LFOSCGOOD interrupt is asserted to alert the application. See the device-specific data sheet for the LFOSC startup time.

2.3.5 Frequency Clock Counter (FCC)

The frequency clock counter (FCC) enables flexible in-system testing and calibration of a variety of oscillators and clocks on the device. The FCC counts the number of clock periods seen on the selected source clock

within a known fixed trigger period (derived from a secondary reference source) to provide an estimation of the frequency of the source clock.

Application software can use the FCC to measure the frequency of the following source oscillators and clocks (selected through the FCCSELCLK field in the GENCLKCFG register):

- MCLK
- SYSOSC
- CLK_OUT
- The external FCC input (FCC_IN)

The reference clock used to set the trigger time over which pulses of the source clock are counted is configurable (through the FCCTRIGSRC field in the GENCLKCFG register), and can be driven by:

- The external FCC input (FCC_IN)
- LFCLK

The trigger time period can be set in one of two ways (through the FCCLVLTRIG field in the GENCLKCFG register):

- Level triggered (one rising edge to one falling edge of the reference clock input). Please note that LFCLK_IN cannot be used as a trigger clock source if using level triggering.
- Rising-edge to rising-edge triggered, for a defined number of clock periods of the reference clock (selectable from 1 to 32 through the FCCTRIGCNT field in the GENCLKCFG register)

When the trigger source is selected as the external FCC input in level-triggered mode, a user-specified counting period can be set by applying a logic high pulse on the FCC_IN pin of the desired trigger length.

The FCC counter is 22 bits and supports counting from 0 up to $2^{22} - 1$ or 4 194 303.

While the external FCC input (FCC_IN function) can be used as either the FCC clock source or the FCC trigger input, it cannot be used for both functions during the same FCC capture. It must be configured as either the FCC clock source or the FCC trigger.

2.3.5.1 Using the FCC

Rising-Edge to Rising-Edge Triggered Mode with FCC_IN Trigger

The following steps describe how to use the FCC to count the number of source clock pulses within the trigger period set by the reference clock, with the FCC_IN pin being selected as the reference clock and the SYSOSC being selected as the source clock. This example would be useful for calibrating the SYSOSC frequency with respect to an accurate clock source provided to the FCC_IN pin externally.

1. Set the source clock to SYSOSC by configuring the FCCSELCLK field in the GENCLKCFG register.
2. Set the reference clock to FCC_IN by clearing the FCCTRIGSRC bit in the GENCLKCFG register.
3. Select rising-edge to rising-edge triggering by clearing the FCCLVLTRIG bit in the GENCLKCFG register.
4. Select the desired number of reference clock periods to count the source clock over in the FCCTRIGCNT field in the GENCLKCFG register.
5. Ensure that SYSOSC is enabled at the desired frequency, and that the external clock source connected to FCC_IN is running correctly before continuing.
6. Write the GO bit and KEY field to the FCCCMD register to start the FCC capture on the next trigger clock period.
7. Poll the FCCDONE status bit in the CLKSTATUS register. When the capture completes, FCCDONE will be set by hardware. FCCDONE is read-only and is automatically cleared by hardware when a new capture is started.
8. Extract the resulting count from the 22-bit DATA field in the FCC register.

Level Triggered Mode with FCC_IN Trigger and HFCLK_IN Clock

The following steps describe how to use the FCC to count the number of source clock pulses within one external reference pulse window, with HFCLK_IN being selected as the source clock. This example would be useful for measuring the frequency of an external clock source with respect to a fixed pulse width driven by an external signal.

1. Set the source clock to HFCLK by configuring the FCCSELCLK field in the GENCLKCFG register.
2. Set the trigger clock to the FCC_IN pin function by clearing the FCCTRIGSRC bit in the GENCLKCFG register.
3. Set level triggering by setting the FCCLVLTRIG bit in the GENCLKCFG register.
4. Ensure that IOMUX is configured for FCC_IN, that HFCLK is configured for HFCLK_IN, and that an external clock is sourcing HFCLK_IN.
5. Write the GO bit and KEY field to the FCCCMD register to start the FCC capture when FCC_IN goes logic high. Note that if FCC_IN is already logic high when GO is asserted, counting starts immediately. When using level mode, FCC_IN should be low when GO is set, and the trigger pulse should be sent to FCC_IN after GO is set.
6. Poll the FCCDONE status bit in the CLKSTATUS register. When the capture completes, FCCDONE will be set by hardware. FCCDONE is read-only and is automatically cleared by hardware when a new capture is started.
7. Extract the resulting count from the 22-bit DATA field in the FCC register.

2.3.5.2 FCC Frequency Computation and Accuracy

The frequency of the source clock can be computed after capture if the trigger time is known. The frequency is computed by dividing the number of source clock cycles captured by the trigger time. For example, if the trigger source was a 32.768kHz clock, the trigger mode was rising-edge to rising-edge, and the period count was 1, then the trigger time is one 32.768kHz clock period (30.5µs). If the captured source count were to come back as 122, the frequency of the source clock is computed as 122 divided by 30.5µs, giving a source clock frequency of approximately 3.99MHz.

$$f_{source} = FCC.DATA / ((GENCLKCFG.FCCTRIGCNT+1) / f_{ref}) \quad (6)$$

The FCC is accuracy is dependent on the trigger clock accuracy as well as the total number of clock cycles captured. The FCC intrinsic error is ≤2 source clock cycles per capture due to synchronization of the trigger to the source clock. Therefore, the impact of these two clock cycles is reduced as more cycles are counted (as the trigger time is increased and/or the source clock frequency is increased). Approximate intrinsic error of the FCC for various source clock frequencies captured against one 32.678kHz period (FCCTRIGCNT=0) and 32 clock periods (FCCTRIGCNT=31) are given in Table 2-3.

Table 2-3. FCC Error

Use Case (Source Clock Frequency)	FCC Trigger Time	FCC Count Result	FCC Count Uncertainty	Approximate FCC Intrinsic Uncertainty Error
4MHz source clock	30.5µs	122	2 cycles	1.6%
	976.6µs	3906		0.05%
16MHz source clock	30.5µs	488		0.4%
	976.6µs	15625		0.01%
24MHz source clock	30.5µs	732		0.27%
	976.6µs	23437		0.01%

Note

When using the FCC_IN signal, it is recommended to have a fast slew rate of 10ns or less on the FCC_IN pin to minimize measurement uncertainty.

2.4 System Controller (SYSCTL)

The system controller (SYSCTL) contains all control logic for managing the configuration and state of the PMU and CKM analog circuitry. SYSCTL also provides reset management, control over NRST and SWD pin muxing, flash bank swap control, and flash ECC error handling.

All power, clock, and reset configuration is done through the SYSCTL memory-mapped register interface.

2.4.1 Resets and Device Initialization

The SYSCTL manages device reset levels and device initialization.

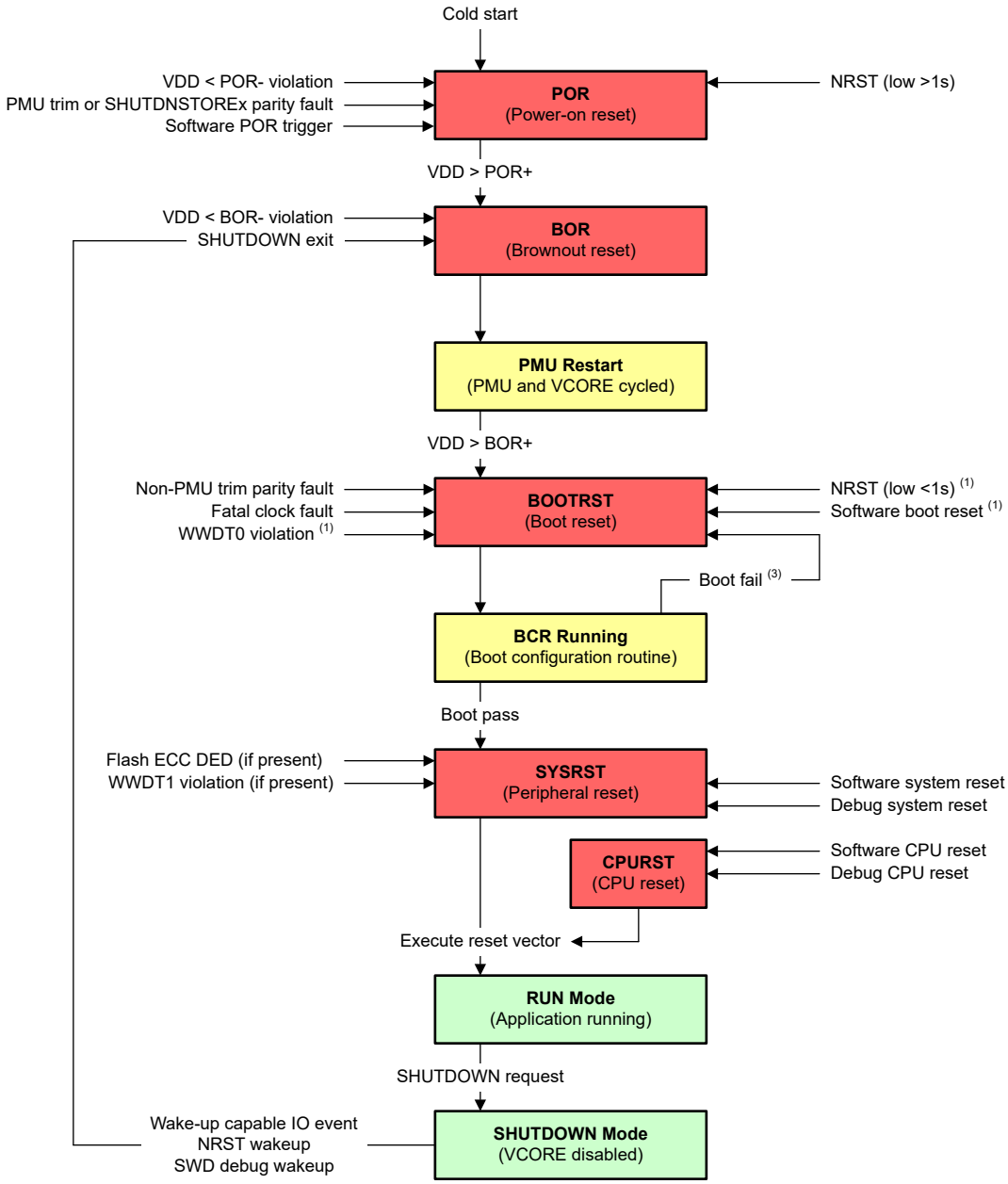
2.4.1.1 Reset Levels

The device has five reset levels:

1. Power-on reset (POR)
2. Brownout reset (BOR)
3. Boot reset (BOOTRST)
4. System reset (SYSRST)
5. CPU reset (CPURST)

The relationships between reset levels are given in Figure 2-7.

Figure 2-7. MSPM0 Reset Levels



(1) An NRST (low <1 s), software boot reset, or WWDT0 violation triggered BOOTRST runs the boot configuration routine but does not reset the LFCLK, LFCLK_IN, and IOMUX configuration of any IO pins used by LFXT or LFCLK_IN..

(3) If a boot fail occurs during execution of the boot configuration routine, a BOOTRST can be generated by SYSCTL to attempt the boot process again from the BOOTRST level. See Section 2.4.1.8.

Note

SLEEP, STOP, and STANDBY operating modes are not shown in this diagram. These modes originate from and return to RUN mode unless an exception occurs which causes a reset level to be asserted or a mode to be suspended.

2.4.1.1.1 Power-on Reset (POR) Reset Level

A power-on reset (POR) is a complete device reset.

The following conditions generate a POR:

- The device powers up (cold start)
- A POR- supply monitor violation(VDD drops below the POR supply monitor negative-going threshold)
- A parity fault on PMU trim data or the shutdown memory
- Software triggers a POR through SYSCTL *(RESETLEVEL 0x03)*
- The NRST pin is held low for more than one second when in NRST mode

A POR always resets the shutdown memory, reenable the NRST/SWD pin functions (if disabled), and trigger a BOR.

2.4.1.1.2 Brownout Reset (BOR) Reset Level

A brownout reset (BOR) resets the device power management unit (PMU). All regulated core logic powered from VCORE is power cycled.

The following conditions generate a BOR:

- A POR
- A BOR0- supply monitor violation (VDD drops below the BOR0- supply monitor negative-going threshold)
- An exit from shutdown mode (through a wakeup-capable IO, NRST, or SWD)

The following are not reset by a BOR:

- The shutdown memory (SHUTDNSTOREx)
- The NRST state, if disabled by software
- The SWD state, if disabled by software
- The latched IO pin state, if the cause of the BOR is an exit from SHUTDOWN mode (see SHUTDOWN mode handling)

A BOR always triggers a BOOTRST when VDD > BOR0+.

2.4.1.1.3 Boot Reset (BOOTRST) Reset Level

A boot reset (BOOTRST) triggers execution of the device boot configuration routine and resets the majority of the core logic, including the SYSOSC FCL mode (if enabled). The system memory (SRAM) is also power cycled and SRAM contents are lost.

The following conditions generate a BOOTRST:

- A BOR
- A parity fault on nonPMU trim data
- A fatal clock failure (see MCLK Monitor)
- A WWDT0 violation
- Software triggers a BOOTRST through SYSCTL *(RESETLEVEL 0x01)*
- The NRST pin is held low for longer than the minimum reset pulse time but less than one second when in NRST mode
- A BOOTRST followed by a boot failure (re-attempt of a failed boot sequence)

The following are not reset by a BOOTRST:

- The shutdown memory (SHUTDNSTOREx)
- The NRST disable state, if disabled by software
- The SWD disable state, if disabled by software

Following a BOOTRST, a SYSRST is always triggered if the boot configuration routine completes successfully. If the boot configuration routine fails to complete successfully, a BOOTRST is again generated and the boot process is attempted again from the BOOTRST point. The boot process attempts to complete successfully up to 3 times, after which the device state locks until a BOR or POR reset occurs (see Section 2.4.1.8).

2.4.1.1.4 System Reset (SYSRST) Reset Level

A system reset clears the state of the CPU and all the peripherals, with the exceptions listed below.

The following conditions generate a SYSRST:

- A BOOTRST followed by a boot pass
- A bootstrap loader (BSL) exit, which is always followed by execution of the boot configuration routine (BCR)
- A flash ECC uncorrectable (DED) error (if present)
- A WWDT1 violation (if present)
- A CPU lockup violation
- Software triggers a SYSRST through SYSCTL *(RESETLEVEL 0x00)*
- The debug subsystem triggers a system reset

The following are not reset by a SYSRST:

- The shutdown memory (SHUTDNSTOREx)
- The NRST state, if disabled by software
- The SWD state, if disabled by software
- The SYSOSC frequency correction loop (FCL), if enabled by software

The device is in RUN mode after a SYSRST, and the CPU executes the reset vector and begins execution of the application software.

2.4.1.1.5 CPU-only Reset (CPURST) Reset Level

A CPU-only reset clears the state of the CPU logic only. Peripheral states are not affected by a CPU reset. A CPU reset is only generated by software through the CPU AIRCR local register or by the debug subsystem.

2.4.1.2 Initial Conditions After POR

After a POR, when the boot process completes and the CPU starts the application, the initial device conditions are as follows:

- The NRST pin is configured in NRST mode
- Serial wire debug (SWD) IO are in SWD mode
- All other configurable I/O pins are high impedance (Hi-Z)
- Peripheral modules are reset as described in their respective chapters of this manual
- The device is in RUN mode
- MCLK is sourced from the internal SYSOSC at base frequency (24MHz)
- LFCLK is sourced from the internal LFOSC (note that LFOSC requires time to start up before LFCLK can be used)
- MFCLK is disabled
- Peripherals are disabled
- Any flash sectors configured to be write protected at boot are write protected

2.4.1.3 NRST Pin

After a cold start, the NRST pin is configured in NRST mode. The NRST pin must be high for the device to boot successfully. There is no internal pullup resistor on NRST. External circuitry (either a pullup resistor to VDD or a reset control circuit) must actively pull NRST high for the device to start. After the device is started, a low pulse on NRST <1 second in duration triggers a BOOTRST. If a low pulse on NRST is held for >1 second, a POR is triggered.

Some low pin count devices support reconfiguring the NRST pin to be a GPIO pin. See the pin configuration of the device-specific data sheet to see if GPIO functionality is shared with NRST. Application software can disable the NRST functionality of the NRST pin, allowing GPIO functionality to be enabled. To disable NRST, set the DISABLE bit in the EXRSTPIN register along with the KEY. Then configure IOMUX for the desired functionality.

After the NRST pin function is disabled, it can only be re-enabled by a POR.

Note

When the NRST pin is shared with the I2C open-drain pin, it is important for the user's system to ensure that the device is powered up and in I2C mode before any transactions occur on the I2C bus. If the device is inadvertently reset due to a low signal on the shared reset or I2C SDA line before this point, it may cause the device reset.

To prevent this, pullups on the NRST pin that is shared with the I2C open-drain IO should be selected to meet the I2C pullup requirements for minimum and maximum values.

2.4.1.4 SWD Pins

There are two serial wire debug (SWD) pins present on all devices:

- SWCLK (serial wire clock)
- SWDIO (serial wire data input/output)

After a cold start, the SWD pins are configured in SWD mode to allow a debug connection to be established. It is possible to re-configure the SWD pins as general purpose IO (GPIO) in software to enable use of these pins in an application when debug support is no longer required. To disable SWD functionality, set the DISABLE bit in the SWDCFG register in SYSCTL along with the KEY. Then configure IOMUX for the desired functionality.

Once the SWD pin functions are disabled, they can only be re-enabled by triggering a POR.

2.4.1.5 Generating Resets in Software

Software can generate a software POR, a software BOOTRST, a software SYSRST with bootstrap loader (BSL) entry, or a software SYSRST by issuing the appropriate command to SYSCTL. To issue a reset, first select the desired reset level in the RESETLEVEL register in SYSCTL. Then set the GO bit in the RESETCMD register along with the KEY value.

Table 2-4. Software Generated SYSCTL Reset Commands

LEVEL	Action
0x0	Software SYSRST
0x1	Software BOOTRST
0x2	Software SYSRST with BSL entry
0x3	Software POR

A CPU-only reset (CPURST) which does not reset the peripherals can also be triggered in software within the Cortex-M0+ CPU by setting the SYSRESETREQ bit in the AIRCR local CPU register. See the CPU Sub System chapter for more information.

Starting the BSL From Software

The software-triggered BSL entry *(RESETLEVEL 0x02)* is a special case of the SYSRST which provides a mechanism for the application software to start the ROM bootstrap loader (BSL). It is not possible to jump to the bootloader code directly during normal software execution in RUN mode. When application software commands a software-triggered BSL entry *(RESETLEVEL 0x02)*, a SYSRST is generated first, followed by execution of the boot configuration routine (for authentication), after which the BSL is started (if the device security policy has the BSL configured to be enabled). Once the BSL has completed execution, a second SYSRST is issued and the BCR will execute. When the BCR completes, a final SYSTRST is asserted to return control of the system back to the application software. Any system configuration which is not reset by a SYSRST will be maintained through this entire process.

2.4.1.6 Reset Cause

After a device reset occurs, the lowest level reset cause which occurred during reset processing is captured in hardware so that application software can interrogate the reason for the reset and take any appropriate action

when starting the application. The lowest level reset cause is encoded into a 5-bit field in the reset cause register in SYSCTL. The contents of the reset cause register are always cleared upon a read, and return zero after being read if no reset has occurred after the read. The reset cause encodings are given in Table 2-5.

Table 2-5. Reset Cause Encoding

Reset				Device Modules Reset										
Reset Level	Cause ID		Reset Cause	NRST/SWD Disables	SHUTDN STOREx	Core Regulator	Debug Subsystem	LFCLK State	SRAM	BCR Execution	IOMUX	EVENT, DMA, FLASHCTL	Peripherals	CPU
	0x00	0	No reset since last read											
POR	0x01	1	VDD < POR- violation	R	R	R	R	R	R	R	R	R	R	R
			PMU trim parity fault											
			SHUTDNSTOREx parity fault											
	0x02	2	NRST pin reset (>1s)	R	R	R	R	R	R	R	R	R	R	R
	0x03	3	Software-triggered POR	R	R	R	R	R	R	R	R	R	R	R
BOR	0x04	4	VDD < BOR- violation			R	R	R	R	R	R	R	R	R
	0x05	5	Wake from SHUTDOWN			R	R	R	R	R	R(1)	R	R	R
	0x06	6	Reserved											
	0x07	7	Reserved											
BOOTRST	0x08	8	Non-PMU trim parity fault					R	R	R	R	R	R	R
	0x09	9	Fatal clock fault					R	R	R	R	R	R	R
	0x0A	10	Reserved											
	0x0B	11	Reserved											
	0x0C	12	NRST pin reset (<1 s)						R	R	R	R	R	R
	0x0D	13	Software-triggered BOOTRST						R	R	R	R	R	R
	0x0E	14	WWDT0 violation						R	R	R	R	R	R
	0x0F	15	Reserved											
SYSRST	0x10	16	BSL exit							R	R	R	R	R
	0x11	17	BSL entry							R	R	R	R	R
	0x12	18	Reserved											
	0x13	19												
	0x14	20	Uncorrectable flash ECC error								R	R	R	R
	0x15	21	CPULOCK violation								R	R	R	R
	0x16	22	Reserved											
	0x17	23	Reserved											
	0x18	24	Reserved											
	0x19	25	Reserved											
	0x1A	26	Debug-triggered SYSRST								R	R	R	R
	0x1B	27	Software-triggered SYSRST								R	R	R	R

Table 2-5. Reset Cause Encoding (continued)

Reset				Device Modules Reset										
Reset Level	Cause ID		Reset Cause	NRST/SWD Disables	SHUTDN STOREx	Core Regulator	Debug Subsystem	LFCLK State	SRAM	BCR Execution	IOMUX	EVENT, DMA, FLASHCTL	Peripherals	CPU
CPURST	0x1C	28	Debug-triggered CPURST											R
	0x1D	29	Software-triggered CPURST											R
	0x1E	30	Reserved											
	0x1F	31	Reserved											

(1) In the case of a SHUTDOWN mode exit, the IOMUX registers are always reset but the IOs themselves retain their last state from the point of entry into SHUTDOWN until the user clears the RELEASE bit in the SHDNIOREL register in SYSCTL. This enables application software to be able to reconfigure IOMUX and any corresponding peripherals before releasing the IO after a SHUTDOWN exit. See shutdown mode handling and IOMUX wake.

If two reset causes occur simultaneously, the lowest cause reset ID value is prioritized and reported. For example, if a WWDT0 violation (cause 0x12) occurs at the same time that a VDD < BOR- violation (cause 0x04) occurs, the reported reset cause is a BOR- violation (cause 0x04), as this is a lower level reset which clears additional aspects of the device state.

The reset cause encoding enables simple software handling during application startup. The reset cause value can be read by application software and tested to be within a certain value range to determine if the following occurred:

- **RESETCAUSE==0x00:** No reset since last read
- **RESETCAUSE<0x04:** The NRST/SWD disable state was reset and can need to be reconfigured
- **RESETCAUSE<0x04:** The SHUTDNSTOREx memory was reset and can need to be reconfigured
- **RESETCAUSE<0x08:** The regulated VCORE domain, including the SRAM, was power cycled
- **RESETCAUSE<0x1C:** The peripherals were reset and can need to be reconfigured

The following example shows how the reset cause can be tested to take specific actions when starting an application after a reset:

```
// Read reset cause into SRAM variable
uint8_t cause = RESETCAUSE;

// Handle device re-configuration based on reset cause level
if (cause!=0)
{
    if (cause<0x04)
    {
        // NRST/SWD disable state was lost
        // SHUTDNSTOREx memory state was lost
        // PMU/VCORE domain state was lost
        // RTC/LFXT/LFCLK state was lost
    }

    if (cause<0x1C)
    {
        // The peripherals were reset
    }
}
```

2.4.1.7 Peripheral Reset Control

Each peripheral on a device contains a reset control register (RSTCTL) and a status register (STAT).

The STAT register is a read-only register which contains a RESETSTKY bit, indicating if the peripheral was reset. This bit can be read by application software to determine if a peripheral was reset and needs to be re-configured. The RESETSTKY bit is cleared by writing the RESETSTKYCLR bit together with the KEY value to the RSTCTL register.

Application software can also force a reset of the peripheral by writing the RESETASSERT bit together with the KEY value to the RSTCTL register. This action resets the peripheral to the default state and sets the RESETSTKY bit in the STAT register.

Note

The RSTCTL register does not reset the FPUB and FSUB registers for a given peripheral. Use SYSRST or directly modify FPUB and FSUB directly of the peripheral to reset the publisher and subscriber event registers.

2.4.1.8 Boot Fail Handling

If a boot fails during execution of the boot configuration routine (BCR), SYSCTL asserts a BOOTRST to attempt another boot. A boot fail can be caused by the following:

- Boot configuration data integrity error
- Device trim integrity error
- BCR timeout (BCR takes significantly longer than expected to complete for any other reason)

Up to three attempts to successfully boot the device are made by hardware. If the first, second, or third boot attempt is successful, the application starts normally. If the third attempt fails, then the boot process fails, no further boot attempts are made, and the application software is not started.

The purpose of the additional boot attempts is to allow the device to boot correctly if a transient (temporary) error was the cause of the boot fail. If three boot attempts are not successful, a steady-state error condition is likely present and the application is not started to prevent unexpected operation.

Note

If a device is locked due to three failed attempts to boot, and a BOR- violation occurs, a BOR and BOOTRST are still generated (by definition) and a single boot attempt is made. Under the same conditions, if power is completely removed from the device (triggering a POR- violation), then the device again attempts to boot up to three times.

2.4.2 Operating Mode Selection

The device operating mode is configured through the use of the following:

1. Policy bits in the SYSOSCCFG and MCLKCFG registers in SYSCTL (to control the behavior of SYSOSC in RUN, SLEEP, and STOP modes)
2. Policy bits in the PMODECFG register in SYSCTL (to set the deep sleep level of STOP, STANDBY or SHUTDOWN)
3. SLEEPDEEP policy bit in the SCR local CPU register (to select whether a WFI instruction triggers SLEEP mode or STOP/STANDBY/SHUTDOWN mode)
4. Use of the Arm WFI (wait for interrupt) CPU instruction (to enter the configured SLEEP/STOP/STANDBY/SHUTDOWN state)

Before entering an operating mode where the CPU is disabled, make sure that the appropriate peripheral that can wake the CPU from sleep has been configured to generate a CPU interrupt on the desired event.

For a detailed description of the behavior of each operating mode, see the operating modes section.

Policy Bit Configuration

Table 2-6 defines how to configure the relevant policy bits for each operating mode. All values are indicated in binary format. A dash (-) indicates that the particular policy bit is a don't care for the specified operating mode.

Table 2-6. Operating Mode Policy Bit Configuration

Operating Mode Policy Control		RUN			SLEEP[2]			STOP			STANDBY		
Register	Bit	RUN0	RUN1	RUN2	SLEEP0	SLEEP1	SLEEP2	STOP0	STOP1	STOP2[3]	STANDBY0	STANDBY1	SHUTDOWN
SYSOSCCFG	DISABLE[1]	0	0	1	0	0	1	-	-	(1)	-	-	-
	USE4MHZSTOP	-	-	-	-	-	-	0	1	0	-	-	-
	DISABLESTOP	-	-	-	-	-	-	0	0	1	-	-	-
MCLKCFG	USELFCLK[1]	0	1	-	0	1	-	0	0	-	-	-	-
	STOPCLKSTBY	-	-	-	-	-	-	-	-	-	0	1	-
PMODECFG	DSLEEP	-	-	-	-	-	-	00	00	00	01	01	10
SCR	SLEEPDEEP	0	0	0	0	0	0	1	1	1	1	1	1

(1) The SYSOSCCFG.DISABLE and MCLKCFG.USELFCLK policy bits take effect immediately after being configured, as these bits affect the RUN mode behavior. Other policy bits only take effect when the CPU is put into deep sleep.
(2) SLEEP mode behavior is always identical to RUN mode, except with the CPUCLK disabled. As such, the SLEEP behavior is determined by the configuration of RUN mode.
(3) The STOP2 policy for STOP mode can be configured by setting the DISABLESTOP bit or DISABLE bit in the SYSOSCCFG register before entering DEEPSLEEP. When DISABLESTOP is set and DISABLE is cleared, SYSOSC is only disabled when DEEPSLEEP is requested. SYSOSC continues to run in RUN and SLEEP modes. When DISABLE is set, DISABLESTOP becomes a don't care, and SYSOSC is disabled immediately and is kept disabled in STOP mode.

Entering SLEEP Mode

Entering **SLEEP** mode disables the CPU, but otherwise maintains the same configuration as **RUN**. To enter **SLEEP** mode:

1. Configure the Cortex-M0+ CPU for SLEEP by clearing the SLEEPDEEP bit in the Cortex-M0+ SCR local register
2. Enter sleep mode by executing a WFI (wait for interrupt) CPU instruction

Entering STOP or STANDBY Modes

To enter **STOP** or **STANDBY** mode:

1. Configure the PMODECFG register in SYSCTL to 0b00 (STOP) or 0b01 (STANDBY)
2. Configure the Cortex-M0+ CPU for DEEP SLEEP by setting the SLEEPDEEP bit in the Cortex-M0+ SCR local register
3. Enter sleep mode by executing a WFI (wait for interrupt) CPU instruction

Entering SHUTDOWN Mode

To enter **SHUTDOWN** mode:

1. Configure the PMODECFG register in SYSCTL to 0b10 (SHUTDOWN)
2. Configure the Cortex-M0+ CPU for DEEP SLEEP by setting the SLEEPDEEP bit in the Cortex-M0+ SCR local register
3. Enter sleep mode by executing a WFI (wait for interrupt) CPU instruction

2.4.3 Asynchronous Fast Clock Requests

Peripherals are configured to asynchronously assert a hardware request to the SYSCTL for a fast clock source, even if the device is operating in STOP or STANDBY mode. This mechanism for applications where the MCLK/ULPCLK tree is normally sourced from either LFCLK (at 32kHz) or SYSOSC, but a faster clock is temporarily needed to quickly handle a peripheral event (for example, a timer IRQ or GPIO IRQ) or peripheral activity (such as serial communication or an ADC conversion).

Asynchronous fast clock requests are also useful for scenarios where the device is running in STANDBY1 mode. In STANDBY1 (when STOPCLKSTBY is set), the ULPCLK and LFCLK are disabled to all peripherals except for a few TIMGx, leaving TIMGx as the only clocked peripherals. To wake up the device from this state where the bus clock (ULPCLK) is disabled, a TIMGx interrupt request forces an asynchronous fast clock request to wake the device to RUN mode. Other peripherals can also wake the device from this state if they support detecting an asynchronous event (for example GPIO, comparator, and serial interfaces).

Asynchronous fast clock requests temporarily provide peripherals with bus clock (MCLK/ULPCLK), sourced from the SYSOSC, for the duration of the request. MFCLK, if enabled for use, is also enabled during the asynchronous request.

Asynchronous Fast Clock Behavior

When configured, SYSCTL will respond to a peripheral fast clock request in the following way:

1. If the device is currently in a STOP or STANDBY mode, the low power state is temporarily suspended to support running the bus clock (ULPCLK) at the SYSOSC base frequency (24MHz)
2. If SYSOSC is disabled, it is forced to be enabled; if SYSOSC is already running but at a different frequency than base frequency, it is forced to base frequency (24MHz)
3. The MCLK/ULPCLK tree is forced to be sourced from SYSOSC at the 24MHz rate; if the device is in RUN mode then the CPUCLK is also switches to the SYSOSC rate (the CPUCLK is always derived from MCLK)
4. If the MFCLK is configured to be used, it will be activated

After the configuration above is applied, it will be held for the duration of time that the asynchronous request remains asserted plus an additional 41 SYSOSC cycles (approximately 1µs). 41 SYSOSC cycles after the request is removed, the system will return to the configuration which existed before the fast clock request, provided the CPU did not change the configuration during the request.

Asynchronous fast clock requests are ignored and will have no effect on the device configuration if any of the following are true:

- MCLK is already sourced from SYSOSC at base frequency (24MHz)
- Asynchronous fast clock requests are globally blocked by setting the BLOCKASYNCALL bit in the SYSOSCCFG register in SYSCTL

Peripheral Support

The , TIMG8, GPIO, SPI, I2C, UART, and ADC peripherals all provide support for generating an asynchronous fast clock request. The purpose, request source, and configuration requirements for these peripherals are given in Table 2-7.

Table 2-7. Peripheral Support for Asynchronous Fast Clock Requests

Peripheral	Purpose	Request Source	Configuration
TIMG8	Fast CPU wake from TIMG8 event	TIMG8 IRQ to CPU	An IRQ event from TIMG8 generates an asynchronous fast clock request when the device is in STANDBY1 mode and the corresponding IMASK interrupt is set in the TIMG registers. This is needed to wake the device as the ULPCLK is disabled to reduce power consumption.
GPIO	Fast CPU wake from GPIO event	GPIO activity	The GPIO generates an asynchronous fast clock request through the GPIO configuration registers. This is for applications where GPIO wake from STANDBY mode is desired, as the fast clock request will cause the GPIO digital glitch filters to run at the rate. In addition to configuring the GPIO registers to request the fast clock, the BLOCKASYNCALL bit must be cleared in the SYSOSCCFG register to allow the request to propagate.
SPI	Temporarily use fast clock for bit clock generation	SPI activity	SPI activity generates an asynchronous fast clock request when the BLOCKASYNC bit is cleared in the CLKCFG register of the respective SPI peripheral.

Table 2-7. Peripheral Support for Asynchronous Fast Clock Requests (continued)

Peripheral	Purpose	Request Source	Configuration
I2C	Temporarily use fast clock for bit clock generation	I2C activity	I2C activity generates an asynchronous fast clock request when the BLOCKASYNC bit is cleared in the CLKCFG register of the respective I2C peripheral.
UART	Temporarily use a fast clock for baud rate generation	UART activity	UART activity generates an asynchronous fast clock request when the BLOCKASYNC bit is cleared in the CLKCFG register of the respective UART peripheral.
ADC	Temporarily run the SYSOSC to support timer-triggered ADC operation from a low-power mode	ADC	If an ADC conversion is triggered when SYSOSC is disabled, an asynchronous fast clock request is generated to enable the SYSOSC (SYSOSC is required for correct ADC operation).

Fast CPU Event Handling

In addition to the peripheral event and activity fast clock request triggers, the SYSCTL can be configured to generate an asynchronous fast clock request upon any IRQ request to the CPU. This provides the lowest latency interrupt handling when the system is running at the LFCLK rate (32kHz), as the IRQ request will propagate through the wake-up logic at the SYSOSC rate () vs. the LFCLK rate (32kHz). When the FASTCPUEVENT bit is set in the SYSOSCCFG register in SYSCTL, any interrupt request to the CPU will also generate a fast clock request.

Asynchronous Fast Clock Request Logic

The logic for asserting a fast clock request is given in the following figure.

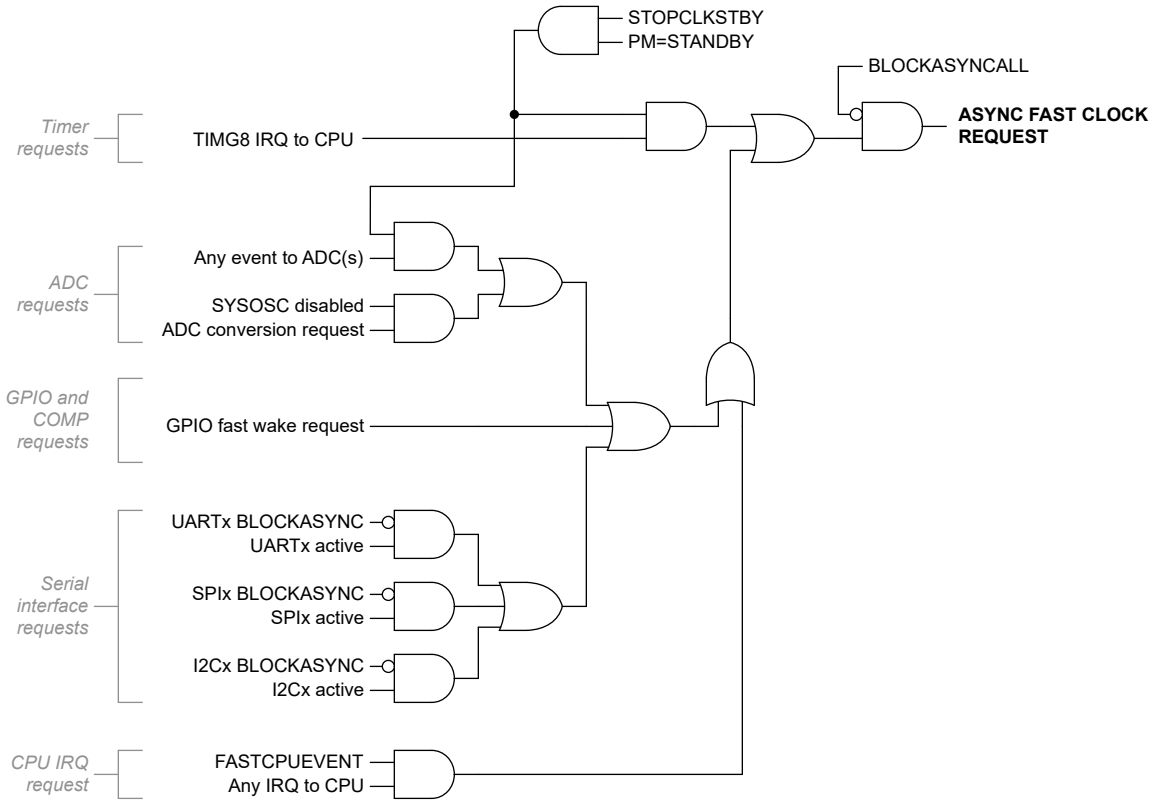


Figure 2-8. MSPM0Cxx Asynchronous Fast Clock Request Logic

2.4.4 SRAM Write Protection

Certain applications need to place read-only data into SRAM. This can occur if code is placed into SRAM (for zero wait state execution) or if critical lookup tables are placed in SRAM (for zero wait state reads). In these cases, especially when code is to be executed from the SRAM, it is desirable to prevent unintentional writes to SRAM addresses that can corrupt executable code in the event of a buffer overrun or a stack overflow. Likewise, it is desirable to prevent execution from nonwrite-protected SRAM addresses. To improve robustness of data in stored in SRAM, SYSCTL provides a write-exclusive-execute boundary mechanism.

To use this feature, first load the read-only data into the desired SRAM address, then configure the SRAM address range to be write protected. SRAM contents which are to be read-execute (no writes) should be placed into the upper portion of SRAM. SRAM contents which are to be read-write (no execute) should be placed into the lower portion of the SRAM. Then, the SRAMBOUNDARY register may be written with the desired boundary to partition the SRAM into two regions, with the lower region being RW and the upper region being RX.

2.4.5 Flash Wait States

Flash wait states are managed automatically by SYSCTL when MCLK is running from SYSOSC or LFCLK.

Refer to the *Recommended Operation Conditions* section of the device specific data sheet to determine the max clock frequency supported with 0 or 1 wait state.

2.4.6 Shutdown Mode Handling

When the device is configured to enter SHUTDOWN mode, the core regulator is powered down and the device register contents and SRAM contents are lost. An exit from SHUTDOWN mode generates a BOR level reset. Two mechanisms are provided to preserve the device state when entering SHUTDOWN mode: IO latching and a small shutdown memory.

Shutdown IO State

The digital IO pin states (output low/high, pullup/pulldown, Hi-Z, drive configuration) are latched and retained upon entry to SHUTDOWN. After exiting SHUTDOWN mode, the IOs are held in the previous state until released by application software setting the RELEASE bit in the SHDNIOREL register along with the matching KEY value. When exiting SHUTDOWN mode, application software must first re-configure the IO to their proper state, then release the IO. To determine at startup if the cause of a reset was an exit from SHUTDOWN mode, application software must read the RSTCAUSE register in SYSCTL.

Note

When exiting SHUTDOWN, **the serial wire debug (SWD) pins also remain locked until application software sets the RELEASE bit**. As a result, a debug connection cannot establish when waking up from SHUTDOWN mode until the IO are released by application software.

Note

When exiting SHUTDOWN, the bootstrap loader (BSL) invoke pin must be held at a logic low level to prevent unintended entry into the BSL during exit from SHUTDOWN. An entry to the BSL during SHUTDOWN exit prevents the application code from starting, the BSL interfaces are not be available, and a SWD connection is possible as the IO states remain latched through SHUTDOWN exit until application software releases the IOs.

Shutdown Memory

To enable saving of application state information before entering SHUTDOWN mode, 4 bytes of shutdown memory are provided in SYSCTL. These memory locations are retained in SHUTDOWN mode and are readable by the application after exiting SHUTDOWN. To save data to the SHUTDOWN memory, write to the SHUTDNSTORE0-SHUTDNSTORE3 registers in SYSCTL.

2.4.7 Configuration Lockout

Configuration registers in SYSCTL can be locked out from writes to add a layer of robustness against unintended changes to the PMCU at runtime. To lock out the configuration registers from writes, set the ACTIVE bit in the WRITELOCK register in SYSCTL.

All SYSCTL registers are protected by the WRITELOCK functionality except for those listed below:

- WRITELOCK
- PMODECFG
- FCC, FCCCMD
- FLBANKSWAP
- RSTCAUSE (read-to-clear), RESETLEVEL, RESETCMD
- BORTHRESHOLD, BORCLRCMD
- SHDNIOREL
- SHUTDNSTOREx

In addition to the overall SYSCTL configuration write lock feature, many SYSCTL registers also require a KEY value to be written in conjunction with the desire configuration data for the write to take effect.

2.4.8 System Status

The status of various aspects of the PMCU can be polled by software by reading the CLKSTATUS and SYSSTATUS registers in SYSCTL.

Checking Clock Status (CLKSTATUS)

The CLKSTATUS register in SYSCTL is a read-only register which indicates the current configuration and status of the clock module. Key status information provided in CLKSTATUS includes:

- The current SYSOSC frequency
- The current MCLK selection
- The LFOSC status
- Error indications if a peripheral requested a clock and the clock cannot be generated

This status information is useful to validate that a requested clock change has completed successfully, or to check the true SYSOSC frequency in applications where SYSOSC can have asynchronous activation or frequency requests issued by peripherals.

Checking System Status (SYSSTATUS)

The SYSSTATUS register in SYSCTL is a read-only register which indicates the peripheral-specific status information.

2.4.9 Error Handling

MSPM0 devices include several diagnostic mechanism to detect errors at runtime. Table 2-8 lists error sources and their corresponding handling mechanism.

Note

Not all MSPM0 devices support all diagnostic features. For example, some devices do not have ECC/parity on memories and some devices do not have dual watchdog timers. Always refer to the device-specific data sheet to understand which diagnostic features are available for a given device. In the PMCU registers section, register maps are also provided for each MCU subfamily detailing the specific registers available for a given device.

Table 2-8. Error Sources and Handling Mechanisms

Error Source	Error	Handling Mechanism
Flash (if device has ECC)	Non-correctable ECC error (if device has ECC)	• For a CPU or DMA request, a FLASHDED nonmaskable interrupt is generated to the processor or a SYSRST is generated depending on configuration of the FLASHECCRSTDIS bit • The FLASHDED sticky bit is set in the SYSSTATUS register in SYSCTL
	Correctable ECC error (if device has ECC)	• A FLASHSEC interrupt is also generated in SYSCTL • The FLASHSEC sticky bit is set in the SYSSTATUS register in SYSCTL
SRAM	Non-correctable ECC error (if device has ECC)	• An SRAMDED nonmaskable interrupt is generated to the processor
	Correctable ECC error (if device has ECC)	• A SYSCTL SRAMSED interrupt is generated to the processor
	Parity error (if device has parity)	• Nonmaskable interrupt is generated to the processor if the request was from the CPU • DMA data error interrupt is generated if the request was from the DMA
	Address error on CPU access	• A hard fault is generated in the CPU
	Address error on DMA access	• A DMA address error interrupt is generated in the DMA controller
	ECC error on CAN SRAM (if device has CAN-FD)	• An interrupt is generated in the CAN-FD peripheral
SHUTDNSTOREx Memory	Parity error	• A POR is generated
CKM	MCLK failure	• A BOOTRST is generated
	LFCLK failure (if present)	• A BOOTRST is generated if LFCLK is sourcing MCLK • An LFCLKFAIL nonmaskable interrupt is generated in the SYSCTL NMI registers.
CPUSS (if device has MPU)	Memory protection unit violation	• A hard fault is generated in the CPU
WWDT	WWDT0 violation	• A BOOTRST is generated or a nonmaskable interrupt is generated in the SYSCTL NMI registers depending on configuration of the WWDTLP0RSTDIS bit
	WWDT1 violation (if present)	• A BOOTRST is generated or a nonmaskable interrupt is generated in the SYSCTL NMI registers depending on configuration of the WWDTLP1RSTDIS bit
PMU	Trim parity error	• A POR is generated
	POR0- supply error	• A POR is generated
	BOR0- supply error	• A BOR is generated
	BOR1/2/3- supply error	• A BORLVL nonmaskable interrupt is generated in the SYSCTL NMI registers
CPUSS	Memory protection unit violation (if present)	• A hard fault is generated in the CPU

Configurable NMI Triggers

Error sources can be configured to trigger either a nonmaskable interrupt or a different handling mechanism. The SYSTEMCFG register in SYSCTL may be used to specify the desired error handling mechanism. For example, the WWDT0 may be configured to generate either a BOOTRST or an NMI, with BOOTRST being the default case. Refer to the SYSTEMCFG register for the relevant device subfamily for the available error handling options.

2.4.10 SYSCTL Events

The SYSCTL module contains two event publishers and no event subscribers. One event publisher manages SYSCTL interrupt requests (IRQs) to the CPU subsystem. The second publisher manages nonmaskable interrupts to the CPU subsystem for critical diagnostics.

The SYSCTL events are summarized in SYSCTL Events.

Table 2-9. SYSCTL Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt	Publisher	SYSCTL	CPU Subsystem	Static route	SYSCTL interrupt registers	Fixed interrupt route from SYSCTL to CPU
CPU nonmaskable interrupt (NMI)	Publisher	SYSCTL	CPU Subsystem	Static route	NMI interrupt registers	Fixed interrupt route from SYSCTL to CPU

2.4.10.1 CPU Interrupt Event (CPU_INT)

The SYSCTL module provides several interrupt sources which can be configured to source a CPU interrupt event. In order of decreasing interrupt priority, the CPU interrupt events from the SYSCTL are given in Table 2-10.

Table 2-10. SYSCTL CPU Interrupt Event Sources

Index (IIDX)	Name	Description
0	NONE	No interrupt pending.
1	LFOSCGOOD	Indicates when LFOSC is ready during startup, as LFOSC takes ≈1ms to start.
2	ANACLKERR	Indicates that an analog function was enabled and expecting a SYSOSC to be operation at a certain frequency, but SYSOSC was either not available or not operating at the required frequency.

The CPU interrupt event configuration is managed with the SYSCTL IIDX, IMASK, RIS, MIS, ISET, and ICLR event management registers. See Section 6.2.5 for guidance on configuring these registers for CPU interrupts.

2.5 Quick Start Reference

The PMCU is designed to provide a simple, easy-to-use power management, clocking, and reset management functionality. This section describes the basic operating principles of the PMCU as well as tips and tricks for taking the default configuration out of reset and optimizing it for particular applications.

2.5.1 Default Device Configuration

The default operating configuration of the device provides basic functionality which can be suitable for many applications without modification.

MSPM0Cxx devices power up and release reset for execution of application code when the external supply (VDD and VSS) reaches 1.62V. When the application code is released for execution, the device is in RUN mode with MCLK, which is sourced from the internal SYSOSC at 24 MHz. The CPUCLK and ULPCLK are also 24 MHz, derived from MCLK. LFCLK starts automatically, sourced from the internal LFOSC. In RUN mode with the default configuration, all peripherals are available to be enabled. Peripherals such as the run directly from

MCLK at the MCLK rate. Other peripherals, such as timers and serial interfaces, can run from the bus clock at 24 MHz or from the low-frequency 32kHz clock (LFCLK) based on their peripheral clock selection.

Power consumption can be reduced by entering SLEEP, STOP, STANDBY, or SHUTDOWN mode. By default, these modes behave in the following way:

- In SLEEP mode, the CPUCLK is disabled but all peripherals, including the DMA, continue to run as configured at either 24 MHz or 32kHz. This power mode is designed for scenarios where PD1 peripherals are used and having the lowest possible wakeup latency is more important than having the lowest power consumption.
- In STOP mode, (STOP0 by default), the MCLK source to PD1 peripherals is disabled and PD1 peripherals are disabled and in retention (unavailable for use). The default STOP mode is designed for scenarios where power optimization is important but the ADC, OPA, or a clock faster than 32kHz is needed.
  - SYSOSC by default will continue to run at , but the MCLK tree will run at 4MHz (SYSOSC/6) and peripherals in PD0 which are still active will see the bus clock (ULPCLK) change from 24 MHz to 4MHz.
  - PD0 peripherals configured to run from LFCLK continue to run at 32kHz.
  - The ADC will always see SYSOSC at 24 MHz for sampling
- In STANDBY mode (STANDBY0 by default), the MCLK tree runs from LFCLK at 32kHz and SYSOSC is disabled. PD0 peripherals running from the bus clock change to 32kHz. PD0 peripherals running from LFCLK continue to run at 32kHz with no change.

2.5.2 Leveraging MFCLK

When running with the default PMCU configuration, timers and serial interfaces can select either the bus clock (MCLK/ULPCLK) or the LFCLK as their clock source. LFCLK is always 32kHz in RUN, SLEEP, STOP, and STANDBY, but MCLK/ULPCLK changes to 4MHz in STOP and to 32kHz in STANDBY, meaning that peripherals running from the bus clock see the source clock frequency change when transitioning power modes.

MFCLK, by contrast, works like LFCLK in that it provides a constant frequency clock source for peripherals across RUN, SLEEP, and STOP modes. MFCLK provides a constant 4MHz as an alternative to LFCLK which runs at 32kHz. The 4MHz time base for MFCLK is always derived from SYSOSC. Peripherals, specifically PD0 peripherals that can be used in STOP mode, can select MFCLK as their clock source instead of ULPCLK. MFCLK is maintained at 4MHz in RUN, SLEEP and STOP for peripherals like UART, I2C, and low-power timers that need a consistent clock but require a clock source greater than 32kHz.

For information on using MFCLK, see the MFCLK section.

2.5.3 Optimizing Power Consumption in STOP Mode

The STOP mode provides considerable flexibility for tailoring the device to an application's specific power and performance requirements. There are several options available for reducing power consumption in STOP mode:

- By default, SYSOSC runs in STOP mode at 24MHz (base frequency) with a divide-by-6 to meet the 4MHz max frequency limit in STOP mode.
- If a 32kHz clock is sufficient to run the needed peripherals, it is possible to run in STOP mode with MCLK sourced from LFCLK at 32kHz. SYSOSC can be disabled to conserve power. To disable SYSOSC in STOP mode and run from LFCLK (STOP2), see disabling SYSOSC and operating mode selection.

2.5.4 Optimizing Power Consumption in STANDBY Mode

In STANDBY mode, if only TIMG8, or asynchronous fast wake from GPIO, comparator (low-power mode), or a serial interface is desired, the lowest possible power consumption can be achieved by configuring the ULPCLK and LFCLK to be disabled when entering STANDBY, leaving only the TIMG0, and TIMG1 running (STANDBY1). See the LFCLK section. In this state, TIMG8, or asynchronous activity/event will trigger an asynchronous fast clock request to wake the system.

2.5.5 Optimizing for Lowest Wakeup Latency

To ensure the lowest possible wakeup latency from STOP or STANDBY mode to RUN mode, set MCLK to SYSOSC with SYSOSC running at base frequency (24MHz) before entering STOP or STANDBY. SYSOSC

always starts at base frequency and latency is minimized if SYSOSC does not need to change to an alternate frequency.

2.5.6 Optimizing for Lowest Peak Current in RUN/SLEEP Mode

In applications which are peak current limited, there are two options for reducing active current in RUN and SLEEP modes:

- If 32kHz provides sufficient performance, run MCLK from LFCLK. MCLK can be selected to run from LFCLK with SYSOSC disabled. If no fast handling of events is needed, SYSOSC asynchronous requests can be disabled to ensure that the device always runs from LFCLK. This provides the lowest possible current with the CPU still running (RUN2). See the MCLK, SYSOSC, and Operating Mode Selection sections.
- If 32kHz does not provide sufficient performance, MCLK can be selected to run from SYSOSC with SYSOSC set to low frequency (4MHz). With MCLK running from SYSOSC, the MDIV divider for MCLK can be applied to reduce current consumption. For example, MCLK can be configured to run as low as 250kHz by setting MDIV to /16 when sourced from SYSOSC running at 4MHz. See the MCLK section.

2.6 SYSCTL_C1103_C1104 Registers

Table 2-11 lists the memory-mapped registers for the SYSCTL_C1103_C1104 registers. All register offset addresses not listed in Table 2-11 should be considered as reserved locations and the register contents should not be modified.

Table 2-11. SYSCTL_C1103_C1104 Registers

Offset	Acronym	Register Name	Section
1020h	IIDX	SYSCTL interrupt index	Section 2.6.1
1028h	IMASK	SYSCTL interrupt mask	Section 2.6.2
1030h	RIS	SYSCTL raw interrupt status	Section 2.6.3
1038h	MIS	SYSCTL masked interrupt status	Section 2.6.4
1040h	ISET	SYSCTL interrupt set	Section 2.6.5
1048h	ICLR	SYSCTL interrupt clear	Section 2.6.6
1050h	NMIIIDX	NMI interrupt index	Section 2.6.7
1060h	NMIRIS	NMI raw interrupt status	Section 2.6.8
1070h	NMIISET	NMI interrupt set	Section 2.6.9
1078h	NMIICLR	NMI interrupt clear	Section 2.6.10
1100h	SYSOSCCFG	SYSOSC configuration	Section 2.6.11
1104h	MCLKCFG	Main clock (MCLK) configuration	Section 2.6.12
1108h	HSCLKEN	High-speed clock (HSCLK) source enable/disable	Section 2.6.13
1138h	GENCLKCFG	General clock configuration	Section 2.6.14
113Ch	GENCLKEN	General clock enable control	Section 2.6.15
1140h	PMODECFG	Power mode configuration	Section 2.6.16
1150h	FCC	Frequency clock counter (FCC) count	Section 2.6.17
1178h	SRAMBOUNDARY	SRAM Write Boundary	Section 2.6.18
1180h	SYSTEMCFG	System configuration	Section 2.6.19
1190h	BEEPCFG	BEEPER Configuration	Section 2.6.20
1200h	WRITELOCK	SYSCTL register write lockout	Section 2.6.21
1204h	CLKSTATUS	Clock module (CKM) status	Section 2.6.22
1208h	SYSSTATUS	System status information	Section 2.6.23
1220h	RSTCAUSE	Reset cause	Section 2.6.24
1300h	RESETLEVEL	Reset level for application-triggered reset command	Section 2.6.25
1304h	RESETCMD	Execute an application-triggered reset command	Section 2.6.26
1308h	BORTHRESHOLD	BOR threshold selection	Section 2.6.27
130Ch	BORCLRCMD	Set the BOR threshold	Section 2.6.28
1310h	SYSOSCFCLCTL	SYSOSC frequency correction loop (FCL) ROSC enable	Section 2.6.29
1318h	EXLFCTL	LFCLK_IN and LFCLK control	Section 2.6.30
131Ch	SHDNIOREL	SHUTDOWN IO release control	Section 2.6.31
1320h	EXRSTPIN	Disable the reset function of the NRST pin	Section 2.6.32
1324h	SYSSTATUSCLR	Clear sticky bits of SYSSTATUS	Section 2.6.33
1328h	SWDCFG	Disable the SWD function on the SWD pins	Section 2.6.34
132Ch	FCCCMD	Frequency clock counter start capture	Section 2.6.35
1400h	SHUTDNSTORE0	Shutdown storage memory (byte 0)	Section 2.6.36
1404h	SHUTDNSTORE1	Shutdown storage memory (byte 1)	Section 2.6.37
1408h	SHUTDNSTORE2	Shutdown storage memory (byte 2)	Section 2.6.38
140Ch	SHUTDNSTORE3	Shutdown storage memory (byte 3)	Section 2.6.39

Complex bit access types are encoded to fit into small table cells. Table 2-12 shows the codes that are used for access types in this section.

Table 2-12. SYSCTL_C1103_C1104 Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value

2.6.1 IIDX Register (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 2-9 and described in Table 2-13.

Return to the Table 2-11.

SYSCTL interrupt index

Figure 2-9. IIDX Register

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

Table 2-13. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1-0	STAT	R	0h	The SYSCTL interrupt index (IIDX) register generates a value corresponding to the highest priority pending interrupt source. This value may be used as an address offset for fast, deterministic handling in the interrupt service routine. A read of the IIDX register will clear the corresponding interrupt status in the RIS and MIS registers. 0h = No interrupt pending 1h = LFOSCGOOD interrupt pending 2h = 2

2.6.2 IMASK Register (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 2-10 and described in Table 2-14.

Return to the Table 2-11.

SYSCTL interrupt mask

Figure 2-10. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ANACLKERR	LFOSCGOOD
R-0h						R/W-0h	R/W-0h

Table 2-14. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	ANACLKERR	R/W	0h	Analog Clocking Consistency Error 0h = 0 1h = 1
0	LFOSCGOOD	R/W	0h	Enable or disable the LFOSCGOOD interrupt. LFOSCGOOD indicates that the LFOSC has started successfully. 0h = Interrupt disabled 1h = Interrupt enabled

2.6.3 RIS Register (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 2-11 and described in Table 2-15.

Return to the Table 2-11.

SYSCTL raw interrupt status

Figure 2-11. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ANACLKERR	LFOSCGOOD
R-0h						R-0h	R-0h

Table 2-15. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	ANACLKERR	R	0h	Analog Clocking Consistency Error 0h = 0 1h = 1
0	LFOSCGOOD	R	0h	Raw status of the LFOSCGOOD interrupt. 0h = No interrupt pending 1h = Interrupt pending

2.6.4 MIS Register (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 2-12 and described in Table 2-16.

Return to the Table 2-11.

SYSCTL masked interrupt status

Figure 2-12. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ANACLKERR	LFOSCGOOD
R-0h						R-0h	R-0h

Table 2-16. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	ANACLKERR	R	0h	Analog Clocking Consistency Error 0h = 0 1h = 1
0	LFOSCGOOD	R	0h	Masked status of the LFOSCGOOD interrupt. 0h = No interrupt pending 1h = Interrupt pending

2.6.5 ISET Register (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 2-13 and described in Table 2-17.

Return to the Table 2-11.

SYSCTL interrupt set

Figure 2-13. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ANACLKERR	LFOSCGOOD
R-0h						W1S-0h	W1S-0h

Table 2-17. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	ANACLKERR	W1S	0h	Analog Clocking Consistency Error 0h = 0 1h = 1
0	LFOSCGOOD	W1S	0h	Set the LFOSCGOOD interrupt. 0h = Writing 0h hs no effect 1h = Set interrupt

2.6.6 ICLR Register (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 2-14 and described in Table 2-18.

Return to the Table 2-11.

SYSCTL interrupt clear

Figure 2-14. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ANACLKERR	LFOSCGOOD
R-0h						W1C-0h	W1C-0h

Table 2-18. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	ANACLKERR	W1C	0h	Analog Clocking Consistency Error 0h = 0 1h = 1
0	LFOSCGOOD	W1C	0h	Clear the LFOSCGOOD interrupt. 0h = Writing 0h has no effect 1h = Clear interrupt

2.6.7 NMIIIDX Register (Offset = 1050h) [Reset = 00000000h]

NMIIIDX is shown in Figure 2-15 and described in Table 2-19.

Return to the Table 2-11.

NMI interrupt index

Figure 2-15. NMIIIDX Register

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

Table 2-19. NMIIIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1-0	STAT	R	0h	The NMI interrupt index (NMIIIDX) register generates a value corresponding to the highest priority pending NMI source. This value may be used as an address offset for fast, deterministic handling in the NMI service routine. A read of the NMIIIDX register will clear the corresponding interrupt status in the NMIRIS register. 0h = No NMI pending 1h = BOR Threshold NMI pending 2h = 2

2.6.8 NMIRIS Register (Offset = 1060h) [Reset = 00000000h]

NMIRIS is shown in Figure 2-16 and described in Table 2-20.

Return to the Table 2-11.

NMI raw interrupt status

Figure 2-16. NMIRIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						WWDT0	BORLVL
R-0h						R-0h	R-0h

Table 2-20. NMIRIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	WWDT0	R	0h	Watch Dog 0 Fault 0h = 0 1h = 1
0	BORLVL	R	0h	Raw status of the BORLVL NMI 0h = No interrupt pending 1h = Interrupt pending

2.6.9 NMIISET Register (Offset = 1070h) [Reset = 00000000h]

NMIISET is shown in Figure 2-17 and described in Table 2-21.

Return to the Table 2-11.

NMI interrupt set

Figure 2-17. NMIISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						WWDT0	BORLVL
R-0h						W1S-0h	W1S-0h

Table 2-21. NMIISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	WWDT0	W1S	0h	Watch Dog 0 Fault 0h = 0 1h = 1
0	BORLVL	W1S	0h	Set the BORLVL NMI 0h = Writing 0h hs no effect 1h = Set interrupt

2.6.10 NMIICLR Register (Offset = 1078h) [Reset = 00000000h]

NMIICLR is shown in Figure 2-18 and described in Table 2-22.

Return to the Table 2-11.

NMI interrupt clear

Figure 2-18. NMIICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						WWDT0	BORLVL
R-0h						W1C-0h	W1C-0h

Table 2-22. NMIICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	WWDT0	W1C	0h	Watch Dog 0 Fault 0h = 0 1h = 1
0	BORLVL	W1C	0h	Clr the BORLVL NMI 0h = Writing 0h hs no effect 1h = Clear interrupt

2.6.11 SYSOSCCFG Register (Offset = 1100h) [Reset = 0002XXXXh]

SYSOSCCFG is shown in Figure 2-19 and described in Table 2-23.

Return to the Table 2-11.

SYSOSC configuration

Figure 2-19. SYSOSCCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						FASTCPUEVENT	BLOCKASYNCALL
R-0h						R/W-1h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED					DISABLE	DISABLESTOP	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED						FREQ	
R-0h						R/W-0h	

Table 2-23. SYSOSCCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	
17	FASTCPUEVENT	R/W	1h	FASTCPUEVENT may be used to assert a fast clock request when an interrupt is asserted to the CPU, reducing interrupt latency. 0h = An interrupt to the CPU will not assert a fast clock request 1h = An interrupt to the CPU will assert a fast clock request
16	BLOCKASYNCALL	R/W	0h	BLOCKASYNCALL may be used to mask block all asynchronous fast clock requests, preventing hardware from dynamically changing the active clock configuration when operating in a given mode. 0h = Asynchronous fast clock requests are controlled by the requesting peripheral 1h = All asynchronous fast clock requests are blocked
15-11	RESERVED	R	0h	
10	DISABLE	R/W	0h	DISABLE sets the SYSOSC enable/disable policy. SYSOSC may be powered off in RUN, SLEEP, and STOP modes to reduce power consumption. When SYSOSC is disabled, MCLK and ULPCLK are sourced from LFCLK. 0h = Do not disable SYSOSC 1h = Disable SYSOSC immediately and source MCLK and ULPCLK from LFCLK
9	DISABLESTOP	R/W	0h	DISABLESTOP sets the SYSOSC stop mode enable/disable policy. When operating in STOP mode, the SYSOSC may be automatically disabled. When set, ULPCLK will run from LFCLK in STOP mode and SYSOSC will be disabled to reduce power consumption. 0h = Do not disable SYSOSC in STOP mode 1h = Disable SYSOSC in STOP mode and source ULPCLK from LFCLK
8-2	RESERVED	R	0h	
1-0	FREQ	R/W	0h	Target operating frequency for the system oscillator (SYSOSC) 0h = Base frequency (32MHz) 1h = Low frequency (4MHz) 2h = User-trimmed frequency (16 or 24 MHz)

2.6.12 MCLKCFG Register (Offset = 1104h) [Reset = 000XXXX0h]

MCLKCFG is shown in Figure 2-20 and described in Table 2-24.

Return to the Table 2-11.

Main clock (MCLK) configuration

Figure 2-20. MCLKCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	MCLKDEADCHK	STOPCLKSTBY	USELFCLK	RESERVED			USEHSCLK
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			R/W-0h
15	14	13	12	11	10	9	8
RESERVED			USEMFTICK	RESERVED			
R-0h			R/W-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED				MDIV			
R-0h				R/W-0h			

Table 2-24. MCLKCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	
22	MCLKDEADCHK	R/W	0h	MCLKDEADCHK enables or disables the continuous MCLK dead check monitor. LFCLK must be running before MCLKDEADCHK is enabled. 0h = The MCLK dead check monitor is disabled 1h = The MCLK dead check monitor is enabled
21	STOPCLKSTBY	R/W	0h	STOPCLKSTBY sets the STANDBY mode policy (STANDBY0 or STANDBY1). When set, ULPCLK and LFCLK are disabled to all peripherals in STANDBY mode, with the exception of TIMG0 and TIMG1 which continue to run. Wake-up is only possible via an asynchronous fast clock request. 0h = ULPCLK/LFCLK runs to all PD0 peripherals in STANDBY mode 1h = ULPCLK/LFCLK is disabled to all peripherals in STANDBY mode except TIMG0 and TIMG1
20	USELFCLK	R/W	0h	USELFCLK sets the MCLK source policy. Set USELFCLK to use LFCLK as the MCLK source. Note that setting USELFCLK does not disable SYSOSC, and SYSOSC remains available for direct use by analog modules. 0h = MCLK will not use the low frequency clock (LFCLK) 1h = MCLK will use the low frequency clock (LFCLK)
19-17	RESERVED	R	0h	
16	USEHSCLK	R/W	0h	USEHSCLK, together with USELFCLK, sets the MCLK source policy. Set USEHSCLK to use HSCLK (HFCLK or SYSPLL) as the MCLK source in RUN and SLEEP modes. 0h = MCLK will not use the high speed clock (HSCLK) 1h = MCLK will use the high speed clock (HSCLK) in RUN and SLEEP mode
15-13	RESERVED	R	0h	

Table 2-24. MCLKCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	USEMFTICK	R/W	0h	USEMFTICK specifies whether the 4MHz constant-rate clock (MFCLK) to peripherals is enabled or disabled. When enabled, MDIV must be disabled (set to 0h=/1). 0h = The 4MHz rate MFCLK to peripherals is enabled 1h = The 4MHz rate MFCLK to peripherals is enabled.
11-4	RESERVED	R	0h	
3-0	MDIV	R/W	0h	MDIV may be used to divide the MCLK frequency when MCLK is sourced from SYSOSC. MDIV=0h corresponds to /1 (no divider). MDIV=1h corresponds to /2 (divide-by-2). MDIV=Fh corresponds to /16 (divide-by-16). MDIV may be set between /1 and /16 on an integer basis.

2.6.13 HSCLKEN Register (Offset = 1108h) [Reset = 0000XXXXh]

HSCLKEN is shown in Figure 2-21 and described in Table 2-25.

Return to the Table 2-11.

High-speed clock (HSCLK) source enable/disable

Figure 2-21. HSCLKEN Register

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

Table 2-25. HSCLKEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	USEEXTHFCLK	R/W	0h	USEEXTHFCLK selects the HFCLK_IN digital clock input to be the source for HFCLK. When disabled, HFXT is the HFCLK source and HFXTEN may be set. Do not set HFXTEN and USEEXTHFCLK simultaneously. 0h = Use HFXT as the HFCLK source 1h = Use the HFCLK_IN digital clock input as the HFCLK source
15-0	RESERVED	R	0h	

2.6.14 GENCLKCFG Register (Offset = 1138h) [Reset = 00000X0Xh]

GENCLKCFG is shown in Figure 2-22 and described in Table 2-26.

Return to the Table 2-11.

General clock configuration

Figure 2-22. GENCLKCFG Register

31	30	29	28	27	26	25	24
RESERVED			FCCTRIGCNT				
R-0h			R/W-0h				
23	22	21	20	19	18	17	16
ANACPUMPCFG		FCCLVLTRIG	FCCTRIGSRC	FCCSELCLK			
R/W-0h		R/W-0h	R/W-0h	R/W-0h			
15	14	13	12	11	10	9	8
HFCLK4MFPCLKDIV				RESERVED		MFPCLKSRC	RESERVED
R/W-0h				R-0h		R/W-0h	R-0h
7	6	5	4	3	2	1	0
EXCLKDIVEN	EXCLKDIVVAL			RESERVED	EXCLKSRC		
R/W-0h	R/W-0h			R-0h	R/W-0h		

Table 2-26. GENCLKCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28-24	FCCTRIGCNT	R/W	0h	FCCTRIGCNT specifies the number of trigger clock periods in the trigger window. FCCTRIGCNT=0h (one trigger clock period) up to 1Fh (32 trigger clock periods) may be specified.
23-22	ANACPUMPCFG	R/W	0h	ANACPUMPCFG selects the analog mux charge pump (VBOOST) enable method. 0h = VBOOST is enabled on request from a COMP, GPAMP, or OPA 1h = VBOOST is enabled when the device is in RUN or SLEEP mode, or when a COMP/GPAMP/OPA is enabled 2h = VBOOST is always enabled
21	FCCLVLTRIG	R/W	0h	FCCLVLTRIG selects the frequency clock counter (FCC) trigger mode. 0h = Rising edge to rising edge triggered 1h = Level triggered
20	FCCTRIGSRC	R/W	0h	FCCTRIGSRC selects the frequency clock counter (FCC) trigger source. 0h = FCC trigger is the external pin 1h = FCC trigger is the LFCLK
19-16	FCCSELCLK	R/W	0h	FCCSELCLK selectes the frequency clock counter (FCC) clock source. 0h = FCC clock is MCLK 1h = FCC clock is SYSOSC 2h = FCC clock is HFCLK 3h = FCC clock is the CLK_OUT selection 7h = FCC clock is the FCCIN external input

Table 2-26. GENCLKCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-12	HFCLK4MFPCLKDIV	R/W	0h	HFCLK4MFPCLKDIV selects the divider applied to HFCLK when HFCLK is used as the MFPCLK source. Integer dividers from /1 to /16 may be selected. 0h = HFCLK is not divided before being used for MFPCLK 1h = HFCLK is divided by 2 before being used for MFPCLK 2h = HFCLK is divided by 3 before being used for MFPCLK 3h = HFCLK is divided by 4 before being used for MFPCLK 4h = HFCLK is divided by 5 before being used for MFPCLK 5h = HFCLK is divided by 6 before being used for MFPCLK 6h = HFCLK is divided by 7 before being used for MFPCLK 7h = HFCLK is divided by 8 before being used for MFPCLK 8h = HFCLK is divided by 9 before being used for MFPCLK 9h = HFCLK is divided by 10 before being used for MFPCLK Ah = HFCLK is divided by 11 before being used for MFPCLK Bh = HFCLK is divided by 12 before being used for MFPCLK Ch = HFCLK is divided by 13 before being used for MFPCLK Dh = HFCLK is divided by 14 before being used for MFPCLK Eh = HFCLK is divided by 15 before being used for MFPCLK Fh = HFCLK is divided by 16 before being used for MFPCLK
11-10	RESERVED	R	0h	
9	MFPCLKSRC	R/W	0h	MFPCLKSRC selects the MFPCLK (middle frequency precision clock) source. 0h = MFPCLK is sourced from SYSOSC 1h = MFPCLK is sourced from HFCLK
8	RESERVED	R	0h	
7	EXCLKDIVEN	R/W	0h	EXCLKDIVEN enables or disables the divider function of the CLK_OUT external clock output block. 0h = CLock divider is disabled (passthrough, EXCLKDIVVAL is not applied) 1h = Clock divider is enabled (EXCLKDIVVAL is applied)
6-4	EXCLKDIVVAL	R/W	0h	EXCLKDIVVAL selects the divider value for the divider in the CLK_OUT external clock output block. 0h = CLK_OUT source is divided by 2 1h = CLK_OUT source is divided by 4 2h = CLK_OUT source is divided by 6 3h = CLK_OUT source is divided by 8 4h = CLK_OUT source is divided by 10 5h = CLK_OUT source is divided by 12 6h = CLK_OUT source is divided by 14 7h = CLK_OUT source is divided by 16
3	RESERVED	R	0h	
2-0	EXCLKSRC	R/W	0h	EXCLKSRC selects the source for the CLK_OUT external clock output block. ULPCLK and MFPCLK require the CLK_OUT divider (EXCLKDIVEN) to be enabled 0h = CLK_OUT is SYSOSC 1h = CLK_OUT is ULPCLK (EXCLKDIVEN must be enabled) 2h = CLK_OUT is LFCLK 3h = CLK_OUT is MFPCLK (EXCLKDIVEN must be enabled) 4h = CLK_OUT is HFCLK

2.6.15 GENCLKEN Register (Offset = 113Ch) [Reset = 0000000Xh]

GENCLKEN is shown in Figure 2-23 and described in Table 2-27.

Return to the Table 2-11.

General clock enable control

Figure 2-23. GENCLKEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			MFPCLKEN	RESERVED			EXCLKEN
R-0h			R/W-0h	R-0h			R/W-0h

Table 2-27. GENCLKEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4	MFPCLKEN	R/W	0h	MFPCLKEN enables the middle frequency precision clock (MFPCLK). 0h = MFPCLK is disabled 1h = MFPCLK is enabled
3-1	RESERVED	R	0h	
0	EXCLKEN	R/W	0h	EXCLKEN enables the CLK_OUT external clock output block. 0h = CLK_OUT block is disabled 1h = CLK_OUT block is enabled

2.6.16 PMODECFG Register (Offset = 1140h) [Reset = 00000000h]

PMODECFG is shown in Figure 2-24 and described in Table 2-28.

Return to the Table 2-11.

Power mode configuration

Figure 2-24. PMODECFG Register

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

Table 2-28. PMODECFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1-0	DSLEEP	R/W	0h	DSLEEP selects the operating mode to enter upon a DEEPSLEEP request from the CPU. 0h = STOP mode is entered 1h = STANDBY mode is entered 2h = SHUTDOWN mode is entered 3h = Reserved

2.6.17 FCC Register (Offset = 1150h) [Reset = 00000000h]

FCC is shown in Figure 2-25 and described in Table 2-29.

Return to the Table 2-11.

Frequency clock counter (FCC) count

Figure 2-25. FCC Register

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
RESERVED										DATA																					
R-0h										R-0h																					

Table 2-29. FCC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	
21-0	DATA	R	0h	Frequency clock counter (FCC) count value.

2.6.18 SRAMBOUNDARY Register (Offset = 1178h) [Reset = 000000XXh]

SRAMBOUNDARY is shown in Figure 2-26 and described in Table 2-30.

Return to the Table 2-11.

SRAM Write Boundary

Figure 2-26. SRAMBOUNDARY Register

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
RESERVED												ADDR															RESERVED				
R-0h												R/W-0h															R-0h				

Table 2-30. SRAMBOUNDARY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	
19-5	ADDR	R/W	0h	SRAM boundary configuration. The value configured into this acts such that: SRAM accesses to addresses less than or equal value will be RW only. SRAM accesses to addresses greater than value will be RX only. Value of 0 is not valid (system will have no stack). If set to 0, the system acts as if the entire SRAM is RWX. Any non-zero value can be configured, including a value = SRAM size.
4-0	RESERVED	R	0h	

2.6.19 SYSTEMCFG Register (Offset = 1180h) [Reset = 00XXXXXXh]

SYSTEMCFG is shown in Figure 2-27 and described in Table 2-31.

Return to the Table 2-11.

System configuration

Figure 2-27. SYSTEMCFG Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WWDTLP0RSTDIS
R-0h							R/W-0h

Table 2-31. SYSTEMCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value of 1Bh (27) must be written to KEY together with contents to be updated. Reads as 0 1Bh = Issue write
23-1	RESERVED	R	0h	
0	WWDTLP0RSTDIS	R/W	0h	WWDTLP0RSTDIS specifies whether a WWDT Error Event will trigger a BOOTRST or an NMI. 0h = WWDTLP0 Error Event will trigger a BOOTRST 1h = WWDTLP0 Error Event will trigger an NMI

2.6.20 BEEPCFG Register (Offset = 1190h) [Reset = 0000000Xh]

BEEPCFG is shown in Figure 2-28 and described in Table 2-32.

Return to the Table 2-11.

BEEPER Configuration

Figure 2-28. BEEPCFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										FREQ		RESERVED			EN
R-0h										R/W-0h		R-0h			R/W-0h

Table 2-32. BEEPCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5-4	FREQ	R/W	0h	Beeper Output Frequency Configuration 0h = Beeper runs at 8KHz 1h = Beeper runs at 4KHz 2h = Beeper runs at 2KHz 3h = Beeper runs at 1KHz
3-1	RESERVED	R	0h	
0	EN	R/W	0h	Beeper Output Enable 0h = Beeper Output Disabled 1h = Beeper Output Enabled

2.6.21 WRITELOCK Register (Offset = 1200h) [Reset = 00000000h]

WRITELOCK is shown in Figure 2-29 and described in Table 2-33.

Return to the Table 2-11.

SYSCTL register write lockout

Figure 2-29. WRITELOCK Register

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

Table 2-33. WRITELOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	ACTIVE	R/W	0h	ACTIVE controls whether critical SYSCTL registers are write protected or not. 0h = Allow writes to lockable registers 1h = Disallow writes to lockable registers

2.6.22 CLKSTATUS Register (Offset = 1204h) [Reset = XXXXXXXXh]

CLKSTATUS is shown in Figure 2-30 and described in Table 2-34.

Return to the Table 2-11.

Clock module (CKM) status

Figure 2-30. CLKSTATUS Register

31	30	29	28	27	26	25	24
ANACLKERR	RESERVED					FCCDONE	FCLMODE
R-0h	R-0h					R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED						CURMCLKSEL	RESERVED
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED				LFOSCGOOD	RESERVED		
R-0h				R-0h	R-0h		
7	6	5	4	3	2	1	0
LFCLKMUX		RESERVED	HSCLKMUX	RESERVED		SYSOSCFREQ	
R-0h		R-0h	R-0h	R-0h		R-0h	

Table 2-34. CLKSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ANACLKERR	R	0h	ANACLKERR is set when the device clock configuration does not support an enabled analog peripheral mode and the analog peripheral may not be functioning as expected. 0h = No analog clock errors detected 1h = Analog clock error detected
30-26	RESERVED	R	0h	
25	FCCDONE	R	0h	FCCDONE indicates when a frequency clock counter capture is complete. 0h = FCC capture is not done 1h = FCC capture is done
24	FCLMODE	R	0h	FCLMODE indicates if the SYSOSC frequency correction loop (FCL) is enabled. 0h = SYSOSC FCL is disabled 1h = SYSOSC FCL is enabled
23-18	RESERVED	R	0h	
17	CURMCLKSEL	R	0h	CURMCLKSEL indicates if MCLK is currently sourced from LFCLK. 0h = MCLK is not sourced from LFCLK 1h = MCLK is sourced from LFCLK
16-12	RESERVED	R	0h	
11	LFOSCGOOD	R	0h	LFOSCGOOD indicates when the LFOSC startup has completed and the LFOSC is ready for use. 0h = LFOSC is not ready 1h = LFOSC is ready
10-8	RESERVED	R	0h	
7-6	LFCLKMUX	R	0h	LFCLKMUX indicates if LFCLK is sourced from the internal LFOSC, the low frequency crystal (LFXT), or the LFCLK_IN digital clock input. 0h = LFCLK is sourced from the internal LFOSC 1h = LFCLK is sourced from the LFXT (crystal) 2h = LFCLK is sourced from LFCLK_IN (external digital clock input)
5	RESERVED	R	0h	

Table 2-34. CLKSTATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HSCLKMUX	R	0h	HSCLKMUX indicates if MCLK is currently sourced from the high-speed clock (HSCLK). 0h = MCLK is not sourced from HSCLK 1h = MCLK is sourced from HSCLK
3-2	RESERVED	R	0h	
1-0	SYSOSCFREQ	R	0h	SYSOSCFREQ indicates the current SYSOSC operating frequency. 0h = SYSOSC is at base frequency (32MHz) 1h = SYSOSC is at low frequency (4MHz) 2h = SYSOSC is at the user-trimmed frequency (16 or 24MHz) 3h = Reserved

2.6.23 SYSSTATUS Register (Offset = 1208h) [Reset = XXXXXXXXh]

SYSSTATUS is shown in Figure 2-31 and described in Table 2-35.

Return to the Table 2-11.

System status information

Figure 2-31. SYSSTATUS Register

31	30	29	28	27	26	25	24
REBOOTATTEMPTS		RESERVED					
R-0h		R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	SHDNIOLOCK	SWDCFGDIS	EXTRSTPINDIS	RESERVED			
R-0h	R-0h	R-0h	R-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED	PMUIREFGOOD	ANACPUMPGOOD	BORLVL	BORCURTHRESHOLD		RESERVED	
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	

Table 2-35. SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REBOOTATTEMPTS	R	0h	REBOOTATTEMPTS indicates the number of boot attempts taken before the user application starts.
29-15	RESERVED	R	0h	
14	SHDNIOLOCK	R	0h	SHDNIOLOCK indicates when IO is locked due to SHUTDOWN 0h = IO IS NOT Locked due to SHUTDOWN 1h = IO IS Locked due to SHUTDOWN
13	SWDCFGDIS	R	0h	SWDCFGDIS indicates when user has disabled the use of SWD Port 0h = SWD Port Enabled 1h = SWD Port Disabled
12	EXTRSTPINDIS	R	0h	EXTRSTPINDIS indicates when user has disabled the use of external reset pin 0h = External Reset Pin Enabled 1h = External Reset Pin Disabled
11-7	RESERVED	R	0h	
6	PMUIREFGOOD	R	0h	PMUIREFGOOD is set by hardware when the PMU current reference is ready. 0h = IREF is not ready 1h = IREF is ready
5	ANACPUMPGOOD	R	0h	ANACPUMPGOOD is set by hardware when the VBOOST analog mux charge pump is ready. 0h = VBOOST is not ready 1h = VBOOST is ready
4	BORLVL	R	0h	BORLVL indicates if a BOR event occured and the BOR threshold was switched to BOR0 by hardware. 0h = No BOR violation occured 1h = A BOR violation occured and the BOR threshold was switched to BOR0

Table 2-35. SYSSTATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	BORCURTHRESHOLD	R	0h	BORCURTHRESHOLD indicates the active brown-out reset supply monitor configuration. 0h = Default minimum threshold; a BOR0- violation triggers a BOR 1h = A BOR1- violation generates a BORLVL interrupt 2h = A BOR2- violation generates a BORLVL interrupt 3h = A BOR3- violation generates a BORLVL interrupt
1-0	RESERVED	R	0h	

2.6.24 RSTCAUSE Register (Offset = 1220h) [Reset = 00000000h]

RSTCAUSE is shown in Figure 2-32 and described in Table 2-36.

Return to the Table 2-11.

Reset cause

Figure 2-32. RSTCAUSE Register

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
RESERVED																											ID				
R-0h																											RC-0h				

Table 2-36. RSTCAUSE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4-0	ID	RC	0h	ID is a read-to-clear field which indicates the lowest level reset cause since the last read. 0h = No reset since last read 1h = POR- violation, SHUTDNSTOREx or PMU trim parity fault 2h = NRST triggered POR (>1s hold) 3h = Software triggered POR 4h = BOR0- violation 5h = SHUTDOWN mode exit 8h = Non-PMU trim parity fault 9h = Fatal clock failure Ch = NRST triggered BOOTRST (<1s hold) Dh = Software triggered BOOTRST Eh = WWDT0 violation 10h = BSL exit 11h = BSL entry 13h = WWDT1 violation 14h = Flash uncorrectable ECC error 15h = CPULOCK violation 1Ah = Debug triggered SYSRST 1Bh = Software triggered SYSRST 1Ch = Debug triggered CPURST 1Dh = Software triggered CPURST

2.6.25 RESETLEVEL Register (Offset = 1300h) [Reset = 00000000h]

RESETLEVEL is shown in Figure 2-33 and described in Table 2-37.

Return to the Table 2-11.

Reset level for application-triggered reset command

Figure 2-33. RESETLEVEL Register

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

Table 2-37. RESETLEVEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	
2-0	LEVEL	R/W	0h	LEVEL is used to specify the type of reset to be issued when RESETCMD is set to generate a software triggered reset. 0h = Issue a SYSRST (CPU plus peripherals only) 1h = Issue a BOOTRST (CPU, peripherals, and boot configuration routine) 2h = Issue a SYSRST and enter the boot strap loader (BSL) 3h = Issue a power-on reset (POR) 4h = Issue a SYSRST and exit the boot strap loader (BSL)

2.6.26 RESETCMD Register (Offset = 1304h) [Reset = 00XXXXXXh]

RESETCMD is shown in Figure 2-34 and described in Table 2-38.

Return to the Table 2-11.

Execute an application-triggered reset command

Figure 2-34. RESETCMD Register

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

Table 2-38. RESETCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value of E4h (228) must be written to KEY together with GO to trigger the reset. E4h = Issue reset
23-1	RESERVED	R	0h	
0	GO	W	0h	Execute the reset specified in RESETLEVEL.LEVEL. Must be written together with the KEY. 1h = Issue reset

2.6.27 BORTHRESHOLD Register (Offset = 1308h) [Reset = 00000000h]

BORTHRESHOLD is shown in Figure 2-35 and described in Table 2-39.

Return to the Table 2-11.

BOR threshold selection

Figure 2-35. BORTHRESHOLD Register

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

Table 2-39. BORTHRESHOLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1-0	LEVEL	R/W	0h	LEVEL specifies the desired BOR threshold and BOR mode. 0h = Default minimum threshold; a BOR0- violation triggers a BOR 1h = A BOR1- violation generates a BORLVL interrupt 2h = A BOR2- violation generates a BORLVL interrupt 3h = A BOR3- violation generates a BORLVL interrupt

2.6.28 BORCLRCMD Register (Offset = 130Ch) [Reset = 00XXXXXXh]

BORCLRCMD is shown in Figure 2-36 and described in Table 2-40.

Return to the Table 2-11.

Set the BOR threshold

Figure 2-36. BORCLRCMD Register

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

Table 2-40. BORCLRCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value of C7h (199) must be written to KEY together with GO to trigger the clear and BOR threshold change. C7h = Issue clear
23-1	RESERVED	R	0h	
0	GO	W	0h	GO clears any prior BOR violation status indications and attempts to change the active BOR mode to that specified in the LEVEL field of the BORTHRESHOLD register. 1h = Issue clear

2.6.29 SYSOSCFCLCTL Register (Offset = 1310h) [Reset = 00XXXXXXh]

SYSOSCFCLCTL is shown in Figure 2-37 and described in Table 2-41.

Return to the Table 2-11.

SYSOSC frequency correction loop (FCL) ROSC enable

Figure 2-37. SYSOSCFCLCTL Register

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

Table 2-41. SYSOSCFCLCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value of 2Ah (42) must be written to KEY together with SETUSEFCL to enable the FCL. 2Ah = Issue Command
23-1	RESERVED	R	0h	
0	SETUSEFCL	W	0h	Set SETUSEFCL to enable the frequency correction loop in SYSOSC. Once enabled, this state is locked until the next BOOTRST. 1h = Enable the SYSOSC FCL

2.6.30 EXLFCTL Register (Offset = 1318h) [Reset = 00XXXXXXh]

EXLFCTL is shown in Figure 2-38 and described in Table 2-42.

Return to the Table 2-11.

LFCLK_IN and LFCLK control

Figure 2-38. EXLFCTL Register

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

Table 2-42. EXLFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value of 36h (54) must be written to KEY together with SETUSEEXLF to set SETUSEEXLF. 36h = Issue command
23-1	RESERVED	R	0h	
0	SETUSEEXLF	W	0h	Set SETUSEEXLF to switch LFCLK to the LFCLK_IN digital clock input. Once set, SETUSEEXLF remains set until the next BOOTRST. 1h = Use LFCLK_IN as the LFCLK source

2.6.31 SHDNIOREL Register (Offset = 131Ch) [Reset = 00XXXXXXh]

SHDNIOREL is shown in Figure 2-39 and described in Table 2-43.

Return to the Table 2-11.

SHUTDOWN IO release control

Figure 2-39. SHDNIOREL Register

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

Table 2-43. SHDNIOREL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value 91h must be written to KEY together with RELEASE to set RELEASE. 91h = Issue command
23-1	RESERVED	R	0h	
0	RELEASE	W	0h	Set RELEASE to release the IO after a SHUTDOWN mode exit. 1h = Release IO

2.6.32 EXRSTPIN Register (Offset = 1320h) [Reset = 00XXXXXXh]

EXRSTPIN is shown in Figure 2-40 and described in Table 2-44.

Return to the Table 2-11.

Disable the reset function of the NRST pin

Figure 2-40. EXRSTPIN Register

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

Table 2-44. EXRSTPIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value 1Eh must be written together with DISABLE to disable the reset function. 1Eh = Issue command
23-1	RESERVED	R	0h	
0	DISABLE	W	0h	Set DISABLE to disable the reset function of the NRST pin. Once set, this configuration is locked until the next POR. 0h = Reset function of NRST pin is enabled 1h = Reset function of NRST pin is disabled

2.6.33 SYSSTATUSCLR Register (Offset = 1324h) [Reset = 00XXXXXXh]

SYSSTATUSCLR is shown in Figure 2-41 and described in Table 2-45.

Return to the Table 2-11.

Clear sticky bits of SYSSTATUS

Figure 2-41. SYSSTATUSCLR Register

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

Table 2-45. SYSSTATUSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value CEh (206) must be written to KEY together with ALLECC to clear the ECC state. CEh = Issue command
23-1	RESERVED	R	0h	
0	ALLECC	W	0h	Set ALLECC to clear all ECC related SYSSTATUS indicators. 1h = Clear ECC error state

2.6.34 SWDCFG Register (Offset = 1328h) [Reset = 00XXXXXXh]

SWDCFG is shown in Figure 2-42 and described in Table 2-46.

Return to the Table 2-11.

Disable the SWD function on the SWD pins

Figure 2-42. SWDCFG Register

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

Table 2-46. SWDCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value 62h (98) must be written to KEY together with DISBALE to disable the SWD functions. 62h = Issue command
23-1	RESERVED	R	0h	
0	DISABLE	W	0h	Set DISABLE to disable the SWD function on SWD pins, allowing the SWD pins to be used as GPIO. 1h = Disable SWD function on SWD pins

2.6.35 FCCCMD Register (Offset = 132Ch) [Reset = 00XXXXXXh]

FCCCMD is shown in Figure 2-43 and described in Table 2-47.

Return to the Table 2-11.

Frequency clock counter start capture

Figure 2-43. FCCCMD Register

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

Table 2-47. FCCCMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	The key value 0Eh (14) must be written with GO to start a capture. 0Eh = Issue command
23-1	RESERVED	R	0h	
0	GO	W	0h	Set GO to start a capture with the frequency clock counter (FCC). 1h = 1

2.6.36 SHUTDNSTORE0 Register (Offset = 1400h) [Reset = 00000000h]

SHUTDNSTORE0 is shown in Figure 2-44 and described in Table 2-48.

Return to the Table 2-11.

Shutdown storage memory (byte 0)

Figure 2-44. SHUTDNSTORE0 Register

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

Table 2-48. SHUTDNSTORE0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
7-0	DATA	R/W	0h	Shutdown storage byte 0

2.6.37 SHUTDNSTORE1 Register (Offset = 1404h) [Reset = 00000000h]

SHUTDNSTORE1 is shown in Figure 2-45 and described in Table 2-49.

Return to the Table 2-11.

Shutdown storage memory (byte 1)

Figure 2-45. SHUTDNSTORE1 Register

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

Table 2-49. SHUTDNSTORE1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
7-0	DATA	R/W	0h	Shutdown storage byte 1

2.6.38 SHUTDNSTORE2 Register (Offset = 1408h) [Reset = 00000000h]

SHUTDNSTORE2 is shown in Figure 2-46 and described in Table 2-50.

Return to the Table 2-11.

Shutdown storage memory (byte 2)

Figure 2-46. SHUTDNSTORE2 Register

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

Table 2-50. SHUTDNSTORE2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
7-0	DATA	R/W	0h	Shutdown storage byte 2

2.6.39 SHUTDNSTORE3 Register (Offset = 140Ch) [Reset = 00000000h]

SHUTDNSTORE3 is shown in Figure 2-47 and described in Table 2-51.

Return to the Table 2-11.

Shutdown storage memory (byte 3)

Figure 2-47. SHUTDNSTORE3 Register

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

Table 2-51. SHUTDNSTORE3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
7-0	DATA	R/W	0h	Shutdown storage byte 3

Chapter 3

CPU

The CPU subsystem (MCPUSS) includes the Arm Cortex-M0+ processor, the interrupt logic, and the prefetch and cache logic.

3.1 Overview

The MSP CPU subsystem (MCPUSS) contains the central processing unit (CPU) along with associated supporting logic and read-only memory (ROM). The functional blocks that comprise the MCPUSS include:

- The Arm Cortex-M0+ 32-bit CPU and internal peripherals
- The CPU bus splitter and router
- The interrupt management logic and DEEPSLEEP entry and exit logic
- The nonvolatile memory system prefetch and cache logic
- The CPU debug interface to the debug subsystem (DEBUGSS)
- The read-only memory (ROM) used for the BCR and BSL

The top level architecture of the MCPUSS is shown in .

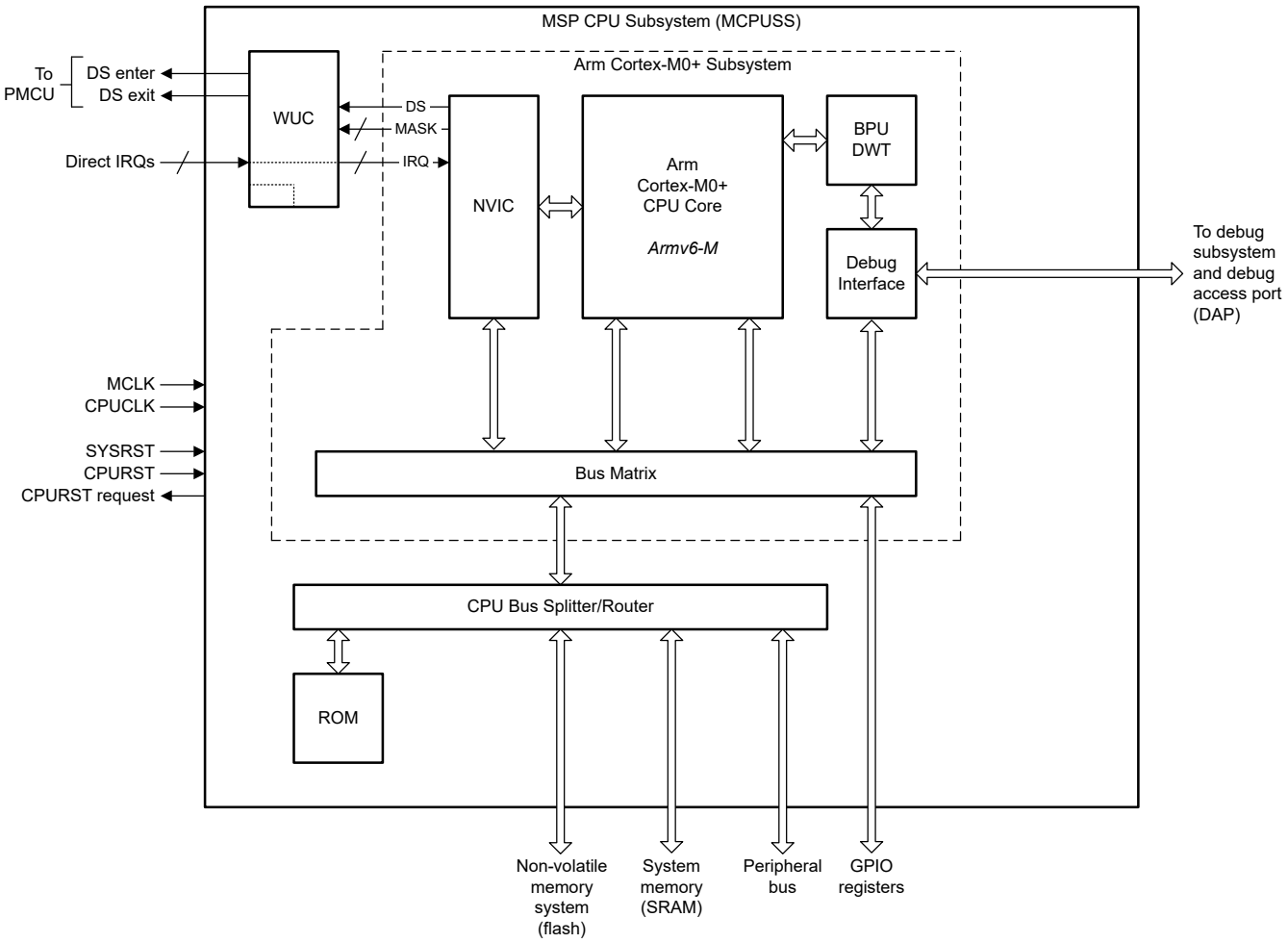


Figure 3-1. MSPM0Cxx MCPUSS Top Level Diagram

3.2 Arm Cortex-M0+ CPU

The MCPUSS contains an energy-efficient Arm Cortex-M0+ CPU implementing the Armv6-M instruction set architecture (ISA) with support for CPU clock speeds up to 24MHz. The Cortex-M0+ is a Von Neumann style 32-bit processor with a 2-stage ultra-low power pipeline and a single-cycle access port to the GPIO registers for efficient GPIO manipulation.

The Cortex-M0+ implementation on MSPM0Cxx devices has the following features:

- Up to 24MHz execution frequency
- Little-endian (least significant byte at lowest byte address location)

- Support for 32-bit word instruction fetches
- Single cycle 32×32 multiply instruction
- User and privileged execution modes
- Two hardware breakpoints and one hardware watchpoint for debug
- Reset-all-registers support
- Vector table offset support

The Cortex-M0+ architecture enables excellent code density, deterministic interrupt handling, and upwards compatibility with other processor architectures in the Arm Cortex-M family.

A general overview of the Arm Cortex-M0+ is given in this section to provide a basic understanding of the features of the processor. For detailed information on developing with the Arm Cortex-M0+ processor, refer to the *Arm Cortex-M0+ Devices Generic User's Guide*.

3.2.1 CPU Register File

The Arm Cortex-M0+ processor instructions operate on registers in the CPU register file. The processor contains a register file consisting of 16 standard registers and 3 special registers as shown in Figure 3-2.

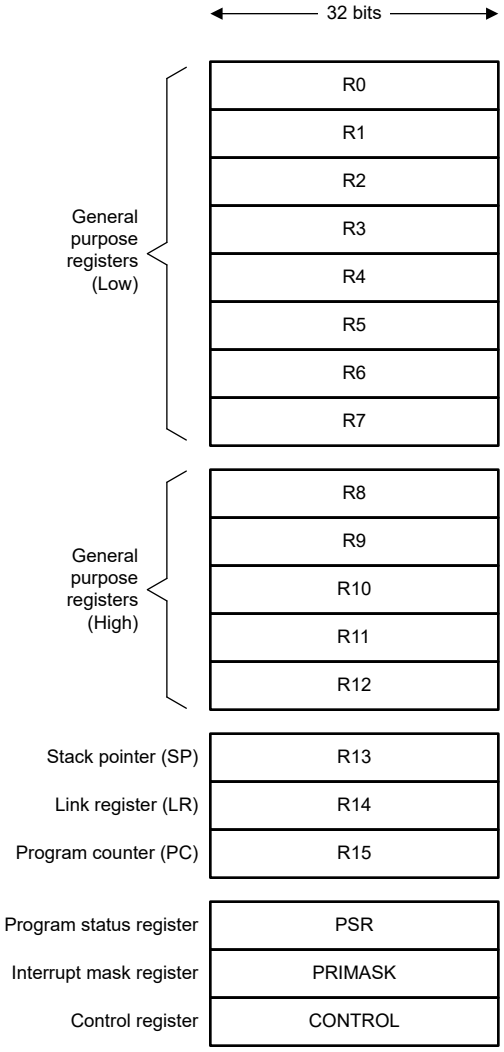


Figure 3-2. CPU Registers

General Purpose Registers (R0-R12)

The processor provides 13 general purpose registers, R0-R12, for operating on data. Registers R0 to R7 (low registers) are accessible by all instructions which specify a general purpose register. Registers R8 to R12 (high registers) are not accessible by 16-bit instructions but are accessible by any 32-bit instructions which specify a general purpose register.

Stack Pointer Register (R13)

The stack pointer is contained in R13, and can contain the main stack pointer (MSP) or the process stack pointer (PSP) . When the processor is running in handler mode, the main stack pointer (MSP) is always used. When the processor is running in thread mode, the MSP or the process stack pointer (PSP) can be used, depending on the configuration of the SPSEL bit in the CONTROL register.

After a CPURST, the processor automatically and unconditionally fetches the default stack pointer from the first address of main flash (0x0000.0000) as the main stack pointer (MSP).

Link Register (R14)

R14 serves as the link register and contains the return value of function calls as well as exceptions. The link register must be set before being used as it is not reset to any known value. It is accessible in privileged and unprivileged mode.

Program Counter Register (R15)

The program counter register (R15) contains the address of the next instruction to be executed. The PC is accessible in privileged and unprivileged mode.

After a CPURST, the processor automatically and unconditionally fetches the default PC from the second word of main flash (0x0000.0004).

Special Registers

Special registers include the program status register (PSR), the interrupt mask register (PRIMASK), and the control register (CONTROL). Special registers are typically accessed by using the CPS, MRS, and MSR system instructions.

- **Program Status Register (PSR):** The PSR is a combination of the application status (APSR), interrupt status (IPSR), and execution status (EPSR) registers. Application software can access the PSR with MRS and MSR instructions, accessing either the complete PSR or a combination of one or more registers, with some restrictions. The PSR registers can be accessed with MRS and MSR instructions using the mnemonics given in Table 3-1.
  - The application status register (APSR) contains the N, Z, C, and V flags which are used by the processor to evaluate conditional branch instructions. These bits are located in BIT31, BIT30, BIT29, and BIT28 of the PSR, respectively.
  - The interrupt status register (IPSR) reports the current exception number for a currently executing exception when in handler mode. In thread mode it reads as zero. The processor ignores writes to this register. The exception number field is presented in from BIT0 to BIT5 of the PSR.
  - The execution status register (EPSR) contains the T bit (BIT24), which defines whether the processor is in Thumb state. This bit cannot be read or written by software, but it is used by the processor.
- **Interrupt Mask Register (PRIMASK):** BIT0 of the PRIMASK register (PM) can be used to mask all interrupts to the processor which have configurable priority (see Section 3.3). This can be thought of as a global peripheral interrupt mask control. The processor ignores unprivileged writes to PRIMASK. Clearing PM to 0 enables interrupts. Setting PM to 1 disables interrupts. The CPS instruction can be used to change the PM bit value in the PRIMASK register.
- **Control Register (CONTROL):** The control register can be used to define whether code executing in thread mode is privileged or unprivileged by clearing or setting the nPRIV bit (BIT0), respectively. It can also be used

to select the stack pointer used in R13 as either the main stack pointer (MSP) or process stack pointer (PSP) by clearing or setting the SPSEL bit (BIT1), respectively. A CPURST clears the CONTROL register to zero. The processor ignores unprivileged writes to the CONTROL register. The SPSEL stack pointer selection bit is updated by the processor automatically when entering and returning from exceptions. Note that software must implement an ISB barrier instruction after writing to CONTROL to ensure that any changes take effect before the next application instruction is executed by the processor.

Table 3-1. Program Status Register (PSR) Access Mnemonics

Mnemonic	Subregisters Included
APSR	APSR
IPSR	IPSR
EPSR	EPSR
IAPSR	IPSR and APSR
EAPSR	EPSR and APSR
XPSR	APSR, IPSR, EPSR
IEPSR	IPSR and EPSR

3.2.2 Stack Behavior

The Arm Cortex-M0+ processor implements a full descending stack protocol. The stack pointer register (SP) always indicates the location of the last stacked data. When new data is added to the call stack, the SP value is decremented and the data is written to the location indicated by the SP after being decremented.

The Arm Cortex-M0+ supports managing two independent stacks with two pointers: the main stack (MSP) and the process stack (PSP).

3.2.3 Execution Modes and Privilege Levels

The processor supports two primary modes of execution:

- **Thread mode** (for executing application software)
- **Handler mode** (for handling processor exceptions and peripheral interrupts)

By default, the processor is in thread mode out of reset. If an exception is issued to the processor, the processor will handle the exception in handler mode and return to thread mode after handler execution is complete. Code running in thread mode can be configured as being privileged or unprivileged, based on the configuration of the processor's internal CONTROL register. Code running in handler mode always executes as privileged.

In general, code which executes as privileged has complete control of the processor configuration, including control of the , NVIC, and SCB. Only privileged code can change the privilege level for code running in thread mode.

Code that is executing in thread mode in an unprivileged state cannot access the previously mentioned resources (NVIC, SCB).

3.2.4 Address Space and Supported Data Sizes

MSPM0 devices implement a flat memory map with a 32-bit byte-addressable address space. Byte addresses are unsigned numbers ranging from zero to 2^{32}-1.

Address Space

The processor sees the address space as containing 2^{30} 32-bit words, with each word being word-aligned (4-byte aligned). Pointers are always 32 bits, and stack operations (for example, push , pop) increment the stack pointer by 4 addresses (4 bytes). Address calculations by the processor wrap around if they overflow or underflow the 32-bit memory space.

Instruction fetches by the processor are always 16-bit half-word aligned.

Data reads by the processor must be naturally aligned (for example, words must be word aligned, half words must be half-word aligned, etc.).

Supported Data Sizes

The processor supports 8-bit byte, 16-bit half-word, and 32-bit word data sizes. Signed and unsigned data is supported, and signed data is stored in CPU registers in 32-bit two's complement format. The Armv6-M instruction set does not provide native instructions supporting operations on 64-bit double-word data.

Load operations from memory to a CPU register can be signed or unsigned when the data size is less than 32 bits. When loading unsigned half-word or byte data to a CPU register, the value is zero-extended to 32 bits automatically. When loading signed half-word or byte data to a CPU register, the value is sign-extended to 32 bits automatically.

Stores from CPU registers to memory are sign agnostic.

All instruction and data accesses use little endian byte order.

3.3 Interrupts and Exceptions

Peripheral interrupt exceptions and system exceptions temporarily pause the processor's normal execution flow so that the processor can be used to handle an event.

The following can cause interruption of normal execution flow:

- A CPURST
- A fault exception in the system (HardFault)
- Execution of a supervisor call instruction (SVCall)
- Setting of a pending supervisor service request (PendSV)
- An enabled peripheral interrupt (IRQ)
- A breakpoint instruction (for debug)

Exception State

Each exception source to the processor will be in one of the below states at any given point in time:

- **Inactive** (not active, not pending)
- **Pending** (waiting to be serviced by the processor)
- **Active** (actively being serviced by the processor)
- **Active and pending** (actively being serviced by the processor when the same source generates another exception)

Exception Prioritization, Entry, and Exit

Exceptions are prioritized by the processor together with the nested vectored interrupt controller (NVIC). Each exception has either a fixed priority (reset, hard fault) or a configurable priority (SVCall, PendSV, peripheral IRQs). Exceptions with configurable priority can be disabled by application software running in privileged mode. Exceptions with fixed priority cannot be disabled.

The processor exception model supports preemption, tail-chaining, and late-arrival features to boost exception handling performance:

- In the **preemption** case, if an exception of higher priority is pending when an exception of lower priority is executing, the higher priority exception will preempt the ongoing handler servicing the lower priority exception.
- In the **tail-chaining** case, if a valid-for-entry exception is pending at the time of completion of an exception handler, then the application context is not restored from the stack and control is given immediately to the pending exception.
- In the **late-arriving** case, if a higher priority exception occurs during entry to a lower priority exception, the higher priority exception will be serviced first after the processor state is saved to the stack. Once the higher

priority exception handling is complete, the lower priority exception (which is still pending) is serviced based on the tail-chaining procedure.

An exception entry is issued if and when all of the following are true:

- An exception is in a **pending** state
- The priority of the pending exception is higher than the limit set by the exception mask register (PRIMASK)
- The processor is currently in either thread mode (not servicing an exception) or the newly pending exception is higher priority than the exception which is currently being serviced (resulting in a preemption)

Processor exceptions are vectored. When an exception occurs, the current processor state is pushed onto the stack which was active at the time of the event, and execution is vectored to the entry point address in the vector table corresponding to the exception which is to be processed.

If the exception is tail-chained to a previous handler which has completed, then there is no need to push any state to the stack and the interrupt service routine can be vectored to immediately. Likewise, if the exception is higher priority than a previous exception which started entry but did not complete entry, then there is no need to save the context again (late arrival).

Upon completion of an exception handler, if there is no exception pending which needs to be handled then the processor will pop the state from the stack and restore the processor to the previous state which it was in when the exception occurred.

3.3.1 Peripheral Interrupts (IRQs)

Peripheral interrupt functionality is managed by several components on the device:

- The nested vectored interrupt controller (NVIC)
- One or more interrupt groups (INT_GROUP)
- The wake-up controller (WUC)

MSPM0 devices include an Arm nested vectored interrupt controller (NVIC) with the Cortex-M0+ CPU for managing peripheral interrupts. The NVIC operation is tightly integrated with the processor and supports up to 32 native peripheral interrupt sources.

In addition to the NVIC, interrupt grouping modules can be present on a device to enable interfacing of more than 32 peripheral interrupts to the NVIC. High-priority interrupt sources that might require preemption capability are mapped directly to the NVIC and behave like normal NVIC interrupts. Lower priority interrupt sources which do not commonly require preemption capability are mapped to an interrupt grouping module, the output of which is then mapped to the NVIC as a native NVIC interrupt source. This routing arrangement is shown in Figure 3-3.

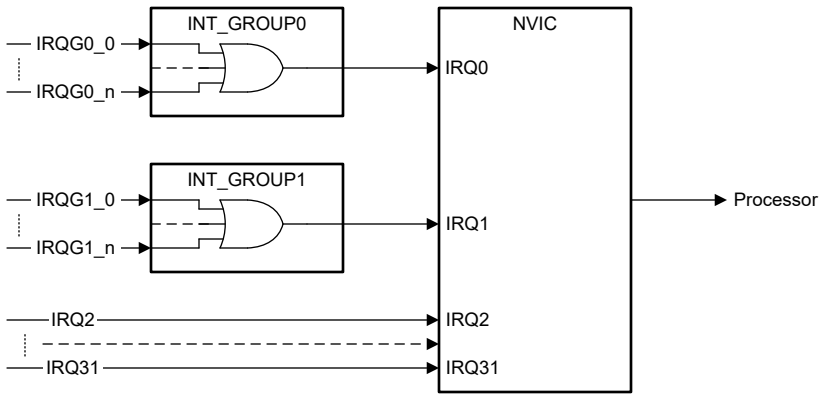


Figure 3-3. Peripheral Interrupt Hierarchy

The wake-up controller (WUC) determines if the PD1 power domain (containing the processor) needs to be powered up to service a peripheral interrupt if the PD1 domain is powered down in STOP or STANDBY mode.

3.3.1.1 Nested Vectored Interrupt Controller (NVIC)

The nested vectored interrupt controller (NVIC) is an industry-standard Arm component which interfaces peripheral interrupts (which are external to the processor) into the CPU. The NVIC supports connection of up to 32 native peripheral interrupt sources.

The NVIC is configured through memory-mapped registers in the system private peripheral bus (PPB) region. See Table 3-2 for the list of NVIC registers. The software development kit (SDK) provided with the devices supports the standard Arm Cortex Microcontroller Software Interface Standard (CMSIS) register access definitions for the NVIC. Application software must use 32-bit aligned, word-size transactions when accessing any NVIC register.

In addition to interfacing peripheral interrupts to the processor, the NVIC also supports programmable priority for each interrupt.

Enabling and Disabling Interrupts

Peripheral interrupt enables can be read, set, and cleared through the interrupt set-enable (ISER) and interrupt clear-enable (ICER) registers in the NVIC. The 32 interrupts are mapped to the ISER and ICER registers with interrupt zero in the BIT0 position (LSB) and interrupt 31 in the BIT31 position (MSB) of each register. To enable an interrupt, set the corresponding enable bit in the ISER register. Writing a '0' to ISER has no effect. It is possible to read the ISER register to determine which interrupts are enabled. Upon a read, a '1' indicates that an interrupt is enabled; a '0' indicates disabled. To disable an interrupt, set the corresponding enable bit in the ICER register. Writing a '0' to ICER has no effect.

Note

In addition to enabling a peripheral interrupt at the NVIC, it is generally necessary to also configure the interrupt configuration of the corresponding peripheral as well. Most peripherals have multiple interrupt sources, which are merged together in the peripheral to source a single NVIC interrupt. Masking of individual peripheral interrupts is done within the peripheral's interrupt management registers.

In the event that an interrupt is disabled in the NVIC, if the interrupt is asserted by the corresponding peripheral then the NVIC interrupt will go to a pending state but the processor is not interrupted. If an interrupt is disabled when in an active state (when a handler is running) it will remain active until the exception handler returns or a reset occurs, but no further activations will happen.

Note

If a peripheral asserts an interrupt to the NVIC, but that peripheral's interrupt in the NVIC is disabled, the device may remain in a higher power mode than expected as the wake-up controller (WUC) is holding an event for the processor. To prevent this situation, ensure peripheral interrupts are masked at the peripheral directly, versus only masking interrupts at the NVIC.

Setting and Clearing Pending Interrupt Status

Pending interrupt status can be read, set, and cleared through the interrupt set-pending (ISPR) and interrupt clear-pending (ICPR) registers in the NVIC. The 32 interrupts are mapped to the ISPR and ICPR registers with interrupt zero in the BIT0 position (LSB) and interrupt 31 in the BIT31 position (MSB) of each register. To read if an interrupt is pending, read either the ISPR or the ICPR. Upon a read, a '1' indicates that an interrupt is pending; a '0' indicates not pending. To set an interrupt to a pending state through software, set the corresponding bit in the ISPR register. Writing a '0' to ISPR has no effect. To clear an interrupt pending state, set the corresponding bit in the ICPR register. Writing a '0' to ICPR has no effect. Note that if a peripheral interrupt condition is still present, the pending state will be set again by hardware even if it is cleared.

Setting Interrupt Priority

Interrupts on the NVIC have programmable priority. There are four priority levels possible. Priority is set by programming the eight IPRx registers in the NVIC. Each priority field is 8 bits in length, and the priority for 4 interrupts is configured per 32-bit register. The Arm Cortex-M0+ only implements the most significant 2 bits of each 8-bit priority field (giving the 4 priority levels). Lower priority values have higher priority. System exceptions (reset, NMI, hard fault) have fixed priorities of -3, -2, and -1, respectively. As such, these exceptions always have higher priority than peripheral interrupts. Peripheral interrupt priorities are programmable as 0, 64, 128, or 192, with 0 being highest priority and 192 being lowest priority.

If the processor is currently handling an exception, it can only be preempted by a higher priority exception. In the event that there are multiple exceptions in a pending state which all have the same priority level assigned, the exception with the lowest exception number is taken first.

Note

Application software must not change the priority of an interrupt while the corresponding interrupt is either active (being handled) or enabled. Doing so can result in unpredictable behavior.

Table 3-2. Arm Cortex-M0+ NVIC Registers

Address	Register	CMSIS	Description
0xE000.E100	NVIC_ISER	NVIC->ISER[0]	Interrupt set-enable register
0xE000.E180	NVIC_ICER	NVIC->ICER[0]	Interrupt clear-enable register
0xE000.E200	NVIC_ISPR	NVIC->ISPR[0]	Interrupt set-pending register
0xE000.E280	NVIC_ICPR	NVIC->ICPR[0]	Interrupt clear-pending register
0xE000.E400	NVIC_IPR0	NVIC->IP[0]	Interrupt priority register (0-3)
0xE000.E404	NVIC_IPR1	NVIC->IP[1]	Interrupt priority register (4-7)
0xE000.E408	NVIC_IPR2	NVIC->IP[2]	Interrupt priority register (8-11)
0xE000.E40C	NVIC_IPR3	NVIC->IP[3]	Interrupt priority register (12-15)
0xE000.E410	NVIC_IPR4	NVIC->IP[4]	Interrupt priority register (16-19)
0xE000.E414	NVIC_IPR5	NVIC->IP[5]	Interrupt priority register (20-23)
0xE000.E418	NVIC_IPR6	NVIC->IP[6]	Interrupt priority register (24-27)
0xE000.E41C	NVIC_IPR7	NVIC->IP[7]	Interrupt priority register (28-31)

3.3.1.2 Interrupt Groups

To support mapping of more than 32 peripheral interrupt sources to the NVIC, certain MSPM0 devices include interrupt grouping logic (INT_GROUP) in the MCPUSS to combine several interrupts together to source one native NVIC interrupt.

The INT_GROUP interrupt grouping uses the MSPM0 event management register structure with the key difference being that all peripheral interrupt sources to the interrupt group are always unmasked (always enabled) such that no additional enable configuration is needed beyond the peripheral interrupt configuration and the NVIC configuration. The IMASK register itself is read-only and hardwired to enable all sources to the interrupt group. Figure 3-4 shows the INT_GROUP structure.

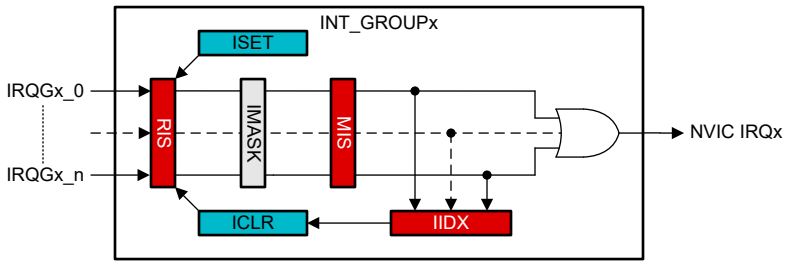


Figure 3-4. Interrupt Group (INT_GROUP)

Because no masking control is required, application software only needs to interface with the interrupt index (IIDX) register to efficiently handle a peripheral interrupt which is registered to the NVIC through an INT_GROUP.

Application software can read the IIDX register in the INT_GROUP to determine and clear the highest priority pending peripheral interrupt in the group. A read to IIDX will return an index corresponding to the highest priority peripheral which set an interrupt. The read action will also simultaneously clear the RIS and MIS bits corresponding to the highest priority interrupt whose index was returned by the read. The value read from the IIDX register can then be used in a case statement, as shown below.

```
void GROUP_HANDLER(void)
{
  switch(IIDX)
  {
    case 0:         // no IRQ pending
      break;
    case 1:         // IRQ[0]
      do_peripheral_1_ISR();
      break;
    case 2:         // IRQ[1]
      do_peripheral_2_ISR();
      break;
    default:        // out of range
      illegal();
  }
}
```

Usage

Because peripheral interrupts that are grouped together into an INT_GROUP source a single NVIC interrupt, it is not possible to have one peripheral interrupt in a group preempt the execution of an active handler for the interrupt group. For example, take the scenario where the WWDT0 interrupt request line and the PMCU interrupt request line are connected to INT_GROUP0, and INT_GROUP0 sources NVIC peripheral interrupt 0. If the WWDT0 interrupt is asserted, INT_GROUP0 will assert an interrupt request to the NVIC. If no higher priority interrupt is active, the NVIC will vector the processor to the INT_GROUP0 handler. Application software can then read the INT_GROUP0 IIDX register to determine that it was the WWDT0 that triggered the INT_GROUP0 interrupt on the NVIC, and software can jump to the WWDT0 handler function.

If the PMCU asserts its interrupt line while the processor is still in handler mode servicing the WWDT0 function (which is really a part of the INT_GROUP0 handler), the PMCU interrupt can not preempt the WWDT0 handler. When the WWDT0 handler completes, The INT_GROUP0 handler will return. At that time, the processor will see that the INT_GROUP0 request was asserted again (this time, due to the PMCU), and it will tail-chain a second entry to the interrupt handler. This time, application software will read the IIDX and determine that the PMCU was the cause of the INT_GROUP0 interrupt being asserted to the NVIC.

If there are two or more peripheral interrupts pending to a single interrupt group, software can set the priority with which the interrupts are handled by first reading the RIS or MIS register to test which peripheral interrupts are asserted, followed by executing the software-determined priority. Alternatively, if the interrupt index (IIDX) register is used, the interrupt group hardware will return the highest priority index based on the index order.

3.3.1.3 Wake Up Controller (WUC)

The wake up controller (WUC) is responsible for monitoring for assertion of interrupts when the processor is powered down in the STOP or STANDBY operating mode. In these modes, the entire PD1 power domain is power gated, and as such, the processor and NVIC are not available to check for interrupts. The WUC retains a copy of which peripheral interrupt sources to the NVIC were enabled when the processor entered STOP or STANDBY mode. In the event that an enabled interrupt is issued, the WUC will handshake with the PMCU to bring the device out of STOP or STANDBY mode so that the CPU can service the interrupt. The WUC will capture the interrupt state and present it to the NVIC and processor when the processor is brought up, such that the processor will see the interrupt even if the raw interrupt status of the peripheral is removed before the processor finishes powering up to service the interrupt.

The WUC requires no configuration by application software upon entry to or exit from low-power modes, and operation is transparent to application software.

3.3.2 Interrupt and Exception Table

MSPM0 devices share a common interrupt and exception mapping across devices. Specific devices do not implement all interrupt sources, but if a peripheral is common to two devices, that peripheral has the same mapping to the NVIC on both devices.

See the device-specific data sheet for a complete list of which interrupts a particular device supports.

The Arm Cortex-M0+ interrupt vector table is 48 words long (192 bytes). The complete platform interrupt and exception table with vector table addresses is given in Table 3-3.

Table 3-3. MSPM0 Platform Processor Interrupt and Exception Table

Exception Number	NVIC Number(1)	Priority Group	Exception or Interrupt	Vector Table Address	Vector Description
-	-	-	-	0x0000.0000	Stack pointer
1	-	-3	Reset	0x0000.0004	Reset vector
2	-	-2	NMI	0x0000.0008	NMI handler
3	-	-1	Hard fault	0x0000.000C	Hard fault handler
4	-	-	Reserved	0x0000.0010	-
5	-	-	Reserved	0x0000.0014	-
6	-	-	Reserved	0x0000.0018	-
7	-	-	Reserved	0x0000.001C	-
8	-	-	Reserved	0x0000.0020	-
9	-	-	Reserved	0x0000.0024	-
10	-	-	Reserved	0x0000.0028	-
11	-	Selectable	SVCall	0x0000.002C	Supervisor call handler
12	-	-	Reserved	0x0000.0030	-
13	-	-	Reserved	0x0000.0034	-
14	-	Selectable	PendSV	0x0000.0038	Pended supervisor handler
15	-	Selectable	SysTick	0x0000.003C	SysTick handler
16	0	Selectable	INT_GROUP0	0x0000.0040	Combined peripheral group 0 handler (see INT_GROUP0 below)
17	1	Selectable	INT_GROUP1	0x0000.0044	Combined peripheral group 1 handler (see INT_GROUP1 below)
18	2	Selectable		0x0000.0048	
19	3	Selectable		0x0000.004C	
20	4	Selectable	ADC0	0x0000.0050	ADC0 interrupt handler
21	5	Selectable		0x0000.0054	
22	6	Selectable		0x0000.0058	
23	7	Selectable		0x0000.005C	
24	8	Selectable	Reserved	0x0000.0060	
25	9	Selectable	SPI0	0x0000.0064	SPI0 interrupt handler
26	10	Selectable		0x0000.0068	
27	11	Selectable	Reserved	0x0000.006C	-
28	12	Selectable	Reserved	0x0000.0070	-
29	13	Selectable	UART1	0x0000.0074	UART1 interrupt handler
30	14	Selectable		0x0000.0078	
31	15	Selectable	UART0	0x0000.007C	UART0 interrupt handler
32	16	Selectable	TIMG0	0x0000.0080	Timer TIMG0 interrupt handler

Table 3-3. MSPM0 Platform Processor Interrupt and Exception Table (continued)

Exception Number	NVIC Number[1]	Priority Group	Exception or Interrupt	Vector Table Address	Vector Description
33	17	Selectable		0x0000.0084	
34	18	Selectable		0x0000.0088	
35	19	Selectable		0x0000.008C	
36	20	Selectable		0x0000.0090	
37	21	Selectable		0x0000.0094	
38	22	Selectable	Reserved	0x0000.0098	-
39	23	Selectable	Reserved	0x0000.009C	-
40	24	Selectable	I2C0	0x0000.00A0	I2C0 interrupt handler
41	25	Selectable	I2C1	0x0000.00A4	I2C1 interrupt handler
42	26	Selectable	Reserved	0x0000.00A8	-
43	27	Selectable	Reserved	0x0000.00AC	-
44	28	Selectable		0x0000.00B0	
45	29	Selectable	Reserved	0x0000.00B4	-
46	30	Selectable		0x0000.00B8	
47	31	Selectable	DMA	0x0000.00BC	DMA interrupt handler

(1) The NVIC number also indicates the relative interrupt priority if multiple NVIC interrupts have the same group priority. However, an interrupt does not preempt an active handler for another interrupt with the same group priority, even if the interrupt has a higher (numerically lower) NVIC position. For preemption to occur, the new interrupt must be configured to a higher priority group (numerically lower).

Non-Maskable Interrupt (NMI)

The CPU implements a nonmaskable interrupt, which handles critical interrupts which must be serviced immediately by the processor. The NMI interrupt sources are managed by SYSCTL. See the corresponding NMI information in the SYSCTL section of the PMCU chapter.

INT_GROUP0 Peripheral Interrupt Group

The INT_GROUP0 peripheral interrupt group sources an interrupt to NVIC0 (exception 16) if any peripheral in the group has a pending interrupt. The peripheral interrupts mapped to INT_GROUP0 are given in Table 3-4.

Table 3-4. INT_GROUP0 Interrupts

Priority	IIDX Index	Interrupt	Description
0	1	WWDT0	WWDT0 interrupt handler
1	2	Reserved	-
2	3	DEBUGSS	Debug subsystem interrupt handler
3	4	FLASHCTL	Flash controller interrupt handler
4	5	WUC FSUB0	Generic event subscriber 0 interrupt handler
5	6	WUC FSUB1	Generic event subscriber 1 interrupt handler
6	7	PMCU (SYSCTL)	PMCU (system controller) interrupt handler
7	8	Reserved	-

INT_GROUP1 Peripheral Interrupt Group

The INT_GROUP1 peripheral interrupt group sources an interrupt to NVIC1 (exception 17) if any peripheral in the group has a pending interrupt. The peripheral interrupts mapped to INT_GROUP1 are given in Table 3-5.

Table 3-5. INT_GROUP1 Interrupts

Priority	IIDX Index	Interrupt	Description
0	1	GPIO0	GPIO0 interrupt handler
1	2	Reserved	-
2	3		
3	4	Reserved	-
4	5	Reserved	-
5	6	Reserved	-
6	7	Reserved	-
7	8	Reserved	-

Note

Not all of the devices support Interrupt Group or NMI (non-maskable interrupt). Refer to the device-specific data sheet to see which devices support them.

3.3.3 Processor Lockup Scenario

There are several exception conditions which can cause the processor to enter a lockup state. On MSPM0 devices, a processor lockup is considered a fatal fault which always triggers a SYSRST to clear the lockup condition and restart the system.

A lockup state is entered by the processor if an SVC (supervisor call) or fault condition occurs while the processor is handling an exception with priority of -1 or higher (numerically lower). Such a fault is considered by the processor to be unexpected under normal operating conditions.

The following examples are conditions that can trigger a lockup state in the processor:

- The processor cannot fetch the stack pointer or reset vector at reset
- The processor cannot fetch the NMI vector
- The processor cannot fetch the hard fault vector
- A memory fault occurs when the processor is already handling an exception with priority of -1 or -2 (hard fault or NMI)
- A supervisor call (SVC) occurs when the processor is already handling an exception with priority of -1 or -2 (hard fault or NMI)
- A usage fault or undefined instruction is fetched when the processor is already handling an exception with priority of -1 or -2 (hard fault or NMI)
- A BKPT instruction is executed when the processor is already handling an exception with priority of -1 or -2 (hard fault or NMI)

3.4 CPU Peripherals

The Arm Cortex-M0+ includes tightly coupled peripherals for system timing and memory protection.

3.4.1 System Control Block (SCB)

The system control block (SCB) provides system implementation information and system control functionality, as well as configuration, control, and reporting of processor exceptions.

The SCB is configured through memory-mapped registers in the system private peripheral bus (PPB) region. See Table 3-6 for the list of SCB registers. The software development kit (SDK) provided with the devices supports the standard Arm Cortex Microcontroller Software Interface Standard (CMSIS) register access definitions for the SCB. Application software must use 32-bit aligned, word-size transactions when accessing any SCB register.

Table 3-6. Arm Cortex-M0+ System Control Block Registers

Address	Register	CMSIS	Description
0xE000.ED00	CPUID	SCB->CPUID	Read-only register indicating the CPU type and revision

Table 3-6. Arm Cortex-M0+ System Control Block Registers (continued)

Address	Register	CMSIS	Description
0xE000.ED04	ICSR	SCB->ICSR	Provides specific interrupt controls and state
0xE000.ED08	VTOR	SCB->VTOR	Used to specify the vector table offset from 0x0000.0000
0xE000.ED0C	AIRCR	SCB->AIRCR	Used to issue a CPU reset request (SYSRESETREQ)
0xE000.ED10	SCR	SCB->SCR	System control register, used to control low-power mode behavior
0xE000.ED14	CCR	SCB->CCR	Read-only register indicating behavior of the processor
0xE000.ED1C	SHPR2	SCB->SHP[2]	Used to configure the priority of the SVCall system handler
0xE000.ED20	SHPR3	SCB->SHP[3]	Used to configure the priority of the and PendSV system handlers

For detailed information on the system control block register configuration, see the SCB section of the Arm Cortex-M0+ devices generic user guide.

3.5 Read-Only Memory (ROM)

The MCPUSS contains a read-only memory which contains the executable code for the boot configuration routine (BCR) and bootstrap loader (BSL). The ROM is active after a BOOTRST, or after a SYSRST with BSL entry/exit. The ROM is disabled automatically when the application is started and is not accessible by application software.

3.6 CPUSS Registers

Table 3-7 lists the memory-mapped registers for the CPUSS registers. All register offset addresses not listed in Table 3-7 should be considered as reserved locations and the register contents should not be modified.

Table 3-7. CPUSS Registers

Offset	Acronym	Register Name	Section
10E0h	EVT_MODE	Event Mode	Section 3.6.1
10FCh	DESC	Module Description	Section 3.6.2
1100h	IIDX	Interrupt index	Section 3.6.3
1108h	IMASK	Interrupt mask	Section 3.6.4
1110h	RIS	Raw interrupt status	Section 3.6.5
1118h	MIS	Masked interrupt status	Section 3.6.6
1120h	ISET	Interrupt set	Section 3.6.7
1128h	ICLR	Interrupt clear	Section 3.6.8

Complex bit access types are encoded to fit into small table cells. Table 3-8 shows the codes that are used for access types in this section.

Table 3-8. CPUSS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value

3.6.1 EVT_MODE Register (Offset = 10E0h) [Reset = 00000000h]

EVT_MODE is shown in Figure 3-5 and described in Table 3-9.

Return to the Table 3-7.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 3-5. EVT_MODE Register

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

Table 3-9. EVT_MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1-0	INT_CFG	R	0h	Event line mode select 0h = The interrupt or event line is disabled. 1h = Event handled by software. Software must clear the associated RIS flag. 2h = Event handled by hardware. The hardware (another module) clears automatically the associated RIS flag.

3.6.2 DESC Register (Offset = 10FCh) [Reset = 00000000h]

DESC is shown in Figure 3-6 and described in Table 3-10.

Return to the Table 3-7.

This register identifies the peripheral and its exact version.

Figure 3-6. DESC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				RESERVED				MAJREV				MINREV			
R-0h				R-0h				R-0h				R-0h			

Table 3-10. DESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	0h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness.
15-12	FEATUREVER	R	0x0	Feature Set for the module *instance*
11-8	RESERVED	R	0h	
7-4	MAJREV	R	0x0	Major rev of the IP
3-0	MINREV	R	0x0	Minor rev of the IP

3.6.3 IIDX Register (Offset = 1100h) [Reset = 00000000h]

IIDX is shown in Figure 3-7 and described in Table 3-11.

Return to the Table 3-7.

Interrupt index register. This read-only register provides the interrupt index of the pending interrupt with the highest priority. It also indicates if no interrupt is pending. The priority order is fixed: lower index equals higher priority. Alternatively to the use of IIDX, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred.
On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flags in the RIS and MIS are cleared as well. After a read from the CPU (not from the debug interface), the register must be updated with the next highest priority interrupt or indicate that no interrupt is pending. Only interrupts which are selected via IMASK are indicated.

Figure 3-7. IIDX Register

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 3-11. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 0h = No pending interrupt 1h = Interrupt 0 2h = Interrupt 1 3h = Interrupt 2 4h = Interrupt 3 5h = Interrupt 4 6h = Interrupt 5 7h = Interrupt 6 8h = Interrupt 7

3.6.4 IMASK Register (Offset = 1108h) [Reset = 000000FFh]

IMASK is shown in Figure 3-8 and described in Table 3-12.

Return to the Table 3-7.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS."

Figure 3-8. IMASK Register

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
RESERVED																								INT							
R-0h																								R-FFh							

Table 3-12. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	INT	R	0xFF	Masks the corresponding interrupt 1h = Interrupt 0 2h = Interrupt 1 4h = Interrupt 2 8h = Interrupt 3 10h = Interrupt 4 20h = Interrupt 5 40h = Interrupt 6 80h = Interrupt 7

3.6.5 RIS Register (Offset = 1110h) [Reset = 00000000h]

RIS is shown in Figure 3-9 and described in Table 3-13.

Return to the Table 3-7.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 3-9. RIS Register

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
RESERVED																								INT							
R-0h																								R-0h							

Table 3-13. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	INT	R	0h	Raw interrupt status for INT 1h = Interrupt 0 2h = Interrupt 1 4h = Interrupt 2 8h = Interrupt 3 10h = Interrupt 4 20h = Interrupt 5 40h = Interrupt 6 80h = Interrupt 7

3.6.6 MIS Register (Offset = 1118h) [Reset = 00000000h]

MIS is shown in Figure 3-10 and described in Table 3-14.

Return to the Table 3-7.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 3-10. MIS Register

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
RESERVED																								INT							
R-0h																								R-0h							

Table 3-14. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	INT	R	0h	Masked interrupt status for INT0 1h = Interrupt 0 2h = Interrupt 1 4h = Interrupt 2 8h = Interrupt 3 10h = Interrupt 4 20h = Interrupt 5 40h = Interrupt 6 80h = Interrupt 7

3.6.7 ISET Register (Offset = 1120h) [Reset = 00000000h]

ISET is shown in Figure 3-11 and described in Table 3-15.

Return to the Table 3-7.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 3-11. ISET Register

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
RESERVED																								INT							
R-0h																								R-0h							

Table 3-15. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	INT	R	0h	Sets INT in RIS register 1h = Interrupt 0 2h = Interrupt 1 4h = Interrupt 2 8h = Interrupt 3 10h = Interrupt 4 20h = Interrupt 5 40h = Interrupt 6 80h = Interrupt 7

3.6.8 ICLR Register (Offset = 1128h) [Reset = 00000000h]

ICLR is shown in Figure 3-12 and described in Table 3-16.

Return to the Table 3-7.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 3-12. ICLR Register

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
RESERVED																								INT							
R-0h																								R-0h							

Table 3-16. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	INT	R	0h	Clears INT in RIS register 1h = Interrupt 0 2h = Interrupt 1 4h = Interrupt 2 8h = Interrupt 3 10h = Interrupt 4 20h = Interrupt 5 40h = Interrupt 6 80h = Interrupt 7

3.7 WUC Registers

Table 3-17 lists the memory-mapped registers for the WUC registers. All register offset addresses not listed in Table 3-17 should be considered as reserved locations and the register contents should not be modified.

Table 3-17. WUC Registers

Offset	Acronym	Register Name	Section
400h	FSUB_0	Subscriber Port 0	FSUB_0 Register (Offset = 400h) [Reset = 00h]
404h	FSUB_1	Subscriber Port 1	FSUB_1 Register (Offset = 404h) [Reset = 00h]

3.7.1 FSUB_0 Register (Offset = 400h) [Reset = 00h]

FSUB_0 is shown in Figure 3-13 and described in Table 3-18.

Return to the Summary Table.

Subscriber port

Figure 3-13. FSUB_0 Register

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

Table 3-18. FSUB_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channel ID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum can be less than 15.

3.7.2 FSUB_1 Register (Offset = 404h) [Reset = 00h]

FSUB_1 is shown in Figure 3-14 and described in Table 3-19.

Return to the Summary Table.

Subscriber port

Figure 3-14. FSUB_1 Register

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

Table 3-19. FSUB_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channel ID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum can be less than15.

Chapter 4

DMA

The direct memory access (DMA) controller module transfers data from one address to another, without CPU intervention. This chapter describes the operation of the DMA controller.

4.1 DMA Overview

The DMA controller transfers data from a source address to a destination address without CPU intervention. For example, the DMA controller can be used to move data from ADC conversion memory to SRAM.

Devices can have up to sixteen DMA channels available. Therefore, depending on the number of DMA channels available, some features described in this chapter are not applicable to all devices. Please refer to the device-specific data sheet for the actual channel count of the DMA.

Using the DMA controller can increase the throughput of peripheral modules. It can also reduce system power consumption by allowing the CPU to remain in a low-power mode, without having to awaken to move data to or from a peripheral.

DMA controller features include:

- Up to sixteen independent transfer channels
- Configurable DMA channel priorities
- Byte (8-bit), short word (16-bit), word (32-bit) and long word (64-bit) or mixed byte and word transfer capability
- Transfer counter block size supports up to 64k transfers of any data type
- Configurable DMA transfer trigger selection
- Six flexible addressing modes
- Single or block transfer modes

The DMA controller block diagram is shown in Figure 4-1.

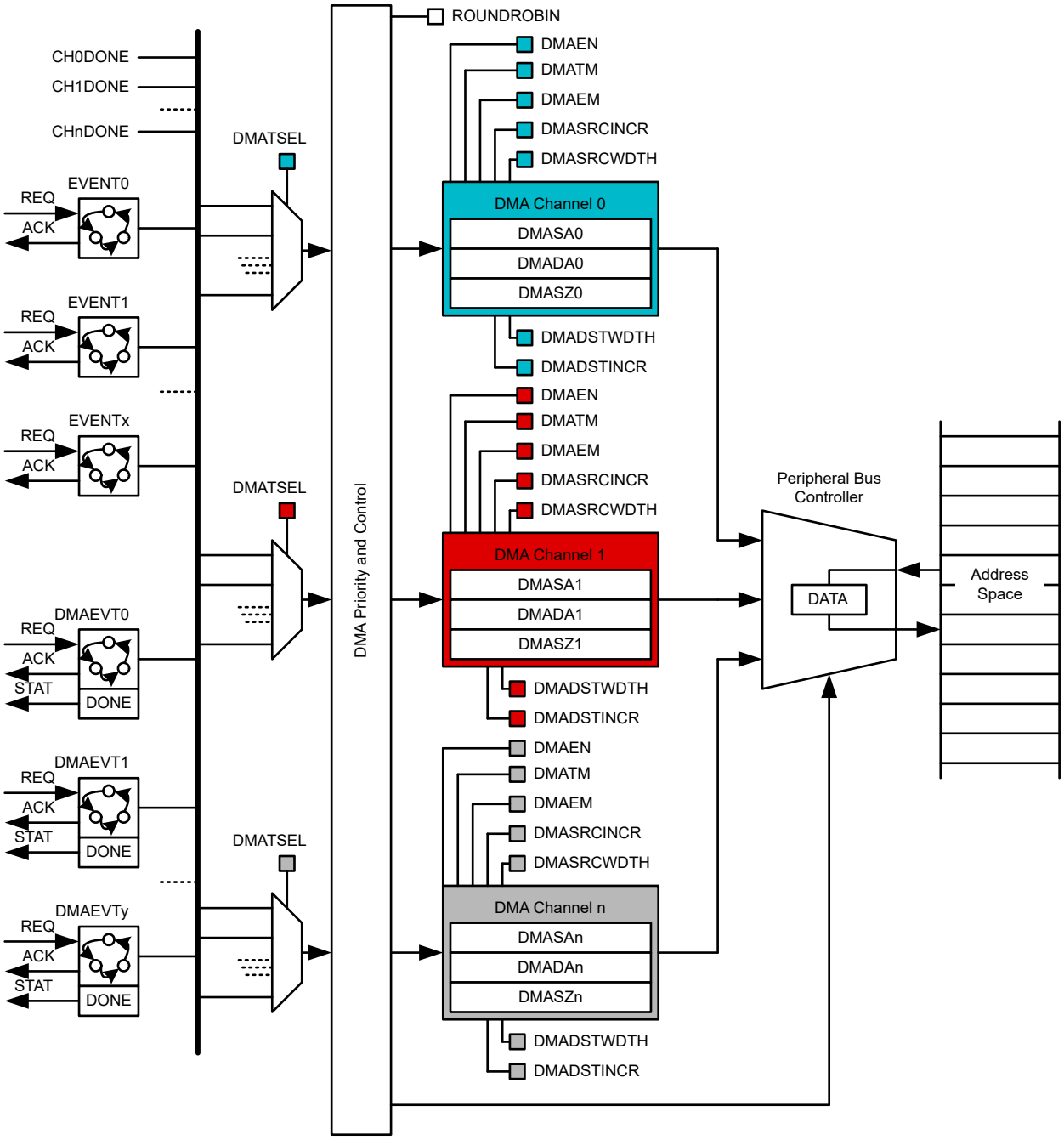


Figure 4-1. DMA Block Diagram

Note

DMA in some MSPM0C devices only support ADC, no supporting for other peripherals. Refer to the device-specific data sheet to see the details.

4.2 DMA Operation

The DMA controller is configured with user software. The setup and operation of the DMA is discussed in the following sections.

4.2.1 Addressing Modes

The DMA controller has six addressing modes. The addressing mode for each DMA channel is independently configurable. For example, channel 0 can transfer between two fixed addresses, while channel 1 transfers between two blocks of addresses. The addressing modes are shown in Figure 4-2.

The addressing modes are:

1. Fixed address to fixed address
2. Fixed address to block of addresses
3. Block of addresses to fixed address
4. Block of addresses to block of addresses
5. Fill data to block of addresses
6. Data table to specific address

Addressing modes 1-4 shown above are simply configured with the DMASRCINCR and DMADSTINCR control bits. The DMASRCINCR bits select if the source address is incremented, decremented, or unchanged after each transfer. The DMADSTINCR bits select if the destination address is incremented, decremented, or unchanged after each transfer.

Addressing modes 5 and 6 shown above are also configured with the DMASRCINCR and DMADSTINCR control bits along with the help of additional parameters such as DMAEM for leveraging the extended modes of the DMA. Refer to Section 4.2.4.1 and Section 4.2.4.2 for more details on how to properly configure and use the DMA in Fill Mode and Table Mode.

Transfers can be byte to byte, short word to short word, word to word, long word to long word, or any combination of the four. When transferring (short or long) word to byte, only the lower byte of the source data transfers. When transferring (long) word to short word, only the lower short word of the source data transfers. When transferring byte to (short or long) word, the upper bytes of the destination word is cleared when the transfer occurs. When transferring short word to (long) word, the upper short word is cleared when the transfer occurs. When transferring word to long word, the upper word is cleared. There is no packing or unpacking support by combining several source byte transfers to one single destination (short) word or the reverse.

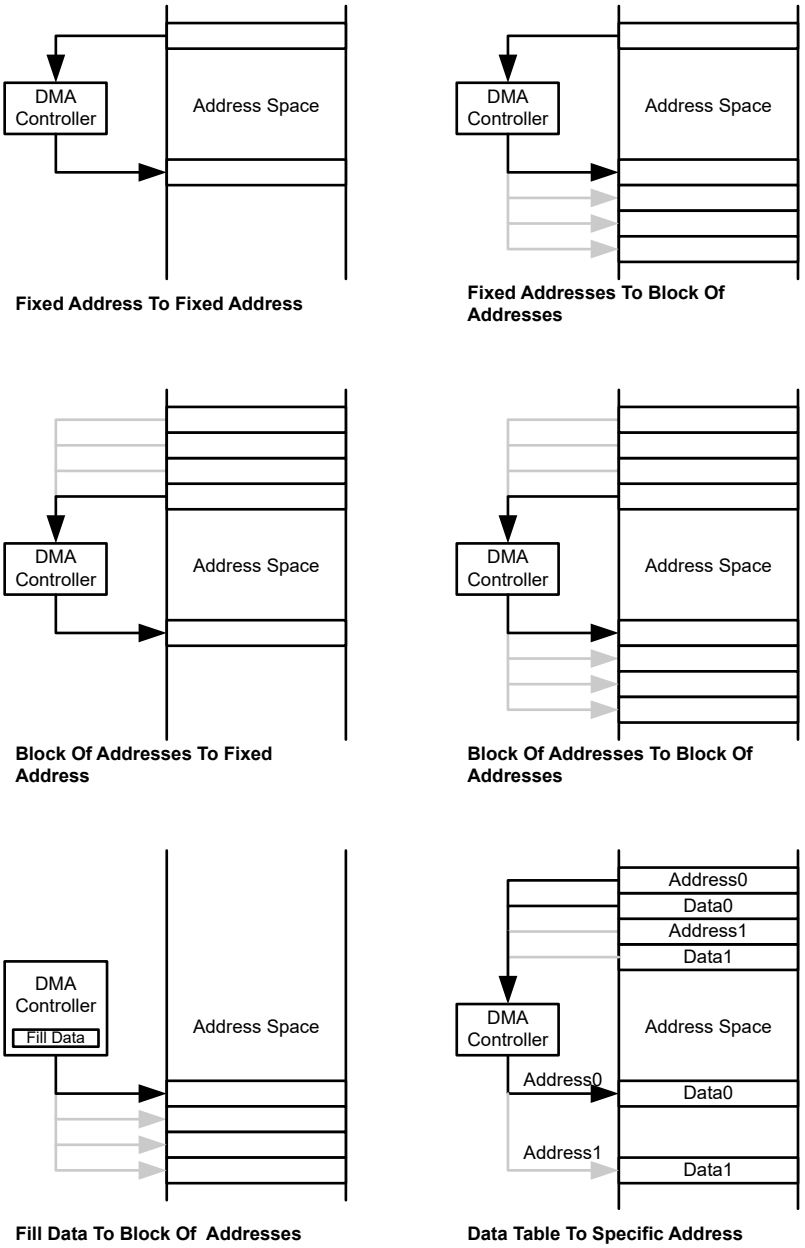


Figure 4-2. DMA Addressing Modes

4.2.2 Channel Types

There are two types of DMA channels: basic (BASIC) and full-featured (FULL) channels. BASIC channels support only single or block transfers, and FULL channels support repeated single and repeated block transfers with additional features such as early interrupt request generation and extended table and fill modes.

The highest priority DMA channels (starting with DMA0) are FULL channels, and the remaining priority channels are BASIC channels.

Note

See the device-specific data sheet to determine how many BASIC and FULL channels are available.

Table 4-1 shows the features supported in basic and full-feature DMA channels.

Table 4-1. Feature Comparison of Basic and Full-Feature DMA Channels

DMA Feature	Full-Feature Channel	Basic Channel
Repeated mode	✓	–
Early IRQ notification	✓	–
Block burst mode	✓	✓
Stride mode	✓	✓
Internal channel as trigger source	✓	✓
Extended Mode (Table and Fill Mode)	✓	–

4.2.3 Transfer Modes

The DMA controller has four transfer modes selected by the DMATM bits as listed in Table 4-2. Each channel is individually configurable for its transfer mode. For example, channel 0 can be configured in repeated block transfer mode, while channel 1 is configured for block transfer mode, and channel 2 operates in single transfer mode. The transfer mode is configured independently from the addressing mode. Any addressing mode can be used with any transfer mode.

Four types of data can be transferred selectable by the DMADSTWDTH and DMASRCWDTH control bits. The source and destination locations can be either byte, short word, word, or long word data. It is also possible to transfer byte to byte, short word to short word, word to word, long word to long word, or any combination.

Additionally, all transfers modes support a stride mode where the DMA source and destination can be incremented to a higher value to support re-organization of data.

Table 4-2. DMA Transfer Modes

DMATM	Transfer Mode	Description	Channel Type
0h	Single transfer	Each transfer requires a trigger. DMAEN is automatically cleared when DMASZx transfers have been made.	Basic
1h	Block transfer	A complete block is transferred with one trigger. DMAEN is automatically cleared at the end of the block transfer.	Basic
2h	Repeated single transfer	Each transfer requires a trigger. DMAEN remains enabled.	Full-feature
3h	Repeated block transfer	A complete block is transferred with one trigger. DMAEN remains enabled.	Full-feature

4.2.3.1 Single Transfer

In single transfer mode (DMATM = 0h), each byte, half-word, word, or long-word transfer requires a separate trigger. Single transfer mode is available in basic and full-feature DMA channels.

The DMASZx register defines the number of transfers to be made. The DMADSTINCR and DMASRCINCR bits select if the destination address and the source address are incremented or decremented after each transfer. If DMASZx = 0, no transfers occur.

The DMASAx, DMADAx, and DMASZx registers are incremented or decremented after each transfer. The DMADSTWDTH will indicate whether the destination address will increment or decrement by 1, 2, 4, 8, or 16 with each transfer cycle. The same is true for the DMASRCWDTH and the source address respectively. When the DMASZx register decrements to zero, the corresponding RIS flag is set.

The DMAEN bit is cleared automatically when DMASZx decrements to zero and must be set again for another transfer to occur.

4.2.3.2 Block Transfer

In block transfer mode (DMATM = 1h), a transfer of a complete block of data occurs after one trigger. Block transfer mode is available in basic DMA channels only.

The DMASZx register defines the size of the block, and the DMADSTINCR and DMASRCINCR bits select if the destination address and the source address are incremented or decremented after each transfer of the block. If DMASZx = 0, no transfers occur.

The DMASAx, DMADAx, and DMASZx registers are copied into temporary registers. The temporary values of DMASAx and DMADAx are incremented or decremented after each transfer in the block. The DMADSTWDTH will indicate whether the destination address will increment or decrement by 1, 2, 4, 8 or 16 with each transfer cycle. The same is true for the DMASRCWDTH and the source address respectively. The DMASZx register is decremented after each transfer of the block and shows the number of blocks remained. When DMATM = 01, the DMAEN bit is cleared automatically when DMASZx decrements to zero and must be set again for another transfer to occur.

The DMAEN bit is cleared after the completion of the block transfer and must be set again before another block transfer can be triggered. After a block transfer has started, another trigger signal that occurs during the block transfer is ignored.

4.2.3.3 Repeated Single Transfer

In repeated single transfer mode (DMATM = 2h), the DMA controller remains enabled with DMAEN = 1, and a transfer occurs every time a trigger occurs. Repeated single transfer modes are available in full-featured DMA channels only.

The DMASAx, DMADAx, and DMASZx registers are copied into temporary registers. The temporary values of DMASAx and DMADAx are incremented or decremented after each transfer. The DMASZx register is decremented after each register. The DMADSTWDTH will indicate whether the destination address will increment or decrement by 1, 2, 4, or 8 with each transfer cycle. The same is true for the DMASRCWDTH and the source address respectively. When the DMASZx register decrements to zero, it is reloaded from its temporary register and the corresponding RIS flag is set.

Note

When using repeated single transfer mode, the DMA does not support pausing and continuing a transfer by disabling a channel (to pause) and then re-enabling the channel (to continue).

4.2.3.4 Repeated Block Transfer

In repeated block transfer mode (DMATM = 11), the DMAEN bit remains set after completion of the block transfer. The next trigger after the completion of a repeated block transfer starts another block transfer. Repeated block transfer modes are available in full-featured DMA channels only.

The DMASAx, DMADAx, and DMASZx registers are copied into temporary registers. The temporary values of DMASAx and DMADAx are incremented or decremented after each transfer in the block. The DMADSTWDTH will indicate whether the destination address will increment or decrement by 1, 2, 4 or 8 with each transfer cycle. The same is true for the DMASRCWDTH and the source address respectively. The DMASZx register is decremented after each transfer of the block and shows the number of transfers remaining in the block.

4.2.3.5 Stride Mode

All transfer modes support a “stride” mode where the DMA source and destination can be incremented to a higher value (rather than +1) after a transfer. This is helpful for re-organizing the order of data between the source and destination.

To support incremental strides, set the DMADSTINCR and/or DMASRCINCR to STRIDE_n, where n is the number of destination and/or source increments. The real increments are based in terms of the definitions DMADSTWDTH and/or DMASRCWDTH, respectively. For example, if external ADC data is transmitted to the MCU as a six-word SPI frame, DMADSTINCR can be set to STRIDE_6 during a block transfer so that the destination address is incremented by 6 and the data is organized to make processing easier.

4.2.4 Extended Modes

In FULL channels, the DMA controller has two extended modes selected by the DMAEM bits as listed in Table 4-3. Each channel is individually configurable for the extended mode. For example, channel 0 can be configured in table mode while channel 1 is configured in fill mode.

Table 4-3. DMA Extended Modes

DMAEM	Extended Mode	Description
00	Normal mode	Operation is defined by DMATM
01	Gather mode	Used to read data from an address table and copy to configurable address
10	Fill mode	Used to fill predefined data patterns into memory
11	Table mode	Used to help configure a table of peripheral control registers

4.2.4.1 Fill Mode

In fill mode (DMAEM = 10b), the DMA controller takes a predefined FILL pattern and writes the pattern to a user defined segment of memory. The DMATM bits are ignored and the automatic transfer mode used is "block transfer".

The DMASAx register is used as the FILL pattern data. The DMASRCINCR bit field is used to indicate whether the FILL pattern data should be constant or incremented/decremented with every write cycle. This feature allows for filling a memory block with a sequential pattern (for example. 0, 1, 2, 3, …). The DMASRCWDTH bit field indicates the magnitude of increment of the FILL mode data. Refer to Table 4-4 for how to use DMASRCWDTH in fill mode.

Table 4-4. DMASRCWDTH in Fill Mode

DMASRCWDTH	FILL Mode Data Increment Value
0	±1
1	±2
2	±4
3	±8

The destination registers and bit fields DMADAx, DMADSTINCR, and DMADSTWDTH all behave as expected and influence where and how in memory the FILL pattern is written.

4.2.4.2 Table Mode

In table mode (DMAEM = 11b), the DMA controller executes 2 reads from the source and one write to a determined destination. This feature can be leveraged to interpret a table of addresses and data and uses the DMA to efficiently program that data to their associated addresses without CPU intervention. Table mode allows you to parse through a table of addresses and data to configure peripheral memory mapped registers in a single block transfer.

The DMASRCWDTH bit field should be set to "3" (64-bit mode) and the DMADSTWDTH bit field should be set to "2" (32-bit mode). The DMASRCINCR bit field can be set to 10b to decrement the source address or 11b to increment the source address after transfers. DMASZ is set to represent the number of entries in the table and DMATM should be set to "01" for block transfer mode. DMADAx and DMADSTINCR are ignored in table mode and can be treated as "don’t care" values.

The DMASAx register needs to be programmed with the start address of the table, which needs to be aligned to 64-bit data (that is, DMASAx[2:0] = "000"). The address stored in the table needs to be on the lower word of a 64-bit data (ADDR[2:0] = "000" while the data needs to be on the upper word of a 64-bit data (ADDR[2:0] = "100"). Table 4-5 is an example of a table in memory compatible with the DMA table mode.

Table 4-5. Example of an Incremental Table Compatible with DMA Table Mode

Table Address	Table Data
0x0000	Address 0
0x0004	Data 0
0x0008	Address 1
0x000C	Data 1

4.2.5 Initiating DMA Transfers

Each DMA channel is independently configured for its trigger source with DMATSEL. The DMATSEL bits should be modified only when the DMACTLx.DMAEN bit is 0; otherwise, unpredictable DMA triggers can occur.

See the device-specific data sheet for the list of triggers available, along with their respective DMATSEL values.

When selecting the trigger, the trigger must not have already occurred, or the transfer does not take place.

DMA channels can be internally triggered upon the completion of activity on another channel to support cascading. Completion of activity occurs when a DMA channel's DMASZ counter reaches zero. This is beneficial for applications where data can be retrieved, transferred, and/or error-checked without an interrupt or event configuration.

Set the DMATINT bit to internally trigger the next DMA channel based on the DMATSEL trigger source. Once the DMATSEL trigger occurs, the next DMA channel begins to automatically execute.

For example, if UART data is received and transmitted to SRAM through DMA channel 0 and DMATSEL is set to UART RX, then DMA channel 1 can be internally triggered when the UART is finished receiving data. If DMA channel 1 is configured to transmit the data from SRAM to CRC, then the DMA transfer will trigger once the UART data is received. In this case, the DMA channels are cascaded from Channel 0 to Channel 1.

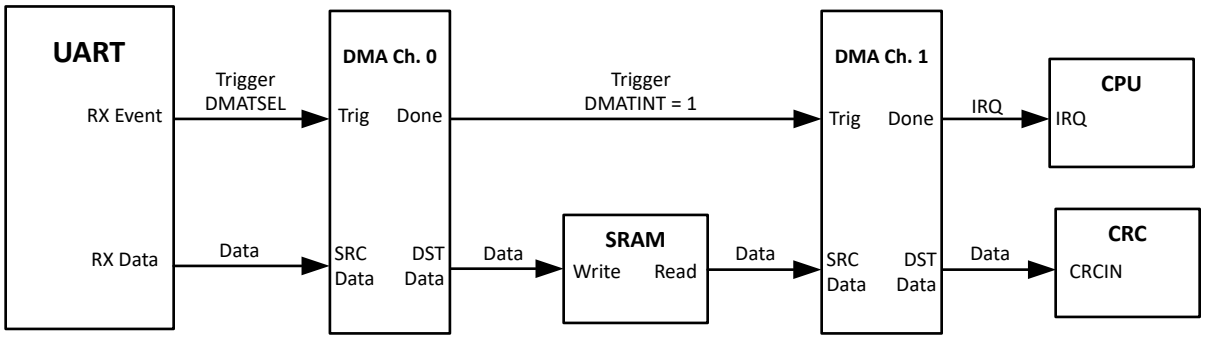


Figure 4-3. DMA Cascading Channels

4.2.6 Stopping DMA Transfers

A DMA block transfer in progress can be stopped by clearing the DMAEN bit. The DMA will stop after the completion of the ongoing transfer cycle and all the channel registers will stay in the current state. The block transfer can be continued as originally configured after setting the DMAEN bit again and resending the trigger. Please note that a trigger is necessary for halted transfer to resume.

Note

A single transfer in progress cannot be interrupted.

4.2.7 Channel Priorities

The default DMA channel priorities are DMA0 through DMA15. If two or three triggers happen simultaneously or are pending, the channel with the highest priority completes its transfer (single or block transfer) first, then the second priority channel, then the third priority channel. Transfers in progress are not halted if a higher-priority channel is triggered. The higher-priority channel waits until the transfer in progress completes before starting.

The DMA channel priorities are configurable with the ROUNDROBIN bit. When the ROUNDROBIN bit is set, the channel that completes a transfer becomes the lowest priority. The order of the priority of the channels always stays the same, DMA0-DMA1-DMA2, for example, for three channels. When the ROUNDROBIN bit is cleared, the channel priority returns to the default priority.

Table 4-6. Round-Robin DMA Priority Example

Current DMA Priority	Transfer Occurs	New DMA Priority
DMA0 - DMA1 - DMA2	DMA1	DMA2 - DMA0 - DMA1
DMA2 - DMA0 - DMA1	DMA2	DMA0 - DMA1 - DMA2
DMA2 - DMA0 - DMA1	DMA0	DMA1 - DMA2 - DMA0

4.2.8 Burst Block Mode

The DMA module supports a burst block mode for suspending an active channel after a configurable number of transfers to service other pending channels. The burst block size is configurable by setting DMAPRIO.BURSTSZ to 8, 16, 32, or an infinite number of transfers. If a higher priority channel is pending after the burst block, the DMA will execute the higher priority channel and resume on the suspended channel once the higher priority channel is complete. If no other channel is pending, the priority logic assigns the control back to the block transfer for the next burst.

4.2.9 Using DMA with System Interrupts

System interrupt service routines are interrupted by DMA transfers. If an interrupt service routine or other routine must execute with no interruptions, the DMA controller should be disabled before executing the routine.

4.2.10 DMA Controller Interrupts

Each DMA channel has its own RIS flag. Each RIS flag is set in any mode when the corresponding DMASZx register counts to zero. If the corresponding MASK and RIS bits are set, an interrupt request is generated.

All RIS flags are prioritized, with DMA0 being the highest, and combined to source a single interrupt vector. The highest-priority enabled interrupt generates a number in the IIDX register. This number can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine.

Any access, read or write, of the IIDX register automatically resets the highest pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. For example, assume that DMA0 has the highest priority. If the DMA0-RIS and DMA2-RIS flags are set when the interrupt service routine accesses the IIDX register, DMA0-RIS is reset automatically. After the interrupt service routine is executed, the DMA2-RIS generates another interrupt.

4.2.11 DMA Trigger Event Status

The DMA controller supports dedicated DMA events. See Section 6.2.2 for details on the DMA event trigger protocol. The idea is that the DMA can inform the event triggering peripheral about the status of the assigned DMA channel. This will allow the triggering peripheral to issue an interrupt itself after the completion of a repeated transfer, instead of the DMA issuing an interrupt event. The advantage is, that the DMA interrupt service routine does not need to keep track of the assigned function of the channel. As a result, the DMA triggering peripheral interrupt service routine will deal with the completion of the DMA transfer.

The status will reflect the value of the DMASZx register. If the last DMA transfer resulted in a size decrement to zero, the DMA will return the status of 1, indicating the end of the transfer. Otherwise the status will be 0.

Additionally, the DMA module can generate an early interrupt request to the CPU to indicate that a transfer will complete within a configurable number of transfers (1, 2, 4, 8, 32, 64, half-DMASZ).

An early IRQ event is enabled by setting DMAPREIRQ to the desired number of transfers. When the DMA has reached the number of transfers, the corresponding DMA channel's PREIRQ interrupt is set.

Early DMA interrupt generation is useful to:

- Reduce the interrupt latency in timing-critical applications where it would be beneficial to let the DMA preemptively generate the IRQ before the DMA transfer is complete
- Serve as a “progress notification” when scheduling other tasks for the CPU to complete
- Transfer weights of a neural network layer and notify the CPU to complete software configuration writes to the IP
- Implement a ping-pong buffer (by setting DMAPREIRQ to half)

Note

This feature is available on repeat-capable channels only.

4.2.12 DMA Operating Mode Support

The DMA supports triggered transfers in RUN mode, as well as in the SLEEP, STOP, and STANDBY low-power modes. Refer to the following sections for more details.

4.2.12.1 Transfer in RUN Mode

In RUN mode the system is fully operational. The CPU and all other peripherals and resources are available, therefore there is no restriction on the DMA functionality in RUN mode.

4.2.12.2 Transfer in SLEEP Mode

In SLEEP mode only the CPU is halted. All other peripherals and resources are available as when in RUN mode, therefore there is no restriction on the DMA functionality in SLEEP mode. All peripherals that can trigger a DMA transfer in RUN mode will also be able to trigger a DMA transfer in SLEEP mode.

4.2.12.3 Transfer in STOP Mode

In STOP mode the CPU is halted and the ULPCLK is limited to 4MHz operation. The event manager will detect a DMA trigger event and request the PMU to enter a "suspended STOP" state. For more info on this state refer to Section 2.1.2.7.While STOP mode is suspended, the DMA is fully functional and will work on the pending DMA trigger request. Once the DMA transfer is complete, the DMA will acknowledge the pending trigger event and the event subsystem removes the power mode request from the PMU. If the PMU has no other pending requests, the SoC will transition back into normal STOP mode.

4.2.12.4 Transfers in STANDBY Mode

In STANDBY mode the CPU is halted and the ULPCLK is limited to 32kHz operation. The event manager will detect a DMA trigger event and request the PMU to enter a "suspended STANDBY" state. For more info on this state refer to Section 2.1.2.7.While STANDBY mode is suspended, the DMA is fully functional and will work on the pending DMA trigger request. Once the DMA transfer is complete, the DMA will acknowledge the pending trigger event and the event subsystem removes the power mode request from the PMU. If the PMU has no other pending requests, the SoC will transition back into normal STANDBY mode.

4.2.13 DMA Address and Data Errors

The DMA itself has the ability to flag address or data errors. Source or destination address errors can come from accessing a protected or nonexisting memory range. If an address error occurs, the interrupt index IIDX[j].STAT flags a DMA address error (11h). Address error interrupts can be masked, set, and cleared using the ADDERR bit.

Note

The DMA itself does not perform range checking. If the DMA transfer occurs over a protected memory range, the destination data will report zeros (0h) for each byte of the DMA transaction that overlaps the protected or nonexisting memory range.

Data errors can occur in SRAM or flash if it has an ECC or parity error. If a data error occurs, the interrupt index IIDX[j].STAT flags a DMA data error (12h). Data error interrupts can be masked, set, and cleared using the DATERR bit.

4.2.14 Interrupt and Event Support

The DMA module contains one event publishers and two event subscribers.

- One event publisher (CPU_INT) manages DMA interrupt requests (IRQs) to the CPU subsystem through a static event route.
- The second and third event (GEN_EVENT) are used to setup the generic event publishers and subscribers through Generic route.

DMA events are summarized in Table 4-7.

Table 4-7. DMA Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt	Publisher	DMA	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from TIMx to CPU
Generic publisher event	Publisher	DMA	Other peripherals	Generic route	GEN_EVENT and FPUB_1 registers	Configurable interrupt route from DMA to other peripherals
Generic subscriber event	Subscriber	Other peripherals	DMA	Generic route	FSUB_0	Configurable interrupt route from other peripherals to DMA
Generic subscriber event	Subscriber	Other peripherals	DMA	Generic route	FSUB_1	Configurable interrupt route from other peripherals to DMA

4.3 DMA Registers

Table 4-8 lists the memory-mapped registers for the DMA registers. All register offset addresses not listed in Table 4-8 should be considered as reserved locations and the register contents should not be modified.

Table 4-8. DMA Registers

Offset	Acronym	Register Name	Group	Section
400h	FSUB_0	Subscriber Port 0		Go
404h	FSUB_1	Subscriber Port 1		Go
444h	FPUB_1	Publisher Port 0		Go
1018h	PDBGCTL	Peripheral Debug Control		Go
1020h	IIDX	Interrupt index	CPU_INT	Go
1028h	IMASK	Interrupt mask	CPU_INT	Go
1030h	RIS	Raw interrupt status	CPU_INT	Go
1038h	MIS	Masked interrupt status	CPU_INT	Go
1040h	ISET	Interrupt set	CPU_INT	Go
1048h	ICLR	Interrupt clear	CPU_INT	Go
1050h	IIDX	Interrupt index	GEN_EVENT	Go
1058h	IMASK	Interrupt mask	GEN_EVENT	Go
1060h	RIS	Raw interrupt status	GEN_EVENT	Go
1068h	MIS	Masked interrupt status	GEN_EVENT	Go
1070h	ISET	Interrupt set	GEN_EVENT	Go
1078h	ICLR	Interrupt clear	GEN_EVENT	Go
10E0h	EVT_MODE	Event Mode		Go
10FCh	DESC	Module Description		Go
1100h	DMAPRIO	DMA Channel Priority Control		Go
1110h + formula	DMATCTL[j]	DMA Trigger Select		Go
1200h + formula	DMACTL[j]	DMA Channel Control		Go
1204h + formula	DMASA[j]	DMA Channel Source Address		Go
1208h + formula	DMADA[j]	DMA Channel Destination Address		Go
120Ch + formula	DMASZ[j]	DMA Channel Size		Go

Complex bit access types are encoded to fit into small table cells. Table 4-9 shows the codes that are used for access types in this section.

Table 4-9. DMA Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 4-9. DMA Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

4.3.1 FSUB_0 (Offset = 400h) [Reset = 00000000h]

FSUB_0 is shown in Figure 4-4 and described in Table 4-10.

Return to the Summary Table.

Subscriber port

Figure 4-4. FSUB_0

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
RESERVED																								CHANID							
R/W-0h																								R/W-0h							

Table 4-10. FSUB_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	
7-0	CHANID	R/W	0h	0 = disconnected. 1-255 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected FFh = Consult your device data sheet as the actual allowed maximum may be less than 255.

4.3.2 FSUB_1 (Offset = 404h) [Reset = 00000000h]

FSUB_1 is shown in Figure 4-5 and described in Table 4-11.

Return to the Summary Table.

Subscriber port

Figure 4-5. FSUB_1

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
RESERVED																								CHANID							
R/W-0h																								R/W-0h							

Table 4-11. FSUB_1 Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	
7-0	CHANID	R/W	0h	0 = disconnected. 1-255 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected FFh = Consult your device data sheet as the actual allowed maximum may be less than 255.

4.3.3 FPUB_1 (Offset = 444h) [Reset = 00000000h]

FPUB_1 is shown in Figure 4-6 and described in Table 4-12.

Return to the Summary Table.

Publisher port

Figure 4-6. FPUB_1

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
RESERVED																								CHANID							
R/W-0h																								R/W-0h							

Table 4-12. FPUB_1 Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	
7-0	CHANID	R/W	0h	0 = disconnected. 1-255 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected FFh = Consult your device data sheet as the actual allowed maximum may be less than 255.

4.3.4 PDBGCTL (Offset = 1018h) [Reset = 00000003h]

PDBGCTL is shown in Figure 4-7 and described in Table 4-13.

Return to the Summary Table.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 4-7. PDBGCTL

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R/W-						R/W-1h	R/W-1h

Table 4-13. PDBGCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1	SOFT	R/W	1h	Soft halt boundary control. This function is only available, if FREE is set to 'STOP' 0h = The peripheral will halt immediately, even if the resultant state will result in corruption if the system is restarted 1h = The peripheral blocks the debug freeze until it has reached a boundary where it can resume without corruption
0	FREE	R/W	1h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

4.3.5 IIDX (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 4-8 and described in Table 4-14.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, . . . IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred.
On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in RIS [RIS] and MIS [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-8. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 4-14. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No bit is set means there is no pending interrupt request 01h = DMA Channel 0 size counter reached zero (DMASZ=0). 02h = DMA Channel 1 size counter reached zero (DMASZ=0). 03h = DMA Channel 2 size counter reached zero (DMASZ=0). 04h = DMA Channel 3 size counter reached zero (DMASZ=0). 05h = DMA Channel 4 size counter reached zero (DMASZ=0). 06h = DMA Channel 5 size counter reached zero (DMASZ=0). 07h = DMA Channel 6 size counter reached zero (DMASZ=0). 08h = DMA Channel 7 size counter reached zero (DMASZ=0). 09h = DMA Channel 8 size counter reached zero (DMASZ=0). 0Ah = DMA Channel 9 size counter reached zero (DMASZ=0). 0Bh = DMA Channel 10 size counter reached zero (DMASZ=0). 0Ch = DMA Channel 11 size counter reached zero (DMASZ=0). 0Dh = DMA Channel 12 size counter reached zero (DMASZ=0). 0Eh = DMA Channel 13 size counter reached zero (DMASZ=0). 0Fh = DMA Channel 14 size counter reached zero (DMASZ=0). 10h = DMA Channel 15 size counter reached zero (DMASZ=0). 11h = PRE-IRQ event for DMA Channel 0. 12h = PRE-IRQ event for DMA Channel 1. 13h = PRE-IRQ event for DMA Channel 2. 14h = PRE-IRQ event for DMA Channel 3. 15h = PRE-IRQ event for DMA Channel 4. 16h = PRE-IRQ event for DMA Channel 5. 17h = PRE-IRQ event for DMA Channel 6. 18h = PRE-IRQ event for DMA Channel 7. 19h = DMA address error, SRC address not reachable. 1Ah = DMA data error, SRC data might be corrupted (PAR or ECC error).

4.3.6 IMASK (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 4-9 and described in Table 4-15.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then the corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX [IIDX], as well as MIS [MIS].
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-9. IMASK

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
R/W-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 4-15. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	
25	DATAERR	R/W	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	R/W	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	R/W	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	R/W	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	R/W	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	R/W	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	R/W	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	R/W	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
17	PREIRQCH1	R/W	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-15. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	PREIRQCH0	R/W	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	R/W	0h	DMA Channel 15 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
14	DMACH14	R/W	0h	DMA Channel 14 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
13	DMACH13	R/W	0h	DMA Channel 13 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
12	DMACH12	R/W	0h	DMA Channel 12 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
11	DMACH11	R/W	0h	DMA Channel 11 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
10	DMACH10	R/W	0h	DMA Channel 10 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
9	DMACH9	R/W	0h	DMA Channel 9 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
8	DMACH8	R/W	0h	DMA Channel 8 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
7	DMACH7	R/W	0h	DMA Channel 7 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
6	DMACH6	R/W	0h	DMA Channel 6 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
5	DMACH5	R/W	0h	DMA Channel 5 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
4	DMACH4	R/W	0h	DMA Channel 4 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
3	DMACH3	R/W	0h	DMA Channel 3 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
2	DMACH2	R/W	0h	DMA Channel 2 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-15. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DMACH1	R/W	0h	DMA Channel 1 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
0	DMACH0	R/W	0h	DMA Channel 0 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

4.3.7 RIS (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 4-10 and described in Table 4-16.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR [ICLR] register bit even if the corresponding IMASK [IMASK] bit is not enabled.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-10. RIS

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 4-16. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	
25	DATAERR	R	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	R	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	R	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	R	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	R	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	R	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	R	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	R	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-16. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	PREIRQCH1	R	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
16	PREIRQCH0	R	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	R	0h	DMA Channel 15 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
14	DMACH14	R	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
13	DMACH13	R	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
12	DMACH12	R	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
11	DMACH11	R	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
10	DMACH10	R	0h	DMA Channel 10 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
9	DMACH9	R	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
8	DMACH8	R	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
7	DMACH7	R	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
6	DMACH6	R	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
5	DMACH5	R	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
4	DMACH4	R	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
3	DMACH3	R	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred

Table 4-16. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DMACH2	R	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
1	DMACH1	R	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
0	DMACH0	R	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred

4.3.8 MIS (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 4-11 and described in Table 4-17.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK [IMASK] and RIS [RIS] registers.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-11. MIS

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 4-17. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	
25	DATAERR	R	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	R	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	R	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	R	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	R	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	R	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	R	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	R	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
17	PREIRQCH1	R	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-17. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	PREIRQCH0	R	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	R	0h	DMA Channel 15 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
14	DMACH14	R	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
13	DMACH13	R	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
12	DMACH12	R	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
11	DMACH11	R	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
10	DMACH10	R	0h	DMA Channel 10 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
9	DMACH9	R	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
8	DMACH8	R	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
7	DMACH7	R	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
6	DMACH6	R	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
5	DMACH5	R	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
4	DMACH4	R	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
3	DMACH3	R	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
2	DMACH2	R	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred

Table 4-17. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DMACH1	R	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
0	DMACH0	R	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred

4.3.9 ISET (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 4-12 and described in Table 4-18.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS [RIS] bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS [MIS] bit is also set.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-12. ISET

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
W-0h						W-0h	W-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 4-18. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	W	0h	
25	DATAERR	W	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	W	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	W	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	W	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	W	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	W	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	W	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	W	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-18. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	PREIRQCH1	W	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
16	PREIRQCH0	W	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
14	DMACH14	W	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
13	DMACH13	W	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
12	DMACH12	W	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
11	DMACH11	W	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
10	DMACH10	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
9	DMACH9	W	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
8	DMACH8	W	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
7	DMACH7	W	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
6	DMACH6	W	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
5	DMACH5	W	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
4	DMACH4	W	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
3	DMACH3	W	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt

Table 4-18. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DMACH2	W	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
1	DMACH1	W	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
0	DMACH0	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt

4.3.10 ICLR (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 4-13 and described in Table 4-19.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-13. ICLR

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
W-0h						W-0h	W-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 4-19. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	W	0h	
25	DATAERR	W	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	W	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	W	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	W	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	W	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	W	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	W	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	W	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
17	PREIRQCH1	W	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-19. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	PREIRQCH0	W	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	W	0h	DMA Channel 15 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
14	DMACH14	W	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
13	DMACH13	W	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
12	DMACH12	W	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
11	DMACH11	W	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
10	DMACH10	W	0h	DMA Channel 10 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
9	DMACH9	W	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
8	DMACH8	W	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
7	DMACH7	W	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
6	DMACH6	W	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
5	DMACH5	W	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
4	DMACH4	W	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
3	DMACH3	W	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
2	DMACH2	W	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt

Table 4-19. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DMACH1	W	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
0	DMACH0	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt

4.3.11 IIDX (Offset = 1050h) [Reset = 00000000h]

IIDX is shown in Figure 4-14 and described in Table 4-20.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, . . . IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred.
On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in RIS [RIS] and MIS [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-14. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 4-20. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No bit is set means there is no pending interrupt request 01h = DMA Channel 0 size counter reached zero (DMASZ=0). 02h = DMA Channel 1 size counter reached zero (DMASZ=0). 03h = DMA Channel 2 size counter reached zero (DMASZ=0). 04h = DMA Channel 3 size counter reached zero (DMASZ=0). 05h = DMA Channel 4 size counter reached zero (DMASZ=0). 06h = DMA Channel 5 size counter reached zero (DMASZ=0). 07h = DMA Channel 6 size counter reached zero (DMASZ=0). 08h = DMA Channel 7 size counter reached zero (DMASZ=0). 09h = DMA Channel 8 size counter reached zero (DMASZ=0). 0Ah = DMA Channel 9 size counter reached zero (DMASZ=0). 0Bh = DMA Channel 10 size counter reached zero (DMASZ=0). 0Ch = DMA Channel 11 size counter reached zero (DMASZ=0). 0Dh = DMA Channel 12 size counter reached zero (DMASZ=0). 0Eh = DMA Channel 13 size counter reached zero (DMASZ=0). 0Fh = DMA Channel 14 size counter reached zero (DMASZ=0). 10h = DMA Channel 15 size counter reached zero (DMASZ=0). 11h = PRE-IRQ event for DMA Channel 0. 12h = PRE-IRQ event for DMA Channel 1. 13h = PRE-IRQ event for DMA Channel 2. 14h = PRE-IRQ event for DMA Channel 3. 15h = PRE-IRQ event for DMA Channel 4. 16h = PRE-IRQ event for DMA Channel 5. 17h = PRE-IRQ event for DMA Channel 6. 18h = PRE-IRQ event for DMA Channel 7. 19h = DMA address error, SRC address not reachable. 1Ah = DMA data error, SRC data might be corrupted (PAR or ECC error).

4.3.12 IMASK (Offset = 1058h) [Reset = 00000000h]

IMASK is shown in Figure 4-15 and described in Table 4-21.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then the corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX [IIDX], as well as MIS [MIS].
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-15. IMASK

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
R/W-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 4-21. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	
25	DATAERR	R/W	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	R/W	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	R/W	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	R/W	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	R/W	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	R/W	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	R/W	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	R/W	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
17	PREIRQCH1	R/W	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-21. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	PREIRQCH0	R/W	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	R/W	0h	DMA Channel 15 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
14	DMACH14	R/W	0h	DMA Channel 14 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
13	DMACH13	R/W	0h	DMA Channel 13 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
12	DMACH12	R/W	0h	DMA Channel 12 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
11	DMACH11	R/W	0h	DMA Channel 11 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
10	DMACH10	R/W	0h	DMA Channel 10 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
9	DMACH9	R/W	0h	DMA Channel 9 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
8	DMACH8	R/W	0h	DMA Channel 8 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
7	DMACH7	R/W	0h	DMA Channel 7 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
6	DMACH6	R/W	0h	DMA Channel 6 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
5	DMACH5	R/W	0h	DMA Channel 5 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
4	DMACH4	R/W	0h	DMA Channel 4 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
3	DMACH3	R/W	0h	DMA Channel 3 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
2	DMACH2	R/W	0h	DMA Channel 2 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-21. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DMACH1	R/W	0h	DMA Channel 1 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
0	DMACH0	R/W	0h	DMA Channel 0 interrupt signal. Size counter reached zero (DMASZ=0). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

4.3.13 RIS (Offset = 1060h) [Reset = 00000000h]

RIS is shown in Figure 4-16 and described in Table 4-22.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR [ICLR] register bit even if the corresponding IMASK [IMASK] bit is not enabled.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-16. RIS

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 4-22. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	
25	DATAERR	R	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	R	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	R	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	R	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	R	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	R	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	R	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	R	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-22. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	PREIRQCH1	R	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
16	PREIRQCH0	R	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	R	0h	DMA Channel 15 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
14	DMACH14	R	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
13	DMACH13	R	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
12	DMACH12	R	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
11	DMACH11	R	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
10	DMACH10	R	0h	DMA Channel 10 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
9	DMACH9	R	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
8	DMACH8	R	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
7	DMACH7	R	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
6	DMACH6	R	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
5	DMACH5	R	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
4	DMACH4	R	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
3	DMACH3	R	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred

Table 4-22. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DMACH2	R	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
1	DMACH1	R	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred
0	DMACH0	R	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur 1h = Interrupt occurred

4.3.14 MIS (Offset = 1068h) [Reset = 00000000h]

MIS is shown in Figure 4-17 and described in Table 4-23.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK [IMASK] and RIS [RIS] registers.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-17. MIS

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 4-23. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	
25	DATAERR	R	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	R	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	R	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	R	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	R	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	R	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	R	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	R	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
17	PREIRQCH1	R	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-23. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	PREIRQCH0	R	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	R	0h	DMA Channel 15 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
14	DMACH14	R	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
13	DMACH13	R	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
12	DMACH12	R	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
11	DMACH11	R	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
10	DMACH10	R	0h	DMA Channel 10 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
9	DMACH9	R	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
8	DMACH8	R	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
7	DMACH7	R	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
6	DMACH6	R	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
5	DMACH5	R	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
4	DMACH4	R	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
3	DMACH3	R	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
2	DMACH2	R	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred

Table 4-23. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DMACH1	R	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred
0	DMACH0	R	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Interrupt did not occur or is masked out 1h = Interrupt occurred

4.3.15 ISET (Offset = 1070h) [Reset = 00000000h]

ISET is shown in Figure 4-18 and described in Table 4-24.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS [RIS] bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS [MIS] bit is also set.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-18. ISET

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
W-0h						W-0h	W-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 4-24. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	W	0h	
25	DATAERR	W	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	W	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	W	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	W	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	W	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	W	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	W	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	W	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-24. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	PREIRQCH1	W	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
16	PREIRQCH0	W	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
14	DMACH14	W	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
13	DMACH13	W	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
12	DMACH12	W	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
11	DMACH11	W	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
10	DMACH10	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
9	DMACH9	W	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
8	DMACH8	W	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
7	DMACH7	W	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
6	DMACH6	W	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
5	DMACH5	W	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
4	DMACH4	W	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
3	DMACH3	W	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt

Table 4-24. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DMACH2	W	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
1	DMACH1	W	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt
0	DMACH0	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Set interrupt

4.3.16 ICLR (Offset = 1078h) [Reset = 00000000h]

ICLR is shown in Figure 4-19 and described in Table 4-25.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.
Note: The number of DMACH is device dependent. Please consult the data sheet of the specific device to map which channel number is implemented.

Figure 4-19. ICLR

31	30	29	28	27	26	25	24
RESERVED						DATAERR	ADDRERR
W-0h						W-0h	W-0h
23	22	21	20	19	18	17	16
PREIRQCH7	PREIRQCH6	PREIRQCH5	PREIRQCH4	PREIRQCH3	PREIRQCH2	PREIRQCH1	PREIRQCH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DMACH15	DMACH14	DMACH13	DMACH12	DMACH11	DMACH10	DMACH9	DMACH8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DMACH7	DMACH6	DMACH5	DMACH4	DMACH3	DMACH2	DMACH1	DMACH0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 4-25. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	W	0h	
25	DATAERR	W	0h	DMA data error, SRC data might be corrupted (PAR or ECC error). 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
24	ADDRERR	W	0h	DMA address error, SRC address not reachable. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
23	PREIRQCH7	W	0h	Pre-IRQ for Channel 7. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
22	PREIRQCH6	W	0h	Pre-IRQ for Channel 6. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
21	PREIRQCH5	W	0h	Pre-IRQ for Channel 5. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
20	PREIRQCH4	W	0h	Pre-IRQ for Channel 4. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
19	PREIRQCH3	W	0h	Pre-IRQ for Channel 3. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
18	PREIRQCH2	W	0h	Pre-IRQ for Channel 2. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
17	PREIRQCH1	W	0h	Pre-IRQ for Channel 1. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit

Table 4-25. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	PREIRQCH0	W	0h	Pre-IRQ for Channel 0. Size counter reached Pre-IRQ threshold. 0h = Clear interrupt mask bit 1h = Set interrupt mask bit
15	DMACH15	W	0h	DMA Channel 15 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
14	DMACH14	W	0h	DMA Channel 14 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
13	DMACH13	W	0h	DMA Channel 13 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
12	DMACH12	W	0h	DMA Channel 12 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
11	DMACH11	W	0h	DMA Channel 11 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
10	DMACH10	W	0h	DMA Channel 10 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
9	DMACH9	W	0h	DMA Channel 9 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
8	DMACH8	W	0h	DMA Channel 8 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
7	DMACH7	W	0h	DMA Channel 7 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
6	DMACH6	W	0h	DMA Channel 6 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
5	DMACH5	W	0h	DMA Channel 5 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
4	DMACH4	W	0h	DMA Channel 4 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
3	DMACH3	W	0h	DMA Channel 3 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
2	DMACH2	W	0h	DMA Channel 2 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt

Table 4-25. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	DMACH1	W	0h	DMA Channel 1 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt
0	DMACH0	W	0h	DMA Channel 0 interrupt signals that size counter reached zero (DMASZ=0). 0h = Writing 0 has no effect 1h = Clear interrupt

4.3.17 EVT_MODE (Offset = 10E0h) [Reset = 00000009h]

EVT_MODE is shown in Figure 4-20 and described in Table 4-26.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 4-20. EVT_MODE

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED				EVT1_CFG		INT0_CFG	
R/W-				R-2h		R-1h	

Table 4-26. EVT_MODE Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-2	EVT1_CFG	R	2h	Event line mode select for event corresponding to generic event GEN_EVENT 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
1-0	INT0_CFG	R	1h	Event line mode select for event corresponding to interrupt event CPU_INT 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

4.3.18 DESC (Offset = 10FCh) [Reset = 2511F000h]

DESC is shown in Figure 4-21 and described in Table 4-27.

Return to the Summary Table.

This register identifies the peripheral and its exact version.

Figure 4-21. DESC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-2511h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				RESERVED				MAJREV				MINREV			
R-Fh				R-				R-0h				R-0h			

Table 4-27. DESC Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	2511h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness. 0h = Smallest value FFFFh = Highest possible value
15-12	FEATUREVER	R	Fh	Feature Set for the DMA: number of DMA channel minus one (for example 0->1ch, 2->3ch, 15->16ch). 0h = Smallest value (1 channel) Fh = Highest value (16 channel)
11-8	RESERVED	R	0h	
7-4	MAJREV	R	0h	Major rev of the IP 0h = Smallest value Fh = Highest possible value
3-0	MINREV	R	0h	Minor rev of the IP 0h = Smallest value Fh = Highest possible value

4.3.19 DMAPRIO (Offset = 1100h) [Reset = 00000000h]

DMAPRIO is shown in Figure 4-22 and described in Table 4-28.

Return to the Summary Table.

DMA Channel Priority Control

Figure 4-22. DMAPRIO

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED						BURSTSZ	
R/W-						R/W-0h	
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED							ROUNDROBIN
R/W-							R/W-0h

Table 4-28. DMAPRIO Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R/W	0h	
17-16	BURSTSZ	R/W	0h	Define the burst size of a block transfer, before the priority is re-evaluated 0h = There is no burst size, the whole block transfer is completed on one transfer without interruption 1h = The burst size is 8, after 8 transfers the block transfer is interrupted and the priority is reevaluated 2h = The burst size is 16, after 16 transfers the block transfer is interrupted and the priority is reevaluated 3h = The burst size is 32, after 32 transfers the block transfer is interrupted and the priority is reevaluated
15-1	RESERVED	R/W	0h	
0	ROUNDROBIN	R/W	0h	Round robin. This bit enables the round-robin DMA channel priorities. 0h = Round robin priority disabled, DMA channel priority is fixed: DMA0-DMA1-DMA2-...-DMA16 1h = Round robin priority enabled, DMA channel priority changes with each transfer

4.3.20 DMATCTL[j] (Offset = 1110h + formula) [Reset = 00000000h]

DMATCTL[j] is shown in Figure 4-23 and described in Table 4-29.

Return to the Summary Table.

DMA Trigger Control

Offset = 1110h + (j * 4h); where j = 0h to Fh

Figure 4-23. DMATCTL[j]

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
DMATINT	RESERVED	DMATSEL					
R/W-0h	R/W-	R/W-0h					

Table 4-29. DMATCTL[j] Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	
7	DMATINT	R/W	0h	DMA Trigger by Internal Channel 0h = DMATSEL will define external trigger select as transfer trigger. 1h = DMATSEL will define internal channel as transfer trigger select. 0-> Channel0-done, 1-> Channel1-done, ...
6	RESERVED	R/W	0h	
5-0	DMATSEL	R/W	0h	DMA Trigger Select Note: Reference the data sheet of the device to see the specific trigger mapping. 00h = Software trigger request 3Fh = Highest possible value

4.3.21 DMACTL[j] (Offset = 1200h + formula) [Reset = 00000000h]

DMACTL[j] is shown in Figure 4-24 and described in Table 4-30.

Return to the Summary Table.

DMA Channel Control

Offset = 1200h + (j * 10h); where j = 0h to Fh

Figure 4-24. DMACTL[j]

31	30	29	28	27	26	25	24
RESERVED		DMATM		RESERVED		DMAEM	
R/W-		R/W-0h		R/W-		R/W-0h	
23	22	21	20	19	18	17	16
DMADSTINCR				DMASRCINCR			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED		DMADSTWDTH		RESERVED		DMASRCWDTH	
R/W-		R/W-0h		R/W-		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	DMAPREIRQ			RESERVED		DMAEN	DMAREQ
R/W-	R/W-0h			R/W-		R/W-0h	R/W-0h

Table 4-30. DMACTL[j] Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	
29-28	DMATM	R/W	0h	DMA transfer mode register Note: The repeat-single (2h) and repeat-block (3h) transfer are only available in a FULL-channel configuration. Please consult the data sheet of the specific device to map which channel number has FULL or BASIC capability. In a BASIC channel configuration only the values for single (0h) and block (1h) transfer can be set. 0h = Single transfer. Each transfers requires a new trigger. When the DMASZ counts down to zero an event can be generated and the DMAEN is cleared. 1h = Block transfer. Each trigger transfers the complete block defined in DMASZ. After the transfer is complete an event can be generated and the DMAEN is cleared. 2h = Repeated single transfer. Each transfers requires a new trigger. When the DMASZ counts down to zero an event can be generated. After the last transfer the DMASA, DMADA, DAMSZ registers are restored to its initial value and the DMAEN stays enabled. 3h = Repeated block transfer. Each trigger transfers the complete block defined in DMASZ. After the last transfer the DMASA, DMADA, DAMSZ registers are restored to its initial value and the DMAEN stays enabled.
27-26	RESERVED	R/W	0h	
25-24	DMAEM	R/W	0h	DMA extended mode Note: The extended transfer modes are only available in a FULL-channel configuration. Please consult the data sheet of the specific device to map which channel number has FULL or BASIC capability. In a BASIC channel configuration this register is a read-only register and reads 0x0. 0h = Normal mode is related to transfers from SRC to DST 2h = Fill mode will copy the SA register content as data to DA 3h = Table mode will read an address and data value from SA and write the data to address

Table 4-30. DMACTL[j] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-20	DMADSTINCR	R/W	0h	DMA destination increment. This bit selects automatic incrementing or decrementing of the destination address DMADA for each transfer. The amount of change to the DMADA is based on the definition in the DMADSTWDTH. For example an increment of 1 (+1) on a WORD transfer will increment the DMADA by 4. 0h = Address is unchanged (+0) 2h = Decremented by 1 (-1 * DMADSTWDTH) 3h = Incremented by 1 (+1 * DMADSTWDTH) 8h = Stride size 2 (+2 * DMADSTWDTH) 9h = Stride size 3 (+3 * DMADSTWDTH) Ah = Stride size 4 (+4 * DMADSTWDTH) Bh = Stride size 5 (+5 * DMADSTWDTH) Ch = Stride size 6 (+6 * DMADSTWDTH) Dh = Stride size 7 (+7 * DMADSTWDTH) Eh = Stride size 8 (+8 * DMADSTWDTH) Fh = Stride size 9 (+9 * DMADSTWDTH)
19-16	DMASRCINCR	R/W	0h	DMA source increment. This bit selects automatic incrementing or decrementing of the source address DMASA for each transfer. The amount of change to the DMASA is based on the definition in the DMASRCWDTH. For example an increment of 1 (+1) on a WORD transfer will increment the DMASA by 4. 0h = Address is unchanged (+0) 2h = Decremented by 1 (-1 * DMASRCWDTH) 3h = Incremented by 1 (+1 * DMASRCWDTH) 8h = Stride size 2 (+2 * DMASRCWDTH) 9h = Stride size 3 (+3 * DMASRCWDTH) Ah = Stride size 4 (+4 * DMASRCWDTH) Bh = Stride size 5 (+5 * DMASRCWDTH) Ch = Stride size 6 (+6 * DMASRCWDTH) Dh = Stride size 7 (+7 * DMASRCWDTH) Eh = Stride size 8 (+8 * DMASRCWDTH) Fh = Stride size 9 (+9 * DMASRCWDTH)
15-14	RESERVED	R/W	0h	
13-12	DMADSTWDTH	R/W	0h	DMA destination width. This bit selects the destination as a byte, half word, word or long word. 0h = Destination data width is BYTE (8-bit) 1h = Destination data width is HALF-WORD (16-bit) 2h = Destination data width is WORD (32-bit) 3h = Destination data width is LONG-WORD (64-bit)
11-10	RESERVED	R/W	0h	
9-8	DMASRCWDTH	R/W	0h	DMA source width. This bit selects the source data width as a byte, half word, word or long word. 0h = Source data width is BYTE (8-bit) 1h = Source data width is HALF-WORD (16-bit) 2h = Source data width is WORD (32-bit) 3h = Source data width is LONG-WORD (64-bit)
7	RESERVED	R/W	0h	

Table 4-30. DMACTL[j] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	DMAPREIRQ	R/W	0h	Enable an early IRQ event. This can help software to react quicker to and DMA done event or allows some additional configuration before the channel is complete. Note: This register is only available in a FULL-channel configuration. Please consult the data sheet of the specific device to map which channel number has FULL or BASIC capability. In a BASIC configuration this register is a read only value and always reads as 0x0. 0h = Pre-IRQ event disabled. 1h = Issue Pre-IRQ event when DMASZ=1 2h = Issue Pre-IRQ event when DMASZ=2 3h = Issue Pre-IRQ event when DMASZ=4 4h = Issue Pre-IRQ event when DMASZ=8 5h = Issue Pre-IRQ event when DMASZ=32 6h = Issue Pre-IRQ event when DMASZ=64 7h = Issue Pre-IRQ event when DMASZ reached the half size point of the original transfer size
3-2	RESERVED	R/W	0h	
1	DMAEN	R/W	0h	DMA enable 0h = DMA channel disabled 1h = DMA channel enabled
0	DMAREQ	R/W	0h	DMA request. Software-controlled DMA start. DMAREQ is reset automatically. 0h = Default read value 1h = DMA transfer request (start DMA)

4.3.22 DMASA[j] (Offset = 1204h + formula) [Reset = 00000000h]

DMASA[j] is shown in Figure 4-25 and described in Table 4-31.

Return to the Summary Table.

DMA Channel Source Address

Offset = 1204h + (j * 10h); where j = 0h to Fh

Figure 4-25. DMASA[j]

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
ADDR																															
R/W-0h																															

Table 4-31. DMASA[j] Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	DMA Channel Source Address 0h = Smallest value FFFFFFFFh = Highest possible value

4.3.23 DMADA[j] (Offset = 1208h + formula) [Reset = 00000000h]

DMADA[j] is shown in Figure 4-26 and described in Table 4-32.

Return to the Summary Table.

DMA Channel Destination Address

Offset = 1208h + (j * 10h); where j = 0h to Fh

Figure 4-26. DMADA[j]

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
ADDR																															
R/W-0h																															

Table 4-32. DMADA[j] Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	DMA Channel Destination Address 0h = Smallest value FFFFFFFFh = Highest possible value

4.3.24 DMASZ[j] (Offset = 120Ch + formula) [Reset = 00000000h]

DMASZ[j] is shown in Figure 4-27 and described in Table 4-33.

Return to the Summary Table.

DMA Channel Size

Offset = 120Ch + (j * 10h); where j = 0h to Fh

Figure 4-27. DMASZ[j]

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
RESERVED																SIZE															
R/W-																R/W-0h															

Table 4-33. DMASZ[j] Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	
15-0	SIZE	R/W	0h	DMA Channel Size in number of transfers 0h = Smallest value FFFFh = Highest possible value

Chapter 5

NVM (Flash)

The nonvolatile memory (NVM) system provides nonvolatile flash memory for storing executable code and data.

5.1 NVM Overview

The nonvolatile memory system provides in-system programmable flash memory for storing executable code and data. This chapter describes the entire functionality provided by the nonvolatile memory system.

5.1.1 Key Features

Key features of the nonvolatile memory system include:

- In-circuit program and erase supported across the entire supply voltage range
- Internal programming voltage generation
- 64-bit flash word size
- Static write protection (latched at boot and held until BOR or POR)
- Dynamic write protection (configurable at runtime)
- Sector (1KB) and bank (up to 256KB) erase
- Automatic hardware preverification to extend flash bank longevity
- Automatic hardware post-verification of program/erase
-
-
-

Note

To determine if a device has any of the optional features described above, review the nonvolatile memory system detailed description in the corresponding device data sheet.

5.1.2 System Components

The nonvolatile memory system consists of three components (listed below):

- One or more flash memory banks (for storing code and data)
- The flash controller (for managing all program/erase operations on the flash banks)
- The read interface (for interfacing the flash banks to the CPU and peripheral bus)

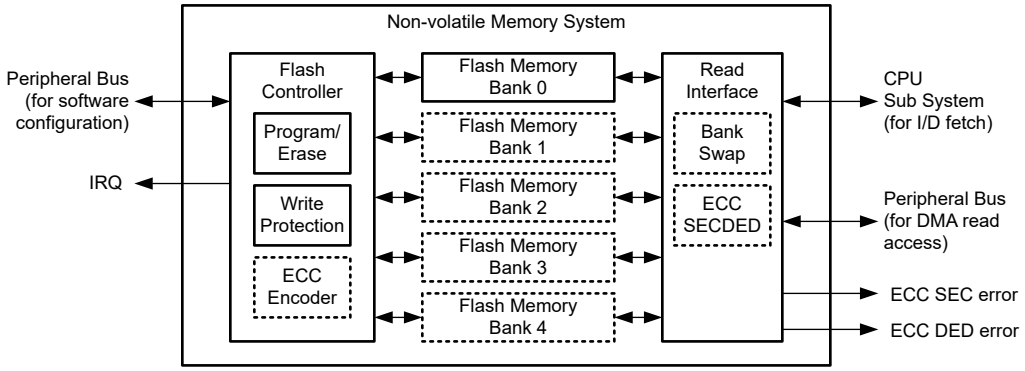


Figure 5-1. Non-volatile Memory System Components

5.1.3 Terminology

Key flash bank terms are defined in this section to be used as a reference for the rest of this chapter.

Table 5-1. NVM System Terminology

Term	Definition	Size
Flash word	Basic data size for program and read operations on the flash memory (also the read bus width to the system)	64 data bits (72 bits with ECC)
Word line	Group of flash words within a sector, with maximum program operation limit before sector erase	16 flash words (128 data bytes, optionally 16 ECC bytes)
Sector	Group of word lines that are erased together (minimum erase resolution of the flash memory)	8 word lines (1024 data bytes, optionally 128 ECC bytes)

Table 5-1. NVM System Terminology (continued)

Term	Definition	Size
Bank	Group of sectors that can be mass erased in one operation. Only one read, program, erase, or verify operation can run concurrently on a given bank.	Variable
Region	Logical assignment of a region of flash memory from a bank.	Variable

5.2 Flash Memory Bank Organization

The flash memory is used for storing application code and data, the device boot configuration, and parameters which are preprogrammed by TI from the factory. The flash memory is organized into one or more banks, and the memory in each bank is further mapped into one or more logical memory regions and assigned system address space for use by the application.

5.2.1 Banks

The nonvolatile memory system provides support for up to 5 flash memory banks (enumerated as BANK0 through BANK4). The number of flash banks present is device dependent. To determine the bank scheme of a particular device, review the detailed description section of the specific device data sheet. Most devices implement a single flash bank (BANK0).

On devices with a single flash bank, an ongoing program/erase operation will stall all read requests to the flash memory until the operation has completed and the flash controller has released control of the bank. On devices with more than one flash bank, a program/erase operation on a bank will also stall read requests issued to the bank which is executing the program/erase operation, but it will not stall read requests issued to any other bank. As such, the presence of multiple banks enables application cases such as:

- Dual-image firmware updates (an application can execute code out of one flash bank while a second image is programmed to a second symmetrical flash bank without stalling the application execution)
- EEPROM emulation (an application can execute code out of one flash bank while a second flash bank is used for writing data without stalling the application execution)

5.2.2 Flash Memory Regions

The memory within each bank is mapped to one or more logical regions based upon the functions that the memory in each bank supports. There are four regions: FACTORY, NONMAIN (Configuration NVM), MAIN (Flash Memory) , and DATA.

Table 5-2. Flash Memory Regions

Flash Memory Region	Region Contents	Executable	Used by	Programmed by
FACTORY	Device ID and other parameters	No	Application	TI only (not modifiable)
NONMAIN (Configuration NVM)	Device boot configuration (BCR and BSL)	No	Boot ROM	TI, User
MAIN (Flash Memory)	Application code and data	Yes	Application	User
DATA	Data, or EEPROM emulation	No	Application	User

Devices with one bank implement the FACTORY, NONMAIN, and MAIN regions on BANK0 (the only bank present), and the data region is not available. Devices with multiple banks also implement FACTORY, NONMAIN, and MAIN regions on BANK0, but include additional banks (BANK1 through BANK4) that can implement MAIN or DATA regions.

For a detailed description of the contents of the read-only FACTORY region, see Section 1.6.

5.2.3 Addressing

The flash memory regions are assigned to address space in the system memory map.

The NONMAIN, DATA, and FACTORY regions are assigned to the peripheral address space (0x4000.0000) as they do not contain any executable code. The CPU should not fetch executable instructions from this region.

The MAIN region is assigned to both the code address space (0x0000.0000) and the peripheral address space (0x4000.0000). Instruction and data fetches are recommended to always be done through the code address space as this gives the best performance. The CPU should not fetch executable instructions from this region.

ECC

On devices which have error correction code (ECC) support, the ECC codes for all memory regions are also assigned to address space and it is possible for software to read the ECC codes as data for diagnostic purposes. It is also possible to read the contents of any of the memory regions without ECC correction applied.

5.2.3.1 Flash Memory Map

The following table lists the system address space assignments. These assignments are consistent for all devices.

NONMAIN, DATA, and FACTORY data reads are processed through the peripheral bus and peripheral address space only. MAIN regions can be accessed through either the CPU bus matrix or through the peripheral bus, depending on whether code address space or peripheral address space is used. The code address space is recommended for CPU accesses (instruction fetches or data reads), as these accesses do not cross the peripheral bus and thus do not compete with the DMA for control of the peripheral bus. See the bus architecture section for a detailed description of the bus interconnect.

On devices that have ECC, an access to an ECC code address returns the 8-bit ECC value for the entire 64-bit flash word that was accessed. On devices that do not have ECC, accesses to the corrected and uncorrected ECC address spaces with the same offset read the same, and ECC code addresses read as 0x0.

5.2.4 Memory Organization Examples

Figure 5-2 is an example of a single bank configuration with a 64KB MAIN region. NONMAIN and FACTORY regions are also included in the single bank (BANK0) with the MAIN region. Most devices with a main region ≤128KB in size implement a single-bank configuration.

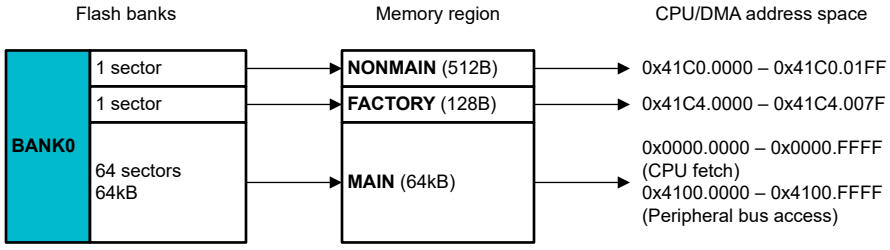


Figure 5-2. Memory Organization Example - Single Bank Configuration

Figure 5-3 is an example of a three bank configuration with a 512KB MAIN region split across BANK0 and BANK1, with a 16KB DATA region provided in BANK2. Like the single bank example, NONMAIN and FACTORY regions are included in BANK0. This example supports EEPROM emulation in the DATA region without stalling fetches to MAIN, and it also supports dual-image applications where BANK0 main can be written to without stalling fetches to BANK1 main (and the reverse). Most devices with a main region ≥256KB in size implement a multibank configuration.

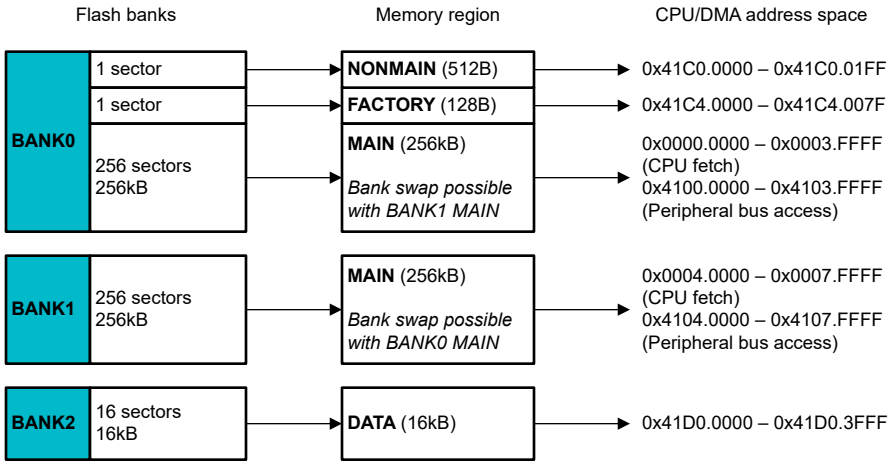


Figure 5-3. Memory Organization Example - Multiple Bank Configuration

5.3 Flash Controller

The flash controller manages all program, erase, and verification operations performed on the nonvolatile memory system. It contains memory-mapped registers in the peripheral region of the device memory map which must be configured by software to perform operations on the flash memory.

TI provides software abstraction for the flash controller as a part of the DriverLib layer of the software development kit (SDK). It is recommended to use the DriverLib abstraction layer when operating on the flash memory with software, but it is not mandatory to do so. To use the DriverLib software abstraction layer to perform operations on the flash memory, review the software development kit (SDK) documentation provided separately from this document. To directly operate on the flash memory with using low level register accesses to the flash controller, review the remainder of this section in detail.

Note

The FLASHCTL registers may not always be configured to default values after a reset. This may occur if the boot configuration routine (BCR) or boot strap loader (BSL) perform an operation on the flash memory during boot. When configuring the FLASHCTL registers for an operation, ensure that all registers which are relevant for the operation are correctly configured.

5.3.1 Overview of Flash Controller Commands

Operations on the flash memory are executed by configuring the CMDTYPE and CMDCTL registers for the desired command, along with any other registers which must be configured for a particular command, and writing 0x01 to the CMDEXEC register to initiate the command.

When 0x01 is set in CMDEXEC, the commanded operation begins executing. While an operation is executing, most configuration registers are blocked for writes until the operation completes. Some registers (for example, mask registers) can change state under hardware control while the operation runs to completion. The flash controller indicates completion of the commanded operation by setting the CMDDONE bit in the STATCMD register. The flash controller also sources an interrupt vector to the CPU subsystem to indicate a “DONE” status when an operation has completed.

The software sequence of setting the CMDEXEC bit and waiting for the CMDDONE response must be executed from either the device SRAM or from a different flash bank from the bank that is being operated on, as the flash controller will take control of the flash bank undergoing the operation. Reads to the flash bank that is being operated on while the flash controller is executing the command are not predictable.

The flash controller provides five basic commands for operating on the flash memory, specified in the COMMAND field of the CMDTYPE register. These commands are described in Table 5-3.

Table 5-3. Flash Controller Commands

Command	Description
NOOP	No operation (default setting).
PROGRAM	Selects a program operation on the flash memory.
ERASE	Selects an erase operation on the flash memory.
READVERIFY	Selects a standalone read verify operation.
BLANKVERIFY	Selects a standalone blank verify operation.

5.3.2 NOOP Command

When not using the flash controller, it is best to set the COMMAND field to NOOP to prevent any unintentional operations on the flash memory in the event that the VAL bit in CMDEXEC is unintentionally set. Executing a NOOP command has no effect on the flash memory.

5.3.3 PROGRAM Command

The program command is used to write (program) the flash memory. Specifically, the purpose of a PROGRAM operation is to configure the flash bits in one or more flash words from the nondeterministic erased state to the deterministic programmed state. Once a byte is programmed using the PROGRAM command, the byte can not be re-programmed unless the sector is erased using the ERASE command.

All devices support single flash word programming of 64 data bits (plus 8 ECC bits on devices with ECC) at a time, with control to limit the scope of a program operation to specific bytes within a 64-bit flash word.

Some devices additionally have support for a multi-word programming mode where 2, 4, or 8 flash words can be written with a single commanded operation. Multi-word programming, when available, significantly speeds up programming when multiple words need to be programmed (for example, during production programming or firmware updates). See the device-specific data sheet to determine if multi-word programming is supported, and if so, how many flash word buffers are provided.

5.3.3.1 Program Bit Masking Behavior

The flash controller provides a program verification mechanism to extend the lifetime of the flash bank. During program operations, the CMDDATAx registers are used by the flash controller as a programming bit mask to indicate which specific bits in the flash word require program pulses. As a result, data which is loaded into the CMDDATAx registers before starting the program operation will be lost from the CMDDATAx registers during and/or upon completion of the program operation. If the same data is to be programmed again, the CMDDATAx registers must be re-loaded by software with the correct data values to be programmed.

5.3.3.2 Programming Less Than One Flash Word

In general, the simplest way to program the flash memory is one flash word at a time (64 bits plus 8 ECC bits if ECC is present). It is possible to program the flash memory with 32-, 16-, or 8-bit (byte) resolution, but special care must be taken when doing so to ensure the following:

1. On devices with ECC, ECC bits must be handled properly to prevent inadvertent ECC errors.
2. The number of program operations applied to a given word line must be monitored to ensure that the maximum word line program limit before erase is not violated.
3. Once a byte has been programmed, the sector containing the byte must be erased before attempting to re-program the same byte.

To program less than one flash word, the CMDBYTEN register must be configured to indicate which bytes in the flash word are to be programmed before starting the program operation. Each bit in CMDBYTEN corresponds to a byte in the flash word to be programmed, including the ECC bits.

Handling ECC

On devices with ECC, programming 64 bits of data at a time ensures that the 8 ECC bits which correspond to the 64-bit data word are also programmed both correctly and at the same time. Doing so prevents ECC errors from occurring if the memory locations are read by the CPU or DMA after programming.

If use of ECC is planned and partial programming is required, one approach is to mask (not program) the ECC bits until all 64 bits of a flash word are programmed, at which time the ECC bits can also be programmed. Programming of ECC bits is masked by clearing BIT8 (0x100) in the CMDBYTEN register. This prevents a situation where the entire sector must be erased each time a program operation is done to re-program ECC bits to match the new 64-bit data. However, in this case, a read to a partially programmed word where the ECC bits are not yet written would result in an ECC error. To avoid an ECC error, the software must either wait until the full 64-bit flash word and the 8 ECC bits are written, or read the data from the uncorrected address space.

Maximum Program Operations per Word Line Before Erase

The device data sheet specifies a maximum limit on the number of program operations per word line before erasure of the sector containing the word line is required. Exceeding this maximum can result in data corruption within the word line.

If 16-bit or greater program operations are performed, and no 16-bit location is programmed more than once before a sector is erased, **the maximum limit will never be reached and thus does not need to be considered**.

If 8-bit (byte) program operations are performed, the maximum program limit per word line must be considered and not exceeded. Program operations performed on ECC locations, if done independently from other program operations, count towards the number of writes before an erase is required.

5.3.3.3 Target Data Alignment (Devices with Single Flash Word Programming Only)

For devices which only support single word programming, only the CMDDATA0, CMDDATA1, and CMDECC0 registers are used to load data to be programmed to the flash memory. CMDDATA0 is always loaded with BIT31-BIT0 of the target data, and CMDDATA1 is always loaded with BIT63-BIT32 of the target data. ECC data, if specified directly and not computed automatically, is loaded into BIT7-BIT0 of CMDECC0. No other CMDDATAx or CMDECCx registers are used, and CMDDATAINDEX is not used. If fewer than 64 data bits are being programmed, see the special handling requirements section above for programming less than one flash word.

Single-word program operations must be flash word (64-bit) aligned. This means that the target system address specified in CMDADDR must be aligned to a 0b000 boundary (for example, the 3 LSBs in CMDADDR must be zero).

5.3.3.4 Target Data Alignment (Devices With Multiword Programming)

For devices that support 2-, 4-, or 8-word programming, there are two options for loading data to be programmed: direct mode or indexed mode. The programmer must select the mode that is most suitable to the application requirements.

Additional alignment rules apply when loading data into the CMDDATAx and CMDECCx registers on devices that support multiword programming, even if the multiword programming feature is not used and only single word programming is not used.

- 1-word program operations must have CMDADDR (the target system address) aligned to a 0b000 boundary (for example, the 3 LSBs in CMDADDR must be zero).
- 2-word program operations must have CMDADDR (the target system address) aligned to a 0b0000 boundary (for example, the 4 LSBs in CMDADDR must be zero).
- 4-word program operations must have CMDADDR (the target system address) aligned to a 0b0.0000 boundary (for example, the 5 LSBs in CMDADDR must be zero).
- 8-word program operations must have CMDADDR (the target system address) aligned to a 0b00.0000 boundary (for example, the 6 LSBs in CMDADDR must be zero).

Direct Data Load

To configure a program operation with direct data loading, the target data is loaded into the appropriate CMDDATAx registers based on the number of flash words supported by the device, the target address alignment, and the target data size. For example, if a 4-word program operation is to be initiated on a device

supporting 4-word programming, the CMDDATA0-CMDDATA7 registers would be populated with the target data. If ECC is being specified directly (rather than automatically calculated by the flash controller) then the appropriate CMDECCx registers also needs to be populated with the ECC values for each data word being programmed.

Indexed Data Load

Rather than buffering data into all the CMDDATAx registers individually, it is possible to use only the CMDDATA0-CMDDATA1 registers in combination with an index register (CMDDATAINDEX) to indicate the flash word offset of the data being loaded. In this way, the index can be adjusted for each word loaded, and each target data word can be written to the same 64-bit space (CMDDATA0-CMDDATA1). When an index is applied, the loaded data will be mapped by the hardware into the appropriate CMDDATAx register. For example, if a 4-word program operation is to be initiated on a device supporting 4-word programming, the CMDDATA1:0 registers are loaded 4 times with the target data, with CMDDATAINDEX being incremented by one before each new word is loaded into CMDDATA1:0.

Alignment Rules

The alignment rules for each device configuration (2, 4, or 8 words) is given in the tables below with guidance on how target data must be placed with the CMDDATAx, CMDECCx, and CMDDATAINDEX registers for 1, 2, 4, or 8 word programming operations.

Table 5-4. Data Load Alignment for Devices Supporting Programming of 2 Flash Words

Direct Load Registers	Indexed Load Index	1 Word Aligned to 0b000	2 Words Aligned to 0b0000	4 Words Aligned to 0b0.0000	8 Words Aligned to 0b00.0000
CMDDATA1:0 / CMDECC0	CMDINDEX = 0	Target data word 0 if the target address ends in 0b0000	Target data word 0	Not supported	Not supported
CMDDATA3:2 / CMDECC1	CMDINDEX = 1	Target data word 0 if the target address ends in 0b1000	Target data word 1		

Table 5-5. Data Load Alignment for Devices Supporting Programming of 4 Flash Words

Direct Load Registers	Indexed Load Index	1 Word Aligned to 0b000	2 Words Aligned to 0b0000	4 Words Aligned to 0b0.0000	8 Words Aligned to 0b00.0000
CMDDATA1:0 / CMDECC0	CMDINDEX= 0	Target data word 0 if the target address ends in 0b0.0000	Target data word 0 if the target address ends in 0b0.0000	Target data word 0	Not supported
CMDDATA3:2 / CMDECC1	CMDINDEX= 1	Target data word 0 if the target address ends in 0b0.1000	Target data word 1 if the target address ends in 0b0.0000	Target data word 1	
CMDDATA5:4 / CMDECC2	CMDINDEX= 2	Target data word 0 if the target address ends in 0b1.0000	Target data word 0 if the target address ends in 0b1.0000	Target data word 2	
CMDDATA7:6 / CMDECC3	CMDINDEX= 3	Target data word 0 if the target address ends in 0b1.1000	Target data word 1 if the target address ends in 0b1.0000	Target data word 3	

Table 5-6. Data Load Alignment for Devices Supporting Programming of 8 Flash Words

Direct Load Registers	Indexed Load Index	1 Word Aligned to 0b000	2 Words Aligned to 0b0000	4 Words Aligned to 0b0.0000	8 Words Aligned to 0b00.0000
CMDDATA1:0 / CMDECC0	CMDINDEX = 0	Target data word 0 if the target address ends in 0b00.0000	Target data word 0 if the target address ends in 0b00.0000	Target data word 0 if the target address ends in 0b00.0000	Target data word 0
CMDDATA3:2 / CMDECC1	CMDINDEX = 1	Target data word 0 if the target address ends in 0b00.1000	Target data word 1 if the target address ends in 0b00.0000	Target data word 1 if the target address ends in 0b00.0000	Target data word 1

Table 5-6. Data Load Alignment for Devices Supporting Programming of 8 Flash Words (continued)

Direct Load Registers	Indexed Load Index	1 Word Aligned to 0b000	2 Words Aligned to 0b0000	4 Words Aligned to 0b0.0000	8 Words Aligned to 0b00.0000
CMDDATA5:4 / CMDECC2	CMDINDEX = 2	Target data word 0 if the target address ends in 0b01.0000	Target data word 0 if the target address ends in 0b01.0000	Target data word 2 if the target address ends in 0b00.0000	Target data word 2
CMDDATA7:6 / CMDECC3	CMDINDEX = 3	Target data word 0 if the target address ends in 0b01.1000	Target data word 1 if the target address ends in 0b01.0000	Target data word 3 if the target address ends in 0b00.0000	Target data word 3
CMDDATA9:8 / CMDECC4	CMDINDEX = 4	Target data word 0 if the target address ends in 0b10.0000	Target data word 0 if the target address ends in 0b10.0000	Target data word 0 if the target address ends in 0b10.0000	Target data word 4
CMDDATA11:10 / CMDECC5	CMDINDEX = 5	Target data word 0 if the target address ends in 0b10.1000	Target data word 1 if the target address ends in 0b10.0000	Target data word 1 if the target address ends in 0b10.0000	Target data word 5
CMDDATA13:12 / CMDECC6	CMDINDEX = 6	Target data word 0 if the target address ends in 0b11.0000	Target data word 0 if the target address ends in 0b11.0000	Target data word 2 if the target address ends in 0b10.0000	Target data word 6
CMDDATA15:14 / CMDECC7	CMDINDEX = 7	Target data word 0 if the target address ends in 0b11.1000	Target data word 1 if the target address ends in 0b11.0000	Target data word 3 if the target address ends in 0b10.0000	Target data word 7

5.3.3.5 Executing a PROGRAM Operation

To program the flash memory:

1. Select the command in the CMDTYPE register:
   a. Set the COMMAND field in the CMDTYPE register to PROGRAM.
   b. Set the SIZE field in the CMDTYPE register to the desired size (1, 2, 4, or 8 flash words). If a device does not support multi-word programming, select ONEWORD. If a device supports multi-word programming, and multi-word programming is desired, select the desired size which is less than or equal to the max size supported by the target device. The hardware will not check for invalid configuration of the SIZE field; software must ensure that the selection option is supported by the device. Note that SECTOR and BANK sizes are not valid sizes for PROGRAM operations. These sizes only apply to erase operations.
2. Configure the program command in the CMDCTL register:
   a. On devices with ECC, the flash controller by default will generate the needed ECC bits from the data during the PROGRAM operation. Optionally, software can override the hardware ECC code generation and manually provide the ECC code to be programmed by setting the ECCGENOVR bit in CMDCTL register.
3. Select the target programming address in the CMDADDR and CMDBYTEN register:
   a. Load the target system address into the CMDADDR register to indicate the base address from which programming will start. The target address must be a flash word address (64-bit aligned). The flash controller will translate the system address into the applicable flash region, bank ID, and bank address. If desired, after the operation completes the flash region, bank ID, and bank address can be read from the STATADDR register. In a multi-word program, STATADDR will indicate the bank ID and the final address which was programmed.
   b. If subword programming (programming of less than the full 64 or 72 bit flash word) is desired, configure the CMDBYTEN register to set the bytes within the addressed flash word which are to be programmed. Each bit in CMDBYTEN corresponds to a byte in the addressed flash word to be programmed, including the ECC byte. For example, programming of the ECC code can be masked by clearing bit 8 in CMDBYTEN while programming the data bytes of the flash word. Note that there is a maximum number of program operations allowed per word line before a sector erase must be applied (see the device specific data sheet for the maximum).
4. Load the data to program into the CMDDATAx registers:
   a. For a single flash word programming operation (64 or 72 bits depending on the presence of ECC), load the data into the CMDDATAx registers consistent with the alignment requirements (for devices which

only support single-word programming, CMDDATA0 and CMDDATA1 are always used regardless of the target address).
   b. For multi-word programming (if available and selected), load data into the CMDDATAx registers consistent with the alignment rules and the size of the multi-word program operation specified in step 1.
   c. If ECCGENOVR in the CMDCTL register was set above (disabling hardware ECC code generation), then write the ECC data in the CMDDATAECC0 register (for single word programming) and optionally additional CMDDATAECCx registers as applicable for multi-word programming.
   d. Note that the CMDDATA registers are used as bit masking registers by the flash controller during the program operation; after the operation completes, the values written to these registers will have been overwritten by the flash controller.
5. Ensure the write protection scheme is configured to allow writes to the target addresses (see the write protection section of this guide for additional information on configuring write protection).
6. Execute the program operation by writing 0x1 to the CMDEXEC register.
7. Monitor for completion of the program operation:
   a. The STATCMD register can be polled to determine the status of the program operation. The CMDINPROGRESS bit will be set by hardware as soon as the command is initiated. The CMDDONE bit will be set when the operation terminates.
   b. When CMDDONE is set, the CMDPASS bit will be reset or set at the same time to indicate whether the operation completed successfully or failed. If a program was attempted on a protected region, the FAILWEPROT bit is asserted. If the program operation cannot be completed successfully within the maximum program pulse count limit FAILVERIFY will be asserted. See the device-specific data sheet for maximum program times.
8. After completion of a program operation, the flash controller will configure several settings:
   a. All dynamic write protection registers are set to a protected state (to protect against inadvertent programming)
   b. All data registers are set to 1.
   c. All program byte enables are cleared to 0.
9. Following programming of the flash memory, it is possible that there may be stale data in the processor's cache and prefetch logic. Before reading locations which were programmed, it is recommended to first flush the cache in the CPU subsystem.

5.3.4 ERASE Command

The erase command is used to erase individual sectors of flash memory (for MAIN, NONMAIN, or DATA regions) or a complete bank of flash memory (for MAIN regions only). From this erased state, bits can later be programmed to a ‘0’ state or a '1' state as desired using the PROGRAM command. It is not possible to erase with a resolution lower than one sector (1KB) with sector alignment. For devices with multiple banks, a bank erase must be executed on all banks to erase the entire MAIN region on the device.

Note

After erasure, the memory contents of a sector are not deterministic until programmed. Erased bits do not always read as 1 after an erase. A memory location must be successfully programmed using the PROGRAM command before the memory location can be considered deterministic.

5.3.4.1 Erase Sector Masking Behavior

The flash controller provides an erase verification mechanism to extend the lifetime of the flash bank. The CMDWEPROTx registers are used as an erase mask and are manipulated by the flash controller during the execution of the erase operation. At the end of all erase operations, the CMDWEPROTx registers are set to a fully protected state to prevent against inadvertent programming and must be re-configured before attempting another program or erase operation.

5.3.4.2 Executing an ERASE Operation

To erase a sector or bank of the flash memory, follow these steps:

1. Select the command in the CMDTYPE register:

a. Set the COMMAND field in the CMDTYPE register to ERASE.
b. Set the SIZE field in the CMDTYPE register to SECTOR or BANK. Sizes other than SECTOR or BANK (for example, ONEWORD) are not supported by the ERASE command. It is the responsibility of software to check the configuration before issuing an ERASE command.

2. Select the target erase address in the CMDADDR register:
   a. Store the target system address into the CMDADDR register to indicate the base address of the sector or bank to be erased. When performing a bank erase with write protection enabled (such that only unprotected sectors are erased), ensure that the address written to CMDADDR is in an unprotected sector. The flash controller will translate the system address into the applicable flash region, bank ID, and bank address. If desired, after the operation completes the flash region, bank ID, and bank address can be read from the STATADDR register.
3. Ensure the write protection scheme is configured to allow writes to the target sectors (see the write protection section of this guide for additional information on configuring write protection).
4. Execute the erase operation by writing 0x1 to the CMDEXEC register.
5. Monitor for completion of the erase operation:
   a. The STATCMD register can be polled to determine the status of the erase operation. The CMDINPROGRESS bit will be set by hardware as soon as the command is initiated. The CMDDONE bit will be set when the operation terminates. When CMDDONE is set, the CMDPASS bit will be reset or set at the same time to indicate whether the operation completed successfully or failed.
   b. If an erase was attempted on a protected region, the FAILWEPROT bit is asserted.
   c. If the erase operation cannot be completed successfully within the maximum erase pulse count limit, FAILVERIFY will be asserted.
6. After completion of the erase operation, the flash controller will configure several settings to protect against inadvertent programming:
   a. All dynamic write protection registers are set to a protected state.

5.3.5 READVERIFY Command

The read verify command can be used to read a flash location and compare the data to data which is preloaded into the CMDDATA registers of the flash controller. The command can be applied to a single flash word, multiple flash words (if the device supports multi-word programming), an entire sector, or an entire bank. When performing a read verify on an entire sector or bank, the data in CMDDATAx will be re-used.

5.3.5.1 Executing a READVERIFY Operation

To execute a read verify command, follow these steps:

1. Select the command in the CMDTYPE register:
   a. Set the COMMAND field in the CMDTYPE register to READVERIFY.
   b. Set the SIZE field in the CMDTYPE register to the desired size.
2. Configure the read verify command in the CMDCTL register:
   a. If the desire is to manually provide ECC bits along with the data, set the ECCGENOVR bit in the CMDCTL register. If ECCGENOVR is cleared, the flash controller will generate ECC bits for comparison based on the provided compare data.
3. Select the target address to verify on the CMDADDR register:
   a. Load the target system address into the CMDADDR register to indicate the base address to be verified. The flash controller will translate the system address into the applicable flash region, bank ID, and bank address. If desired, after the operation completes the flash region, bank ID, and bank address can be read from the STATADDR register.
4. Load the data to verify into the CMDDATAx registers:
   a. For single word verification, write the data to be verified to the CMDDATA0 and CMDDATA1 registers. For multi-word verification, if available on the target device, write the data to be verified to the appropriate CMDDATAx registers beyond CMDDATA0 and CMDDATA1.
5. Configure the byte enable settings in the CMDBYTEN register:

   a. Any CMDBYTEN bit set to logic “0” will mask the associated data byte from being compared during the execution of the READVERIFY command. This can be used to verify data less than one flash word (less than 64 bits).
6. Execute the read verify operation by writing 0x1 to the CMDEXEC register.
7. Monitor for completion of the read verify operation:
   a. The STATCMD register can be polled to determine the status of the erase operation. The CMDINPROGRESS bit will be set by hardware as soon as the command is initiated. The CMDDONE bit will be set when the operation terminates. When CMDDONE is set, the CMDPASS bit will be reset or set at the same time to indicate whether the read verification passed or failed.
   b. The FAILVERIFY bit in STATCMD will be set if any data read from the flash did not match the expected data loaded in CMDDATAx.
8. After completion of the read verify operation, the flash controller will configure several settings:
   a. All dynamic write protection registers are set to a protected state (to protect against inadvertent programming).
   b. All data registers are set to ‘1’s.
   c. All program byte enables are cleared to ‘0’s.

5.3.6 BLANKVERIFY Command

The blank verify (BLANKVERIFY) command can be used by application software to verify that a flash word is blank. A blank flash word is defined as a flash word which has been successfully erased with the ERASE command and not yet programmed away from that nonerased state with the PROGRAM command.

After erase, a flash word is not in a deterministic state until it is programmed using the PROGRAM command. This means that application software can not expect that erased bits will read back as '1's' after an erase. A memory location must be successfully programmed using the PROGRAM command before a read of that memory location can be considered deterministic and used by application software.

Because it is not possible to determine if a flash word is in an erased state by simply reading the location directly (as an erased location will return nondeterministic data when read), the BLANKVERIFY command can be used to test if a flash word is in a blank state, indicating it has not yet been programmed away from an erased state.

The BLANKVERIFY command can only be applied to a single flash word at a time.

5.3.6.1 Executing a BLANKVERIFY Operation

To execute a blank verify operation, follow these steps:

1. Select the command in the CMDTYPE register:
   a. Set the COMMAND field in the CMDTYPE register to BLANKVERIFY.
   b. Set the SIZE field in the CMDTYPE register to one word.
2. Select the target address to verify in the CMDADDR register:
   a. Store the target system address into the CMDADDR register to indicate the base address to be verified. The flash controller will translate the system address into the applicable flash region, bank ID, and bank address. If desired, after the operation completes the flash region, bank ID, and bank address can be read from the STATADDR register. All 8 data bytes in the specified flash word, and the corresponding ECC byte, will be checked by BLANKVERIFY.
3. Execute the blank verify operation by writing 0x1 to the CMDEXEC register.
4. Monitor for completion of the blank verify operation:
   a. The STATCMD register can be polled to determine the status of the erase operation. The CMDINPROGRESS bit will be set by hardware as soon as the command is initiated. The CMDDONE bit will be set when the operation terminates. When CMDDONE is set, the CMDPASS bit will be reset or set at the same time to indicate whether the blank verification passed or failed.
   b. The FAILVERIFY bit in STATCMD will be set if the flash location is not erased.
5. After completion of the blank verify operation, the flash controller will configure several settings:

a. All dynamic write protection registers are set to a protected state (to protect against inadvertent programming)
b. All data registers are set to ‘1’s.
c. All program byte enables are cleared to ‘0’s.

5.3.7 Command Diagnostics

The flash controller updates several software-readable registers to communicate information about an initiated operation.

5.3.7.1 Command Status

The STATCMD register is a read-only register which provides diagnostic information about an operation which is been initiated or completed. The CMDINPROGRESS bit indicates that an operation is currently ongoing. The CMDDONE bit indicates that an operation has completed. These bits can be polled by software to determine the state of the flash controller during operations.

5.3.7.2 Address Translation

The STATADDR register is a read-only register which can be read to determine the current bank ID, region ID, and bank address which the flash controller is pointing to. These values can increment during execution of certain commands, in which case the value present after the completion of a command indicates the last address touched by the flash controller.

5.3.7.3 Pulse Counts

The STATPCNT register is a read-only register which can be read to determine the current pulse count applied during a program or erase operation.

5.3.8 Overriding the System Address With a Bank ID, Region ID, and Bank Address

Normally, flash controller commands are targeted to a specific flash location by loading a system memory map address into the CMDADDR register. This is the recommended way to specify the target address for a PROGRAM, ERASE, READVERIFY, or BLANKVERIFY command. In this mode of operation, the flash controller will automatically translate the system address into the corresponding bank ID, region ID, and bank address which are used to execute the command on the flash memory. Application software does not need to specify these items individually; only the system address is needed.

However, in some circumstances it can be desirable to directly specify the target flash bank, region, and address in the specified bank/region. For example, if the desire is to erase a complete bank when doing a mass erase operation on the device, application software actually does not need to have any knowledge of the system address of the bank to be erased- it only needs to specify the bank ID and region ID to the flash controller to erase the bank.

To use the flash controller to execute a command in address override mode, set the ADDRXLATEOVR bit in the CMDCTL register, and specify the bank ID, region, and bank address before executing the command. To return to system addressed mode, clear ADDRXLATEOVR. ADDRXLATEOVR is cleared by default (supporting system address operation).

Example Case - Bank Erase with ADDRXLATEOVR

To erase the MAIN region of BANK0 by specifying the bank ID and region instead of the system address, follow the steps in Section 5.3.4.2, but replace step 2 with the alternate steps given below:

1. Set the ADDRXLATEOVR bit in CMDCTL to enable address translation override mode
2. Specify BANK0 by setting the BANKSEL field to 0x1 in the CMDCTL register
3. Specify the MAIN region by setting the REGIONSEL field to 0x1 in the CMDCTL register
4. Set the CMDADDR register to 0x0000.0000

5.3.9 FLASHCTL Events

The flash controller contains one event publisher and no event subscribers. One event publisher (CPU_INT) manages FLASHCTL interrupt requests (IRQs) to the CPU subsystem through a static event route.

Table 5-7 summarizes the FLASHCTL events.

Table 5-7. FLASHCTL Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU Interrupt Event	Publisher	FLASHCTL	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from FLASHCTL to CPU

5.3.9.1 CPU Interrupt Event Publisher

The FLASHCTL module provides one interrupt source which can be configured to source a CPU interrupt event. The FLASHCTL interrupt conditions are given in Table 5-8.

Table 5-8. FLASHCTL CPU Interrupt Conditions

Index (IIDX)	Name	Description
0	DONE	Indicating that the FLASHCTL operation has completed

The CPU interrupt event configuration is managed with the CPU_INT event management registers. See Section 6.2.5 for guidance on configuring these registers.

5.4 Write Protection

The flash controller provides two write protection mechanisms (one static, one dynamic) which are applied in parallel (logical OR) to protect user-specified sectors during any attempted program or erase operation. If a program or erase operation is issued to a write protected flash sector, the operation will terminate with a FAILWEPROT error reported in the STATCMD register.

5.4.1 Write Protection Resolution

The write protection resolution for both the static and dynamic write protection mechanisms is dependent on the flash bank and memory region which is being protected.

Table 5-9. Write Protection Resolution by Region

Flash Bank	Memory Region	Write Protection Resolution
0	NONMAIN	512B (entire region)
	MAIN	1KB (1 sector) for first 32KB (32 sectors) 8KB (8 sectors) for remaining sectors
1-4	MAIN	8KB (8 sectors)
	DATA	1KB (1 sector)

5.4.2 Static Write Protection

The static write protection scheme is configured and latched during device boot by the immutable ROM boot code before the main application in flash can execute. After a flash sector is configured to be write protected through the static write protection mechanism, software cannot remove the protection at runtime without a reboot. Thus, sectors protected by static write protection can be thought of as immutable after boot. This type of protection is useful for protecting a custom bootloader in a dual-image application, or for extending the secure root of trust from the ROM boot code into a portion of the main flash region to enable a secure boot manager with locked public keys.

The static write protection scheme is configured by programming the appropriate bits in the NONMAIN flash region, which is read by the boot code before the main application starts. The NONMAIN flash sector can be statically write protected, resulting in a system in which the static write protection scheme is fully permanent and cannot be modified. If the NONMAIN sector is configured to be statically protected, along with any other flash

sectors, all statically protected flash sectors can be functionally viewed as read only memory which cannot be updated by any means.

Static write protection can be configured for all sectors of flash memory across all banks present on a device. Instructions for configuring the static write protection, along with the rest of the NONMAIN region, are given in Section 1.4.

5.4.3 Dynamic Write Protection

The dynamic write protection scheme is intended to be configured at runtime by software. This scheme provides a simple way for application software to specify sectors to protect from modification by any program or erase operations that are issued with the flash controller. Unlike the static write protection mechanism, the dynamic mechanism is not lockable and thus does not provide any level of data security.

There are two primary uses for dynamic write protection. First, it provides an extra level of robustness against unintentional program or erase of specified sectors in applications that involve in-system programming for either firmware updates or EEPROM emulation. Second, it provides a way to simplify a situation where a bank erase is desired but a small number of sectors should not be erased when a bank erase command is issued. One example would be an application running on a single-bank device in which most of the MAIN region sectors are used to store the executable image, but a few sectors are used to store device-specific data that should not be erased during a firmware update. In that case, the sectors containing the device-specific data can be protected with dynamic write protection, and a bank erase command can be issued to erase all other sectors. This has the benefit that the majority of the MAIN region can be erased with a single command (a bank erase) rather than individual sector-by-sector commands, which would have a longer overall erase time and use more energy.

The dynamic write protection scheme is configured by setting up the CMDWEPROTx registers in the flash controller. The CMDWEPROTx registers cover one flash bank at a time. This means that these registers must be configured with knowledge of which flash bank an attempted program or erase operation will be applied to. Note that the CMDWEPROTx registers are reset to a protected state at the end of all program and erase operations. These registers must be re-configured by software before a new operation is initiated.

5.4.3.1 Configuring Protection for the MAIN Region

The CMDWEPROTA register is used to configure the dynamic write protection for the first 32 sectors (32KB) of the MAIN region in BANK0. Each bit corresponds to one sector of the MAIN region, starting from the beginning of the MAIN region. CMDWEPROTA is only applicable to operations on the lower 32 sectors of the MAIN region of BANK0 (sectors 0-31). It is not used during program/erase operations applied to other sectors.

The CMDWEPROTB register is also used to configure the dynamic write protection for the MAIN region. There are two modes in which CMDWEPROTB is applied, depending on whether the program/erase operation is being applied to BANK0 or BANK1-4. In the case of a program/erase operation on BANK0, CMDWEPROTB protects sectors 32-255 in 8-sector increments (1 bit per 8KB), starting from BIT4. BIT0-BIT3 in CMDWEPROTB are ignored. The lower 32 sectors (sectors 0-31 of BANK0) are protected by CMDWEPROTA. In the case of a program/erase operation on BANK1-4, CMDWEPROTB protects sectors 0-255 in 8-sector increments (1 bit per 8 sectors), starting from BIT0 in CMDWEPROTB.

The CMDWEPROTC register configures the dynamic write protection for the MAIN region. For BANK0-4, CMDWEPROTC protects sectors 256-511 in 8-sector increments (1 bit per 8 sectors).

5.4.3.2 Configuring Protection for the NONMAIN Region

The CMDWEPROTNM register protects the NONMAIN region from program and erase. One protection bit is provided per sector. Devices only have one NONMAIN sector and it will be in BANK0.

5.5 Read Interface

The read interface provides the read path to the CPU subsystem (for instruction/data fetch), the read path to the peripheral bus (for use by the DMA controller or CPU), main bank address swapping, and detection and reporting of ECC SEC or DED errors.

5.5.1 Bank Address Swapping

Devices that contain more than one bank support swapping of the MAIN regions of the banks within the address space. This mechanism enables two versions of application firmware to be programmed into the device without the firmware needing to know which physical bank it exists in.

Table 5-10 gives an example of the mapping before and after a bank swap is requested for a device with a 512KB main flash split across 2 banks (256KB each).

Table 5-10. Bank Address Swap Translation

Bank and Region	Address Space Before Swap	Address Space After Swap
BANK0 MAIN	0x0000.0000 – 0x0003.FFFF	0x0004.0000 – 0x0007.FFFF
BANK1 MAIN	0x0004.0000 – 0x0007.FFFF	0x0000.0000 – 0x0003.FFFF

After a device reset, the MAIN region of the lower bank is always mapped to the lowest address space. The application software is responsible for determining if a bank address swap is to be applied. The bank address swap control is contained within the SYSCTL module; see the SYSCTL chapter for register and bit definitions. During an address swap, the application software must meet the following constraints:

1. The software must disable interrupts before issuing the bank swap command.
2. The software must poll the bank swap status after issuing the bank swap command before proceeding with execution.
3. If the swap command and poll status routines are executing from flash, they must exist at the exact same location in both banks such that execution resumes from where it left off after the bank swap. This restriction does not apply if the bank swap and status polling is done from SRAM and not from flash memory.

5.6 FLASHCTL Registers

Table 5-11 lists the memory-mapped registers for the FLASHCTL registers. All register offset addresses not listed in Table 5-11 should be considered as reserved locations and the register contents should not be modified.

Table 5-11. FLASHCTL Registers

Offset	Acronym	Register Name	Group	Section
1020h	IIDX	Interrupt Index Register		Go
1028h	IMASK	Interrupt Mask Register		Go
1030h	RIS	Raw Interrupt Status Register		Go
1038h	MIS	Masked Interrupt Status Register		Go
1040h	ISET	Interrupt Set Register		Go
1048h	ICLR	Interrupt Clear Register		Go
1100h	CMDEXEC	Command Execute Register		Go
1104h	CMDTYPE	Command Type Register		Go
1108h	CMDCTL	Command Control Register		Go
1120h	CMDADDR	Command Address Register		Go
1124h	CMDBYTEN	Command Program Byte Enable Register		Go
112Ch	CMDDATAINDEX	Command Data Index Register		Go
1130h	CMDDATA0	Command Data Register 0		Go
1134h	CMDDATA1	Command Data Register 1		Go
1138h	CMDDATA2	Command Data Register 2		Go
113Ch	CMDDATA3	Command Data Register Bits 127:96		Go
1140h	CMDDATA4	Command Data Register 4		Go
1144h	CMDDATA5	Command Data Register 5		Go
1148h	CMDDATA6	Command Data Register 6		Go
114Ch	CMDDATA7	Command Data Register 7		Go
1150h	CMDDATA8	Command Data Register 8		Go
1154h	CMDDATA9	Command Data Register 9		Go
1158h	CMDDATA10	Command Data Register 10		Go
115Ch	CMDDATA11	Command Data Register 11		Go
1160h	CMDDATA12	Command Data Register 12		Go
1164h	CMDDATA13	Command Data Register 13		Go
1168h	CMDDATA14	Command Data Register 14		Go
116Ch	CMDDATA15	Command Data Register 15		Go
1170h	CMDDATA16	Command Data Register 16		Go
1174h	CMDDATA17	Command Data Register 17		Go
1178h	CMDDATA18	Command Data Register 18		Go
117Ch	CMDDATA19	Command Data Register 19		Go
1180h	CMDDATA20	Command Data Register 20		Go
1184h	CMDDATA21	Command Data Register 21		Go
1188h	CMDDATA22	Command Data Register 22		Go
118Ch	CMDDATA23	Command Data Register 23		Go
1190h	CMDDATA24	Command Data Register 24		Go
1194h	CMDDATA25	Command Data Register 25		Go
1198h	CMDDATA26	Command Data Register 26		Go
119Ch	CMDDATA27	Command Data Register 27		Go
11A0h	CMDDATA28	Command Data Register 28		Go

Table 5-11. FLASHCTL Registers (continued)

Offset	Acronym	Register Name	Group	Section
11A4h	CMDDATA29	Command Data Register 29		Go
11A8h	CMDDATA30	Command Data Register 30		Go
11ACh	CMDDATA31	Command Data Register 31		Go
11B0h	CMDDATAECC0	Command Data Register ECC 0		Go
11B4h	CMDDATAECC1	Command Data Register ECC 1		Go
11B8h	CMDDATAECC2	Command Data Register ECC 2		Go
11BCh	CMDDATAECC3	Command Data Register ECC 3		Go
11C0h	CMDDATAECC4	Command Data Register ECC 4		Go
11C4h	CMDDATAECC5	Command Data Register ECC 5		Go
11C8h	CMDDATAECC6	Command Data Register ECC 6		Go
11CCh	CMDDATAECC7	Command Data Register ECC 7		Go
11D0h	CMDWEPROTA	Command Write Erase Protect A Register		Go
11D4h	CMDWEPROTB	Command Write Erase Protect B Register		Go
11D8h	CMDWEPROTC	Command Write Erase Protect C Register		Go
1210h	CMDWEPROTNM	Command Write Erase Protect Non-Main Register		Go
13B4h	CFGPCNT	Pulse Counter Configuration Register		Go
13D0h	STATCMD	Command Status Register		Go
13D4h	STATADDR	Address Status Register		Go
13D8h	STATPCNT	Pulse Count Status Register		Go

Complex bit access types are encoded to fit into small table cells. Table 5-12 shows the codes that are used for access types in this section.

Table 5-12. FLASHCTL Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value

5.6.1 IIDX (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 5-4 and described in Table 5-13.

Return to the Summary Table.

The interrupt index (IIDX) register provides the index of the highest priority pending and enabled interrupt.

Figure 5-4. IIDX

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

Table 5-13. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	STAT	R	0h	Index corresponding to the highest priority pending interrupt source. This value may be used as an address offset for fast, deterministic handling in the interrupt service routine. A read of the IIDX register will clear the corresponding interrupt status in the RIS and MIS registers. 0h (R/W) = No Interrupt Pending 1h (R/W) = DONE Interrupt Pending

5.6.2 IMASK (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 5-5 and described in Table 5-14.

Return to the Summary Table.

The interrupt mask (IMASK) register holds the current interrupt mask settings.

Figure 5-5. IMASK

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

Table 5-14. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Reserved
0	DONE	R/W	0h	Enable or disable the DONE interrupt. 0h (R/W) = Interrupt is masked out 1h (R/W) = Interrupt will request an interrupt service routine and corresponding bit in MIS will be set

5.6.3 RIS (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 5-6 and described in Table 5-15.

Return to the Summary Table.

The raw interrupt status (RIS) register holds the current raw interrupt status.

Figure 5-6. RIS

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

Table 5-15. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	DONE	R	0h	Raw status of the DONE interrupt. 0h (R/W) = Interrupt did not occur 1h (R/W) = Interrupt occurred

5.6.4 MIS (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 5-7 and described in Table 5-16.

Return to the Summary Table.

The masked interrupt status (MIS) register holds the current masked interrupt status.

Figure 5-7. MIS

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

Table 5-16. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	DONE	R	0h	Masked status of the DONE interrupt. 0h (R/W) = Masked interrupt did not occur 1h (R/W) = Masked interrupt occurred

5.6.5 ISET (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 5-8 and described in Table 5-17.

Return to the Summary Table.

The interrupt set (ISET) register may be used to set an interrupt to pending from software.

Figure 5-8. ISET

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

Table 5-17. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	Reserved
0	DONE	W	0h	Set the DONE interrupt. 0h (R/W) = Writing a 0 has no effect 1h (R/W) = Set RIS bit

5.6.6 ICLR (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 5-9 and described in Table 5-18.

Return to the Summary Table.

The interrupt clear (ICLR) register may be used to clear a pending interrupt.

Figure 5-9. ICLR

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

Table 5-18. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	Reserved
0	DONE	W	0h	Clear the DONE interrupt. 0h (R/W) = Writing a 0 has no effect 1h (R/W) = Clear RIS bit

5.6.7 CMDEXEC (Offset = 1100h) [Reset = 00000000h]

CMDEXEC is shown in Figure 5-10 and described in Table 5-19.

Return to the Summary Table.

Command Execute Register
Initiates execution of the command specified in the CMDTYPE register. This register is blocked for writes after being written to 1 and prior to STATCMD.DONE being set by hardware. Hardware clears this register after the processing of the command has completed.

Figure 5-10. CMDEXEC

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

Table 5-19. CMDEXEC Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	Reserved
0	VAL	R/W	0h	Command Execute value Initiates execution of the command specified in the CMDTYPE register. 0h (R/W) = Command will not execute or is not executing in hardware 1h (R/W) = Command will execute or is executing in hardware

5.6.8 CMDTYPE (Offset = 1104h) [Reset = 00000000h]

CMDTYPE is shown in Figure 5-11 and described in Table 5-20.

Return to the Summary Table.

Command Type Register
Specifies the type of command to be executed by hardware. This register is blocked for writes after CMDEXEC is written to a 1 and prior to STATCMD.DONE being set by the hardware to indicate that command execution has completed.

Figure 5-11. CMDTYPE

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED	SIZE			RESERVED	COMMAND		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		

Table 5-20. CMDTYPE Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R/W	0h	Reserved
6-4	SIZE	R/W	0h	Command size 0h (R/W) = Operate on 1 flash word 1h (R/W) = Operate on 2 flash words 2h (R/W) = Operate on 4 flash words 3h (R/W) = Operate on 8 flash words 4h (R/W) = Operate on a flash sector 5h (R/W) = Operate on an entire flash bank
3	RESERVED	R/W	0h	Reserved
2-0	COMMAND	R/W	0h	Command type 0h (R/W) = No Operation 1h (R/W) = Program 2h (R/W) = Erase 3h (R/W) = Read Verify - Perform a standalone read verify operation. 6h (R/W) = Blank Verify - Check whether a flash word is in the erased state. This command may only be used with CMDTYPE.SIZE = ONEWORD

5.6.9 CMDCTL (Offset = 1108h) [Reset = 00000000h]

CMDCTL is shown in Figure 5-12 and described in Table 5-21.

Return to the Summary Table.

Command Control Register This register configures specific capabilities of the state machine for related to the execution of a command. This register is blocked for writes after CMDEXEC is written to a 1 and prior to STATCMD.DONE being set by the hardware to indicate that command execution has completed.

Figure 5-12. CMDCTL

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED		RESERVED	SSERASEDIS	RESERVED		ECCGENOVR	ADDRXLATEOVR
R/W-0h		R/W-	R/W-0h	R/W-		R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		RESERVED	REGIONSEL				RESERVED
R/W-		R/W-0h	R/W-0h				R/W-
7	6	5	4	3	2	1	0
RESERVED			BANKSEL	RESERVED			
R/W-			R/W-0h	R/W-			

Table 5-21. CMDCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R/W	0h	Reserved
21	RESERVED	R/W	0h	
20	SSERASEDIS	R/W	0h	Disable Stair-Step Erase. If set, the default VHV trim voltage setting will be used for all erase pulses. By default, this bit is reset, meaning that the VHV voltage will be stepped during successive erase pulses. The step count, step voltage, begin and end voltages are all hard-wired. 0h (R/W) = Enable 1h (R/W) = Disable
19-18	RESERVED	R/W	0h	
17	ECCGENOVR	R/W	0h	Override hardware generation of ECC data for program. Use data written to CMDDATAECC*. 0h (R/W) = Do not override 1h (R/W) = Override
16	ADDRXLATEOVR	R/W	0h	Override hardware address translation of address in CMDADDR from a system address to the corresponding bank address and bank ID. When set, CMDADDR will be used directly as the bank address, CMDCTL.REGIONSEL will be used directly as the region ID, and CMDCTL.BANKSEL will be used directly as the bank ID (if the device contains multiple banks). 0h (R/W) = Do not override 1h (R/W) = Override
15-14	RESERVED	R/W	0h	
13	RESERVED	R/W	0h	Reserved
12-9	REGIONSEL	R/W	0h	Bank Region A specific region ID can be written to this field to indicate to which region an operation should be applied if CMDCTL.ADDRXLATEOVR is set. 1h (R/W) = Main Region 2h (R/W) = Non-Main Region

Table 5-21. CMDCTL Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-5	RESERVED	R/W	0h	
4	BANKSEL	R/W	0h	Bank Select A specific Bank ID can be written to this field to indicate to which bank an operation should be applied if CMDCTL.ADDRXLATEOVR is set. 1h (R/W) = Bank 0 2h (R/W) = Bank 1 4h (R/W) = Bank 2 8h (R/W) = Bank 3 10h (R/W) = Bank 4
3-0	RESERVED	R/W	0h	

5.6.10 CMDADDR (Offset = 1120h) [Reset = 00000000h]

CMDADDR is shown in Figure 5-13 and described in Table 5-22.

Return to the Summary Table.

Command Address Register:
This register forms the target address of a command. The use cases are as follows:
1) For single-word program, this address indicates the flash bank word to be programmed.
2) For multi-word program, this address indicates the first flash bank address for the program. The address will be incremented for further words.
3) For sector erase, this address indicates the sector to be erased.
4) For bank erase, the address indicates the bank to be erased.
5) For read verify, the address indications follow program/erase listed above.
Note the address written to this register will be submitted for translation to the flash address translation interface, and the translated address will be used to access the bank. However, if the CMDCTL.ADDRXLATEOVR bit is set, then the address written to this register will be used directly as the bank address.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.

Figure 5-13. CMDADDR

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
VAL																															
R/W-0h																															

Table 5-22. CMDADDR Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	0h	Address value 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.11 CMDBYTEN (Offset = 1124h) [Reset = 00000000h]

CMDBYTEN is shown in Figure 5-14 and described in Table 5-23.

Return to the Summary Table.

Command Program Byte Enable Register:
This register forms a per-byte enable for programming data. For data bytes to be programmed, a 1 must be written to the corresponding bit in this register. Normally, all bits are written to 1, allowing program of full flash words. However, leaving some bits 0 allows programming of 8-bit, 16-bit, 32-bit or 64-bit portions of a flash word. In addition, the read verify command will ignore data bytes read from the flash in its comparison if the corresponding CMDBYTEN bit is 0.
For 64-bit flash word size devices, the CMDBYTEN register uses BIT7-0 to enable each data byte and BIT8 to enable the ECC code byte.
For 128-bit flash word size devices, the CMDBYTEN register uses BIT15-0 to enable each data byte and BIT17-16 to enable each ECC code byte.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is written to all 0 after the completion of all commands.

Figure 5-14. CMDBYTEN

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
RESERVED																RESERVED								VAL							
R/W-0h																R/W-								R/W-0h							

Table 5-23. CMDBYTEN Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R/W	0h	Reserved
17-8	RESERVED	R/W	0h	
7-0	VAL	R/W	0h	Command Byte Enable value. A 1-bit per flash word byte value is placed in this register. 0h = Minimum value of [VAL] 0003FFFFh = Maximum value of [VAL]

5.6.12 CMDDATAINDEX (Offset = 112Ch) [Reset = 00000000h]

CMDDATAINDEX is shown in Figure 5-15 and described in Table 5-24.

Return to the Summary Table.

Command Program Data Index Register:
When multiple data registers are available for multi-word program, this register can be written with an index which points to one of the data registers. When a write to CMDDATA* is done, the data will be written to the physical data register indexed by the value in this register.
Up to 8 data registers can be present, so this register can be written with 0x0 to 0x7. If less than 8 data registers are present, successive MSB bits of this register are ignored when indexing the CMDDATA* registers.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.

Figure 5-15. CMDDATAINDEX

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
RESERVED																													VAL		
R/W-0h																													R/W-0h		

Table 5-24. CMDDATAINDEX Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R/W	0h	Reserved
2-0	VAL	R/W	0h	Data register index 0h = Minimum value of [VAL] 7h = Maximum value of [VAL]

5.6.13 CMDDATA0 (Offset = 1130h) [Reset = FFFFFFFFh]

CMDDATA0 is shown in Figure 5-16 and described in Table 5-25.

Return to the Summary Table.

Command Data Register 0
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 0.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-16. CMDDATA0

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-25. CMDDATA0 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.14 CMDDATA1 (Offset = 1134h) [Reset = FFFFFFFFh]

CMDDATA1 is shown in Figure 5-17 and described in Table 5-26.

Return to the Summary Table.

Command Data Register 1
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 0.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-17. CMDDATA1

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-26. CMDDATA1 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.15 CMDDATA2 (Offset = 1138h) [Reset = FFFFFFFFh]

CMDDATA2 is shown in Figure 5-18 and described in Table 5-27.

Return to the Summary Table.

Command Data Register 2
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 1.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-18. CMDDATA2

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-27. CMDDATA2 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.16 CMDDATA3 (Offset = 113Ch) [Reset = FFFFFFFFh]

CMDDATA3 is shown in Figure 5-19 and described in Table 5-28.

Return to the Summary Table.

Command Data Register 3
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 1.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-19. CMDDATA3

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-28. CMDDATA3 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.17 CMDDATA4 (Offset = 1140h) [Reset = FFFFFFFFh]

CMDDATA4 is shown in Figure 5-20 and described in Table 5-29.

Return to the Summary Table.

Command Data Register 4
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 2.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-20. CMDDATA4

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-29. CMDDATA4 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. T 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.18 CMDDATA5 (Offset = 1144h) [Reset = FFFFFFFFh]

CMDDATA5 is shown in Figure 5-21 and described in Table 5-30.

Return to the Summary Table.

Command Data Register 5
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 2.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-21. CMDDATA5

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-30. CMDDATA5 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.19 CMDDATA6 (Offset = 1148h) [Reset = FFFFFFFFh]

CMDDATA6 is shown in Figure 5-22 and described in Table 5-31.

Return to the Summary Table.

Command Data Register 6
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 3.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-22. CMDDATA6

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-31. CMDDATA6 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.20 CMDDATA7 (Offset = 114Ch) [Reset = FFFFFFFFh]

CMDDATA7 is shown in Figure 5-23 and described in Table 5-32.

Return to the Summary Table.

Command Data Register 7
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 3.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-23. CMDDATA7

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-32. CMDDATA7 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.21 CMDDATA8 (Offset = 1150h) [Reset = FFFFFFFFh]

CMDDATA8 is shown in Figure 5-24 and described in Table 5-33.

Return to the Summary Table.

Command Data Register 8
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 4.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-24. CMDDATA8

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-33. CMDDATA8 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.22 CMDDATA9 (Offset = 1154h) [Reset = FFFFFFFFh]

CMDDATA9 is shown in Figure 5-25 and described in Table 5-34.

Return to the Summary Table.

Command Data Register 9
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 4.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-25. CMDDATA9

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-34. CMDDATA9 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.23 CMDDATA10 (Offset = 1158h) [Reset = FFFFFFFFh]

CMDDATA10 is shown in Figure 5-26 and described in Table 5-35.

Return to the Summary Table.

Command Data Register 10
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 5.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-26. CMDDATA10

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-35. CMDDATA10 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.24 CMDDATA11 (Offset = 115Ch) [Reset = FFFFFFFFh]

CMDDATA11 is shown in Figure 5-27 and described in Table 5-36.

Return to the Summary Table.

Command Data Register 11
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 5.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-27. CMDDATA11

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-36. CMDDATA11 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.25 CMDDATA12 (Offset = 1160h) [Reset = FFFFFFFFh]

CMDDATA12 is shown in Figure 5-28 and described in Table 5-37.

Return to the Summary Table.

Command Data Register 12
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 6.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-28. CMDDATA12

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-37. CMDDATA12 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.26 CMDDATA13 (Offset = 1164h) [Reset = FFFFFFFFh]

CMDDATA13 is shown in Figure 5-29 and described in Table 5-38.

Return to the Summary Table.

Command Data Register 13
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 6.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-29. CMDDATA13

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-38. CMDDATA13 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.27 CMDDATA14 (Offset = 1168h) [Reset = FFFFFFFFh]

CMDDATA14 is shown in Figure 5-30 and described in Table 5-39.

Return to the Summary Table.

Command Data Register 14
This register contains the data for a command.
This register represents bits 31:0 of flash word data register 7.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-30. CMDDATA14

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-39. CMDDATA14 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.28 CMDDATA15 (Offset = 116Ch) [Reset = FFFFFFFFh]

CMDDATA15 is shown in Figure 5-31 and described in Table 5-40.

Return to the Summary Table.

Command Data Register 15
This register contains the data for a command.
This register represents bits 63:32 of flash word data register 7.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all NoWrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-31. CMDDATA15

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-40. CMDDATA15 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.29 CMDDATA16 (Offset = 1170h) [Reset = FFFFFFFFh]

CMDDATA16 is shown in Figure 5-32 and described in Table 5-41.

Return to the Summary Table.

Command Data Register 16
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 31:0 of flash word data register 4.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-32. CMDDATA16

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-41. CMDDATA16 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.30 CMDDATA17 (Offset = 1174h) [Reset = FFFFFFFFh]

CMDDATA17 is shown in Figure 5-33 and described in Table 5-42.

Return to the Summary Table.

Command Data Register 17
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 63:32 of flash word data register 4.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-33. CMDDATA17

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-42. CMDDATA17 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.31 CMDDATA18 (Offset = 1178h) [Reset = FFFFFFFFh]

CMDDATA18 is shown in Figure 5-34 and described in Table 5-43.

Return to the Summary Table.

Command Data Register 18
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 95:64 of flash word data register 4.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-34. CMDDATA18

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-43. CMDDATA18 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.32 CMDDATA19 (Offset = 117Ch) [Reset = FFFFFFFFh]

CMDDATA19 is shown in Figure 5-35 and described in Table 5-44.

Return to the Summary Table.

Command Data Register 19
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 127:96 of flash word data register 4.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-35. CMDDATA19

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-44. CMDDATA19 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.33 CMDDATA20 (Offset = 1180h) [Reset = FFFFFFFFh]

CMDDATA20 is shown in Figure 5-36 and described in Table 5-45.

Return to the Summary Table.

Command Data Register 20
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 31:0 of flash word data register 5.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-36. CMDDATA20

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-45. CMDDATA20 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.34 CMDDATA21 (Offset = 1184h) [Reset = FFFFFFFFh]

CMDDATA21 is shown in Figure 5-37 and described in Table 5-46.

Return to the Summary Table.

Command Data Register 21
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 63:32 of flash word data register 5.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-37. CMDDATA21

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-46. CMDDATA21 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.35 CMDDATA22 (Offset = 1188h) [Reset = FFFFFFFFh]

CMDDATA22 is shown in Figure 5-38 and described in Table 5-47.

Return to the Summary Table.

Command Data Register 22
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 95:64 of flash word data register 5.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-38. CMDDATA22

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-47. CMDDATA22 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.36 CMDDATA23 (Offset = 118Ch) [Reset = FFFFFFFFh]

CMDDATA23 is shown in Figure 5-39 and described in Table 5-48.

Return to the Summary Table.

Command Data Register 23
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 127:96 of flash word data register 5.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-39. CMDDATA23

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-48. CMDDATA23 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.37 CMDDATA24 (Offset = 1190h) [Reset = FFFFFFFFh]

CMDDATA24 is shown in Figure 5-40 and described in Table 5-49.

Return to the Summary Table.

Command Data Register 24
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 31:0 of flash word data register 6.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-40. CMDDATA24

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-49. CMDDATA24 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.38 CMDDATA25 (Offset = 1194h) [Reset = FFFFFFFFh]

CMDDATA25 is shown in Figure 5-41 and described in Table 5-50.

Return to the Summary Table.

Command Data Register 25
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 63:32 of flash word data register 6.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-41. CMDDATA25

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-50. CMDDATA25 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.39 CMDDATA26 (Offset = 1198h) [Reset = FFFFFFFFh]

CMDDATA26 is shown in Figure 5-42 and described in Table 5-51.

Return to the Summary Table.

Command Data Register 26
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 95:64 of flash word data register 6.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-42. CMDDATA26

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-51. CMDDATA26 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.40 CMDDATA27 (Offset = 119Ch) [Reset = FFFFFFFFh]

CMDDATA27 is shown in Figure 5-43 and described in Table 5-52.

Return to the Summary Table.

Command Data Register 27
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 127:96 of flash word data register 6.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-43. CMDDATA27

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-52. CMDDATA27 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.41 CMDDATA28 (Offset = 11A0h) [Reset = FFFFFFFFh]

CMDDATA28 is shown in Figure 5-44 and described in Table 5-53.

Return to the Summary Table.

Command Data Register 28
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 31:0 of flash word data register 7.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-44. CMDDATA28

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-53. CMDDATA28 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.42 CMDDATA29 (Offset = 11A4h) [Reset = FFFFFFFFh]

CMDDATA29 is shown in Figure 5-45 and described in Table 5-54.

Return to the Summary Table.

Command Data Register 29
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 63:32 of flash word data register 7.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-45. CMDDATA29

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-54. CMDDATA29 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.43 CMDDATA30 (Offset = 11A8h) [Reset = FFFFFFFFh]

CMDDATA30 is shown in Figure 5-46 and described in Table 5-55.

Return to the Summary Table.

Command Data Register 30
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 95:64 of flash word data register 7.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-46. CMDDATA30

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-55. CMDDATA30 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.44 CMDDATA31 (Offset = 11ACh) [Reset = FFFFFFFFh]

CMDDATA31 is shown in Figure 5-47 and described in Table 5-56.

Return to the Summary Table.

Command Data Register 31
This register forms the data for a command.
For DATAWIDTH == 128: This register represents bits 127:96 of flash word data register 7.
This register is blocked for writes after a 1 is written to the CMDEXEC
register and prior to STATCMD.DONE being set by the flash wrapper
hardware.
This register is used to aggregate masking for bits that do not
require additional program pulses during program operations, and will be
written to all 1 after the completion of all flash wrapper commands.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.

Figure 5-47. CMDDATA31

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-56. CMDDATA31 Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	A 32-bit data value is placed in this field. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.45 CMDDATAECC0 (Offset = 11B0h) [Reset = 0000FFFFh]

CMDDATAECC0 is shown in Figure 5-48 and described in Table 5-57.

Return to the Summary Table.

Command Data Register 0
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 0.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-48. CMDDATAECC0

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-57. CMDDATAECC0 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.46 CMDDATAECC1 (Offset = 11B4h) [Reset = 0000FFFFh]

CMDDATAECC1 is shown in Figure 5-49 and described in Table 5-58.

Return to the Summary Table.

Command Data Register 1
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 1.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-49. CMDDATAECC1

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-58. CMDDATAECC1 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.47 CMDDATAECC2 (Offset = 11B8h) [Reset = 0000FFFFh]

CMDDATAECC2 is shown in Figure 5-50 and described in Table 5-59.

Return to the Summary Table.

Command Data Register 2
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 2.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-50. CMDDATAECC2

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-59. CMDDATAECC2 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.48 CMDDATAECC3 (Offset = 11BCh) [Reset = 0000FFFFh]

CMDDATAECC3 is shown in Figure 5-51 and described in Table 5-60.

Return to the Summary Table.

Command Data Register 3
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 3.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-51. CMDDATAECC3

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-60. CMDDATAECC3 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.49 CMDDATAECC4 (Offset = 11C0h) [Reset = 0000FFFFh]

CMDDATAECC4 is shown in Figure 5-52 and described in Table 5-61.

Return to the Summary Table.

Command Data Register 4
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 4.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-52. CMDDATAECC4

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-61. CMDDATAECC4 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.50 CMDDATAECC5 (Offset = 11C4h) [Reset = 0000FFFFh]

CMDDATAECC5 is shown in Figure 5-53 and described in Table 5-62.

Return to the Summary Table.

Command Data Register 5
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 5.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-53. CMDDATAECC5

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-62. CMDDATAECC5 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.51 CMDDATAECC6 (Offset = 11C8h) [Reset = 0000FFFFh]

CMDDATAECC6 is shown in Figure 5-54 and described in Table 5-63.

Return to the Summary Table.

Command Data Register 6
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 6.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-54. CMDDATAECC6

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-63. CMDDATAECC6 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.52 CMDDATAECC7 (Offset = 11CCh) [Reset = 0000FFFFh]

CMDDATAECC7 is shown in Figure 5-55 and described in Table 5-64.

Return to the Summary Table.

Command Data Register 7
This register forms the ECC portion of the data for a command. This ECC data in this register covers flash data register 7.
The hardware ECC generation can be overridden and ECC data developed elsewhere can be used. ECC data is placed in this register.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
Use cases for the CMDDATA* registers are as follows:
1) Program - These registers contain the data to be programmed.
2) Erase - These registers are not used.
3) Read Verify - These registers contain data to be verified.
This register is used to aggregate masking for bits that do not require additional program pulses during program operations, and will be written to all 1 after the completion of all commands.

Figure 5-55. CMDDATAECC7

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
RESERVED																VAL1								VAL0							
R/W-0h																R/W-FFh								R/W-FFh							

Table 5-64. CMDDATAECC7 Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-8	VAL1	R/W	FFh	ECC data for bits 127:64 of the data is placed here. 0h = Minimum value FFh = Maximum value
7-0	VAL0	R/W	FFh	ECC data for bits 63:0 of the data is placed here. 0h = Minimum value FFh = Maximum value

5.6.53 CMDWEPROTA (Offset = 11D0h) [Reset = FFFFFFFFh]

CMDWEPROTA is shown in Figure 5-56 and described in Table 5-65.

Return to the Summary Table.

Command WriteErase Protect A Register
This register allows the first 32 sectors of the main region to be protected from program or erase, with 1 bit protecting each sector. If the main region size is smaller than 32 sectors, then this register provides protection for the whole region.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
In addition, this register is used to aggregate masking for sectors that do not require additional erase pulses during bank erase operations, and will be written to all 1 after the completion of all commands.

Figure 5-56. CMDWEPROTA

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-65. CMDWEPROTA Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	Each bit protects 1 sector. bit [0]: When 1, sector 0 of the flash memory will be protected from program and erase. bit [1]: When 1, sector 1 of the flash memory will be protected from program and erase. : : bit [31]: When 1, sector 31 of the flash memory will be protected from program and erase. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.54 CMDWEPROTB (Offset = 11D4h) [Reset = FFFFFFFFh]

CMDWEPROTB is shown in Figure 5-57 and described in Table 5-66.

Return to the Summary Table.

Command WriteErase Protect B Register
This register allows main region sectors to be protected from program and erase. Each bit corresponds to a group of 8 sectors. There are 3 cases for how these protect bits are applied:
1. Single-bank system:
In the case where only a single flash bank is present, the first 32 sectors are protected via the CMDWEPROTA register. Thus, the protection give by the bits in CMDWEPROTB begin with sector 32.
2. Multi-bank system, Bank 0:
When multiple flash banks are present, the first 32 sectors of bank 0 are protected via the CMDWEPROTA register. Thus, only bits 4 and above of CMDWEPROTB would be applicable to bank 0. The protection of bit 4 and above would begin at sector 32. Bits 3:0 of WEPROTB are ignored for bank 0.
3. Multi-bank system, Banks 1-N:
For banks other than bank 0 in a multi-bank system, CMDWEPROTA has no effect, so the bits in CMDWEPROTB will protect these banks starting from sector 0.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
In addition, this register is used to aggregate masking for sectors that do not require additional erase pulses during bank erase operations, and will be written to all 1 after the completion of all commands.

Figure 5-57. CMDWEPROTB

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-66. CMDWEPROTB Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	Each bit protects a group of 8 sectors. When a bit is 1, the associated 8 sectors in the flash will be protected from program and erase. A maximum of 256 sectors can be protected with this register. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.55 CMDWEPROTC (Offset = 11D8h) [Reset = FFFFFFFFh]

CMDWEPROTC is shown in Figure 5-58 and described in Table 5-67.

Return to the Summary Table.

Command WriteErase Protect C Register
This register allows main region sectors to be protected from program and erase. Each bit corresponds to a group of 8 sectors.
This register extends the protection bits from the CMDWEPROTB register to cover bank sizes larger than 32*8=256 sectors. This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
In addition, this register is used to aggregate masking for sectors that do not require additional erase pulses during bank erase operations, and will be written to all 1 after the completion of all commands.

Figure 5-58. CMDWEPROTC

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-67. CMDWEPROTC Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	Each bit protects a group of 8 sectors. When a bit is 1, the associated 8 sectors in the flash will be protected from program and erase. Note that the sectors protected with this register start at sector 256 in the flash, where the sectors protected by the CMDWEPROTB register end. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.56 CMDWEPROTNM (Offset = 1210h) [Reset = FFFFFFFFh]

CMDWEPROTNM is shown in Figure 5-59 and described in Table 5-68.

Return to the Summary Table.

Command WriteErase Protect Non-Main Register
This register allows nonmain region sectors to be protected from program and erase. Each bit corresponds to 1 sector.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.
In addition, this register is used to aggregate masking for sectors that do not require additional erase pulses during bank erase operations, and will be written to all 1 after the completion of all commands.

Figure 5-59. CMDWEPROTNM

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
VAL																															
R/W-FFFFFFFFh																															

Table 5-68. CMDWEPROTNM Field Descriptions

Bit	Field	Type	Reset	Description
31-0	VAL	R/W	FFFFFFFFh	Each bit protects 1 sector. bit [0]: When 1, sector 0 of the nonmain region will be protected from program and erase. bit [1]: When 1, sector 1 of the nonmain region will be protected from program and erase. : : bit [31]: When 1, sector 31 of the nonmain will be protected from program and erase. 0h = Minimum value of [VAL] FFFFFFFFh = Maximum value of [VAL]

5.6.57 CFGPCNT (Offset = 13B4h) [Reset = 00000000h]

CFGPCNT is shown in Figure 5-60 and described in Table 5-69.

Return to the Summary Table.

Pulse Counter Configuration Register
This register allows further configuration of maximum pulse counts for program and erase operations.
This register is blocked for writes after a 1 is written to the CMDEXEC register and prior to STATCMD.DONE being set by hardware.

Figure 5-60. CFGPCNT

31	30	29	28	27	26	25	24
MAXERSPCNTVAL							
R/W-0h							
23	22	21	20	19	18	17	16
MAXERSPCNTVAL				RESERVED			MAXERSPCNTOVR
R/W-0h				R/W-0h			R/W-0h
15	14	13	12	11	10	9	8
RESERVED				MAXPCNTVAL			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
MAXPCNTVAL				RESERVED			MAXPCNTOVR
R/W-0h				R/W-0h			R/W-0h

Table 5-69. CFGPCNT Field Descriptions

Bit	Field	Type	Reset	Description
31-20	MAXERSPCNTVAL	R/W	0h	Override maximum pulse count for erase with this value. If MAXERSPCNTOVR = 0, then this field is ignored. If MAXERSPCNTOVR = 1, then this value will be used to override the max pulse count for erase. 0h = Minimum value FFFh = Maximum value
19-17	RESERVED	R/W	0h	Reserved
16	MAXERSPCNTOVR	R/W	0h	Override hard-wired maximum pulse count for erase. If set, then the value in MAXERSPCNTVAL will be used as the max pulse count for erase operations. By default, this bit is 0, and a hard-wired max pulse count is used. 0h = Use hard-wired (default) value for maximum pulse count 1h = Use value from MAXERSPCNTVAL field as maximum erase pulse count
15-12	RESERVED	R/W	0h	Reserved
11-4	MAXPCNTVAL	R/W	0h	Override maximum pulse counter with this value. If MAXPCNTOVR = 0, then this field is ignored. If MAXPCNTOVR = 1 and MAXERSPCNTOVR = 0, then this value will be used to override the max pulse count for both program and erase. Full max value will be {4'h0, MAXPCNTVAL} . If MAXPCNTOVR = 1 and MAXERSPCNTOVR = 1, then this value will be used to override the max pulse count for program only. Full max value will be {4'h0, MAXPCNTVAL}. 0h = Minimum value FFh = Maximum value
3-1	RESERVED	R/W	0h	Reserved

Table 5-69. CFGPCNT Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	MAXPCNTOVR	R/W	0h	Override hard-wired maximum pulse count. If MAXERSPCNTOVR is not set, then setting this value alone will override the max pulse count for both program and erase. If MAXERSPCNTOVR is set, then this bit will only control the max pulse count setting for program. By default, this bit is 0, and a hard-wired max pulse count is used. 0h = Use hard-wired (default) value for maximum pulse count 1h = Use value from MAXPCNTVAL field as maximum pulse count

5.6.58 STATCMD (Offset = 13D0h) [Reset = 00000000h]

STATCMD is shown in Figure 5-61 and described in Table 5-70.

Return to the Summary Table.

Command Status Register This register contains status regarding completion and errors of command execution.

Figure 5-61. STATCMD

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			FAILMISC	RESERVED			RESERVED
R-0h			R-0h	R-0h			R-
7	6	5	4	3	2	1	0
FAILMODE	FAILILLADDR	FAILVERIFY	FAILWEPROT	RESERVED	CMDINPROGRESS	CMDPASS	CMDDONE
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-70. STATCMD Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	FAILMISC	R	0h	Command failed due to error other than write/erase protect violation or verify error. This is an extra bit in case a new failure mechanism is added which requires a status bit. 0h = No Fail 1h = Fail
11-9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	
7	FAILMODE	R	0h	Command failed because a bank has been set to a mode other than READ. Program and Erase commands cannot be initiated unless all banks are in READ mode. 0h = No Fail 1h = Fail
6	FAILILLADDR	R	0h	Command failed due to the use of an illegal address 0h = No Fail 1h = Fail
5	FAILVERIFY	R	0h	Command failed due to verify error 0h = No Fail 1h = Fail
4	FAILWEPROT	R	0h	Command failed due to Write/Erase Protect Sector Violation 0h = No Fail 1h = Fail
3	RESERVED	R	0h	Reserved
2	CMDINPROGRESS	R	0h	Command In Progress 0h = Complete 1h = In Progress
1	CMDPASS	R	0h	Command Pass - valid when CMD_DONE field is 1 0h = Fail 1h = Pass

Table 5-70. STATCMD Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CMDDONE	R	0h	Command Done 0h = Not Done 1h = Done

5.6.59 STATADDR (Offset = 13D4h) [Reset = 00010000h]

STATADDR is shown in Figure 5-62 and described in Table 5-71.

Return to the Summary Table.

Current Address Counter Value Read only register giving read access to the state machine current address. A bank id, region id and address are stored in this register and are incremented as necessary during execution of a command.

Figure 5-62. STATADDR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED						BANKID					REGIONID				
R-0h						R-0h					R-1h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BANKADDR															
R-0h															

Table 5-71. STATADDR Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25-21	BANKID	R	0h	Current Bank ID A bank indicator is stored in this register which represents the current bank on which the state machine is operating. There is 1 bit per bank. 1h (R/W) = Bank 0 2h (R/W) = Bank 1 4h (R/W) = Bank 2 8h (R/W) = Bank 3 10h (R/W) = Bank 4
20-16	REGIONID	R	1h	Current Region ID A region indicator is stored in this register which represents the current flash region on which the state machine is operating. 1h (R/W) = Main Region 2h (R/W) = Non-Main Region 4h (R/W) = Trim Region 8h (R/W) = Engr Region
15-0	BANKADDR	R	0h	Current Bank Address A bank offset address is stored in this register. 0h = Minimum value FFFFh = Maximum value

5.6.60 STATPCNT (Offset = 13D8h) [Reset = 00000000h]

STATPCNT is shown in Figure 5-63 and described in Table 5-72.

Return to the Summary Table.

Current Pulse Count Register: Read only register giving read access to the state machine current pulse count value for program/erase operations.

Figure 5-63. STATPCNT

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
RESERVED																				PULSECNT											
R-0h																				R-0h											

Table 5-72. STATPCNT Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11-0	PULSECNT	R	0h	Current Pulse Counter Value 0h = Minimum value FFFh = Maximum value

Chapter 6

Events

The event manager provides the peripheral-to-peripheral, peripheral-to-DMA, and peripheral-to-CPU (IRQ) event connections.

6.1 Events Overview

The event manager transfers digital events from one entity (for example, a peripheral) to another (for example, a second peripheral, the DMA, or the CPU). The event manager implements event transfer through a defined set of event publishers (generators) and subscribers (receivers) which are interconnected through an event fabric containing a combination of fixed (static) and programmable routes.

Events which are transferred by the event manager include:

- Peripheral event transferred to the CPU as an interrupt request (IRQ)
  - Example: RTC interrupt is sent to the CPU
- Peripheral event transferred to the DMA as a DMA trigger
  - Example: UART data receive trigger to DMA to request a DMA transfer
- Peripheral event transferred to another peripheral to directly trigger an action in hardware
  - Example: TIMx timer peripheral publishes a periodic event to the ADC subscriber port, and the ADC uses the event to trigger start-of-sampling

In addition to providing the event transfer logic, the event manager also interfaces with the power management and clock unit (PMCU) if an event requires the power and/or clock configuration of the device to change to handle the event properly. For example, if a peripheral asserts an event that targets the DMA, and the device is in a STOP or STANDBY operating mode (DMA is disabled), the event manager will handshake with the PMCU to suspend the low power operating mode state temporarily and enable the DMA such that the DMA transfer can be processed.

The event manager configuration is device dependent, as different devices support different peripherals. See the device-specific data sheet for information on the device-specific event implementation.

6.1.1 Event Publisher

An event publisher is the source of an event which is propagated on the event fabric. Peripherals contain event publishers for publishing CPU interrupts, DMA triggers and generic events to the event fabric through the publishing port FPUB_x. Publisher behavior is configured with standardized event management registers.

6.1.2 Event Subscriber

Event subscribers are included within the processor, the DMA, and certain peripherals (see Section 6.1.4). Event subscribers subscribe the events through the subscribing port FSUB_x. Event subscribers enable modules to be able to subscribe to, and take a predefined action upon, events which are published to the event fabric by an event publisher.

6.1.3 Event Fabric Routing

There are three different types of routes through the event fabric which are used to connect a publisher to a subscriber: CPU interrupt events, DMA trigger events, and generic events.

6.1.3.1 CPU Interrupt Event Route (CPU_INT)

A CPU interrupt event route is a fixed, point-to-point connection between one event publisher (inside a peripheral module) and one event subscriber (the CPU subsystem) used to propagate CPU interrupts.

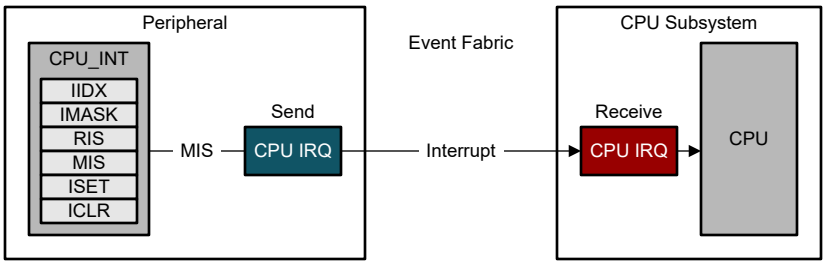


Figure 6-1. CPU Interrupt (Fixed Event Route)

For each peripheral which is capable of generating a CPU interrupt, a fixed route is provided from the peripheral's masked interrupt status (MIS) register to the CPU subsystem's interrupt management logic.

If software does not clear the interrupt request in the peripheral's event management registers, the request will remain pending to the CPU subsystem. See Section 6.2.5.3 for guidance on setting and clearing interrupt status with the event management registers.

6.1.3.2 DMA Trigger Event Route (DMA_TRIGx)

A DMA route is a fixed route between a peripheral and the DMA controller, which optionally has additional side-band signals to pass a DMA done condition from the DMA controller back to the triggering peripheral to indicate when a DMA activity has run to completion. The DMA trigger route is shown in Figure 6-2.

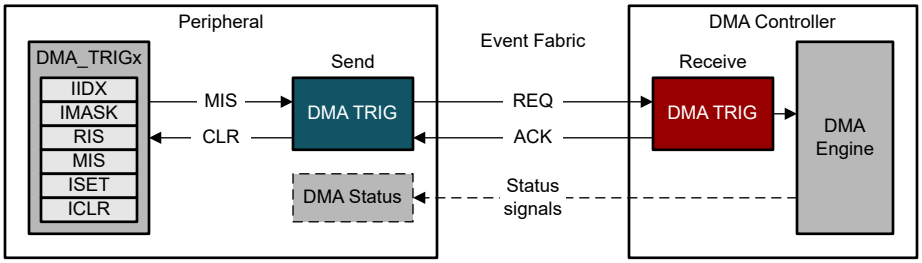


Figure 6-2. DMA Route

Most peripherals capable of generating a DMA trigger have an additional set of event management registers (in addition to the CPU_INT registers used for the CPU interrupt and any GEN_EVENTx generic route publishers). These registers can be used to select which peripheral condition to use for generating the DMA trigger.

When a trigger is received by the DMA, the DMA acknowledges the request and the peripheral clears the request. The DMA also acknowledges the cleared request, after which a new request can be asserted by the peripheral.

The DMA route can also contain status signals (for specific peripherals) to indicate to the triggering peripheral that a DMA transfer sequence has completed. For example, the DMA can be set up to transfer *N* number of bytes from an SRAM buffer into the UART TX data register based on the UART TX DMA trigger. Upon each trigger from the UART, the DMA will acknowledge that the transfer was successful. On the *N*th byte, the DMA will send a complete status signal to the UART, which the UART can use to propagate a transfer completion interrupt to the CPU.

Special Cases

Certain peripherals (for example, the 12-bit DAC) do not implement an event management register set for managing their DMA triggers. In these cases, the peripheral implements specific DMA configuration logic such that the management registers are not needed to interface with the DMA. Figure 6-3 shows the model when the event management registers are not implemented. See the peripheral-specific section of this document for guidance on how to configure DMA channels in this case.

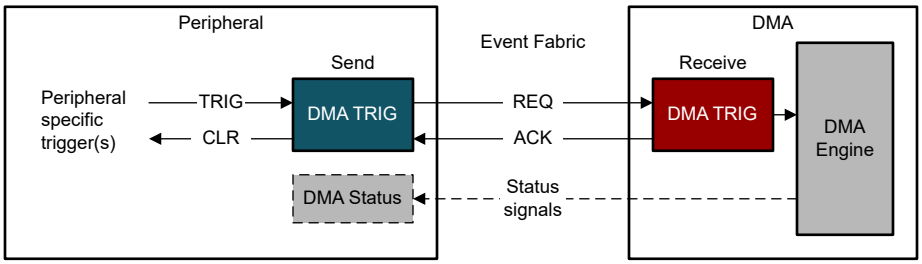


Figure 6-3. DMA Route without Event Management Registers

6.1.3.3 Generic Event Route (GEN_EVENTx)

A generic route is either a point-to-point (1:1) route or a point-to-two (1:2) splitter route in which the peripheral publishing the event uses one of several available generic route channels to publish its event to another entity (or entities, in the case of a splitter route), where an entity can be another peripheral, a generic DMA trigger event, or a generic CPU event, as shown in Figure 6-4.

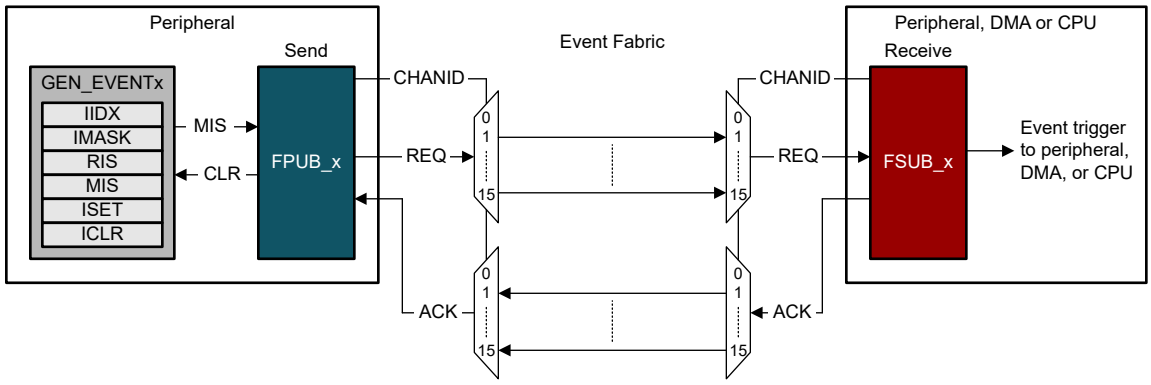


Figure 6-4. Generic Route

Peripherals capable of generating a generic event have an additional group (our groups) of GEN_EVENTx event management registers (in addition to the CPU_INT registers used for the CPU interrupt or DMA_TRIGx for DMA, if present). These registers can be used to select the peripheral condition to use for publishing a generic event. When configured, the event will broadcast out to the generic route channel selected by the FPUB_x register. A second peripheral, the DMA, or the CPU can subscribe to this event by configuring its generic subscriber port (FSUB_x) to listen on the same generic route channel to which the publishing peripheral is connected to.

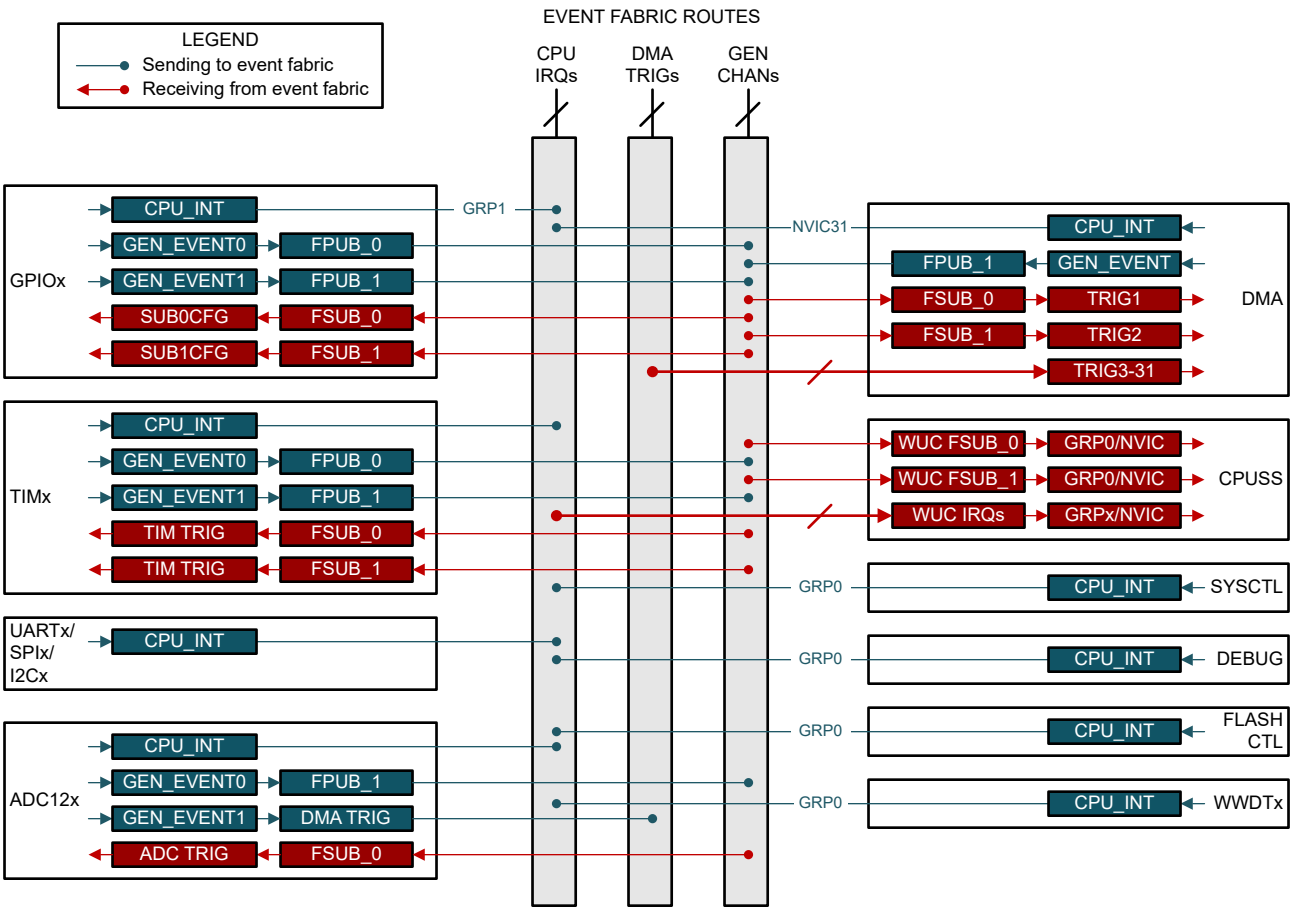
Generic route channels can be configured with one subscriber (1:1 route) or two subscribers (1:2 splitter route), depending on which channel is selected. See the device data sheet for a complete listing of the available generic route channels and their type (1:1 or 1:2). Generic route channels can only be configured with one publishing peripheral at a time. Once a peripheral subscribes to a 1:1 generic route channel, no other peripheral will be able to select that channel to subscribe to, unless the originally connected peripheral is disconnected first. Generic route channels with splitter capability (1:2) enable exactly two peripherals to subscribe to the channel, after which additional attempts to add subscribers will be blocked by hardware until both of the two connected peripherals are disconnected from the splitter channel.

Each peripheral type has unique capabilities in terms of what can generate an event to publish, and what a subscribed event is capable of triggering within the peripheral. Review the chapter of this guide which corresponds to the peripheral of interest to understand what the publisher and subscriber ports on a given peripheral are capable of.

6.1.4 Event Routing Map

The event capabilities of each peripheral type are shown in Figure 6-5. Peripherals such as UART, SPI, and I2C generate CPU interrupt events which are routed to the CPU, and they also generate DMA trigger events which are routed to the DMA. Peripherals such as GPIO and ADC generate CPU interrupt events as well, but they also support generating and receiving events routed through a generic channel. For example, through the use of a generic event channel, it is possible to directly start an ADC conversion from a GPIO event by connecting a GPIO FPUB_x and ADC FSUB_0 to the same generic event channel.

Figure 6-5. Event Map



6.1.5 Event Propagation Latency

Generic route channels implement a four-way hardware handshake between the publishing entity and the subscribing entity. This handshake requires four ULPCLK cycles to complete:

1. Request from publisher to subscriber
2. Acknowledge from subscriber to publisher
3. De-assert of request from publisher to subscriber
4. Acknowledge of de-assert from subscriber to publisher

If the publishing peripheral sends two requests and the first request has not cleared the handshake, the second request is dropped.

6.2 Events Operation

This section describes how to configure peripherals to use the event manager. Note that the event manager does not contain any configuration registers. All event configuration is done through the publishing and subscribing peripherals.

6.2.1 CPU Interrupt

Peripheral interrupt requests (IRQs) are propagated to the CPU subsystem through the event manager. Peripheral interrupt requests use fixed routes, but in addition the CPU subsystem may provide two generic event subscriber ports which can be used to trigger CPU interrupts through a generic route. See the device-specific data sheet for the complete list of interrupt assignments for a given device.

Standard CPU Interrupt Requests

No special configuration is required for fixed route interrupts. Interrupts can be managed through the peripheral's CPU_INT event management registers (IIDX, IMASK, RIS, MIS, ISET, and ICLR) and through the CPU subsystem interrupt configuration (see Section 3.3).

Generic Event Based CPU Interrupt Requests

The CPU subsystem contains two generic event subscriber ports (FSUB_x) which can be used to source a CPU interrupt from any of the device's generic event channels. This can be used to enable special cases where a particular function on a peripheral generates a dedicated interrupt to the CPU subsystem which is independent from, and in addition to, that peripheral's standard interrupt mechanism.

Consider the GPIO peripheral, which has a standard interrupt request as well as 2 publishers which can route a GPIO event to any of the generic event channels based on a defined state in the GPIO. For example, it can be desirable to have most GPIO events configured to source the standard interrupt, while a single specific GPIO event sources a second dedicated CPU interrupt through a generic route. This enables the application software to have two completely independent interrupt handlers for the GPIO.

To configure the event manager to trigger a CPU interrupt from a generic route, follow the steps below:

1. Configure the GEN_EVENT registers of the peripheral generating the event to select the desired peripheral state as an event generator.
2. Configure the FPUB_x register of the peripheral generating the event with the generic route channel ID which is to be used. This channel must not be in use by another peripheral.
3. Configure the FSUB_x register of the wake up controller (WUC), which captures generic route channel events to forward to the CPU subsystem.
4. Configure the CPU subsystem interrupt management to enable the GENSUBx interrupt.

Note that when generating a CPU interrupt through a generic route, the generic event logic will automatically clear the pending interrupt request as a part of the four-way event handshake. Application software will not be able to read the cause of the interrupt from the peripheral registers, and it does not need to clear any interrupt status bits. Software can only read that the FSUB_x generic event generated an interrupt. This reduces the interrupt overhead.

6.2.2 DMA Trigger

DMA triggers are propagated to the DMA through the event manager. Most DMA triggers use fixed routes, but the DMA does provide two generic event subscriber ports which can be used to trigger DMA transfers through a generic route channel. See the device-specific data sheet for the complete list of DMA trigger assignments for a given device.

Standard DMA Triggers

To determine if a particular peripheral on a device provides a fixed DMA trigger (DMA_TRIGx) from the peripheral directly to the DMA, review the DMA triggers table in the detailed description section of the device-specific data sheet. Certain peripherals can have more than one DMA trigger (for example, to enable a TX trigger and an RX trigger on a serial communication peripheral).

To select the specific peripheral event which triggers a static DMA route, configure the peripheral's DMA_TRIGx event management register set (IIDX, IMASK, RIS, MIS, ISET, and ICLR) which corresponds to the targeted DMA route. To determine which DMA_TRIGx register set corresponds with which DMA trigger, review the relevant chapter of this guide for the corresponding peripheral, or review Section 6.1.4.

Certain peripherals (such as the 12-bit DAC) do not implement a DMA_TRIGx register set for managing DMA triggers. In these cases, the DMA trigger configuration is done through peripheral-specific configuration registers.

Generic Event Based DMA Triggers

The DMA contains two generic event subscriber ports (FSUB_x) which can be used to source a DMA trigger from any of the device's generic event channels. This can be used to enable special cases where a particular function on a peripheral generates a DMA trigger. For example, it can be desirable to trigger a DMA transfer from a timer.

To configure the event manager to trigger a DMA channel from a generic route, follow the steps below:

1. Configure the GEN_EVENTx registers of the peripheral generating the event to select the desired peripheral state as an event generator.
2. Configure the FPUB_x register of the peripheral generating the event with the generic event channel ID which is to be used. This channel must not be in use by another peripheral.
3. Configure the FSUB_x register of the DMA, which captures generic event channel events to be used as triggers in the DMA.
4. Configure the DMA according to the configuration instructions in the DMA chapter.

6.2.3 Peripheral to Peripheral Event

Peripheral to peripheral events enable a condition in one peripheral to trigger an action in a second (or third) peripheral, completely in hardware without any CPU interaction. The device provides a certain number of generic route channels which can be either published to or subscribed to by peripherals which include publisher and subscriber ports. Before establishing a configuration, follow these steps:

1. Review the device specific data sheet to determine the generic route channel count and channel type available on the target device. Select an appropriate channel type (point to point or splitter) based on the desired functionality, and determine the channel number to use for the connection (the channel must not already be used by other peripherals).
2. Review the publisher and subscriber capabilities of the peripherals which are to be connected. Some peripherals have more than one publisher and/or more than one subscriber port, and some peripherals have no publisher or subscriber ports. To understand the available ports for a peripheral, review the peripheral's reference chapter in this guide, or check the generic event channel connections in Section 6.1.4.

Once the channel to be used is determined, and both the publisher and subscriber ports for the peripherals being connected are known, use the steps below to establish the event connection. In this example, a timer triggered ADC application will be configured, using TIMG0 to publish an event to generic channel 1, with ADC0 subscribing to generic channel 1 as a start-of-conversion trigger.

1. Configure the GEN_EVENTx event management registers of TIMG0 to set the event request based on the appropriate timer event (for example, a zero event).
2. Store 0x1 into the FPUB_0 register of TIMG0 to publish the TIMG0 event selected by the GEN_EVENTx registers to generic route channel 1. Channel 1 must not be in use by another peripheral.
3. Store 0x1 the FSUB_0 register of ADC0 so that ADC0 is listening for events published by the timer to channel 1.
4. Configure ADC0 to trigger from the subscriber port according to the configuration instructions in Section 9.2.8.
5. Configure and enable TIMG0.

6.2.4 Extended Module Description Register

The DESC_EX register is a read-only register in the event manager which can be read by application software to determine how many point to point (single) generic route channels and splitter (dual) generic route channels are available on a given device.

6.2.5 Using Event Registers

The event management register group is a set of standard registers which are implemented by all peripherals capable of generating events (CPU interrupts, DMA triggers, or generic events). Each event generator in a peripheral contains its own event management register set. For example, if a peripheral supports generating a CPU interrupt and a DMA trigger, it will have an event management register set for the CPU interrupt (with the

group name of CPU_INT) as well as a second event management register set for the DMA trigger (with the group name of DMA_TRIG).

The event management registers are used to:

- Configure which peripheral conditions are used to generate the event (masking)
- Communicate raw and masked peripheral event status
- Set or clear peripheral event status by software

In the "Registers" section for a given peripheral, the "Group" column displays the Group Name to indicate what functionality is mapped to each event management register group. See Table 6-1 for which event management groups are mapped to specific functions in the group name for a peripheral's "Registers" section.

Table 6-1. Event Management Group Functionality and Mapping

Group Name (in Registers)	Functionality
CPU_INT	CPU interrupt (fixed route to the CPU subsystem)
DMA_TRIGx	DMA trigger (fixed route to the DMA controller)
GEN_EVENT	Generic event (programmable route for other module-to-module connections)

6.2.5.1 Event Registers

The event management register set contains 6 standard registers: RIS, IMASK, MIS, ISET, ICLR, and IIDX, given in Table 6-2. The event registers are interconnected as shown in Figure 6-6.

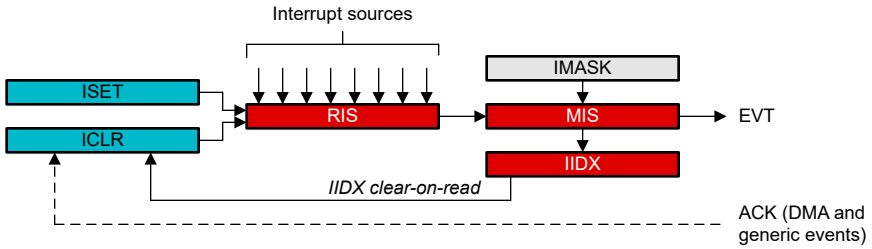


Figure 6-6. Event Management Register Relationship

The peripheral generating the event will contain one or more interrupt source signals which connect to the raw interrupt status (RIS) register. Software can poll RIS at any time to check the raw interrupt status. Software can also clear pending interrupts in the RIS register by writing to the corresponding bit position in the ICLR register. The RIS and IMASK registers are combined through a bit-wise AND function in the MIS register (masked interrupt status). To unmask an interrupt, set the corresponding bit in the IMASK register. Once unmasked, a pending interrupt will be indicated in both the RIS and MIS registers, and an event will be generated. The IIDX register will also be updated with the index of the highest priority pending interrupt.

In the case of a CPU interrupt (CPU_INT) with a CPU interrupt event route, a read of the IIDX register will clear the highest priority pending interrupt in the RIS and MIS registers and return the index of the highest priority pending interrupt to application software.

In the case of a hardware event DMA trigger route (DMA_TRIGx) or generic event route (GEN_EVENTx), the hardware four-way handshake will send an ACK signal to the ICLR mechanism which will clear the pending interrupt in the RIS and MIS registers.

Table 6-2. Standardized Event Management Registers (Used for CPU_INT, DMA_TRIGx, GEN_EVENTx Configuration)

Register	Description	R/W	Functionality
RIS	Raw interrupt status	R	Indicates the current pending interrupt status, with one bit provided per interrupt condition. Writing to ICLR will clear the corresponding bit in the RIS register if the interrupt condition is no longer present.

Table 6-2. Standardized Event Management Registers (Used for CPU_INT, DMA_TRIGx, GEN_EVENTx Configuration) (continued)

Register	Description	R/W	Functionality
IMASK	Interrupt mask	RW	Used by application software to configure which interrupt conditions propagate into an event, with one bit provided per interrupt condition.
MIS	Masked interrupt status	R	Indicates the current pending masked interrupt status to software and hardware, with one bit provided per interrupt condition. MIS is the bit-wise AND of the RIS and IMASK registers. Writing to ICLR will clear the corresponding bit in the RIS register if the interrupt condition is no longer present. If RIS is cleared, the corresponding bit in the MIS register is also automatically cleared.
ISET	Software interrupt set control	W	Used by application software to force an interrupt condition for diagnostics. Writing to ISET will set the corresponding bit in the RIS register. If the interrupt condition is enabled in IMASK, the corresponding bit in the MIS register is also set. Writing a '1' to a bit location in ISET sets the respective interrupt status.
ICLR	Software interrupt clear control	W	Used by application software to clear a pending interrupt status in RIS. Writing a '1' to a bit location in ICLR clears the respective interrupt status. If an interrupt is enabled in IMASK, the corresponding bit location in MIS is also cleared automatically when RIS clears. If the interrupt condition is still present, clearing the status has no effect and the RIS will remain set.
IIDX	Pending interrupt index	R	Used by application software to read the highest priority pending interrupt while simultaneously clearing the highest priority interrupt status in RIS and MIS. A read of IIDX returns 0 if no unmasked interrupts are pending (MIS==0), else it returns an index value indicating the highest priority pending interrupt.

6.2.5.2 Configuring Events

To configure which peripheral interrupt source is to be used to trigger an event, set the bit which corresponds to the desired interrupt source in the IMASK register which corresponds to the desired event. Setting a bit in IMASK will cause the raw interrupt status in the RIS register to propagate to the MIS register. When an interrupt status bit in the MIS register is set (due to the interrupt being unmasked in the IMASK register and a raw interrupt being pending in the RIS register), an event is generated.

Multiple interrupt sources can be enabled for CPU interrupt events, as application software can determine the cause of the interrupt by reading the IIDX or MIS register.

For hardware events such as DMA triggers and generic event publishers, only one interrupt source should be unmasked in IMASK.

6.2.5.3 Responding to CPU Interrupts in Application Software

In the case of an event which generates a CPU interrupt, application software can determine which peripheral interrupt triggered the generation of the event by either reading the IIDX register or by reading the MIS and writing the ICLR registers.

CPU IRQ Interrupt Service Routine using CPU_INT IIDX Register

Application software can read the IIDX register in CPU_INT group to determine and clear the highest priority pending interrupt. A read to IIDX will return an index corresponding to the highest priority interrupt which was both set and unmasked. The read action will also simultaneously clear the RIS and MIS bits corresponding to the highest priority interrupt whose index was returned by the read. The read value from the IIDX register can then be used in a case statement, as shown below.

```
void ISR_IIDX(void)
{
  switch(IIDX)
  {
    case 0:         // no IRQ pending
      break;
    case 1:         // IRQ[0]
      do_irq0();
      break;
```

```
    case 2:         // IRQ[1]
      do_irq1();
      break;
    default:        // out of range
      illegal();
  }
}
```

CPU IRQ Interrupt Service Routine using CPU_INT MIS and ICLR Registers

Alternatively, application software can read the MIS register to determine which bits are set, followed by using the ICLR register to clear the pending interrupt status bits.

```
void ISR(void)
{
  uint32_t pending = MIS;
  ICLR = pending;           // clear pending IRQ
  if (pending & 0x01)       // IRQ[0]
  {
      do_irq0();
  }
  if (pending & 0x02)       // IRQ[1]
  {
      do_irq1();
  }
}
```

6.2.5.4 Hardware Event Handling

In the case of an event which sources a DMA trigger (DMA_TRIG) or a generic event (GEN_EVENT), the IIDX register is not used. A four-way event handshake is performed between the peripheral generating the event and the hardware entity which is subscribed to the event (for example, the DMA or a secondary peripheral). The four-way event handshake will clear the corresponding interrupt status bits in the RIS and MIS registers automatically.

Chapter 7

IOMUX

The IOMUX controls the configuration of all device pins with digital input-output (IO) functions, including: digital function selection, inversion control, drive strength (if applicable), the pullup or pulldown resistor (if applicable), and wake-up configuration (if applicable on certain IOs for wakeup from SHUTDOWN mode).

7.1 IOMUX Overview

The IOMUX manages the configuration of the digital IO. Key functions configured by IOMUX include:

- Selection of which peripheral is multiplexed to each digital IO pin (for example, a GPIO or UART peripheral)
- Digital input path configuration
  - Hysteresis control
  - Input path enable/disable
  - Input logic inversion control
- Digital output path configuration
  - Drive strength control
  - Output connection enable/disable
  - Output logic inversion (control shared with input logic inversion)
  - Logic-high to High-Z output conversion (for open-drain style interfaces)
  - Retention of "last state" when a peripheral connected to an IO is disabled
- Wakeup configuration (for wakeup from SHUTDOWN mode)
  - Read wake up source from the wake stat bit from the PINCM Register
  - Wake up compare level
  - Release SHDNIOREL
  - Wakeup enable/disable
- Pullup and pulldown resistor control

7.1.1 IO Types and Analog Sharing

The IOMUX manages the selection of which peripheral function is to be used on a digital IO. It also provides the controls for the output driver, input path, and the wakeup logic for wakeup from SHUTDOWN mode.

Digital IO Types

There are several digital IO types which can be included on a given device. Each digital IO type supports different features. Table 7-1 lists the features which are included with each IO type. See the device-specific data sheet for which IO type is used on a given package pin.

Table 7-1. Digital IO Features by IO Type

IO Structure	Inversion Control	Drive Strength Control	Hysteresis Control	Pullup Resistor	Pulldown Resistor	Wakeup Logic
Standard-drive	Y			Y	Y	
Standard-drive with wake	Y			Y	Y	Y
High-drive	Y	Y		Y	Y	Y
High-speed	Y	Y		Y	Y	
5V tolerant open drain	Y		Y		Y	Y

Please note that the IOMUX will not support the inversion control and pesudo-open drain (output-high translated to high-impedance) setting on the SPI POCI pins. Also the IOMUX will not support inversion control on the SPI SCLK pins connected on an HSIO pin.

Digital IO Shared with Analog Functions

Certain pins on a device will be digital only and will not have any analog functions connected to the pin. Other pins can have one or more analog functions connected to the pin in addition to the digital IO functions. Analog functions are never selected within the IOMUX; they are always configured within of the respective analog peripheral. Analog peripherals have no knowledge of, or interaction with, the IOMUX.

In general, when analog functionality is used on a pin which also has digital functions, the IOMUX configuration for that pin should be left in its default (high-Z) state so as to not interfere with the proper operation of the analog

function. However, it is possible to have the IOMUX active on a pin when an analog peripheral is also interacting with the pin, provided that the application software ensures that there is not a conflict between the functions. For example, it is possible to have the pullup or pulldown resistor on an IO enabled at the same time that the ADC is running a conversion on the same IO. However, an invalid configuration would be enabling the output driver on an IO at the same time that an analog peripheral is driving the IO (for example, a DAC or OPA output). This would create an IO conflict.

Application software is responsible for ensuring that the IOMUX settings do not conflict with any analog peripheral functions which can be enabled on a shared pad.

IO Slice

The mixed-signal IO pin slice diagram for a full featured IO pin is shown in Figure 7-1. Not all pins will have analog functions, wake-up logic, drive strength control, and pullup or pulldown resistors available. See the device-specific data sheet for detailed information on what features are supported for a specific pin.

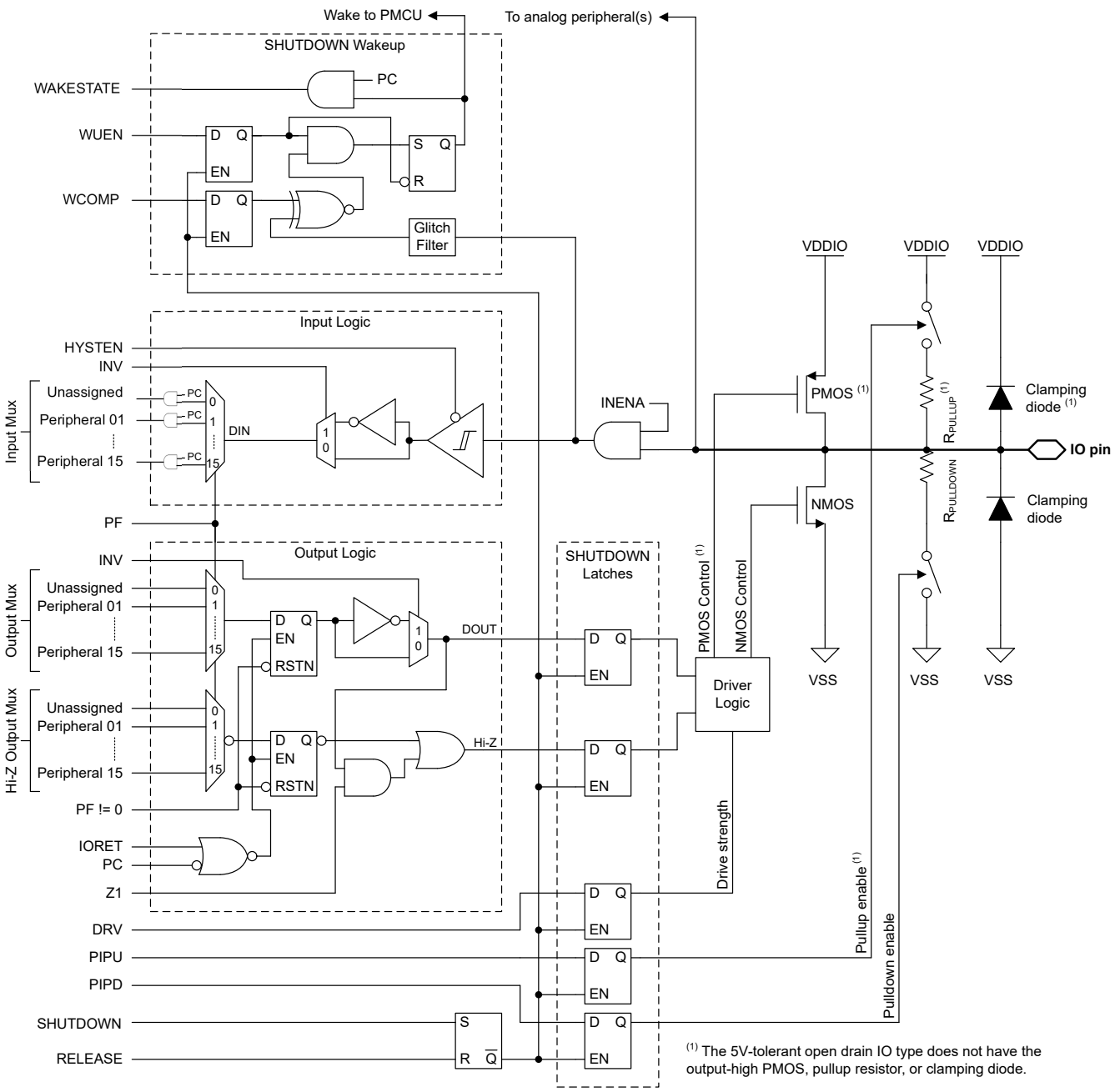


Figure 7-1. Superset IO Slice

Default IOMUX State

The initial state of the IOMUX pin slice for all digital IO after a BOOTRST is as follows:

- The digital IO is in a high-impedance state
- The peripheral function selection field (PF) is cleared (no peripheral function selected) and the peripheral connect (PC) and input enable states are cleared (disabled)
- The inversion logic is disabled
- The Hi-Z output high mode is disabled
- The pullup/pulldown resistors (if present) are disabled
- The input hysteresis control (if present) is disabled to save power
- The drive strength control (if present) is reset

- The wakeup logic (if present) is disabled

Note

The SWD pins are a specific exception to the above default state. The SWD debug pins are configured in SWD mode by default, and may be switched to an alternate setting after start-up. See Section 2.4.1.4 in the SYSCTL section of the PMCU chapter.

7.2 IOMUX Operation

Each digital IO on a device has a dedicated 32-bit PINCM register in the IOMUX peripheral register space which is used to configure the digital functions of the respective IO. See the device specific data sheet for determining the PINCM register index which corresponds to the IO to be configured.

7.2.1 Peripheral Function (PF) Assignment

When setting up the initial IOMUX configuration for an IO after a BOOTRST, application software can select which digital peripheral from the supported options is to be connect to an IO by writing the appropriate peripheral select value to the PF field while simultaneously setting the PC and INENA bits in the PINCMx register corresponding to the targeted pin. The IOMUX configurations for a given peripheral must be set before the peripheral connected to the IOs has been initialized for operation.

To change the peripheral function selection for a digital IO at runtime after a peripheral function has already been configured for that IO, the following procedure should be followed:

1. Disable the currently connected peripheral function
2. Clear the PC bit (input/output connect bit) and INENA (input connect bit) in the corresponding PINCMx register
3. Write 0x0 to the PF field in the PINCMx to clear the logic in the data path
4. Select the new peripheral function by writing the peripheral function ID to the PF register
5. Set the PC and INENA bits in the PINCMx register to connect the newly selected peripheral
6. Enable the newly selected peripheral for operation

At runtime, the INENA bit can be used to mask the input from the IO to the peripheral, if desired. When INENA is cleared, a connected peripheral function will see logic low (0) from the IO, regardless of the external state of the IO. If an IO supports wakeup from SHUTDOWN mode, the INENA bit also controls propagation of the IO state to the SHUTDOWN mode wakeup logic.

If a peripheral is assigned to an IO, but the peripheral is itself in a disabled state, the last valid output conditions (output logic level and Hi-Z state) are latched in the IOMUX output logic. When the peripheral is enabled, the IOMUX will release the latched state to allow the (now enabled) peripheral's output state to propagate to the IO. The PMCU indicates to the IOMUX when a peripheral is entering a disabled state via the IORET signal, which is combined with the PC signal via a logic OR to control the output state latches. This mechanism handles preservation of the last valid output state of peripherals in power domain 1 (PD1) when entering STOP or STANDBY mode, as PD1 peripherals are always temporarily disabled upon entry to STOP/STANDBY, and re-enabled upon exit from STOP/STANDBY modes.

When no peripheral function is selected (PF==0) the output latches are put into a reset state, causing the output NMOS and PMOS to be disabled (leaving the IO pin in a Hi-Z state with the exception of any enabled pullup/pulldown resistors). Note that the pullup/pulldown resistors are never controlled by either a connected peripheral or the peripheral muxing logic. They are only controlled by the IOMUX control bits (see pullup/pulldown).

7.2.2 Logic High to Hi-Z Conversion

The IOMUX supports translating an output high signal from a connected peripheral into a Hi-Z output state at the IO pin. This functionality is particularly useful for open-drain digital input/output applications. When this functionality is enabled, the IO pin state as a function of the peripheral output is as shown in Table 7-2.

Table 7-2. Logic High to Hi-Z Truth Table

Connected Peripheral Output	IO Pin State (Z1 = 0x0)	IO Pin State (Z1 = 0x1)
Logic low (0)	Output low	Output low

Table 7-2. Logic High to Hi-Z Truth Table (continued)

Connected Peripheral Output	IO Pin State (Z1 = 0x0)	IO Pin State (Z1 = 0x1)
Logic high (1)	Output high	High impedance (Hi-Z)

To enable logic high to Hi-Z conversion on a digital IO, set the Z1 bit in the corresponding PINCMx register.

Note that for 5V tolerant open-drain IO pins, the Z1 control has no effect as there is no high-side driver present. On these pins, a logic high output from the peripheral to the IO pin always results in a Hi-Z state.

7.2.3 Logic Inversion

The IOMUX supports logic inversion of the digital input/output path. Logic inversion is useful for scenarios where opposite polarity is required for UART functions or SPI chip select functions.

To enable logic inversion on a digital IO, set the INV bit in the corresponding PINCMx register. To disable logic inversion, clear the corresponding bit. Logic inversion is disabled by default.

When logic inversion is enabled for a 5V tolerant open drain IO, a connected peripheral which outputs a logic low state will cause the IO pin to go to a Hi-Z state. When the peripheral applies a logic high state, the IO pin will go to an output low state.

7.2.4 SHUTDOWN Mode Wakeup Logic

In SHUTDOWN mode, the entire regulated core supply of the device is disabled and the device wakes only from a wake-capable IO that is configured for wakeup, from NRST, or from a debug connection. The IO wake mechanism for exiting SHUTDOWN is managed by IOMUX and is level based. The 5V-tolerant open-drain IOs, high-drive IOs, and certain standard-drive IOs include the additional wakeup logic that can be used to wake the device from a SHUTDOWN operating mode upon a level match.

To configure a wake-capable IO for wakeup from SHUTDOWN mode:

1. Set the INENA bit to let the input state propagate from the IO to the wakeup logic.
2. Select the compare level to use for wake by setting or clearing the WCOMP bit in the PINCMx register corresponding to the targeted pin.
3. Enable wakeup by setting the WUEN bit in the PINCMx register corresponding to the targeted pin.

After the previous configuration, SHUTDOWN mode can be entered through the appropriate command in SYSCTL. Pins on the device that contain digital IO controlled by IOMUX retain their current state when the device enters into SHUTDOWN. While the digital IO state is latched upon entry into SHUTDOWN mode, the IOMUX configuration registers (all PINCMx registers) lose their contents as the regulated core supply is shut down.

After SHUTDOWN is entered, a level match on any pin configured for wakeup triggers the exit sequence from SHUTDOWN. When the device exits SHUTDOWN, a BOR-level reset occurs but the state of the digital IO remains latched through the reset, keeping the IO state that was present upon entry into SHUTDOWN. This state is held until the IO are released in SYSCTL. After the BOR, SYSCTL captures the cause of the reset as a SHUTDOWN exit so that software can identify this and take appropriate action to reconfigure the device.

If multiple pins were configured for wakeup from SHUTDOWN, application software can determine which wakeup-configured IO generated the wake by polling the WAKESTATE bit in any IOs that were enabled for wake before the SHUTDOWN exit.

Application software must apply the following process to restore the IO state upon exit from SHUTDOWN:

1. Check which IO triggered the wakeup from SHUTDOWN, if necessary, as follows:
   a. Reconfigure the PINCMx register corresponding to the IO to be tested for wakeup status and set the peripheral connect (PC) bit (the PC bit gates the WAKESTATE indication).
   b. Test the WAKESTATE bit in the PINCMx register corresponding to the IO to be tested to determine if that particular IO received a WAKE status based on the previously configured WCOMP and WUEN configuration.
2. Reconfigure any remaining IOMUX PINCM registers to the correct states.

3. Reconfigure the peripherals that are connected to pins through IOMUX, and enable them.
4. Release the SHUTDOWN IO lock in SYSCTL.
5. Clear the WUEN bit in the PINCMx register to reset the WAKESTATE status.

Note

After waking from SHUTDOWN, if the WUEN bit not cleared and the shutdown release bit in SYSCTL is not set, then reentering SHUTDOWN results in an immediate wake event, because the WAKESTATE status was not cleared from the previous wake event.

7.2.5 Pullup/Pulldown Resistors

Programmable pullup/pulldown resistors are provided on most digital IO types, and are connected to VDD/VSS, respectively. The 5V tolerant open drain digital IO does not provide a pullup resistor due to the open drain configuration.

To enable the pullup or pulldown resistor on a digital IO, set the PIPU or PIPD bit, respectively, in the corresponding PINCMx register. To disable the pullup or pulldown resistor, clear the corresponding bit.

The pullup/pulldown resistors can be enabled at any time, and their configuration is independent from the peripheral function configuration. It is possible to enable a pullup/pulldown resistor while changing the selected peripheral function.

7.2.6 Drive Strength Control

The high-drive and high-speed digital IO types have programmable drive strength (low drive and high drive). The default drive strength is low drive. Application software can request high drive by setting the DRV bit in the PINCMx register corresponding to the target digital IO. Drive strength control is not available for standard drive and open drain IO types.

The drive strength control is completely independent of the selected peripheral function (PF) and can be changed by application software at any time.

For detailed electrical specifications on the drive performance in each drive mode for a given IO, see the Digital IO parameters in the device-specific data sheet.

7.2.7 Hysteresis and Logic Level Control

The 5V-tolerant open drain digital IOs provide a hysteresis and logic level control to enable operation in input mode with standard CMOS logic (hysteresis enabled, CMOS logic levels) and TTL logic (hysteresis disabled, TTL logic levels).

The default mode for the 5V-tolerant open drain digital IO is TLL mode (HYSTEN bit in the PINCMx register is cleared). To use a 5V-tolerant open drain digital IO in CMOS mode with hysteresis enabled, set the HYSTEN bit in the PINCMx register which corresponds to the targeted IO.

The input logic level differences between TTL mode (left) and CMOS mode (right) are shown in Figure 7-2.

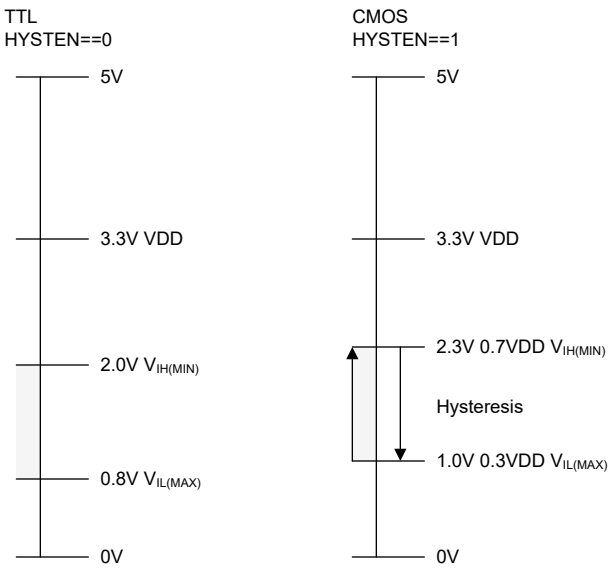


Figure 7-2. Input Logic Levels - 5V Tolerant Open Drain Digital IO

7.3 IOMUX (PINCMx) Register Format

7.3.1 PINCM (Offset = 4h) [Reset = X]

Pin Control Management Register

Figure 7-3. PINCM

31	30	29	28	27	26	25	24
RESERVED			WCOMP	WUEN	INV	HIZ1	RESERVED
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED			DRV	HYSTEN	INENA	PIPU	PIPD
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		WAKESTAT	RESERVED				
R/W-0h		R-0h	R/W-0h				
7	6	5	4	3	2	1	0
PC	RESERVED	PF					
R/W-0h	R/W-0h	R/W-0h					

Table 7-3. PINCM Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R/W	0h	
28	WCOMP	R/W	0h	Wakeup Compare Value bit 0h = Wakeup on a match of 0 1h = Wakeup on a match of 1
27	WUEN	R/W	0h	Wakeup Enable bit 0h = wakeup is disabled. 1h = wakeup is enabled
26	INV	R/W	0h	Data inversion selection 0h = Data inversion is disabled. 1h = Data inversion is enabled
25	HIZ1	R/W	0h	High output value will tri-state the output when this bit is enabled 0h = open-drain is disabled. 1h = open-drain is enabled.
24-21	RESERVED	R/W	0h	
20	DRV	R/W	0h	Drive strength control selection, for HS IOCELL only 0h = Drive setting of 0 selected 1h = Drive setting of 1 selected
19	HYSTEN	R/W	0h	Hysteresis Enable Control Selection 0h = hysteresis is disabled. 1h = hysteresis is enabled
18	INENA	R/W	0h	Input Enable Control Selection 0h = Input enable is disabled. 1h = Input enable is enabled.
17	PIPU	R/W	0h	Pull Up control selection 0h = Pull up is disabled. 1h = Pull up is enabled
16	PIPD	R/W	0h	Pull Down control selection 0h = Pull down is disabled. 1h = Pull down is enabled
15-14	RESERVED	R/W	0h	

Table 7-3. PINCM Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	WAKESTAT	R	0h	This has the IOPAD WAKEUP signal as status bit. 0h = wakeup source is NOT from this IOCELL 1h = wakeup source is from this IOCELL
12-8	RESERVED	R/W	0h	
7	PC	R/W	0h	Peripheral is “Connected” 0h = The output of the peripheral (and its output enable) will not propagate to the IOCELL 1h = The output latch of the dataflow will be “transparent”
6	RESERVED	R/W	0h	
5-0	PF	R/W	0h	P channel Function selection bits 0h = Reserved as unconnected 3Fh = An encoding per function that can be connected to this pin.

7.4 IOMUX Registers

Table 7-4 lists the memory-mapped registers for the IOMUX registers. All register offset addresses not listed in Table 7-4 should be considered as reserved locations and the register contents should not be modified.

Table 7-4. IOMUX Registers

Offset	Acronym	Register Name	Group	Section
4h	PINCM	Pin Control Management Register in SECCFG region		Go

Complex bit access types are encoded to fit into small table cells. Table 7-5 shows the codes that are used for access types in this section.

Table 7-5. IOMUX Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value

7.4.1 PINCM (Offset = 4h) [Reset = X]

PINCM is shown in Figure 7-4 and described in Table 7-6.

Return to the Summary Table.

Pin Control Management Register

Figure 7-4. PINCM

31	30	29	28	27	26	25	24
RESERVED			WCOMP	WUEN	INV	HIZ1	RESERVED
R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
		RESERVED	DRV	HYSTEN	INENA	PIPU	PIPD
		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		WAKESTAT	RESERVED				
R/W-0h		R-0h	R/W-0h				
7	6	5	4	3	2	1	0
PC	RESERVED	PF					
R/W-0h	R/W-0h	R/W-0h					

Table 7-6. PINCM Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R/W	0h	
28	WCOMP	R/W	0h	Wakeup Compare Value bit 0h = Wakeup on a match of 0 1h = Wakeup on a match of 1
27	WUEN	R/W	0h	Wakeup Enable bit 0h = wakeup is disabled. 1h = wakeup is enabled
26	INV	R/W	0h	Data inversion selection 0h = Data inversion is disabled. 1h = Data inversion is enabled
25	HIZ1	R/W	0h	High output value will tri-state the output when this bit is enabled 0h = open-drain is disabled. 1h = open-drain is enabled.
24	RESERVED	R/W	0h	
21	RESERVED	R/W	0h	
20	DRV	R/W	0h	Drive strength control selection, for HS IOCELL only 0h = Drive setting of 0 selected 1h = Drive setting of 1 selected
19	HYSTEN	R/W	0h	Hysteresis Enable Control Selection 0h = hysteresis is disabled. 1h = hysteresis is enabled
18	INENA	R/W	0h	Input Enable Control Selection 0h = Input enable is disabled. 1h = Input enable is enabled.
17	PIPU	R/W	0h	Pull Up control selection 0h = Pull up is disabled. 1h = Pull up is enabled
16	PIPD	R/W	0h	Pull Down control selection 0h = Pull down is disabled. 1h = Pull down is enabled
15-14	RESERVED	R/W	0h	

Table 7-6. PINCM Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	WAKESTAT	R	0h	This has the IOPAD WAKEUP signal as status bit. 0h = wakeup source is NOT from this IOCELL 1h = wakeup source is from this IOCELL
12-8	RESERVED	R/W	0h	
7	PC	R/W	0h	Peripheral is “Connected” 0h = The output of the peripheral (and its output enable) will not propagate to the IOCELL 1h = The output latch of the dataflow will be “transparent”
6	RESERVED	R/W	0h	
5-0	PF	R/W	0h	Peripheral Function selection bits 0h = Reserved as unconnected 3Fh = An encoding per function that can be connected to this pin.

Chapter 8
GPIO

The GPIO peripheral provides the user with a means to write data out and read data in to and from the device pins. It also provides a way to detect wakeup events while the device is in a low power state. This chapter describes the operation of the GPIO peripheral.

8.1 GPIO Overview

The GPIO is used to read in digital data from the device pins and to send digital data out to the device pins.

The GPIO module features include:

- Zero wait state MMR access from CPU
- Set/clear/toggle multiple bits without the need of a read-modify-write construct in software
- Direct writes to individual GPIO output bits (DOUT) without the need of a read-modify-write construct in software
- Direct read comparisons of individual GPIO input bits (DIN) without the need to use masking in software
- DOUT serviceable by DMA to generate a predefined output sequence on specified pins
- "FastWake" feature enables low-power wakeup from STOP and STANDBY modes for any GPIO port
- User controlled input filtering (configurable per IO)
- Interconnection to the device event fabric through event publishers and event subscribers (GPIOA instance only)

Figure 8-1 shows the block diagram of the GPIO peripheral.

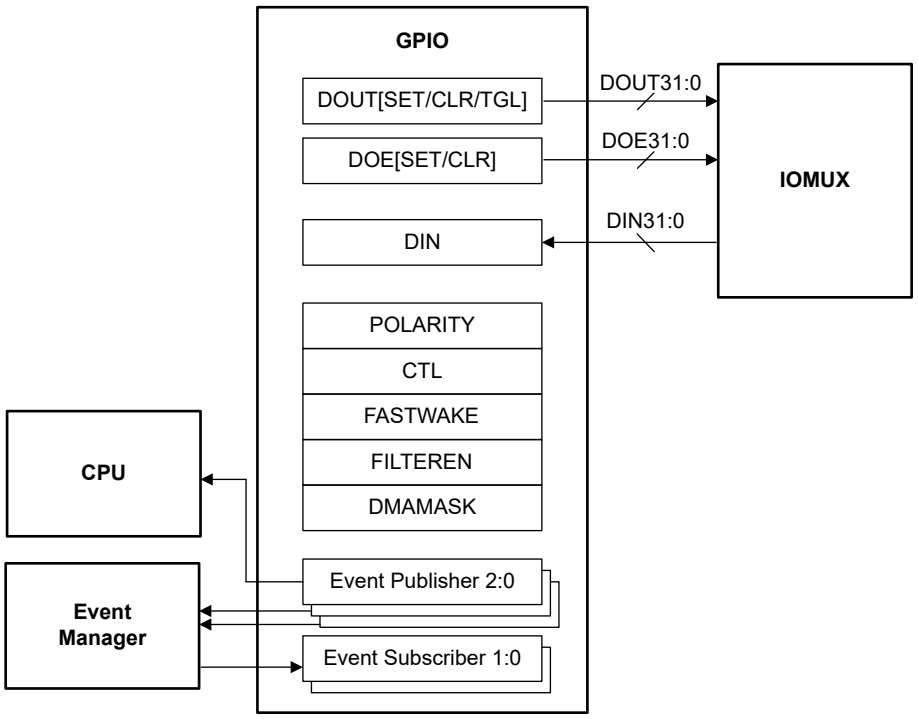


Figure 8-1. GPIO Block Diagram

Note

The GPIO module for the MSPM0 platform does not manage the complete digital IO functionality (for example, pullup, pulldown, or other functional muxing). For complete digital IO control details, refer to Chapter 7. Similar to any other peripheral, the GPIO has inputs and outputs (with output enable) that allow the GPIO to interface with the IOMUX to make connections to the IO pins.

8.2 GPIO Operation

The GPIO peripheral is configured with user software. The setup and operation of the GPIO is discussed in the following sections.

8.2.1 GPIO Ports

An instance of the GPIO peripheral in the MSPM0 platform supports up to 32 data input/output (DIO) bits. For devices with greater than 32 GPIOs, multiple instances of the GPIO peripheral are used to address all of the device pins. The GPIO port and bit names are directly mapped to the signal names associated with each device pin in the *Pin Configuration and Functions* section of the device data sheet.

Table 8-1. GPIO Port and Device Pin Mapping

GPIO Port and Bit Name	Device Pin Signal Name
GPIO Port A Bit 0 (DIO0)	PA0
GPIO Port A Bit 1 (DIO1)	PA1
GPIO Port B Bit 0 (DIO0)	PB0
GPIO Port B Bit 1 (DIO1)	PB1
GPIO Port x Bit y (DIOy)	Pxy

8.2.2 GPIO Read/Write Interface

The GPIO peripheral has features and dedicated registers to allow for advanced bit manipulations without the need to execute a read-modify-write construct in software. These features are outlined below.

8.2.3 GPIO Input Glitch Filtering and Synchronization

The GPIO module evaluates the state of input pins at the ULPCLK (PD0 bus clock) rate, synchronizing the pin state to ULPCLK through a 2-stage synchronizer before passing the GPIO state to the input glitch filter.

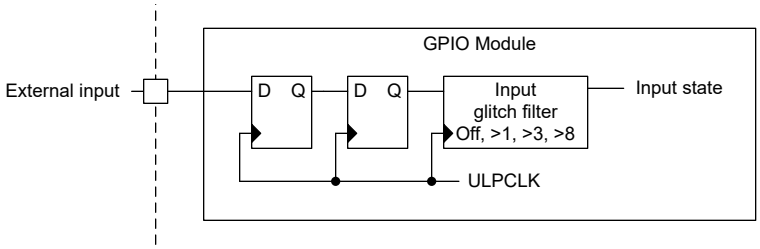


Figure 8-2. GPIO Input Synchronizer

A programmable input glitch filter is provided for suppressing noise on digital input pins. The glitch filter runs at ULPCLK rate. Four levels of user-specified input filtering are possible:

- Sampled input without filtering (the minimum reliably detected pulse width is one ULPCLK cycle due to synchronization of the pin state with ULPCLK +Delay time from edge of asynchronous request to first MCLK edge in case of fast wake enable for STANDBY0/1, STOP1/2 and SLEEP2 modes)
- Synchronized inputs which are not greater than 1 ULPCLK periods are filtered out
- Synchronized inputs which are not greater than 3 ULPCLK periods are filtered out
- Synchronized inputs which are not greater than 8 ULPCLK periods are filtered out

This feature allows users to easily implement input filtering in hardware for cases where fast switching on the input pin is needed to be filtered out. The bit fields in the FILTEREN31_16 and FILTEREN15_0 registers allow users to configure the level of filtering needed for the corresponding GPIO bit.

Input pulses of the same pulse length can be passed in some cases while being filtered in other cases, due to 1 ULPCLK cycle of uncertainty in the synchronization.

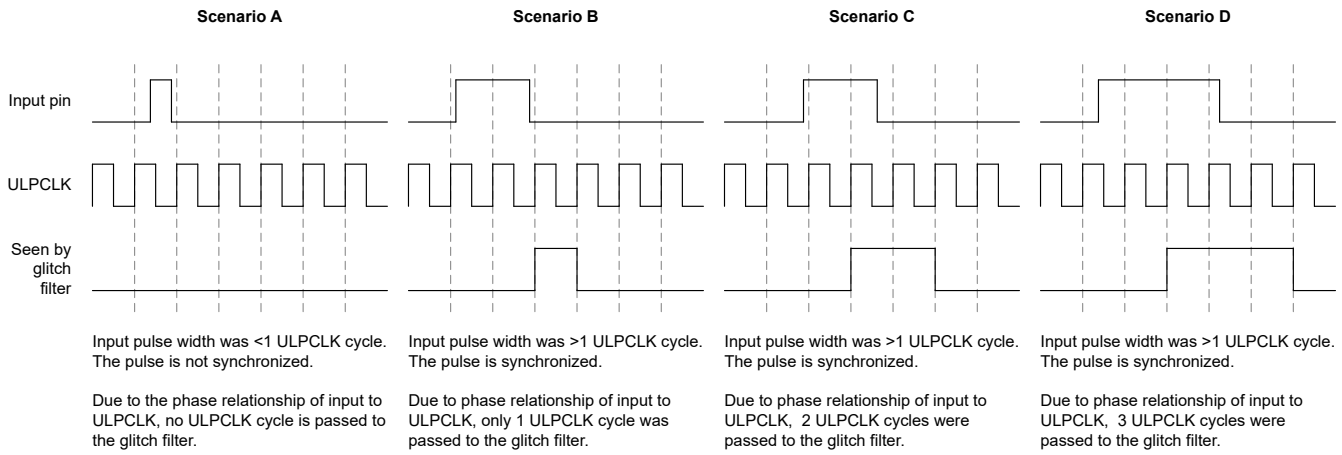


Figure 8-3. GPIO Input Synchronizer and Glitch Filtering Scenarios

- In Scenario A, the input pulse is less than one ULPCLK cycle. Pulses less than one ULPCLK cycle may not be captured. To ensure that GPIO inputs are always captured, the GPIO input pulse width must be greater than the ULPCLK period.
- In Scenario B, the input pulse is nearly two ULPCLK cycles in length, but because the rising edge occurs just after the ULPCLK edge, the GPIO synchronizer only views the input pin as having been high for 1 ULPCLK period. This scenario would not be filtered out by the glitch filter when the glitch filter is disabled, but a glitch filter value of >1 would result in this pulse being filtered. Conversely, the same input pulse width in Scenario C results in the input pin being considered high for two ULPCLK periods, as the rising edge occurred just before the ULPCLK edge. In this case, this scenario would not be filtered out when a glitch filter value of >1 is specified.
- In Scenario D, three ULPCLK cycles are passed to the glitch filter. In this case, the scenario would not be filtered out when a glitch filter value of >1 is specified, but it would be filtered out for a glitch filter value of >3 or >8.

Note

When the fast wake mode is enabled (SYSOSC is requested asynchronously upon input pin activity), the ULPCLK will switch from off (as would be the case in STANDBY1) or 32kHz (as would be the case in STANDBY0) to 24Mhz/32MHz, resulting in the input synchronization logic and glitch filter running at 24Mhz/32MHz after some latency. See the device specific data sheet for the asynchronous fast clock request wake time, and budget this time into any minimum pulse width calculations when using fast wake.

8.2.4 GPIO Fast Wake

The fast wake feature in the MSPM0 GPIO peripheral allows the GPIO module to stay in a low-power state and detect interrupt events on the device pins without requiring a high-speed clock. This allows the device to support fast wakeup from low-power modes, such as STOP and STANDBY, on any GPIO pin.

Fast wake can be enabled on a bit-wise basis using the FASTWAKE register. Setting a bit in the FASTWAKE register enables the corresponding device pin signal to support fast wakeup functionality. The CTL register contains a bit field named FASTWAKEONLY which allows for global control of the fast wake feature. Setting the FASTWAKEONLY bit enables all of the bits in the corresponding GPIO port to support fast wakeup functionality.

Note

Do not enable fast wake in the GPIO while simultaneously blocking asynchronous fast clock requests in SYSCTL. When fast wake is enabled, the GPIO expects to handshake with SYSCTL for the fast clock. If SYSCTL ignores the request, the GPIO does not receive a clock until SYSCTL completes the asynchronous fast clock request handshake.

8.2.5 GPIO DMA Interface

The GPIO peripheral allows the DMA write-access to the DOUT31_0 register. This functionality allows users to generate predefined output sequences on specified device pins. Some applications require preloaded sequences of GPIO pin changes and the MSPM0 platform allows for the DMA to run that sequence so that the CPU can remain asleep and conserve energy.

The DMAMASK register is used to indicate which GPIO bits the DMA is allowed to modify. Setting a bit in the DMAMASK register enables the corresponding DOUT bit to be modified by the DMA.

Note

The CPU can write to any DOUT31_0 bit regardless of the DMAMASK value.

In cases where the DMA and the CPU both attempt to access and modify the DOUT31_0 register concurrently, it is the user's responsibility to manage the DMA and CPU bus transactions that are targeting the same bit to be modified.

- If a DMAMASK bit is set, the DMA will be prioritized to modify the corresponding DOUT bit.
- If a DMAMASK bit is cleared, the CPU will be prioritized to modify the corresponding DOUT bit.

8.2.6 Event Publishers and Subscribers

There are three independent event publishers available for GPIOx peripherals:

1. First Event Publisher (CPU_INT)
   - Used for generating CPU interrupt
   - Interrupt (RIS) flags are cleared upon software reading the IIDX register or writing to the respective ICLR register bits.
   - An event to the CPU can be individually specified for each GPIO bit through the POLARITY register:
     - 0: Disabled
     - 1: Rise Event
     - 2: Fall Event
     - 3: Rise or Fall Event
2. Second Event Publisher (GEN_EVENT0)
   - Uses the same POLARITY register definition as CPU_INT
   - Applies to GPIO bits 15 down to 0 (DIO15:0)
3. Third Event Publisher (GEN_EVENT1),
   - Uses the same POLARITY register definition as CPU_INT
   - Applies to GPIO bits 31 down to 16 (DIO31:16)

There are **two event subscribers**

1. First Event Subscriber (FSUB_0)
   - A specific pin can be directed to change state on an event
   - A subscriber event can only cause one single bit to have an action
   - Applies to GPIO bits 15 down to 0 (DIO15:0)
   - SUB0CFG register is used to enable the FSUB_0 event and define the output policy for a specific GPIO pin
2. Second Event Subscriber (FSUB_1)
   - A specific pin can be directed to change state on an event
   - A subscriber event can only cause one single bit to have an action
   - Applies to GPIO bits 31 down to 16 (DIO31:16)
   - SUB1CFG register is used to enable the FSUB_1 event and define the output policy for a specific GPIO pin

8.3 GPIO Registers

Table 8-2 lists the memory-mapped registers for the GPIO registers. All register offset addresses not listed in Table 8-2 should be considered as reserved locations and the register contents should not be modified.

Table 8-2. GPIO Registers

Offset	Acronym	Register Name	Group	Section
400h	FSUB_0	Subsciber Port 0		Go
404h	FSUB_1	Subscriber Port 1		Go
444h	FPUB_0	Publisher Port 0		Go
448h	FPUB_1	Publisher Port 1		Go
800h	PWREN	Power enable		Go
804h	RSTCTL	Reset Control		Go
814h	STAT	Status Register		Go
1010h	CLKOVR	Clock Override		Go
1018h	PDBGCTL	Peripheral Debug Control		Go
1020h	IIDX	Interrupt index	CPU_INT	Go
1028h	IMASK	Interrupt mask	CPU_INT	Go
1030h	RIS	Raw interrupt status	CPU_INT	Go
1038h	MIS	Masked interrupt status	CPU_INT	Go
1040h	ISET	Interrupt set	CPU_INT	Go
1048h	ICLR	Interrupt clear	CPU_INT	Go
1050h	IIDX	Interrupt index	GEN_EVENT 0	Go
1058h	IMASK	Interrupt mask	GEN_EVENT 0	Go
1060h	RIS	Raw interrupt status	GEN_EVENT 0	Go
1068h	MIS	Masked interrupt status	GEN_EVENT 0	Go
1070h	ISET	Interrupt set	GEN_EVENT 0	Go
1078h	ICLR	Interrupt clear	GEN_EVENT 0	Go
1080h	IIDX	Interrupt index	GEN_EVENT 1	Go
1088h	IMASK	Interrupt mask	GEN_EVENT 1	Go
1090h	RIS	Raw interrupt status	GEN_EVENT 1	Go
1098h	MIS	Masked interrupt status	GEN_EVENT 1	Go
10A0h	ISET	Interrupt set	GEN_EVENT 1	Go
10A8h	ICLR	Interrupt clear	GEN_EVENT 1	Go
10E0h	EVT_MODE	Event Mode		Go
10FCh	DESC	Module Description		Go
1200h	DOUT3_0	Data output 3 to 0		Go
1204h	DOUT7_4	Data output 7 to 4		Go
1208h	DOUT11_8	Data output 11 to 8		Go
120Ch	DOUT15_12	Data output 15 to 12		Go

Table 8-2. GPIO Registers (continued)

Offset	Acronym	Register Name	Group	Section
1210h	DOUT19_16	Data output 19 to 16		Go
1214h	DOUT23_20	Data output 23 to 20		Go
1218h	DOUT27_24	Data output 27 to 24		Go
121Ch	DOUT31_28	Data output 31 to 28		Go
1280h	DOUT31_0	Data output 31 to 0		Go
1290h	DOUTSET31_0	Data output set 31 to 0		Go
12A0h	DOUTCLR31_0	Data output clear 31 to 0		Go
12B0h	DOUTTGL31_0	Data output toggle 31 to 0		Go
12C0h	DOE31_0	Data output enable 31 to 0		Go
12D0h	DOESET31_0	Data output enable set 31 to 0		Go
12E0h	DOECLR31_0	Data output enable clear 31 to 0		Go
1300h	DIN3_0	Data input 3 to 0		Go
1304h	DIN7_4	Data input 7 to 4		Go
1308h	DIN11_8	Data input 11 to 8		Go
130Ch	DIN15_12	Data input 15 to 12		Go
1310h	DIN19_16	Data input 19 to 16		Go
1314h	DIN23_20	Data input 23 to 20		Go
1318h	DIN27_24	Data input 27 to 24		Go
131Ch	DIN31_28	Data input 31 to 28		Go
1380h	DIN31_0	Data input 31 to 0		Go
1390h	POLARITY15_0	Polarity 15 to 0		Go
13A0h	POLARITY31_16	Polarity 31 to 16		Go
1400h	CTL	FAST WAKE GLOBAL EN		Go
1404h	FASTWAKE	FAST WAKE ENABLE		Go
1500h	SUB0CFG	Subscriber 0 configuration		Go
1508h	FILTEREN15_0	Filter Enable 15 to 0		Go
150Ch	FILTEREN31_16	Filter Enable 31 to 16		Go
1510h	DMAMASK	DMA Write MASK		Go
1520h	SUB1CFG	Subscriber 1 configuration		Go

Complex bit access types are encoded to fit into small table cells. Table 8-3 shows the codes that are used for access types in this section.

Table 8-3. GPIO Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
K	K	Write protected by a key
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

8.3.1 FSUB_0 (Offset = 400h) [Reset = 00000000h]

FSUB_0 is shown in Figure 8-4 and described in Table 8-4.

Return to the Summary Table.

Subscriber port

Figure 8-4. FSUB_0

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

Table 8-4. FSUB_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

8.3.2 FSUB_1 (Offset = 404h) [Reset = 00000000h]

FSUB_1 is shown in Figure 8-5 and described in Table 8-5.

Return to the Summary Table.

Subscriber port

Figure 8-5. FSUB_1

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

Table 8-5. FSUB_1 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

8.3.3 FPUB_0 (Offset = 444h) [Reset = 00000000h]

FPUB_0 is shown in Figure 8-6 and described in Table 8-6.

Return to the Summary Table.

Publisher port

Figure 8-6. FPUB_0

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

Table 8-6. FPUB_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

8.3.4 FPUB_1 (Offset = 448h) [Reset = 00000000h]

FPUB_1 is shown in Figure 8-7 and described in Table 8-7.

Return to the Summary Table.

Publisher port

Figure 8-7. FPUB_1

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

Table 8-7. FPUB_1 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

8.3.5 PWREN (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 8-8 and described in Table 8-8.

Return to the Summary Table.

Register to control the power state

Figure 8-8. PWREN

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R/W-							K-0h

Table 8-8. PWREN Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R/W	0h	
0	ENABLE	K	0h	Enable the power KEY must be set to 26h to write to this bit. 0h = Disable Power 1h = Enable Power

8.3.6 RSTCTL (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 8-9 and described in Table 8-9.

Return to the Summary Table.

Register to control reset assertion and de-assertion

Figure 8-9. RSTCTL

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-							
15	14	13	12	11	10	9	8
RESERVED							
W-							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
W-						WK-0h	WK-0h

Table 8-9. RSTCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	W	0h	
1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Clear reset sticky bit
0	RESETASSERT	WK	0h	Assert reset to the peripheral KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Assert reset

8.3.7 STAT (Offset = 814h) [Reset = 00000000h]

STAT is shown in Figure 8-10 and described in Table 8-10.

Return to the Summary Table.

peripheral enable and reset status

Figure 8-10. STAT

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

Table 8-10. STAT Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	0h	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

8.3.8 CLKOVR (Offset = 1010h) [Reset = 00000000h]

CLKOVR is shown in Figure 8-11 and described in Table 8-11.

Return to the Summary Table.

This register overrides the functional clock request by this peripheral to the system

Figure 8-11. CLKOVR

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						RUN_STOP	OVERRIDE
R/W-0h						R/W-0h	R/W-0h

Table 8-11. CLKOVR Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1	RUN_STOP	R/W	0h	If OVERRIDE is enabled, this register is used to manually control the peripheral's clock request to the system 0h = Run/ungate functional clock 1h = Stop/gate functional clock
0	OVERRIDE	R/W	0h	Unlocks the functionality of RUN_STOP to override the automatic peripheral clock request 0h = Override disabled 1h = Override enabled

8.3.9 PDBGCTL (Offset = 1018h) [Reset = 00000001h]

PDBGCTL is shown in Figure 8-12 and described in Table 8-12.

Return to the Summary Table.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 8-12. PDBGCTL

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

Table 8-12. PDBGCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	FREE	R/W	1h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

8.3.10 IIDX (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 8-13 and described in Table 8-13.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 8-13. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 8-13. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 0h = No bit is set means there is no pending interrupt request 1h = DIO0 interrupt 2h = DIO1 interrupt 3h = DIO2 interrupt 4h = DIO3 interrupt 5h = DIO4 interrupt 6h = DIO5 interrupt 7h = DIO6 interrupt 8h = DIO7 interrupt 9h = DIO8 interrupt Ah = DIO9 interrupt Bh = DIO10 interrupt Ch = DIO11 interrupt Dh = DIO12 interrupt Eh = DIO13 interrupt Fh = DIO14 interrupt 10h = DIO15 interrupt 11h = DIO16 interrupt 12h = DIO17 interrupt 13h = DIO18 interrupt 14h = DIO19 interrupt 15h = DIO20 interrupt 16h = DIO21 interrupt 17h = DIO22 interrupt 18h = DIO23 interrupt 19h = DIO24 interrupt 1Ah = DIO25 interrupt 1Bh = DIO26 interrupt 1Ch = DIO27 interrupt 1Dh = DIO28 interrupt 1Eh = DIO29 interrupt 1Fh = DIO30 interrupt 20h = DIO31 interrupt

8.3.11 IMASK (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 8-14 and described in Table 8-14.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 8-14. IMASK

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-14. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	DIO31 event mask 0h = Event is masked 1h = Event is unmasked
30	DIO30	R/W	0h	DIO30 event mask 0h = Event is masked 1h = Event is unmasked
29	DIO29	R/W	0h	DIO29 event mask 0h = Event is masked 1h = Event is unmasked
28	DIO28	R/W	0h	DIO28 event mask 0h = Event is masked 1h = Event is unmasked
27	DIO27	R/W	0h	DIO27 event mask 0h = Event is masked 1h = Event is unmasked
26	DIO26	R/W	0h	DIO26 event mask 0h = Event is masked 1h = Event is unmasked
25	DIO25	R/W	0h	DIO25 event mask 0h = Event is masked 1h = Event is unmasked
24	DIO24	R/W	0h	DIO24 event mask 0h = Event is masked 1h = Event is unmasked
23	DIO23	R/W	0h	DIO23 event mask 0h = Event is masked 1h = Event is unmasked
22	DIO22	R/W	0h	DIO22 event mask 0h = Event is masked 1h = Event is unmasked

Table 8-14. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R/W	0h	DIO21 event mask 0h = Event is masked 1h = Event is unmasked
20	DIO20	R/W	0h	DIO20 event mask 0h = Event is masked 1h = Event is unmasked
19	DIO19	R/W	0h	DIO19 event mask 0h = Event is masked 1h = Event is unmasked
18	DIO18	R/W	0h	DIO18 event mask 0h = Event is masked 1h = Event is unmasked
17	DIO17	R/W	0h	DIO17 event mask 0h = Event is masked 1h = Event is unmasked
16	DIO16	R/W	0h	DIO16 event mask 0h = Event is masked 1h = Event is unmasked
15	DIO15	R/W	0h	DIO15 event mask 0h = Event is masked 1h = Event is unmasked
14	DIO14	R/W	0h	DIO14 event mask 0h = Event is masked 1h = Event is unmasked
13	DIO13	R/W	0h	DIO13 event mask 0h = Event is masked 1h = Event is unmasked
12	DIO12	R/W	0h	DIO12 event mask 0h = Event is masked 1h = Event is unmasked
11	DIO11	R/W	0h	DIO11 event mask 0h = Event is masked 1h = Event is unmasked
10	DIO10	R/W	0h	DIO10 event mask 0h = Event is masked 1h = Event is unmasked
9	DIO9	R/W	0h	DIO9 event mask 0h = Event is masked 1h = Event is unmasked
8	DIO8	R/W	0h	DIO8 event mask 0h = Event is masked 1h = Event is unmasked
7	DIO7	R/W	0h	DIO7 event mask 0h = Event is masked 1h = Event is unmasked
6	DIO6	R/W	0h	DIO6 event mask 0h = Event is masked 1h = Event is unmasked
5	DIO5	R/W	0h	DIO5 event mask 0h = Event is masked 1h = Event is unmasked
4	DIO4	R/W	0h	DIO4 event mask 0h = Event is masked 1h = Event is unmasked
3	DIO3	R/W	0h	DIO3 event mask 0h = Event is masked 1h = Event is unmasked

Table 8-14. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	R/W	0h	DIO2 event mask 0h = Event is masked 1h = Event is unmasked
1	DIO1	R/W	0h	DIO1 event mask 0h = Event is masked 1h = Event is unmasked
0	DIO0	R/W	0h	DIO0 event mask 0h = Event is masked 1h = Event is unmasked

8.3.12 RIS (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 8-15 and described in Table 8-15.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 8-15. RIS

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-15. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R	0h	DIO31 event 0h = DIO31 event did not occur 1h = DIO31 event occurred
30	DIO30	R	0h	DIO30 event 0h = DIO30 event did not occur 1h = DIO30 event occurred
29	DIO29	R	0h	DIO29 event 0h = DIO29 event did not occur 1h = DIO29 event occurred
28	DIO28	R	0h	DIO28 event 0h = DIO28 event did not occur 1h = DIO28 event occurred
27	DIO27	R	0h	DIO27 event 0h = DIO27 event did not occur 1h = DIO27 event occurred
26	DIO26	R	0h	DIO26 event 0h = DIO26 event did not occur 1h = DIO26 event occurred
25	DIO25	R	0h	DIO25 event 0h = DIO25 event did not occur 1h = DIO25 event occurred
24	DIO24	R	0h	DIO24 event 0h = DIO24 event did not occur 1h = DIO24 event occurred
23	DIO23	R	0h	DIO23 event 0h = DIO23 event did not occur 1h = DIO23 event occurred
22	DIO22	R	0h	DIO22 event 0h = DIO22 event did not occur 1h = DIO22 event occurred

Table 8-15. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R	0h	DIO21 event 0h = DIO21 event did not occur 1h = DIO21 event occurred
20	DIO20	R	0h	DIO20 event 0h = DIO20 event did not occur 1h = DIO20 event occurred
19	DIO19	R	0h	DIO19 event 0h = DIO19 event did not occur 1h = DIO19 event occurred
18	DIO18	R	0h	DIO18 event 0h = DIO18 event did not occur 1h = DIO18 event occurred
17	DIO17	R	0h	DIO17 event 0h = DIO17 event did not occur 1h = DIO17 event occurred
16	DIO16	R	0h	DIO16 event 0h = DIO16 event did not occur 1h = DIO16 event occurred
15	DIO15	R	0h	DIO15 event 0h = DIO15 event did not occur 1h = DIO15 event occurred
14	DIO14	R	0h	DIO14 event 0h = DIO14 event did not occur 1h = DIO14 event occurred
13	DIO13	R	0h	DIO13 event 0h = DIO13 event did not occur 1h = DIO13 event occurred
12	DIO12	R	0h	DIO12 event 0h = DIO12 event did not occur 1h = DIO12 event occurred
11	DIO11	R	0h	DIO11 event 0h = DIO11 event did not occur 1h = DIO11 event occurred
10	DIO10	R	0h	DIO10 event 0h = DIO10 event did not occur 1h = DIO10 event occurred
9	DIO9	R	0h	DIO9 event 0h = DIO9 event did not occur 1h = DIO9 event occurred
8	DIO8	R	0h	DIO8 event 0h = DIO8 event did not occur 1h = DIO8 event occurred
7	DIO7	R	0h	DIO7 event 0h = DIO7 event did not occur 1h = DIO7 event occurred
6	DIO6	R	0h	DIO6 event 0h = DIO6 event did not occur 1h = DIO6 event occurred
5	DIO5	R	0h	DIO5 event 0h = DIO5 event did not occur 1h = DIO5 event occurred
4	DIO4	R	0h	DIO4 event 0h = DIO4 event did not occur 1h = DIO4 event occurred
3	DIO3	R	0h	DIO3 event 0h = DIO3 event did not occur 1h = DIO3 event occurred

Table 8-15. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	R	0h	DIO2 event 0h = DIO2 event did not occur 1h = DIO2 event occurred
1	DIO1	R	0h	DIO1 event 0h = DIO1 event did not occur 1h = DIO1 event occurred
0	DIO0	R	0h	DIO0 event 0h = DIO0 event did not occur 1h = DIO0 event occurred

8.3.13 MIS (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 8-16 and described in Table 8-16.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 8-16. MIS

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-16. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R	0h	DIO31 event 0h = DIO31 event did not occur 1h = DIO31 event occurred
30	DIO30	R	0h	DIO30 event 0h = DIO30 event did not occur 1h = DIO30 event occurred
29	DIO29	R	0h	DIO29 event 0h = DIO29 event did not occur 1h = DIO29 event occurred
28	DIO28	R	0h	DIO28 event 0h = DIO28 event did not occur 1h = DIO28 event occurred
27	DIO27	R	0h	DIO27 event 0h = DIO27 event did not occur 1h = DIO27 event occurred
26	DIO26	R	0h	DIO26 event 0h = DIO26 event did not occur 1h = DIO26 event occurred
25	DIO25	R	0h	DIO25 event 0h = DIO25 event did not occur 1h = DIO25 event occurred
24	DIO24	R	0h	DIO24 event 0h = DIO24 event did not occur 1h = DIO24 event occurred
23	DIO23	R	0h	DIO23 event 0h = DIO23 event did not occur 1h = DIO23 event occurred
22	DIO22	R	0h	DIO22 event 0h = DIO22 event did not occur 1h = DIO22 event occurred

Table 8-16. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R	0h	DIO21 event 0h = DIO21 event did not occur 1h = DIO21 event occurred
20	DIO20	R	0h	DIO20 event 0h = DIO20 event did not occur 1h = DIO20 event occurred
19	DIO19	R	0h	DIO19 event 0h = DIO19 event did not occur 1h = DIO19 event occurred
18	DIO18	R	0h	DIO18 event 0h = DIO18 event did not occur 1h = DIO18 event occurred
17	DIO17	R	0h	DIO17 event 0h = DIO17 event did not occur 1h = DIO17 event occurred
16	DIO16	R	0h	DIO16 event 0h = DIO16 event did not occur 1h = DIO16 event occurred
15	DIO15	R	0h	DIO15 event 0h = DIO15 event did not occur 1h = DIO15 event occurred
14	DIO14	R	0h	DIO14 event 0h = DIO14 event did not occur 1h = DIO14 event occurred
13	DIO13	R	0h	DIO13 event 0h = DIO13 event did not occur 1h = DIO13 event occurred
12	DIO12	R	0h	DIO12 event 0h = DIO12 event did not occur 1h = DIO12 event occurred
11	DIO11	R	0h	DIO11 event 0h = DIO11 event did not occur 1h = DIO11 event occurred
10	DIO10	R	0h	DIO10 event 0h = DIO10 event did not occur 1h = DIO10 event occurred
9	DIO9	R	0h	DIO9 event 0h = DIO9 event did not occur 1h = DIO9 event occurred
8	DIO8	R	0h	DIO8 event 0h = DIO8 event did not occur 1h = DIO8 event occurred
7	DIO7	R	0h	DIO7 event 0h = DIO7 event did not occur 1h = DIO7 event occurred
6	DIO6	R	0h	DIO6 event 0h = DIO6 event did not occur 1h = DIO6 event occurred
5	DIO5	R	0h	DIO5 event 0h = DIO5 event did not occur 1h = DIO5 event occurred
4	DIO4	R	0h	DIO4 event 0h = DIO4 event did not occur 1h = DIO4 event occurred
3	DIO3	R	0h	DIO3 event 0h = DIO3 event did not occur 1h = DIO3 event occurred

Table 8-16. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	R	0h	DIO2 event 0h = DIO2 event did not occur 1h = DIO2 event occurred
1	DIO1	R	0h	DIO1 event 0h = DIO1 event did not occur 1h = DIO1 event occurred
0	DIO0	R	0h	DIO0 event 0h = DIO0 event did not occur 1h = DIO0 event occurred

8.3.14 ISET (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 8-17 and described in Table 8-17.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 8-17. ISET

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-17. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	DIO31 event 0h = No effect 1h = Sets DIO31 in RIS register
30	DIO30	W	0h	DIO30 event 0h = No effect 1h = Sets DIO30 in RIS register
29	DIO29	W	0h	DIO29 event 0h = No effect 1h = Sets DIO29 in RIS register
28	DIO28	W	0h	DIO28 event 0h = No effect 1h = Sets DIO28 in RIS register
27	DIO27	W	0h	DIO27 event 0h = No effect 1h = Sets DIO27 in RIS register
26	DIO26	W	0h	DIO26 event 0h = No effect 1h = Sets DIO26 in RIS register
25	DIO25	W	0h	DIO25 event 0h = No effect 1h = Sets DIO25 in RIS register
24	DIO24	W	0h	DIO24 event 0h = No effect 1h = Sets DIO24 in RIS register
23	DIO23	W	0h	DIO23 event 0h = No effect 1h = Sets DIO23 in RIS register
22	DIO22	W	0h	DIO22 event 0h = No effect 1h = Sets DIO22 in RIS register

Table 8-17. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	W	0h	DIO21 event 0h = No effect 1h = Sets DIO21 in RIS register
20	DIO20	W	0h	DIO20 event 0h = No effect 1h = Sets DIO20 in RIS register
19	DIO19	W	0h	DIO19 event 0h = No effect 1h = Sets DIO19 in RIS register
18	DIO18	W	0h	DIO18 event 0h = No effect 1h = Sets DIO18 in RIS register
17	DIO17	W	0h	DIO17 event 0h = No effect 1h = Sets DIO17 in RIS register
16	DIO16	W	0h	DIO16 event 0h = No effect 1h = Sets DIO16 in RIS register
15	DIO15	W	0h	DIO15 event 0h = No effect 1h = Sets DIO15 in RIS register
14	DIO14	W	0h	DIO14 event 0h = No effect 1h = Sets DIO14 in RIS register
13	DIO13	W	0h	DIO13 event 0h = No effect 1h = Sets DIO13 in RIS register
12	DIO12	W	0h	DIO12 event 0h = No effect 1h = Sets DIO12 in RIS register
11	DIO11	W	0h	DIO11 event 0h = No effect 1h = Sets DIO11 in RIS register
10	DIO10	W	0h	DIO10 event 0h = No effect 1h = Sets DIO10 in RIS register
9	DIO9	W	0h	DIO9 event 0h = No effect 1h = Sets DIO9 in RIS register
8	DIO8	W	0h	DIO8 event 0h = No effect 1h = Sets DIO8 in RIS register
7	DIO7	W	0h	DIO7 event 0h = No effect 1h = Sets DIO7 in RIS register
6	DIO6	W	0h	DIO6 event 0h = No effect 1h = Sets DIO6 in RIS register
5	DIO5	W	0h	DIO5 event 0h = No effect 1h = Sets DIO5 in RIS register
4	DIO4	W	0h	DIO4 event 0h = No effect 1h = Sets DIO4 in RIS register
3	DIO3	W	0h	DIO3 event 0h = No effect 1h = Sets DIO3 in RIS register

Table 8-17. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	W	0h	DIO2 event 0h = No effect 1h = Sets DIO2 in RIS register
1	DIO1	W	0h	DIO1 event 0h = No effect 1h = Sets DIO1 in RIS register
0	DIO0	W	0h	DIO0 event 0h = No effect 1h = Sets DIO0 in RIS register

8.3.15 ICLR (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 8-18 and described in Table 8-18.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 8-18. ICLR

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-18. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	DIO31 event 0h = No effect 1h = Clears DIO31 in RIS register
30	DIO30	W	0h	DIO30 event 0h = No effect 1h = Clears DIO30 in RIS register
29	DIO29	W	0h	DIO29 event 0h = No effect 1h = Clears DIO29 in RIS register
28	DIO28	W	0h	DIO28 event 0h = No effect 1h = Clears DIO28 in RIS register
27	DIO27	W	0h	DIO27 event 0h = No effect 1h = Clears DIO27 in RIS register
26	DIO26	W	0h	DIO26 event 0h = No effect 1h = Clears DIO26 in RIS register
25	DIO25	W	0h	DIO25 event 0h = No effect 1h = Clears DIO25 in RIS register
24	DIO24	W	0h	DIO24 event 0h = No effect 1h = Clears DIO24 in RIS register
23	DIO23	W	0h	DIO23 event 0h = No effect 1h = Clears DIO23 in RIS register
22	DIO22	W	0h	DIO22 event 0h = No effect 1h = Clears DIO22 in RIS register

Table 8-18. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	W	0h	DIO21 event 0h = No effect 1h = Clears DIO21 in RIS register
20	DIO20	W	0h	DIO20 event 0h = No effect 1h = Clears DIO20 in RIS register
19	DIO19	W	0h	DIO19 event 0h = No effect 1h = Clears DIO19 in RIS register
18	DIO18	W	0h	DIO18 event 0h = No effect 1h = Clears DIO18 in RIS register
17	DIO17	W	0h	DIO17 event 0h = No effect 1h = Clears DIO17 in RIS register
16	DIO16	W	0h	DIO16 event 0h = No effect 1h = Clears DIO16 in RIS register
15	DIO15	W	0h	DIO15 event 0h = No effect 1h = Clears DIO15 in RIS register
14	DIO14	W	0h	DIO14 event 0h = No effect 1h = Clears DIO14 in RIS register
13	DIO13	W	0h	DIO13 event 0h = No effect 1h = Clears DIO13 in RIS register
12	DIO12	W	0h	DIO12 event 0h = No effect 1h = Clears DIO12 in RIS register
11	DIO11	W	0h	DIO11 event 0h = No effect 1h = Clears DIO11 in RIS register
10	DIO10	W	0h	DIO10 event 0h = No effect 1h = Clears DIO10 in RIS register
9	DIO9	W	0h	DIO9 event 0h = No effect 1h = Clears DIO9 in RIS register
8	DIO8	W	0h	DIO8 event 0h = No effect 1h = Clears DIO8 in RIS register
7	DIO7	W	0h	DIO7 event 0h = No effect 1h = Clears DIO7 in RIS register
6	DIO6	W	0h	DIO6 event 0h = No effect 1h = Clears DIO6 in RIS register
5	DIO5	W	0h	DIO5 event 0h = No effect 1h = Clears DIO5 in RIS register
4	DIO4	W	0h	DIO4 event 0h = No effect 1h = Clears DIO4 in RIS register
3	DIO3	W	0h	DIO3 event 0h = No effect 1h = Clears DIO3 in RIS register

Table 8-18. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	W	0h	DIO2 event 0h = No effect 1h = Clears DIO2 in RIS register
1	DIO1	W	0h	DIO1 event 0h = No effect 1h = Clears DIO1 in RIS register
0	DIO0	W	0h	DIO0 event 0h = No effect 1h = Clears DIO0 in RIS register

8.3.16 IIDX (Offset = 1050h) [Reset = 00000000h]

IIDX is shown in Figure 8-19 and described in Table 8-19.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 8-19. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 8-19. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 0h = No bit is set means there is no pending interrupt request 1h = DIO0 interrupt 2h = DIO1 interrupt 3h = DIO2 interrupt 4h = DIO3 interrupt 5h = DIO4 interrupt 6h = DIO5 interrupt 7h = DIO6 interrupt 8h = DIO7 interrupt 9h = DIO8 interrupt Ah = DIO9 interrupt Bh = DIO10 interrupt Ch = DIO11 interrupt Dh = DIO12 interrupt Eh = DIO13 interrupt Fh = DIO14 interrupt 10h = DIO15 interrupt

8.3.17 IMASK (Offset = 1058h) [Reset = 00000000h]

IMASK is shown in Figure 8-20 and described in Table 8-20.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 8-20. IMASK

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-20. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	
15	DIO15	R/W	0h	DIO15 event mask 0h = Event is masked 1h = Event is unmasked
14	DIO14	R/W	0h	DIO14 event mask 0h = Event is masked 1h = Event is unmasked
13	DIO13	R/W	0h	DIO13 event mask 0h = Event is masked 1h = Event is unmasked
12	DIO12	R/W	0h	DIO12 event mask 0h = Event is masked 1h = Event is unmasked
11	DIO11	R/W	0h	DIO11 event mask 0h = Event is masked 1h = Event is unmasked
10	DIO10	R/W	0h	DIO10 event mask 0h = Event is masked 1h = Event is unmasked
9	DIO9	R/W	0h	DIO9 event mask 0h = Event is masked 1h = Event is unmasked
8	DIO8	R/W	0h	DIO8 event mask 0h = Event is masked 1h = Event is unmasked
7	DIO7	R/W	0h	DIO7 event mask 0h = Event is masked 1h = Event is unmasked
6	DIO6	R/W	0h	DIO6 event mask 0h = Event is masked 1h = Event is unmasked

Table 8-20. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	DIO5	R/W	0h	DIO5 event mask 0h = Event is masked 1h = Event is unmasked
4	DIO4	R/W	0h	DIO4 event mask 0h = Event is masked 1h = Event is unmasked
3	DIO3	R/W	0h	DIO3 event mask 0h = Event is masked 1h = Event is unmasked
2	DIO2	R/W	0h	DIO2 event mask 0h = Event is masked 1h = Event is unmasked
1	DIO1	R/W	0h	DIO1 event mask 0h = Event is masked 1h = Event is unmasked
0	DIO0	R/W	0h	DIO0 event mask 0h = Event is masked 1h = Event is unmasked

8.3.18 RIS (Offset = 1060h) [Reset = 00000000h]

RIS is shown in Figure 8-21 and described in Table 8-21.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 8-21. RIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-21. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	DIO15	R	0h	DIO15 event 0h = DIO15 event did not occur 1h = DIO15 event occurred
14	DIO14	R	0h	DIO14 event 0h = DIO14 event did not occur 1h = DIO14 event occurred
13	DIO13	R	0h	DIO13 event 0h = DIO13 event did not occur 1h = DIO13 event occurred
12	DIO12	R	0h	DIO12 event 0h = DIO12 event did not occur 1h = DIO12 event occurred
11	DIO11	R	0h	DIO11 event 0h = DIO11 event did not occur 1h = DIO11 event occurred
10	DIO10	R	0h	DIO10 event 0h = DIO10 event did not occur 1h = DIO10 event occurred
9	DIO9	R	0h	DIO9 event 0h = DIO9 event did not occur 1h = DIO9 event occurred
8	DIO8	R	0h	DIO8 event 0h = DIO8 event did not occur 1h = DIO8 event occurred
7	DIO7	R	0h	DIO7 event 0h = DIO7 event did not occur 1h = DIO7 event occurred

Table 8-21. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DIO6	R	0h	DIO6 event 0h = DIO6 event did not occur 1h = DIO6 event occurred
5	DIO5	R	0h	DIO5 event 0h = DIO5 event did not occur 1h = DIO5 event occurred
4	DIO4	R	0h	DIO4 event 0h = DIO4 event did not occur 1h = DIO4 event occurred
3	DIO3	R	0h	DIO3 event 0h = DIO3 event did not occur 1h = DIO3 event occurred
2	DIO2	R	0h	DIO2 event 0h = DIO2 event did not occur 1h = DIO2 event occurred
1	DIO1	R	0h	DIO1 event 0h = DIO1 event did not occur 1h = DIO1 event occurred
0	DIO0	R	0h	DIO0 event 0h = DIO0 event did not occur 1h = DIO0 event occurred

8.3.19 MIS (Offset = 1068h) [Reset = 00000000h]

MIS is shown in Figure 8-22 and described in Table 8-22.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 8-22. MIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-22. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	DIO15	R	0h	DIO15 event 0h = DIO15 event did not occur 1h = DIO15 event occurred
14	DIO14	R	0h	DIO14 event 0h = DIO14 event did not occur 1h = DIO14 event occurred
13	DIO13	R	0h	DIO13 event 0h = DIO13 event did not occur 1h = DIO13 event occurred
12	DIO12	R	0h	DIO12 event 0h = DIO12 event did not occur 1h = DIO12 event occurred
11	DIO11	R	0h	DIO11 event 0h = DIO11 event did not occur 1h = DIO11 event occurred
10	DIO10	R	0h	DIO10 event 0h = DIO10 event did not occur 1h = DIO10 event occurred
9	DIO9	R	0h	DIO9 event 0h = DIO9 event did not occur 1h = DIO9 event occurred
8	DIO8	R	0h	DIO8 event 0h = DIO8 event did not occur 1h = DIO8 event occurred
7	DIO7	R	0h	DIO7 event 0h = DIO7 event did not occur 1h = DIO7 event occurred
6	DIO6	R	0h	DIO6 event 0h = DIO6 event did not occur 1h = DIO6 event occurred

Table 8-22. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	DIO5	R	0h	DIO5 event 0h = DIO5 event did not occur 1h = DIO5 event occurred
4	DIO4	R	0h	DIO4 event 0h = DIO4 event did not occur 1h = DIO4 event occurred
3	DIO3	R	0h	DIO3 event 0h = DIO3 event did not occur 1h = DIO3 event occurred
2	DIO2	R	0h	DIO2 event 0h = DIO2 event did not occur 1h = DIO2 event occurred
1	DIO1	R	0h	DIO1 event 0h = DIO1 event did not occur 1h = DIO1 event occurred
0	DIO0	R	0h	DIO0 event 0h = DIO0 event did not occur 1h = DIO0 event occurred

8.3.20 ISET (Offset = 1070h) [Reset = 00000000h]

ISET is shown in Figure 8-23 and described in Table 8-23.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 8-23. ISET

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-23. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	W	0h	
15	DIO15	W	0h	DIO15 event 0h = No effect 1h = Sets DIO15 in RIS register
14	DIO14	W	0h	DIO14 event 0h = No effect 1h = Sets DIO14 in RIS register
13	DIO13	W	0h	DIO13 event 0h = No effect 1h = Sets DIO13 in RIS register
12	DIO12	W	0h	DIO12 event 0h = No effect 1h = Sets DIO12 in RIS register
11	DIO11	W	0h	DIO11 event 0h = No effect 1h = Sets DIO11 in RIS register
10	DIO10	W	0h	DIO10 event 0h = No effect 1h = Sets DIO10 in RIS register
9	DIO9	W	0h	DIO9 event 0h = No effect 1h = Sets DIO9 in RIS register
8	DIO8	W	0h	DIO8 event 0h = No effect 1h = Sets DIO8 in RIS register
7	DIO7	W	0h	DIO7 event 0h = No effect 1h = Sets DIO7 in RIS register

Table 8-23. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DIO6	W	0h	DIO6 event 0h = No effect 1h = Sets DIO6 in RIS register
5	DIO5	W	0h	DIO5 event 0h = No effect 1h = Sets DIO5 in RIS register
4	DIO4	W	0h	DIO4 event 0h = No effect 1h = Sets DIO4 in RIS register
3	DIO3	W	0h	DIO3 event 0h = No effect 1h = Sets DIO3 in RIS register
2	DIO2	W	0h	DIO2 event 0h = No effect 1h = Sets DIO2 in RIS register
1	DIO1	W	0h	DIO1 event 0h = No effect 1h = Sets DIO1 in RIS register
0	DIO0	W	0h	DIO0 event 0h = No effect 1h = Sets DIO0 in RIS register

8.3.21 ICLR (Offset = 1078h) [Reset = 00000000h]

ICLR is shown in Figure 8-24 and described in Table 8-24.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 8-24. ICLR

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-24. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	W	0h	
15	DIO15	W	0h	DIO15 event 0h = No effect 1h = Clears DIO15 in RIS register
14	DIO14	W	0h	DIO14 event 0h = No effect 1h = Clears DIO14 in RIS register
13	DIO13	W	0h	DIO13 event 0h = No effect 1h = Clears DIO13 in RIS register
12	DIO12	W	0h	DIO12 event 0h = No effect 1h = Clears DIO12 in RIS register
11	DIO11	W	0h	DIO11 event 0h = No effect 1h = Clears DIO11 in RIS register
10	DIO10	W	0h	DIO10 event 0h = No effect 1h = Clears DIO10 in RIS register
9	DIO9	W	0h	DIO9 event 0h = No effect 1h = Clears DIO9 in RIS register
8	DIO8	W	0h	DIO8 event 0h = No effect 1h = Clears DIO8 in RIS register
7	DIO7	W	0h	DIO7 event 0h = No effect 1h = Clears DIO7 in RIS register
6	DIO6	W	0h	DIO6 event 0h = No effect 1h = Clears DIO6 in RIS register

Table 8-24. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	DIO5	W	0h	DIO5 event 0h = No effect 1h = Clears DIO5 in RIS register
4	DIO4	W	0h	DIO4 event 0h = No effect 1h = Clears DIO4 in RIS register
3	DIO3	W	0h	DIO3 event 0h = No effect 1h = Clears DIO3 in RIS register
2	DIO2	W	0h	DIO2 event 0h = No effect 1h = Clears DIO2 in RIS register
1	DIO1	W	0h	DIO1 event 0h = No effect 1h = Clears DIO1 in RIS register
0	DIO0	W	0h	DIO0 event 0h = No effect 1h = Clears DIO0 in RIS register

8.3.22 IIDX (Offset = 1080h) [Reset = 00000000h]

IIDX is shown in Figure 8-25 and described in Table 8-25.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 8-25. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 8-25. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 0h = No bit is set means there is no pending interrupt request 1h = DIO0 interrupt 2h = DIO1 interrupt 3h = DIO2 interrupt 4h = DIO3 interrupt 5h = DIO4 interrupt 6h = DIO5 interrupt 7h = DIO6 interrupt 8h = DIO7 interrupt 9h = DIO8 interrupt Ah = DIO9 interrupt Bh = DIO10 interrupt Ch = DIO11 interrupt Dh = DIO12 interrupt Eh = DIO13 interrupt Fh = DIO14 interrupt 10h = DIO15 interrupt

8.3.23 IMASK (Offset = 1088h) [Reset = 00000000h]

IMASK is shown in Figure 8-26 and described in Table 8-26.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 8-26. IMASK

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

Table 8-26. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	DIO31 event mask 0h = Event is masked 1h = Event is unmasked
30	DIO30	R/W	0h	DIO30 event mask 0h = Event is masked 1h = Event is unmasked
29	DIO29	R/W	0h	DIO29 event mask 0h = Event is masked 1h = Event is unmasked
28	DIO28	R/W	0h	DIO28 event mask 0h = Event is masked 1h = Event is unmasked
27	DIO27	R/W	0h	DIO27 event mask 0h = Event is masked 1h = Event is unmasked
26	DIO26	R/W	0h	DIO26 event mask 0h = Event is masked 1h = Event is unmasked
25	DIO25	R/W	0h	DIO25 event mask 0h = Event is masked 1h = Event is unmasked
24	DIO24	R/W	0h	DIO24 event mask 0h = Event is masked 1h = Event is unmasked
23	DIO23	R/W	0h	DIO23 event mask 0h = Event is masked 1h = Event is unmasked
22	DIO22	R/W	0h	DIO22 event mask 0h = Event is masked 1h = Event is unmasked

Table 8-26. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R/W	0h	DIO21 event mask 0h = Event is masked 1h = Event is unmasked
20	DIO20	R/W	0h	DIO20 event mask 0h = Event is masked 1h = Event is unmasked
19	DIO19	R/W	0h	DIO19 event mask 0h = Event is masked 1h = Event is unmasked
18	DIO18	R/W	0h	DIO18 event mask 0h = Event is masked 1h = Event is unmasked
17	DIO17	R/W	0h	DIO17 event mask 0h = Event is masked 1h = Event is unmasked
16	DIO16	R/W	0h	DIO16 event mask 0h = Event is masked 1h = Event is unmasked
15-0	RESERVED	R/W	0h	

8.3.24 RIS (Offset = 1090h) [Reset = 00000000h]

RIS is shown in Figure 8-27 and described in Table 8-27.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 8-27. RIS

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 8-27. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R	0h	DIO31 event 0h = DIO31 event did not occur 1h = DIO31 event occurred
30	DIO30	R	0h	DIO30 event 0h = DIO30 event did not occur 1h = DIO30 event occurred
29	DIO29	R	0h	DIO29 event 0h = DIO29 event did not occur 1h = DIO29 event occurred
28	DIO28	R	0h	DIO28 event 0h = DIO28 event did not occur 1h = DIO28 event occurred
27	DIO27	R	0h	DIO27 event 0h = DIO27 event did not occur 1h = DIO27 event occurred
26	DIO26	R	0h	DIO26 event 0h = DIO26 event did not occur 1h = DIO26 event occurred
25	DIO25	R	0h	DIO25 event 0h = DIO25 event did not occur 1h = DIO25 event occurred
24	DIO24	R	0h	DIO24 event 0h = DIO24 event did not occur 1h = DIO24 event occurred
23	DIO23	R	0h	DIO23 event 0h = DIO23 event did not occur 1h = DIO23 event occurred
22	DIO22	R	0h	DIO22 event 0h = DIO22 event did not occur 1h = DIO22 event occurred

Table 8-27. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R	0h	DIO21 event 0h = DIO21 event did not occur 1h = DIO21 event occurred
20	DIO20	R	0h	DIO20 event 0h = DIO20 event did not occur 1h = DIO20 event occurred
19	DIO19	R	0h	DIO19 event 0h = DIO19 event did not occur 1h = DIO19 event occurred
18	DIO18	R	0h	DIO18 event 0h = DIO18 event did not occur 1h = DIO18 event occurred
17	DIO17	R	0h	DIO17 event 0h = DIO17 event did not occur 1h = DIO17 event occurred
16	DIO16	R	0h	DIO16 event 0h = DIO16 event did not occur 1h = DIO16 event occurred
15-0	RESERVED	R	0h	

8.3.25 MIS (Offset = 1098h) [Reset = 00000000h]

MIS is shown in Figure 8-28 and described in Table 8-28.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 8-28. MIS

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 8-28. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R	0h	DIO31 event 0h = DIO31 event did not occur 1h = DIO31 event occurred
30	DIO30	R	0h	DIO30 event 0h = DIO30 event did not occur 1h = DIO30 event occurred
29	DIO29	R	0h	DIO29 event 0h = DIO29 event did not occur 1h = DIO29 event occurred
28	DIO28	R	0h	DIO28 event 0h = DIO28 event did not occur 1h = DIO28 event occurred
27	DIO27	R	0h	DIO27 event 0h = DIO27 event did not occur 1h = DIO27 event occurred
26	DIO26	R	0h	DIO26 event 0h = DIO26 event did not occur 1h = DIO26 event occurred
25	DIO25	R	0h	DIO25 event 0h = DIO25 event did not occur 1h = DIO25 event occurred
24	DIO24	R	0h	DIO24 event 0h = DIO24 event did not occur 1h = DIO24 event occurred
23	DIO23	R	0h	DIO23 event 0h = DIO23 event did not occur 1h = DIO23 event occurred
22	DIO22	R	0h	DIO22 event 0h = DIO22 event did not occur 1h = DIO22 event occurred

Table 8-28. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R	0h	DIO21 event 0h = DIO21 event did not occur 1h = DIO21 event occurred
20	DIO20	R	0h	DIO20 event 0h = DIO20 event did not occur 1h = DIO20 event occurred
19	DIO19	R	0h	DIO19 event 0h = DIO19 event did not occur 1h = DIO19 event occurred
18	DIO18	R	0h	DIO18 event 0h = DIO18 event did not occur 1h = DIO18 event occurred
17	DIO17	R	0h	DIO17 event 0h = DIO17 event did not occur 1h = DIO17 event occurred
16	DIO16	R	0h	DIO16 event 0h = DIO16 event did not occur 1h = DIO16 event occurred
15-0	RESERVED	R	0h	

8.3.26 ISET (Offset = 10A0h) [Reset = 00000000h]

ISET is shown in Figure 8-29 and described in Table 8-29.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 8-29. ISET

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							
W-0h							

Table 8-29. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	DIO31 event 0h = No effect 1h = Sets DIO31 in RIS register
30	DIO30	W	0h	DIO30 event 0h = No effect 1h = Sets DIO30 in RIS register
29	DIO29	W	0h	DIO29 event 0h = No effect 1h = Sets DIO29 in RIS register
28	DIO28	W	0h	DIO28 event 0h = No effect 1h = Sets DIO28 in RIS register
27	DIO27	W	0h	DIO27 event 0h = No effect 1h = Sets DIO27 in RIS register
26	DIO26	W	0h	DIO26 event 0h = No effect 1h = Sets DIO26 in RIS register
25	DIO25	W	0h	DIO25 event 0h = No effect 1h = Sets DIO25 in RIS register
24	DIO24	W	0h	DIO24 event 0h = No effect 1h = Sets DIO24 in RIS register
23	DIO23	W	0h	DIO23 event 0h = No effect 1h = Sets DIO23 in RIS register
22	DIO22	W	0h	DIO22 event 0h = No effect 1h = Sets DIO22 in RIS register

Table 8-29. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	W	0h	DIO21 event 0h = No effect 1h = Sets DIO21 in RIS register
20	DIO20	W	0h	DIO20 event 0h = No effect 1h = Sets DIO20 in RIS register
19	DIO19	W	0h	DIO19 event 0h = No effect 1h = Sets DIO19 in RIS register
18	DIO18	W	0h	DIO18 event 0h = No effect 1h = Sets DIO18 in RIS register
17	DIO17	W	0h	DIO17 event 0h = No effect 1h = Sets DIO17 in RIS register
16	DIO16	W	0h	DIO16 event 0h = No effect 1h = Sets DIO16 in RIS register
15-0	RESERVED	W	0h	

8.3.27 ICLR (Offset = 10A8h) [Reset = 00000000h]

ICLR is shown in Figure 8-30 and described in Table 8-30.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 8-30. ICLR

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							
W-0h							

Table 8-30. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	DIO31 event 0h = No effect 1h = Clears DIO31 in RIS register
30	DIO30	W	0h	DIO30 event 0h = No effect 1h = Clears DIO30 in RIS register
29	DIO29	W	0h	DIO29 event 0h = No effect 1h = Clears DIO29 in RIS register
28	DIO28	W	0h	DIO28 event 0h = No effect 1h = Clears DIO28 in RIS register
27	DIO27	W	0h	DIO27 event 0h = No effect 1h = Clears DIO27 in RIS register
26	DIO26	W	0h	DIO26 event 0h = No effect 1h = Clears DIO26 in RIS register
25	DIO25	W	0h	DIO25 event 0h = No effect 1h = Clears DIO25 in RIS register
24	DIO24	W	0h	DIO24 event 0h = No effect 1h = Clears DIO24 in RIS register
23	DIO23	W	0h	DIO23 event 0h = No effect 1h = Clears DIO23 in RIS register
22	DIO22	W	0h	DIO22 event 0h = No effect 1h = Clears DIO22 in RIS register

Table 8-30. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	W	0h	DIO21 event 0h = No effect 1h = Clears DIO21 in RIS register
20	DIO20	W	0h	DIO20 event 0h = No effect 1h = Clears DIO20 in RIS register
19	DIO19	W	0h	DIO19 event 0h = No effect 1h = Clears DIO19 in RIS register
18	DIO18	W	0h	DIO18 event 0h = No effect 1h = Clears DIO18 in RIS register
17	DIO17	W	0h	DIO17 event 0h = No effect 1h = Clears DIO17 in RIS register
16	DIO16	W	0h	DIO16 event 0h = No effect 1h = Clears DIO16 in RIS register
15-0	RESERVED	W	0h	

8.3.28 EVT_MODE (Offset = 10E0h) [Reset = 00000029h]

EVT_MODE is shown in Figure 8-31 and described in Table 8-31.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 8-31. EVT_MODE

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED		EVT2_CFG		EVT1_CFG		INT0_CFG	
R/W-		R-2h		R-2h		R-1h	

Table 8-31. EVT_MODE Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	
5-4	EVT2_CFG	R	2h	Event line mode select for event corresponding to none.GEN_EVENT1 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
3-2	EVT1_CFG	R	2h	Event line mode select for event corresponding to none.GEN_EVENT0 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
1-0	INT0_CFG	R	1h	Event line mode select for event corresponding to none.CPU_INT 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

8.3.29 DESC (Offset = 10FCh) [Reset = 16110000h]

DESC is shown in Figure 8-32 and described in Table 8-32.

Return to the Summary Table.

This register identifies the peripheral and its exact version.

Figure 8-32. DESC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-1611h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				RESERVED				MAJREV				MINREV			
R-				R-				R-				R-			

Table 8-32. DESC Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	1611h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness. 0h = Smallest value FFFFh = Highest possible value
15-12	FEATUREVER	R	0h	Feature Set for the module *instance* 0h = Smallest value Fh = Highest possible value
11-8	RESERVED	R	0h	
7-4	MAJREV	R	0h	Major rev of the IP 0h = Smallest value Fh = Highest possible value
3-0	MINREV	R	0h	Minor rev of the IP 0h = Smallest value Fh = Highest possible value

8.3.30 DOUT3_0 (Offset = 1200h) [Reset = 00000000h]

DOUT3_0 is shown in Figure 8-33 and described in Table 8-33.

Return to the Summary Table.

Data output for pins configured as DIO3 to DIO0. This is an alias register for byte access to bits 3 to 0 in DOUT31_0 register.

Figure 8-33. DOUT3_0

31	30	29	28	27	26	25	24
RESERVED							DIO3
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO2
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO1
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO0
W-0h							W-0h

Table 8-33. DOUT3_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO3	W	0h	This bit sets the value of the pin configured as DIO3 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO2	W	0h	This bit sets the value of the pin configured as DIO2 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO1	W	0h	This bit sets the value of the pin configured as DIO1 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO0	W	0h	This bit sets the value of the pin configured as DIO0 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.31 DOUT7_4 (Offset = 1204h) [Reset = 00000000h]

DOUT7_4 is shown in Figure 8-34 and described in Table 8-34.

Return to the Summary Table.

Data output for pins configured as DIO7 to DIO4. This is an alias register for byte access to bits 7 to 4 in DOUT31_0 register.

Figure 8-34. DOUT7_4

31	30	29	28	27	26	25	24
RESERVED							DIO7
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO6
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO5
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO4
W-0h							W-0h

Table 8-34. DOUT7_4 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO7	W	0h	This bit sets the value of the pin configured as DIO7 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO6	W	0h	This bit sets the value of the pin configured as DIO6 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO5	W	0h	This bit sets the value of the pin configured as DIO5 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO4	W	0h	This bit sets the value of the pin configured as DIO4 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.32 DOUT11_8 (Offset = 1208h) [Reset = 00000000h]

DOUT11_8 is shown in Figure 8-35 and described in Table 8-35.

Return to the Summary Table.

Data output for pins configured as DIO11 to DIO8. This is an alias register for byte access to bits 11 to 8 in DOUT31_0 register.

Figure 8-35. DOUT11_8

31	30	29	28	27	26	25	24
RESERVED							DIO11
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO10
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO9
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO8
W-0h							W-0h

Table 8-35. DOUT11_8 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO11	W	0h	This bit sets the value of the pin configured as DIO11 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO10	W	0h	This bit sets the value of the pin configured as DIO10 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO9	W	0h	This bit sets the value of the pin configured as DIO9 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO8	W	0h	This bit sets the value of the pin configured as DIO8 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.33 DOUT15_12 (Offset = 120Ch) [Reset = 00000000h]

DOUT15_12 is shown in Figure 8-36 and described in Table 8-36.

Return to the Summary Table.

Data output for pins configured as DIO15 to DIO12. This is an alias register for byte access to bits 15 to 12 in DOUT31_0 register.

Figure 8-36. DOUT15_12

31	30	29	28	27	26	25	24
RESERVED							DIO15
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO14
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO13
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO12
W-0h							W-0h

Table 8-36. DOUT15_12 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO15	W	0h	This bit sets the value of the pin configured as DIO15 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO14	W	0h	This bit sets the value of the pin configured as DIO14 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO13	W	0h	This bit sets the value of the pin configured as DIO13 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO12	W	0h	This bit sets the value of the pin configured as DIO12 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.34 DOUT19_16 (Offset = 1210h) [Reset = 00000000h]

DOUT19_16 is shown in Figure 8-37 and described in Table 8-37.

Return to the Summary Table.

Data output for pins configured as DIO19 to DIO16. This is an alias register for byte access to bits 19 to 16 in DOUT31_0 register.

Figure 8-37. DOUT19_16

31	30	29	28	27	26	25	24
RESERVED							DIO19
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO18
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO17
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO16
W-0h							W-0h

Table 8-37. DOUT19_16 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO19	W	0h	This bit sets the value of the pin configured as DIO19 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO18	W	0h	This bit sets the value of the pin configured as DIO18 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO17	W	0h	This bit sets the value of the pin configured as DIO17 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO16	W	0h	This bit sets the value of the pin configured as DIO16 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.35 DOUT23_20 (Offset = 1214h) [Reset = 00000000h]

DOUT23_20 is shown in Figure 8-38 and described in Table 8-38.

Return to the Summary Table.

Data output for pins configured as DIO23 to DIO20. This is an alias register for byte access to bits 23 to 20 in DOUT31_0 register.

Figure 8-38. DOUT23_20

31	30	29	28	27	26	25	24
RESERVED							DIO23
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO22
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO21
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO20
W-0h							W-0h

Table 8-38. DOUT23_20 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO23	W	0h	This bit sets the value of the pin configured as DIO23 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO22	W	0h	This bit sets the value of the pin configured as DIO22 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO21	W	0h	This bit sets the value of the pin configured as DIO21 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO20	W	0h	This bit sets the value of the pin configured as DIO20 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.36 DOUT27_24 (Offset = 1218h) [Reset = 00000000h]

DOUT27_24 is shown in Figure 8-39 and described in Table 8-39.

Return to the Summary Table.

Data output for pins configured as DIO27 to DIO24. This is an alias register for byte access to bits 27 to 24 in DOUT31_0 register.

Figure 8-39. DOUT27_24

31	30	29	28	27	26	25	24
RESERVED							DIO27
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO26
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO25
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO24
W-0h							W-0h

Table 8-39. DOUT27_24 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO27	W	0h	This bit sets the value of the pin configured as DIO27 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO26	W	0h	This bit sets the value of the pin configured as DIO26 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO25	W	0h	This bit sets the value of the pin configured as DIO25 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO24	W	0h	This bit sets the value of the pin configured as DIO24 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.37 DOUT31_28 (Offset = 121Ch) [Reset = 00000000h]

DOUT31_28 is shown in Figure 8-40 and described in Table 8-40.

Return to the Summary Table.

Data output for pins configured as DIO31 to DIO28. This is an alias register for byte access to bits 31 to 28 in DOUT31_0 register.

Figure 8-40. DOUT31_28

31	30	29	28	27	26	25	24
RESERVED							DIO31
W-0h							W-0h
23	22	21	20	19	18	17	16
RESERVED							DIO30
W-0h							W-0h
15	14	13	12	11	10	9	8
RESERVED							DIO29
W-0h							W-0h
7	6	5	4	3	2	1	0
RESERVED							DIO28
W-0h							W-0h

Table 8-40. DOUT31_28 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	W	0h	
24	DIO31	W	0h	This bit sets the value of the pin configured as DIO31 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
23-17	RESERVED	W	0h	
16	DIO30	W	0h	This bit sets the value of the pin configured as DIO30 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15-9	RESERVED	W	0h	
8	DIO29	W	0h	This bit sets the value of the pin configured as DIO29 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7-1	RESERVED	W	0h	
0	DIO28	W	0h	This bit sets the value of the pin configured as DIO28 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.38 DOUT31_0 (Offset = 1280h) [Reset = 00000000h]

DOUT31_0 is shown in Figure 8-41 and described in Table 8-41.

Return to the Summary Table.

Data output for pins configured as DIO31 to DIO0.

Figure 8-41. DOUT31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-41. DOUT31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	This bit sets the value of the pin configured as DIO31 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
30	DIO30	R/W	0h	This bit sets the value of the pin configured as DIO30 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
29	DIO29	R/W	0h	This bit sets the value of the pin configured as DIO29 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
28	DIO28	R/W	0h	This bit sets the value of the pin configured as DIO28 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
27	DIO27	R/W	0h	This bit sets the value of the pin configured as DIO27 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
26	DIO26	R/W	0h	This bit sets the value of the pin configured as DIO26 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
25	DIO25	R/W	0h	This bit sets the value of the pin configured as DIO25 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
24	DIO24	R/W	0h	This bit sets the value of the pin configured as DIO24 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

Table 8-41. DOUT31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	DIO23	R/W	0h	This bit sets the value of the pin configured as DIO23 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
22	DIO22	R/W	0h	This bit sets the value of the pin configured as DIO22 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
21	DIO21	R/W	0h	This bit sets the value of the pin configured as DIO21 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
20	DIO20	R/W	0h	This bit sets the value of the pin configured as DIO20 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
19	DIO19	R/W	0h	This bit sets the value of the pin configured as DIO19 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
18	DIO18	R/W	0h	This bit sets the value of the pin configured as DIO18 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
17	DIO17	R/W	0h	This bit sets the value of the pin configured as DIO17 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
16	DIO16	R/W	0h	This bit sets the value of the pin configured as DIO16 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
15	DIO15	R/W	0h	This bit sets the value of the pin configured as DIO15 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
14	DIO14	R/W	0h	This bit sets the value of the pin configured as DIO14 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
13	DIO13	R/W	0h	This bit sets the value of the pin configured as DIO13 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
12	DIO12	R/W	0h	This bit sets the value of the pin configured as DIO12 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
11	DIO11	R/W	0h	This bit sets the value of the pin configured as DIO11 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
10	DIO10	R/W	0h	This bit sets the value of the pin configured as DIO10 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

Table 8-41. DOUT31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DIO9	R/W	0h	This bit sets the value of the pin configured as DIO9 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
8	DIO8	R/W	0h	This bit sets the value of the pin configured as DIO8 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
7	DIO7	R/W	0h	This bit sets the value of the pin configured as DIO7 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
6	DIO6	R/W	0h	This bit sets the value of the pin configured as DIO6 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
5	DIO5	R/W	0h	This bit sets the value of the pin configured as DIO5 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
4	DIO4	R/W	0h	This bit sets the value of the pin configured as DIO4 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
3	DIO3	R/W	0h	This bit sets the value of the pin configured as DIO3 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
2	DIO2	R/W	0h	This bit sets the value of the pin configured as DIO2 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
1	DIO1	R/W	0h	This bit sets the value of the pin configured as DIO1 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1
0	DIO0	R/W	0h	This bit sets the value of the pin configured as DIO0 when the output is enabled through DOE31_0 register. 0h = Output is set to 0 1h = Output is set to 1

8.3.39 DOUTSET31_0 (Offset = 1290h) [Reset = 00000000h]

DOUTSET31_0 is shown in Figure 8-42 and described in Table 8-42.

Return to the Summary Table.

Writing 1 to a bit position in this register sets the corresponding bit in the DOUT31_0 register.

Figure 8-42. DOUTSET31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-42. DOUTSET31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	Writing 1 to this bit sets the DIO31 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO31 in DOUT31_0
30	DIO30	W	0h	Writing 1 to this bit sets the DIO30 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO30 in DOUT31_0
29	DIO29	W	0h	Writing 1 to this bit sets the DIO29 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO29 in DOUT31_0
28	DIO28	W	0h	Writing 1 to this bit sets the DIO28 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO28 in DOUT31_0
27	DIO27	W	0h	Writing 1 to this bit sets the DIO27 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO27 in DOUT31_0
26	DIO26	W	0h	Writing 1 to this bit sets the DIO26 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO26 in DOUT31_0
25	DIO25	W	0h	Writing 1 to this bit sets the DIO25 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO25 in DOUT31_0
24	DIO24	W	0h	Writing 1 to this bit sets the DIO24 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO24 in DOUT31_0

Table 8-42. DOUTSET31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	DIO23	W	0h	Writing 1 to this bit sets the DIO23 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO23 in DOUT31_0
22	DIO22	W	0h	Writing 1 to this bit sets the DIO22 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO22 in DOUT31_0
21	DIO21	W	0h	Writing 1 to this bit sets the DIO21 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO21 in DOUT31_0
20	DIO20	W	0h	Writing 1 to this bit sets the DIO20 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO20 in DOUT31_0
19	DIO19	W	0h	Writing 1 to this bit sets the DIO19 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO19 in DOUT31_0
18	DIO18	W	0h	Writing 1 to this bit sets the DIO18 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO18 in DOUT31_0
17	DIO17	W	0h	Writing 1 to this bit sets the DIO17 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO17 in DOUT31_0
16	DIO16	W	0h	Writing 1 to this bit sets the DIO16 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO16 in DOUT31_0
15	DIO15	W	0h	Writing 1 to this bit sets the DIO15 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO15 in DOUT31_0
14	DIO14	W	0h	Writing 1 to this bit sets the DIO14 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO14 in DOUT31_0
13	DIO13	W	0h	Writing 1 to this bit sets the DIO13 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO13 in DOUT31_0
12	DIO12	W	0h	Writing 1 to this bit sets the DIO12 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO12 in DOUT31_0
11	DIO11	W	0h	Writing 1 to this bit sets the DIO11 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO11 in DOUT31_0
10	DIO10	W	0h	Writing 1 to this bit sets the DIO10 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO10 in DOUT31_0

Table 8-42. DOUTSET31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DIO9	W	0h	Writing 1 to this bit sets the DIO9 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO9 in DOUT31_0
8	DIO8	W	0h	Writing 1 to this bit sets the DIO8 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO8 in DOUT31_0
7	DIO7	W	0h	Writing 1 to this bit sets the DIO7 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO7 in DOUT31_0
6	DIO6	W	0h	Writing 1 to this bit sets the DIO6 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO6 in DOUT31_0
5	DIO5	W	0h	Writing 1 to this bit sets the DIO5 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO5 in DOUT31_0
4	DIO4	W	0h	Writing 1 to this bit sets the DIO4 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO4 in DOUT31_0
3	DIO3	W	0h	Writing 1 to this bit sets the DIO3 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO3 in DOUT31_0
2	DIO2	W	0h	Writing 1 to this bit sets the DIO2 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO2 in DOUT31_0
1	DIO1	W	0h	Writing 1 to this bit sets the DIO1 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO1 in DOUT31_0
0	DIO0	W	0h	Writing 1 to this bit sets the DIO0 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO0 in DOUT31_0

8.3.40 DOUTCLR31_0 (Offset = 12A0h) [Reset = 00000000h]

DOUTCLR31_0 is shown in Figure 8-43 and described in Table 8-43.

Return to the Summary Table.

Writing 1 to a bit position in this register clears the corresponding bit in the DOUT31_0 register.

Figure 8-43. DOUTCLR31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-43. DOUTCLR31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	Writing 1 to this bit clears the DIO31 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO31 in DOUT31_0
30	DIO30	W	0h	Writing 1 to this bit clears the DIO30 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO30 in DOUT31_0
29	DIO29	W	0h	Writing 1 to this bit clears the DIO29 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO29 in DOUT31_0
28	DIO28	W	0h	Writing 1 to this bit clears the DIO28 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO28 in DOUT31_0
27	DIO27	W	0h	Writing 1 to this bit clears the DIO27 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO27 in DOUT31_0
26	DIO26	W	0h	Writing 1 to this bit clears the DIO26 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO26 in DOUT31_0
25	DIO25	W	0h	Writing 1 to this bit clears the DIO25 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO25 in DOUT31_0
24	DIO24	W	0h	Writing 1 to this bit clears the DIO24 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO24 in DOUT31_0

Table 8-43. DOUTCLR31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	DIO23	W	0h	Writing 1 to this bit clears the DIO23 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO23 in DOUT31_0
22	DIO22	W	0h	Writing 1 to this bit clears the DIO22 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO22 in DOUT31_0
21	DIO21	W	0h	Writing 1 to this bit clears the DIO21 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO21 in DOUT31_0
20	DIO20	W	0h	Writing 1 to this bit clears the DIO20 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO20 in DOUT31_0
19	DIO19	W	0h	Writing 1 to this bit clears the DIO19 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO19 in DOUT31_0
18	DIO18	W	0h	Writing 1 to this bit clears the DIO18 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO18 in DOUT31_0
17	DIO17	W	0h	Writing 1 to this bit clears the DIO17 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO17 in DOUT31_0
16	DIO16	W	0h	Writing 1 to this bit clears the DIO16 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO16 in DOUT31_0
15	DIO15	W	0h	Writing 1 to this bit clears the DIO15 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO15 in DOUT31_0
14	DIO14	W	0h	Writing 1 to this bit clears the DIO14 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO14 in DOUT31_0
13	DIO13	W	0h	Writing 1 to this bit clears the DIO13 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO13 in DOUT31_0
12	DIO12	W	0h	Writing 1 to this bit clears the DIO12 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO12 in DOUT31_0
11	DIO11	W	0h	Writing 1 to this bit clears the DIO11 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO11 in DOUT31_0
10	DIO10	W	0h	Writing 1 to this bit clears the DIO10 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO10 in DOUT31_0

Table 8-43. DOUTCLR31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DIO9	W	0h	Writing 1 to this bit clears the DIO9 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO9 in DOUT31_0
8	DIO8	W	0h	Writing 1 to this bit clears the DIO8 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO8 in DOUT31_0
7	DIO7	W	0h	Writing 1 to this bit clears the DIO7 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO7 in DOUT31_0
6	DIO6	W	0h	Writing 1 to this bit clears the DIO6 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO6 in DOUT31_0
5	DIO5	W	0h	Writing 1 to this bit clears the DIO5 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO5 in DOUT31_0
4	DIO4	W	0h	Writing 1 to this bit clears the DIO4 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO4 in DOUT31_0
3	DIO3	W	0h	Writing 1 to this bit clears the DIO3 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO3 in DOUT31_0
2	DIO2	W	0h	Writing 1 to this bit clears the DIO2 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO2 in DOUT31_0
1	DIO1	W	0h	Writing 1 to this bit clears the DIO1 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO1 in DOUT31_0
0	DIO0	W	0h	Writing 1 to this bit clears the DIO0 bit in the DOUT31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO0 in DOUT31_0

8.3.41 DOUTTGL31_0 (Offset = 12B0h) [Reset = 00000000h]

DOUTTGL31_0 is shown in Figure 8-44 and described in Table 8-44.

Return to the Summary Table.

Writing 1 to a bit position in this register will invert the corresponding DIO output.

Figure 8-44. DOUTTGL31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-44. DOUTTGL31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	This bit is used to toggle DIO31 output. 0h = No effect 1h = Toggle output
30	DIO30	W	0h	This bit is used to toggle DIO30 output. 0h = No effect 1h = Toggle output
29	DIO29	W	0h	This bit is used to toggle DIO29 output. 0h = No effect 1h = Toggle output
28	DIO28	W	0h	This bit is used to toggle DIO28 output. 0h = No effect 1h = Toggle output
27	DIO27	W	0h	This bit is used to toggle DIO27 output. 0h = No effect 1h = Toggle output
26	DIO26	W	0h	This bit is used to toggle DIO26 output. 0h = No effect 1h = Toggle output
25	DIO25	W	0h	This bit is used to toggle DIO25 output. 0h = No effect 1h = Toggle output
24	DIO24	W	0h	This bit is used to toggle DIO24 output. 0h = No effect 1h = Toggle output
23	DIO23	W	0h	This bit is used to toggle DIO23 output. 0h = No effect 1h = Toggle output
22	DIO22	W	0h	This bit is used to toggle DIO22 output. 0h = No effect 1h = Toggle output

Table 8-44. DOUTTGL31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	W	0h	This bit is used to toggle DIO21 output. 0h = No effect 1h = Toggle output
20	DIO20	W	0h	This bit is used to toggle DIO20 output. 0h = No effect 1h = Toggle output
19	DIO19	W	0h	This bit is used to toggle DIO19 output. 0h = No effect 1h = Toggle output
18	DIO18	W	0h	This bit is used to toggle DIO18 output. 0h = No effect 1h = Toggle output
17	DIO17	W	0h	This bit is used to toggle DIO17 output. 0h = No effect 1h = Toggle output
16	DIO16	W	0h	This bit is used to toggle DIO16 output. 0h = No effect 1h = Toggle output
15	DIO15	W	0h	This bit is used to toggle DIO15 output. 0h = No effect 1h = Toggle output
14	DIO14	W	0h	This bit is used to toggle DIO14 output. 0h = No effect 1h = Toggle output
13	DIO13	W	0h	This bit is used to toggle DIO13 output. 0h = No effect 1h = Toggle output
12	DIO12	W	0h	This bit is used to toggle DIO12 output. 0h = No effect 1h = Toggle output
11	DIO11	W	0h	This bit is used to toggle DIO11 output. 0h = No effect 1h = Toggle output
10	DIO10	W	0h	This bit is used to toggle DIO10 output. 0h = No effect 1h = Toggle output
9	DIO9	W	0h	This bit is used to toggle DIO9 output. 0h = No effect 1h = Toggle output
8	DIO8	W	0h	This bit is used to toggle DIO8 output. 0h = No effect 1h = Toggle output
7	DIO7	W	0h	This bit is used to toggle DIO7 output. 0h = No effect 1h = Toggle output
6	DIO6	W	0h	This bit is used to toggle DIO6 output. 0h = No effect 1h = Toggle output
5	DIO5	W	0h	This bit is used to toggle DIO5 output. 0h = No effect 1h = Toggle output
4	DIO4	W	0h	This bit is used to toggle DIO4 output. 0h = No effect 1h = Toggle output
3	DIO3	W	0h	This bit is used to toggle DIO3 output. 0h = No effect 1h = Toggle output

Table 8-44. DOUTTGL31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	W	0h	This bit is used to toggle DIO2 output. 0h = No effect 1h = Toggle output
1	DIO1	W	0h	This bit is used to toggle DIO1 output. 0h = No effect 1h = Toggle output
0	DIO0	W	0h	This bit is used to toggle DIO0 output. 0h = No effect 1h = Toggle output

8.3.42 DOE31_0 (Offset = 12C0h) [Reset = 00000000h]

DOE31_0 is shown in Figure 8-45 and described in Table 8-45.

Return to the Summary Table.

This register is used to enable the data outputs for DIO31 to DIO0.

Figure 8-45. DOE31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-45. DOE31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R/W	0h	Enables data output for DIO31. 0h = Output disabled 1h = Output enabled
30	DIO30	R/W	0h	Enables data output for DIO30. 0h = Output disabled 1h = Output enabled
29	DIO29	R/W	0h	Enables data output for DIO29. 0h = Output disabled 1h = Output enabled
28	DIO28	R/W	0h	Enables data output for DIO28. 0h = Output disabled 1h = Output enabled
27	DIO27	R/W	0h	Enables data output for DIO27. 0h = Output disabled 1h = Output enabled
26	DIO26	R/W	0h	Enables data output for DIO26. 0h = Output disabled 1h = Output enabled
25	DIO25	R/W	0h	Enables data output for DIO25. 0h = Output disabled 1h = Output enabled
24	DIO24	R/W	0h	Enables data output for DIO24. 0h = Output disabled 1h = Output enabled
23	DIO23	R/W	0h	Enables data output for DIO23. 0h = Output disabled 1h = Output enabled
22	DIO22	R/W	0h	Enables data output for DIO22. 0h = Output disabled 1h = Output enabled

Table 8-45. DOE31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R/W	0h	Enables data output for DIO21. 0h = Output disabled 1h = Output enabled
20	DIO20	R/W	0h	Enables data output for DIO20. 0h = Output disabled 1h = Output enabled
19	DIO19	R/W	0h	Enables data output for DIO19. 0h = Output disabled 1h = Output enabled
18	DIO18	R/W	0h	Enables data output for DIO18. 0h = Output disabled 1h = Output enabled
17	DIO17	R/W	0h	Enables data output for DIO17. 0h = Output disabled 1h = Output enabled
16	DIO16	R/W	0h	Enables data output for DIO16. 0h = Output disabled 1h = Output enabled
15	DIO15	R/W	0h	Enables data output for DIO15. 0h = Output disabled 1h = Output enabled
14	DIO14	R/W	0h	Enables data output for DIO14. 0h = Output disabled 1h = Output enabled
13	DIO13	R/W	0h	Enables data output for DIO13. 0h = Output disabled 1h = Output enabled
12	DIO12	R/W	0h	Enables data output for DIO12. 0h = Output disabled 1h = Output enabled
11	DIO11	R/W	0h	Enables data output for DIO11. 0h = Output disabled 1h = Output enabled
10	DIO10	R/W	0h	Enables data output for DIO10. 0h = Output disabled 1h = Output enabled
9	DIO9	R/W	0h	Enables data output for DIO9. 0h = Output disabled 1h = Output enabled
8	DIO8	R/W	0h	Enables data output for DIO8. 0h = Output disabled 1h = Output enabled
7	DIO7	R/W	0h	Enables data output for DIO7. 0h = Output disabled 1h = Output enabled
6	DIO6	R/W	0h	Enables data output for DIO6. 0h = Output disabled 1h = Output enabled
5	DIO5	R/W	0h	Enables data output for DIO5. 0h = Output disabled 1h = Output enabled
4	DIO4	R/W	0h	Enables data output for DIO4. 0h = Output disabled 1h = Output enabled
3	DIO3	R/W	0h	Enables data output for DIO3. 0h = Output disabled 1h = Output enabled

Table 8-45. DOE31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	R/W	0h	Enables data output for DIO2. 0h = Output disabled 1h = Output enabled
1	DIO1	R/W	0h	Enables data output for DIO1. 0h = Output disabled 1h = Output enabled
0	DIO0	R/W	0h	Enables data output for DIO0. 0h = Output disabled 1h = Output enabled

8.3.43 DOESET31_0 (Offset = 12D0h) [Reset = 00000000h]

DOESET31_0 is shown in Figure 8-46 and described in Table 8-46.

Return to the Summary Table.

Writing 1 to a bit position in this register sets the corresponding bit in the DOE31_0 register.

Figure 8-46. DOESET31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-46. DOESET31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	Writing 1 to this bit sets the DIO31 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO31 in DOE31_0
30	DIO30	W	0h	Writing 1 to this bit sets the DIO30 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO30 in DOE31_0
29	DIO29	W	0h	Writing 1 to this bit sets the DIO29 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO29 in DOE31_0
28	DIO28	W	0h	Writing 1 to this bit sets the DIO28 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO28 in DOE31_0
27	DIO27	W	0h	Writing 1 to this bit sets the DIO27 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO27 in DOE31_0
26	DIO26	W	0h	Writing 1 to this bit sets the DIO26 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO26 in DOE31_0
25	DIO25	W	0h	Writing 1 to this bit sets the DIO25 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO25 in DOE31_0
24	DIO24	W	0h	Writing 1 to this bit sets the DIO24 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO24 in DOE31_0

Table 8-46. DOESET31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	DIO23	W	0h	Writing 1 to this bit sets the DIO23 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO23 in DOE31_0
22	DIO22	W	0h	Writing 1 to this bit sets the DIO22 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO22 in DOE31_0
21	DIO21	W	0h	Writing 1 to this bit sets the DIO21 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO21 in DOE31_0
20	DIO20	W	0h	Writing 1 to this bit sets the DIO20 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO20 in DOE31_0
19	DIO19	W	0h	Writing 1 to this bit sets the DIO19 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO19 in DOE31_0
18	DIO18	W	0h	Writing 1 to this bit sets the DIO18 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO18 in DOE31_0
17	DIO17	W	0h	Writing 1 to this bit sets the DIO17 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO17 in DOE31_0
16	DIO16	W	0h	Writing 1 to this bit sets the DIO16 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO16 in DOE31_0
15	DIO15	W	0h	Writing 1 to this bit sets the DIO15 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO15 in DOE31_0
14	DIO14	W	0h	Writing 1 to this bit sets the DIO14 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO14 in DOE31_0
13	DIO13	W	0h	Writing 1 to this bit sets the DIO13 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO13 in DOE31_0
12	DIO12	W	0h	Writing 1 to this bit sets the DIO12 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO12 in DOE31_0
11	DIO11	W	0h	Writing 1 to this bit sets the DIO11 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO11 in DOE31_0
10	DIO10	W	0h	Writing 1 to this bit sets the DIO10 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO10 in DOE31_0

Table 8-46. DOESET31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DIO9	W	0h	Writing 1 to this bit sets the DIO9 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO9 in DOE31_0
8	DIO8	W	0h	Writing 1 to this bit sets the DIO8 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO8 in DOE31_0
7	DIO7	W	0h	Writing 1 to this bit sets the DIO7 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO7 in DOE31_0
6	DIO6	W	0h	Writing 1 to this bit sets the DIO6 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO6 in DOE31_0
5	DIO5	W	0h	Writing 1 to this bit sets the DIO5 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO5 in DOE31_0
4	DIO4	W	0h	Writing 1 to this bit sets the DIO4 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO4 in DOE31_0
3	DIO3	W	0h	Writing 1 to this bit sets the DIO3 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO3 in DOE31_0
2	DIO2	W	0h	Writing 1 to this bit sets the DIO2 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO2 in DOE31_0
1	DIO1	W	0h	Writing 1 to this bit sets the DIO1 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO1 in DOE31_0
0	DIO0	W	0h	Writing 1 to this bit sets the DIO0 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Sets DIO0 in DOE31_0

8.3.44 DOECLR31_0 (Offset = 12E0h) [Reset = 00000000h]

DOECLR31_0 is shown in Figure 8-47 and described in Table 8-47.

Return to the Summary Table.

Writing 1 to a bit position in this register clears the corresponding bit in the DOE31_0 register.

Figure 8-47. DOECLR31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 8-47. DOECLR31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	W	0h	Writing 1 to this bit clears the DIO31 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO31 in DOE31_0
30	DIO30	W	0h	Writing 1 to this bit clears the DIO30 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO30 in DOE31_0
29	DIO29	W	0h	Writing 1 to this bit clears the DIO29 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO29 in DOE31_0
28	DIO28	W	0h	Writing 1 to this bit clears the DIO28 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO28 in DOE31_0
27	DIO27	W	0h	Writing 1 to this bit clears the DIO27 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO27 in DOE31_0
26	DIO26	W	0h	Writing 1 to this bit clears the DIO26 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO26 in DOE31_0
25	DIO25	W	0h	Writing 1 to this bit clears the DIO25 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO25 in DOE31_0
24	DIO24	W	0h	Writing 1 to this bit clears the DIO24 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO24 in DOE31_0

Table 8-47. DOECLR31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	DIO23	W	0h	Writing 1 to this bit clears the DIO23 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO23 in DOE31_0
22	DIO22	W	0h	Writing 1 to this bit clears the DIO22 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO22 in DOE31_0
21	DIO21	W	0h	Writing 1 to this bit clears the DIO21 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO21 in DOE31_0
20	DIO20	W	0h	Writing 1 to this bit clears the DIO20 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO20 in DOE31_0
19	DIO19	W	0h	Writing 1 to this bit clears the DIO19 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO19 in DOE31_0
18	DIO18	W	0h	Writing 1 to this bit clears the DIO18 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO18 in DOE31_0
17	DIO17	W	0h	Writing 1 to this bit clears the DIO17 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO17 in DOE31_0
16	DIO16	W	0h	Writing 1 to this bit clears the DIO16 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO16 in DOE31_0
15	DIO15	W	0h	Writing 1 to this bit clears the DIO15 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO15 in DOE31_0
14	DIO14	W	0h	Writing 1 to this bit clears the DIO14 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO14 in DOE31_0
13	DIO13	W	0h	Writing 1 to this bit clears the DIO13 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO13 in DOE31_0
12	DIO12	W	0h	Writing 1 to this bit clears the DIO12 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO12 in DOE31_0
11	DIO11	W	0h	Writing 1 to this bit clears the DIO11 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO11 in DOE31_0
10	DIO10	W	0h	Writing 1 to this bit clears the DIO10 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO10 in DOE31_0

Table 8-47. DOECLR31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	DIO9	W	0h	Writing 1 to this bit clears the DIO9 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO9 in DOE31_0
8	DIO8	W	0h	Writing 1 to this bit clears the DIO8 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO8 in DOE31_0
7	DIO7	W	0h	Writing 1 to this bit clears the DIO7 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO7 in DOE31_0
6	DIO6	W	0h	Writing 1 to this bit clears the DIO6 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO6 in DOE31_0
5	DIO5	W	0h	Writing 1 to this bit clears the DIO5 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO5 in DOE31_0
4	DIO4	W	0h	Writing 1 to this bit clears the DIO4 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO4 in DOE31_0
3	DIO3	W	0h	Writing 1 to this bit clears the DIO3 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO3 in DOE31_0
2	DIO2	W	0h	Writing 1 to this bit clears the DIO2 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO2 in DOE31_0
1	DIO1	W	0h	Writing 1 to this bit clears the DIO1 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO1 in DOE31_0
0	DIO0	W	0h	Writing 1 to this bit clears the DIO0 bit in the DOE31_0 register. Writing 0 has no effect. 0h = No effect 1h = Clears DIO0 in DOE31_0

8.3.45 DIN3_0 (Offset = 1300h) [Reset = 00000000h]

DIN3_0 is shown in Figure 8-48 and described in Table 8-48.

Return to the Summary Table.

Data input from pins configured as DIO3 to DIO0. This is an alias register for byte access to bits 3 to 0 in DIN31_0 register.

Figure 8-48. DIN3_0

31	30	29	28	27	26	25	24
RESERVED							DIO3
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO2
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO1
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO0
R-0h							R-0h

Table 8-48. DIN3_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO3	R	0h	This bit reads the data input value of DIO3. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO2	R	0h	This bit reads the data input value of DIO2. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO1	R	0h	This bit reads the data input value of DIO1. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO0	R	0h	This bit reads the data input value of DIO0. 0h = Input value is 0 1h = Input value is 1

8.3.46 DIN7_4 (Offset = 1304h) [Reset = 00000000h]

DIN7_4 is shown in Figure 8-49 and described in Table 8-49.

Return to the Summary Table.

Data input from pins configured as DIO7 to DIO4. This is an alias register for byte access to bits 7 to 4 in DIN31_0 register.

Figure 8-49. DIN7_4

31	30	29	28	27	26	25	24
RESERVED							DIO7
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO6
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO5
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO4
R-0h							R-0h

Table 8-49. DIN7_4 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO7	R	0h	This bit reads the data input value of DIO7. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO6	R	0h	This bit reads the data input value of DIO6. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO5	R	0h	This bit reads the data input value of DIO5. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO4	R	0h	This bit reads the data input value of DIO4. 0h = Input value is 0 1h = Input value is 1

8.3.47 DIN11_8 (Offset = 1308h) [Reset = 00000000h]

DIN11_8 is shown in Figure 8-50 and described in Table 8-50.

Return to the Summary Table.

Data input from pins configured as DIO11 to DIO8. This is an alias register for byte access to bits 11 to 8 in DIN31_0 register.

Figure 8-50. DIN11_8

31	30	29	28	27	26	25	24
RESERVED							DIO11
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO10
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO9
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO8
R-0h							R-0h

Table 8-50. DIN11_8 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO11	R	0h	This bit reads the data input value of DIO11. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO10	R	0h	This bit reads the data input value of DIO10. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO9	R	0h	This bit reads the data input value of DIO9. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO8	R	0h	This bit reads the data input value of DIO8. 0h = Input value is 0 1h = Input value is 1

8.3.48 DIN15_12 (Offset = 130Ch) [Reset = 00000000h]

DIN15_12 is shown in Figure 8-51 and described in Table 8-51.

Return to the Summary Table.

Data input from pins configured as DIO15 to DIO12. This is an alias register for byte access to bits 15 to 12 in DIN31_0 register.

Figure 8-51. DIN15_12

31	30	29	28	27	26	25	24
RESERVED							DIO15
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO14
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO13
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO12
R-0h							R-0h

Table 8-51. DIN15_12 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO15	R	0h	This bit reads the data input value of DIO15. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO14	R	0h	This bit reads the data input value of DIO14. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO13	R	0h	This bit reads the data input value of DIO13. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO12	R	0h	This bit reads the data input value of DIO12. 0h = Input value is 0 1h = Input value is 1

8.3.49 DIN19_16 (Offset = 1310h) [Reset = 00000000h]

DIN19_16 is shown in Figure 8-52 and described in Table 8-52.

Return to the Summary Table.

Data input from pins configured as DIO19 to DIO16. This is an alias register for byte access to bits 19 to 16 in DIN31_0 register.

Figure 8-52. DIN19_16

31	30	29	28	27	26	25	24
RESERVED							DIO19
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO18
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO17
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO16
R-0h							R-0h

Table 8-52. DIN19_16 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO19	R	0h	This bit reads the data input value of DIO19. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO18	R	0h	This bit reads the data input value of DIO18. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO17	R	0h	This bit reads the data input value of DIO17. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO16	R	0h	This bit reads the data input value of DIO16. 0h = Input value is 0 1h = Input value is 1

8.3.50 DIN23_20 (Offset = 1314h) [Reset = 00000000h]

DIN23_20 is shown in Figure 8-53 and described in Table 8-53.

Return to the Summary Table.

Data input from pins configured as DIO23 to DIO20. This is an alias register for byte access to bits 23 to 20 in DIN31_0 register.

Figure 8-53. DIN23_20

31	30	29	28	27	26	25	24
RESERVED							DIO23
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO22
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO21
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO20
R-0h							R-0h

Table 8-53. DIN23_20 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO23	R	0h	This bit reads the data input value of DIO23. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO22	R	0h	This bit reads the data input value of DIO22. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO21	R	0h	This bit reads the data input value of DIO21. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO20	R	0h	This bit reads the data input value of DIO20. 0h = Input value is 0 1h = Input value is 1

8.3.51 DIN27_24 (Offset = 1318h) [Reset = 00000000h]

DIN27_24 is shown in Figure 8-54 and described in Table 8-54.

Return to the Summary Table.

Data input from pins configured as DIO27 to DIO24. This is an alias register for byte access to bits 27 to 24 in DIN31_0 register.

Figure 8-54. DIN27_24

31	30	29	28	27	26	25	24
RESERVED							DIO27
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO26
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO25
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO24
R-0h							R-0h

Table 8-54. DIN27_24 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO27	R	0h	This bit reads the data input value of DIO27. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO26	R	0h	This bit reads the data input value of DIO26. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO25	R	0h	This bit reads the data input value of DIO25. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO24	R	0h	This bit reads the data input value of DIO24. 0h = Input value is 0 1h = Input value is 1

8.3.52 DIN31_28 (Offset = 131Ch) [Reset = 00000000h]

DIN31_28 is shown in Figure 8-55 and described in Table 8-55.

Return to the Summary Table.

Data input from pins configured as DIO31 to DIO28. This is an alias register for byte access to bits 31 to 28 in DIN31_0 register.

Figure 8-55. DIN31_28

31	30	29	28	27	26	25	24
RESERVED							DIO31
R-0h							R-0h
23	22	21	20	19	18	17	16
RESERVED							DIO30
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED							DIO29
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							DIO28
R-0h							R-0h

Table 8-55. DIN31_28 Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	
24	DIO31	R	0h	This bit reads the data input value of DIO31. 0h = Input value is 0 1h = Input value is 1
23-17	RESERVED	R	0h	
16	DIO30	R	0h	This bit reads the data input value of DIO30. 0h = Input value is 0 1h = Input value is 1
15-9	RESERVED	R	0h	
8	DIO29	R	0h	This bit reads the data input value of DIO29. 0h = Input value is 0 1h = Input value is 1
7-1	RESERVED	R	0h	
0	DIO28	R	0h	This bit reads the data input value of DIO28. 0h = Input value is 0 1h = Input value is 1

8.3.53 DIN31_0 (Offset = 1380h) [Reset = 00000000h]

DIN31_0 is shown in Figure 8-56 and described in Table 8-56.

Return to the Summary Table.

Data input value for pins configured as DIO31 to DIO0.

Figure 8-56. DIN31_0

31	30	29	28	27	26	25	24
DIO31	DIO30	DIO29	DIO28	DIO27	DIO26	DIO25	DIO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
DIO23	DIO22	DIO21	DIO20	DIO19	DIO18	DIO17	DIO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
DIO15	DIO14	DIO13	DIO12	DIO11	DIO10	DIO9	DIO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 8-56. DIN31_0 Field Descriptions

Bit	Field	Type	Reset	Description
31	DIO31	R	0h	This bit reads the data input value of DIO31. 0h = Input value is 0 1h = Input value is 1
30	DIO30	R	0h	This bit reads the data input value of DIO30. 0h = Input value is 0 1h = Input value is 1
29	DIO29	R	0h	This bit reads the data input value of DIO29. 0h = Input value is 0 1h = Input value is 1
28	DIO28	R	0h	This bit reads the data input value of DIO28. 0h = Input value is 0 1h = Input value is 1
27	DIO27	R	0h	This bit reads the data input value of DIO27. 0h = Input value is 0 1h = Input value is 1
26	DIO26	R	0h	This bit reads the data input value of DIO26. 0h = Input value is 0 1h = Input value is 1
25	DIO25	R	0h	This bit reads the data input value of DIO25. 0h = Input value is 0 1h = Input value is 1
24	DIO24	R	0h	This bit reads the data input value of DIO24. 0h = Input value is 0 1h = Input value is 1
23	DIO23	R	0h	This bit reads the data input value of DIO23. 0h = Input value is 0 1h = Input value is 1
22	DIO22	R	0h	This bit reads the data input value of DIO22. 0h = Input value is 0 1h = Input value is 1

Table 8-56. DIN31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIO21	R	0h	This bit reads the data input value of DIO21. 0h = Input value is 0 1h = Input value is 1
20	DIO20	R	0h	This bit reads the data input value of DIO20. 0h = Input value is 0 1h = Input value is 1
19	DIO19	R	0h	This bit reads the data input value of DIO19. 0h = Input value is 0 1h = Input value is 1
18	DIO18	R	0h	This bit reads the data input value of DIO18. 0h = Input value is 0 1h = Input value is 1
17	DIO17	R	0h	This bit reads the data input value of DIO17. 0h = Input value is 0 1h = Input value is 1
16	DIO16	R	0h	This bit reads the data input value of DIO16. 0h = Input value is 0 1h = Input value is 1
15	DIO15	R	0h	This bit reads the data input value of DIO15. 0h = Input value is 0 1h = Input value is 1
14	DIO14	R	0h	This bit reads the data input value of DIO14. 0h = Input value is 0 1h = Input value is 1
13	DIO13	R	0h	This bit reads the data input value of DIO13. 0h = Input value is 0 1h = Input value is 1
12	DIO12	R	0h	This bit reads the data input value of DIO12. 0h = Input value is 0 1h = Input value is 1
11	DIO11	R	0h	This bit reads the data input value of DIO11. 0h = Input value is 0 1h = Input value is 1
10	DIO10	R	0h	This bit reads the data input value of DIO10. 0h = Input value is 0 1h = Input value is 1
9	DIO9	R	0h	This bit reads the data input value of DIO9. 0h = Input value is 0 1h = Input value is 1
8	DIO8	R	0h	This bit reads the data input value of DIO8. 0h = Input value is 0 1h = Input value is 1
7	DIO7	R	0h	This bit reads the data input value of DIO7. 0h = Input value is 0 1h = Input value is 1
6	DIO6	R	0h	This bit reads the data input value of DIO6. 0h = Input value is 0 1h = Input value is 1
5	DIO5	R	0h	This bit reads the data input value of DIO5. 0h = Input value is 0 1h = Input value is 1
4	DIO4	R	0h	This bit reads the data input value of DIO4. 0h = Input value is 0 1h = Input value is 1
3	DIO3	R	0h	This bit reads the data input value of DIO3. 0h = Input value is 0 1h = Input value is 1

Table 8-56. DIN31_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIO2	R	0h	This bit reads the data input value of DIO2. 0h = Input value is 0 1h = Input value is 1
1	DIO1	R	0h	This bit reads the data input value of DIO1. 0h = Input value is 0 1h = Input value is 1
0	DIO0	R	0h	This bit reads the data input value of DIO0. 0h = Input value is 0 1h = Input value is 1

8.3.54 POLARITY15_0 (Offset = 1390h) [Reset = 00000000h]

POLARITY15_0 is shown in Figure 8-57 and described in Table 8-57.

Return to the Summary Table.

This register is used to enable and configure the polarity for input edge detection on DIO15 to DIO0. The corresponding DIO bits in RIS register will be set when the input event matches the configured polarity.

Figure 8-57. POLARITY15_0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIO15		DIO14		DIO13		DIO12		DIO11		DIO10		DIO9		DIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIO7		DIO6		DIO5		DIO4		DIO3		DIO2		DIO1		DIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 8-57. POLARITY15_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DIO15	R/W	0h	Enables and configures edge detection polarity for DIO15. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
29-28	DIO14	R/W	0h	Enables and configures edge detection polarity for DIO14. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
27-26	DIO13	R/W	0h	Enables and configures edge detection polarity for DIO13. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
25-24	DIO12	R/W	0h	Enables and configures edge detection polarity for DIO12. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
23-22	DIO11	R/W	0h	Enables and configures edge detection polarity for DIO11. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
21-20	DIO10	R/W	0h	Enables and configures edge detection polarity for DIO10. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
19-18	DIO9	R/W	0h	Enables and configures edge detection polarity for DIO9. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
17-16	DIO8	R/W	0h	Enables and configures edge detection polarity for DIO8. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event

Table 8-57. POLARITY15_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DIO7	R/W	0h	Enables and configures edge detection polarity for DIO7. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
13-12	DIO6	R/W	0h	Enables and configures edge detection polarity for DIO6. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
11-10	DIO5	R/W	0h	Enables and configures edge detection polarity for DIO5. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
9-8	DIO4	R/W	0h	Enables and configures edge detection polarity for DIO4. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
7-6	DIO3	R/W	0h	Enables and configures edge detection polarity for DIO3. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
5-4	DIO2	R/W	0h	Enables and configures edge detection polarity for DIO2. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
3-2	DIO1	R/W	0h	Enables and configures edge detection polarity for DIO1. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
1-0	DIO0	R/W	0h	Enables and configures edge detection polarity for DIO0. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event

8.3.55 POLARITY31_16 (Offset = 13A0h) [Reset = 00000000h]

POLARITY31_16 is shown in Figure 8-58 and described in Table 8-58.

Return to the Summary Table.

This register is used to enable and configure the polarity for input edge detection on DIO31 to DIO16. The corresponding DIO bits in RIS register will be set when the input event matches the configured polarity.

Figure 8-58. POLARITY31_16

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIO31		DIO30		DIO29		DIO28		DIO27		DIO26		DIO25		DIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIO23		DIO22		DIO21		DIO20		DIO19		DIO18		DIO17		DIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 8-58. POLARITY31_16 Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DIO31	R/W	0h	Enables and configures edge detection polarity for DIO31. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
29-28	DIO30	R/W	0h	Enables and configures edge detection polarity for DIO30. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
27-26	DIO29	R/W	0h	Enables and configures edge detection polarity for DIO29. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
25-24	DIO28	R/W	0h	Enables and configures edge detection polarity for DIO28. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
23-22	DIO27	R/W	0h	Enables and configures edge detection polarity for DIO27. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
21-20	DIO26	R/W	0h	Enables and configures edge detection polarity for DIO26. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
19-18	DIO25	R/W	0h	Enables and configures edge detection polarity for DIO25. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
17-16	DIO24	R/W	0h	Enables and configures edge detection polarity for DIO24. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event

Table 8-58. POLARITY31_16 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DIO23	R/W	0h	Enables and configures edge detection polarity for DIO23. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
13-12	DIO22	R/W	0h	Enables and configures edge detection polarity for DIO22. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
11-10	DIO21	R/W	0h	Enables and configures edge detection polarity for DIO21. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
9-8	DIO20	R/W	0h	Enables and configures edge detection polarity for DIO20. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
7-6	DIO19	R/W	0h	Enables and configures edge detection polarity for DIO19. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
5-4	DIO18	R/W	0h	Enables and configures edge detection polarity for DIO18. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
3-2	DIO17	R/W	0h	Enables and configures edge detection polarity for DIO17. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event
1-0	DIO16	R/W	0h	Enables and configures edge detection polarity for DIO16. 0h = Edge detection disabled 1h = Detects rising edge of input event 2h = Detects falling edge of input event 3h = Detects both rising and falling edge of input event

8.3.56 CTL (Offset = 1400h) [Reset = 00000000h]

CTL is shown in Figure 8-59 and described in Table 8-59.

Return to the Summary Table.

GPIO Control Register

Figure 8-59. CTL

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

Table 8-59. CTL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	FASTWAKEONLY	R/W	0h	FASTWAKEONLY for the global control of fastwake 0h = The global control of fastwake is not enabled, per bit fast wake feature depends on FASTWAKE.DIN 1h = The global control of fastwake is enabled

8.3.57 FASTWAKE (Offset = 1404h) [Reset = 00000000h]

FASTWAKE is shown in Figure 8-60 and described in Table 8-60.

Return to the Summary Table.

This is per bit fast wake enable for the bit slice, allows the GPIO module to stay in a low power state and not require high speed clocking of the input synchronizer or filter

Figure 8-60. FASTWAKE

31	30	29	28	27	26	25	24
DIN31	DIN30	DIN29	DIN28	DIN27	DIN26	DIN25	DIN24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DIN23	DIN22	DIN21	DIN20	DIN19	DIN18	DIN17	DIN16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DIN15	DIN14	DIN13	DIN12	DIN11	DIN10	DIN9	DIN8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DIN7	DIN6	DIN5	DIN4	DIN3	DIN2	DIN1	DIN0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-60. FASTWAKE Field Descriptions

Bit	Field	Type	Reset	Description
31	DIN31	R/W	0h	Enable fastwake feature for DIN31 0h = fastwake feature is disabled 1h = fastwake feature is enabled
30	DIN30	R/W	0h	Enable fastwake feature for DIN30 0h = fastwake feature is disabled 1h = fastwake feature is enabled
29	DIN29	R/W	0h	Enable fastwake feature for DIN29 0h = fastwake feature is disabled 1h = fastwake feature is enabled
28	DIN28	R/W	0h	Enable fastwake feature for DIN29 0h = fastwake feature is disabled 1h = fastwake feature is enabled
27	DIN27	R/W	0h	Enable fastwake feature for DIN27 0h = fastwake feature is disabled 1h = fastwake feature is enabled
26	DIN26	R/W	0h	Enable fastwake feature for DIN26 0h = fastwake feature is disabled 1h = fastwake feature is enabled
25	DIN25	R/W	0h	Enable fastwake feature for DIN25 0h = fastwake feature is disabled 1h = fastwake feature is enabled
24	DIN24	R/W	0h	Enable fastwake feature for DIN24 0h = fastwake feature is disabled 1h = fastwake feature is enabled
23	DIN23	R/W	0h	Enable fastwake feature for DIN23 0h = fastwake feature is disabled 1h = fastwake feature is enabled
22	DIN22	R/W	0h	Enable fastwake feature for DIN22 0h = fastwake feature is disabled 1h = fastwake feature is enabled

Table 8-60. FASTWAKE Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DIN21	R/W	0h	Enable fastwake feature for DIN21 0h = fastwake feature is disabled 1h = fastwake feature is enabled
20	DIN20	R/W	0h	Enable fastwake feature for DIN20 0h = fastwake feature is disabled 1h = fastwake feature is enabled
19	DIN19	R/W	0h	Enable fastwake feature for DIN19 0h = fastwake feature is disabled 1h = fastwake feature is enabled
18	DIN18	R/W	0h	Enable fastwake feature for DIN18 0h = fastwake feature is disabled 1h = fastwake feature is enabled
17	DIN17	R/W	0h	Enable fastwake feature for DIN17 0h = fastwake feature is disabled 1h = fastwake feature is enabled
16	DIN16	R/W	0h	Enable fastwake feature for DIN16 0h = fastwake feature is disabled 1h = fastwake feature is enabled
15	DIN15	R/W	0h	Enable fastwake feature for DIN15 0h = fastwake feature is disabled 1h = fastwake feature is enabled
14	DIN14	R/W	0h	Enable fastwake feature for DIN14 0h = fastwake feature is disabled 1h = fastwake feature is enabled
13	DIN13	R/W	0h	Enable fastwake feature for DIN13 0h = fastwake feature is disabled 1h = fastwake feature is enabled
12	DIN12	R/W	0h	Enable fastwake feature for DIN12 0h = fastwake feature is disabled 1h = fastwake feature is enabled
11	DIN11	R/W	0h	Enable fastwake feature for DIN11 0h = fastwake feature is disabled 1h = fastwake feature is enabled
10	DIN10	R/W	0h	Enable fastwake feature for DIN10 0h = fastwake feature is disabled 1h = fastwake feature is enabled
9	DIN9	R/W	0h	Enable fastwake feature for DIN9 0h = fastwake feature is disabled 1h = fastwake feature is enabled
8	DIN8	R/W	0h	Enable fastwake feature for DIN8 0h = fastwake feature is disabled 1h = fastwake feature is enabled
7	DIN7	R/W	0h	Enable fastwake feature for DIN7 0h = fastwake feature is disabled 1h = fastwake feature is enabled
6	DIN6	R/W	0h	Enable fastwake feature for DIN6 0h = fastwake feature is disabled 1h = fastwake feature is enabled
5	DIN5	R/W	0h	Enable fastwake feature for DIN5 0h = fastwake feature is disabled 1h = fastwake feature is enabled
4	DIN4	R/W	0h	Enable fastwake feature for DIN4 0h = fastwake feature is disabled 1h = fastwake feature is enabled
3	DIN3	R/W	0h	Enable fastwake feature for DIN3 0h = fastwake feature is disabled 1h = fastwake feature is enabled

Table 8-60. FASTWAKE Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DIN2	R/W	0h	Enable fastwake feature for DIN2 0h = fastwake feature is disabled 1h = fastwake feature is enabled
1	DIN1	R/W	0h	Enable fastwake feature for DIN1 0h = fastwake feature is disabled 1h = fastwake feature is enabled
0	DIN0	R/W	0h	Enable fastwake feature for DIN0 0h = fastwake feature is disabled 1h = fastwake feature is enabled

8.3.58 SUB0CFG (Offset = 1500h) [Reset = 00000000h]

SUB0CFG is shown in Figure 8-61 and described in Table 8-61.

Return to the Summary Table.

This register is used to enable the subscriber 0 event and define the output policy on the selected DIO 0-15 pins.

Figure 8-61. SUB0CFG

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				INDEX			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED						OUTPOLICY	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R/W-0h							R/W-0h

Table 8-61. SUB0CFG Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	
19-16	INDEX	R/W	0h	Indicates the specific bit among lower 16 bits that is targeted by the subscriber action 0h = specific bit targeted by the subscriber action is bit0 Fh = specific bit targeted by the subscriber action is bit15
15-10	RESERVED	R/W	0h	
9-8	OUTPOLICY	R/W	0h	These bits configure the output policy for subscriber 0 event. 0h = Selected DIO pins are set 1h = Selected DIO pins are cleared 2h = Selected DIO pins are toggled
7-1	RESERVED	R/W	0h	
0	ENABLE	R/W	0h	This bit is used to enable subscriber 0 event. 0h = Subscriber 0 event is disabled 1h = Subscriber 0 event is enabled

8.3.59 FILTEREN15_0 (Offset = 1508h) [Reset = 00000000h]

FILTEREN15_0 is shown in Figure 8-62 and described in Table 8-62.

Return to the Summary Table.

Programmable counter length of digital glitch filter for DIN0-DIN15

Figure 8-62. FILTEREN15_0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIN15		DIN14		DIN13		DIN12		DIN11		DIN10		DIN9		DIN8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIN7		DIN6		DIN5		DIN4		DIN3		DIN2		DIN1		DIN0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 8-62. FILTEREN15_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DIN15	R/W	0h	Programmable counter length of digital glitch filter for DIN15 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
29-28	DIN14	R/W	0h	Programmable counter length of digital glitch filter for DIN14 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
27-26	DIN13	R/W	0h	Programmable counter length of digital glitch filter for DIN13 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
25-24	DIN12	R/W	0h	Programmable counter length of digital glitch filter for DIN12 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
23-22	DIN11	R/W	0h	Programmable counter length of digital glitch filter for DIN11 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
21-20	DIN10	R/W	0h	Programmable counter length of digital glitch filter for DIN10 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
19-18	DIN9	R/W	0h	Programmable counter length of digital glitch filter for DIN9 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
17-16	DIN8	R/W	0h	Programmable counter length of digital glitch filter for DIN8 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample

Table 8-62. FILTEREN15_0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DIN7	R/W	0h	Programmable counter length of digital glitch filter for DIN7 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
13-12	DIN6	R/W	0h	Programmable counter length of digital glitch filter for DIN6 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
11-10	DIN5	R/W	0h	Programmable counter length of digital glitch filter for DIN5 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
9-8	DIN4	R/W	0h	Programmable counter length of digital glitch filter for DIN4 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
7-6	DIN3	R/W	0h	Programmable counter length of digital glitch filter for DIN3 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
5-4	DIN2	R/W	0h	Programmable counter length of digital glitch filter for DIN2 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
3-2	DIN1	R/W	0h	Programmable counter length of digital glitch filter for DIN1 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
1-0	DIN0	R/W	0h	Programmable counter length of digital glitch filter for DIN0 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample

8.3.60 FILTEREN31_16 (Offset = 150Ch) [Reset = 00000000h]

FILTEREN31_16 is shown in Figure 8-63 and described in Table 8-63.

Return to the Summary Table.

Programmable counter length of digital glitch filter for DIN16-DIN31

Figure 8-63. FILTEREN31_16

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIN31		DIN30		DIN29		DIN28		DIN27		DIN26		DIN25		DIN24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIN23		DIN22		DIN21		DIN20		DIN19		DIN18		DIN17		DIN16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 8-63. FILTEREN31_16 Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DIN31	R/W	0h	Programmable counter length of digital glitch filter for DIN31 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
29-28	DIN30	R/W	0h	Programmable counter length of digital glitch filter for DIN30 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
27-26	DIN29	R/W	0h	Programmable counter length of digital glitch filter for DIN29 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
25-24	DIN28	R/W	0h	Programmable counter length of digital glitch filter for DIN28 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
23-22	DIN27	R/W	0h	Programmable counter length of digital glitch filter for DIN27 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
21-20	DIN26	R/W	0h	Programmable counter length of digital glitch filter for DIN26 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
19-18	DIN25	R/W	0h	Programmable counter length of digital glitch filter for DIN25 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
17-16	DIN24	R/W	0h	Programmable counter length of digital glitch filter for DIN24 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample

Table 8-63. FILTEREN31_16 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-14	DIN23	R/W	0h	Programmable counter length of digital glitch filter for DIN23 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
13-12	DIN22	R/W	0h	Programmable counter length of digital glitch filter for DIN22 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
11-10	DIN21	R/W	0h	Programmable counter length of digital glitch filter for DIN21 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
9-8	DIN20	R/W	0h	Programmable counter length of digital glitch filter for DIN20 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
7-6	DIN19	R/W	0h	Programmable counter length of digital glitch filter for DIN19 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
5-4	DIN18	R/W	0h	Programmable counter length of digital glitch filter for DIN18 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
3-2	DIN17	R/W	0h	Programmable counter length of digital glitch filter for DIN17 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample
1-0	DIN16	R/W	0h	Programmable counter length of digital glitch filter for DIN16 0h = No additional filter beyond the CDC synchronization sample 1h = 1 ULPCLK minimum sample 2h = 3 ULPCLK minimum sample 3h = 8 ULPCLK minimum sample

8.3.61 DMAMASK (Offset = 1510h) [Reset = 00000000h]

DMAMASK is shown in Figure 8-64 and described in Table 8-64.

Return to the Summary Table.

DMA MASK which indicates which bit lanes the DMA is allowed to modify.

Figure 8-64. DMAMASK

31	30	29	28	27	26	25	24
DOUT31	DOUT30	DOUT29	DOUT28	DOUT27	DOUT26	DOUT25	DOUT24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
DOUT23	DOUT22	DOUT21	DOUT20	DOUT19	DOUT18	DOUT17	DOUT16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DOUT15	DOUT14	DOUT13	DOUT12	DOUT11	DOUT10	DOUT9	DOUT8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOUT7	DOUT6	DOUT5	DOUT4	DOUT3	DOUT2	DOUT1	DOUT0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 8-64. DMAMASK Field Descriptions

Bit	Field	Type	Reset	Description
31	DOUT31	R/W	0h	DMA is allowed to modify DOUT31 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
30	DOUT30	R/W	0h	DMA is allowed to modify DOUT30 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
29	DOUT29	R/W	0h	DMA is allowed to modify DOUT29 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
28	DOUT28	R/W	0h	DMA is allowed to modify DOUT28 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
27	DOUT27	R/W	0h	DMA is allowed to modify DOUT27 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
26	DOUT26	R/W	0h	DMA is allowed to modify DOUT26 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
25	DOUT25	R/W	0h	DMA is allowed to modify DOUT25 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
24	DOUT24	R/W	0h	DMA is allowed to modify DOUT24 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
23	DOUT23	R/W	0h	DMA is allowed to modify DOUT23 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
22	DOUT22	R/W	0h	DMA is allowed to modify DOUT22 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane

Table 8-64. DMAMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	DOUT21	R/W	0h	DMA is allowed to modify DOUT21 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
20	DOUT20	R/W	0h	DMA is allowed to modify DOUT20 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
19	DOUT19	R/W	0h	DMA is allowed to modify DOUT19 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
18	DOUT18	R/W	0h	DMA is allowed to modify DOUT18 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
17	DOUT17	R/W	0h	DMA is allowed to modify DOUT17 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
16	DOUT16	R/W	0h	DMA is allowed to modify DOUT16 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
15	DOUT15	R/W	0h	DMA is allowed to modify DOUT15 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
14	DOUT14	R/W	0h	DMA is allowed to modify DOUT14 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
13	DOUT13	R/W	0h	DMA is allowed to modify DOUT13 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
12	DOUT12	R/W	0h	DMA is allowed to modify DOUT12 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
11	DOUT11	R/W	0h	DMA is allowed to modify DOUT11 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
10	DOUT10	R/W	0h	DMA is allowed to modify DOUT10 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
9	DOUT9	R/W	0h	DMA is allowed to modify DOUT9 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
8	DOUT8	R/W	0h	DMA is allowed to modify DOUT8 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
7	DOUT7	R/W	0h	DMA is allowed to modify DOUT7 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
6	DOUT6	R/W	0h	DMA is allowed to modify DOUT6 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
5	DOUT5	R/W	0h	DMA is allowed to modify DOUT5 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
4	DOUT4	R/W	0h	DMA is allowed to modify DOUT4 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
3	DOUT3	R/W	0h	DMA is allowed to modify DOUT3 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane

Table 8-64. DMAMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	DOUT2	R/W	0h	DMA is allowed to modify DOUT2 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
1	DOUT1	R/W	0h	DMA is allowed to modify DOUT1 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane
0	DOUT0	R/W	0h	DMA is allowed to modify DOUT0 0h = DMA is not allowed to modify this bit lane 1h = DMA is allowed to modify this bit lane

8.3.62 SUB1CFG (Offset = 1520h) [Reset = 00000000h]

SUB1CFG is shown in Figure 8-65 and described in Table 8-65.

Return to the Summary Table.

This register is used to enable the subscriber 1 event and define the output policy on the selected DIO 16-31 pins.

Figure 8-65. SUB1CFG

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				INDEX			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED						OUTPOLICY	
R/W-0h						R/W-0h	
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R/W-0h							R/W-0h

Table 8-65. SUB1CFG Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	
19-16	INDEX	R/W	0h	indicates the specific bit in the upper 16 bits that is targeted by the subscriber action 0h = specific bit targeted by the subscriber action is bit16 Fh = specific bit targeted by the subscriber action is bit31
15-10	RESERVED	R/W	0h	
9-8	OUTPOLICY	R/W	0h	These bits configure the output policy for subscriber 1 event. 0h = Selected DIO pins are set 1h = Selected DIO pins are cleared 2h = Selected DIO pins are toggled
7-1	RESERVED	R/W	0h	
0	ENABLE	R/W	0h	This bit is used to enable subscriber 1 event. 0h = Subscriber 1 event is disabled 1h = Subscriber 1 event is enabled

Chapter 9
ADC

The ADC is a high-performance successive-approximation-register (SAR) analog-to-digital converter. This chapter describes the features and operation of the ADC peripheral.

9.1 ADC Overview

The ADC supports measure analog signals and convert them to a digital representation with minimal CPU intervention.

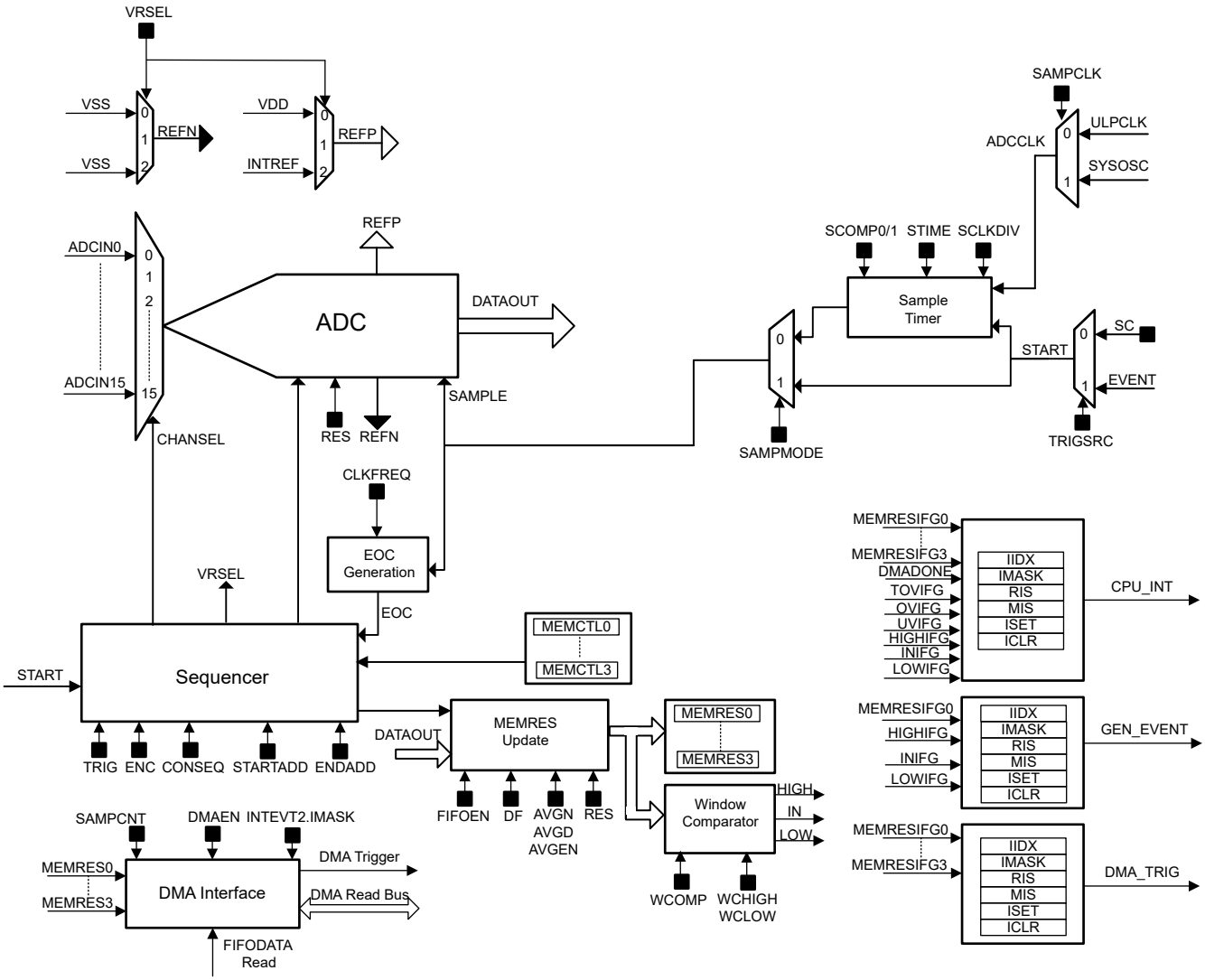
The ADC supports fast 12-, 10-, and 8-bit analog-to-digital conversions. It implements a 12-bit SAR core, sample and conversion mode control, and up to 4 independent conversion-and-control buffers. The ADC allows up to 4 independent analog-to-digital converter (ADC) samples to be converted and stored without any CPU intervention.

ADC features include:

- 1.5-Msps conversion rate at a resolution of 12 bits
- Integrated hardware oversampling for averaging up to 128 samples
- Full-scale ADC operating voltage range
- Sample-and-hold with programmable sampling periods controlled by software or timers
- Two sampling trigger sources: software trigger and event trigger
- Software-selectable on-chip reference voltage of 1.4V or 2.5V
- Configurable ADC reference source: VDD, internal reference (VREF)
- Up to 10 individually configurable analog input channels
- Internal conversion channels for temperature sensing, supply monitoring, and analog signal chain (see the device-specific data sheet for availability and channel mapping)
- Configurable ADC clock source
- Single-channel, repeat-single-channel, sequence (autoscan), and repeat-sequence (repeated autoscan) conversion modes
- 4 conversion-result storage registers (MEMRES0:3)
- Window comparator for low-power monitoring of input signals from conversion-result registers
- DMA support with interrupt event generation on completion of transfer
- Operates in RUN, SLEEP, and STOP modes
- Can be triggered in any operating mode except for SHUTDOWN
- Automatic power, reference, and clock control for low-power operation
- Semi-automatic calibration of CDAC trim values

Figure 9-1 shows the functional block diagram of the ADC peripheral.

Figure 9-1. ADC Block Diagram



9.2 ADC Operation

The ADC is configured with user software. The following sections describe the setup and operation of the ADC.

Note

The ADC result registers (MEMRES) are aliased between two address regions on the device:

- The primary region, through which all ADC registers can be accessed by the CPU or DMA at the ULPCLK rate
- The aliased region, through which the ADC MEMRES registers can be accessed by the CPU or DMA at the MCLK rate for fast read-out of ADC results

On devices which run MCLK and ULPCLK at the same frequency, there is not a performance benefit to accessing the ADC MEMRES registers through the aliased region. On device configurations where MCLK is greater than ULPCLK, the aliased region is recommended to be used. However, application software can use the aliased region on all devices for MEMRES access to keep the software implementation consistent.

9.2.1 ADC Core

The ADC core converts an analog input to its digital representation. The core uses two voltage levels (V_{R+} and V_{R-}) to define the upper and lower limits of the conversion. The digital output (N_{ADC}) is full scale when the input

signal is equal to or higher than V_{R+}, and is zero when the input signal is equal to or lower than V_{R-}. The input channel and the positive reference voltage level (V_{R+}) are defined in the conversion-control memory.

Equation 7 below shows the conversion formula for the ADC result, N_{ADC}, for n-bit resolution mode.

$$N_{ADC} = \left(2^n - 1\right) \times \frac{(V_{in} + 0.5LSB) - V_{R-}}{V_{R+} - V_{R-}} \qquad \text{Where,} \quad LSB = \frac{V_{R+} - V_{R-}}{2^n} \tag{7}$$

Note

The supported V_{R-} for this ADC is 0V. All subsequent equations and sections will reflect this inherent property.

Given that V_{R-} is 0V in this ADC, the equation for N_{ADC} becomes:

$$N_{ADC} = \left(2^n - 1\right) \times \frac{(V_{in} + 0.5LSB)}{V_{R+}} \qquad \text{Where,} \quad LSB = \frac{V_{R+}}{2^n} \tag{8}$$

Equation 9 below describes the input voltage at which the ADC output saturates:

$$V_{in} = V_{R+} - 1.5LSB \tag{9}$$

Note

The ADC is NOT functional in STANDBY or SHUTDOWN mode.

9.2.2 Voltage Reference Options

The ADC voltage reference (V_{R+}) can be configured through the VRSEL bits in the MEMCTL register. Different reference sources can be selected for conversion on different channels. There are three options available for supplying a reference voltage to the ADC:

- MCU supply voltage (VDD)
- Configurable internal reference voltage (INTREF) of 1.4V and 2.5V from

  VREF module

When supplying an external reference to the ADC, connect the VREF+ pin to the reference source with the appropriate decoupling circuitry, and connect the VREF- pin to ground.

The internal reference (VREF) is a dedicated voltage reference module for the ADC peripheral and does not require external decoupling circuitry to operate. When supplying the internal reference as the reference to the ADC, the maximum ADC sampling rate is limited to approximately 1 Msps.

9.2.3 Generic Resolution Modes

The ADC supports operation in 12-bit (default), 10-bit, and 8-bit resolution modes. The resolution mode is configured using the RES bits in the CTL2 register.

- When 12-bit mode is selected, the conversion phase requires a total of 14 conversion clock cycles
- When 10-bit mode is selected, the conversion phase requires a total of 12 conversion clock cycles
- When 8-bit mode is selected, the conversion phase requires a total of 9 conversion clock cycles

9.2.4 Hardware Averaging

This ADC implements digital sample averaging in hardware (HW averaging) to efficiently increase the effective resolution of the ADC without the need for SW or CPU intervention. The HW averaging functionality is configured using the AVGN and AVGD bits in the CTL1 register.

- AVGN defines the number of conversions to accumulate for the current MEMCTLx
- AVGD defines what the accumulated value gets divided by using bit shifting

Note

The MEMRES result register is a maximum of 16 bits long. If not shifted appropriately, the result will be truncated.

Table 9-1. Available Hardware Averaging Settings

Bit Field Value	AVGN Settings (number of samples accumulated)	AVGD Settings (number of bits to right shift)
0x0	0	0
0x1	2	1
0x2	4	2
0x3	8	3
0x4	16	4
0x5	32	5
0x6	64	6
0x7	128	7

The averaging configuration is global and it holds for any channel that enables the averaging feature. It is not possible to have different averaging configurations defined per channel. The averaging feature for each individual channel can be enabled though the AVGEN bit in the MEMCTL register. When the sample trigger is received for a channel with averaging enabled, the required number of conversions are performed automatically back-to-back and the final averaged value is stored in the MEMRES register or FIFODAT.

Note

The data format must be selected as unsigned binary while using the hardware averaging feature.

9.2.5 ADC Clocking

The ADC peripheral clock (ADCCLK) is provided by the Section 2.4 and is used for both sampling and conversion. SYSOSC , HFCLK and ULPCLK are the available clock sources available for ADCCLK, which can support up to 24Mhz. Refer to the device-specific data sheet for supported ADCCLK frequencies. Using the ULPCLK, which is the bus clock for all peripherals, is very useful for deterministic start of sampling. The ADC clock source can be selected by programming the SAMPCLK bits in the CLKCFG register.

SYSOSC needs to be active for the ADC to operate properly. If SYSOSC is not running and the ADC is triggered, the ADC will automatically request SYSCTL to enable and set SYSOSC to base frequency during the conversion. If SYSOSC is already enabled, it will remain the same frequency. The only exception to this is in STOP1 operating mode where SYSOSC will go to base frequency when the ADC is triggered.

In order to provide a way to ensure predictable sample rate operation between power modes, the CCONRUN and CCONSTOP bits can be set to signal the ADC that it can expect that the SYSOSC will already be ON when the device is in RUN and STOP modes respectively. When these bits are set, the ADC will not wait for an ACK from SYSCTL to make sure SYSOSC is running before starting sampling. This feature gives users the flexibility to save power in applications where deterministic sample timing is not a requirement. Refer to Section 9.2.6 for examples on how to properly use the CCONRUN and CCONSTOP control bits.

The user must configure the FRANGE bits in the CLKFREQ register to the appropriate setting based on the expected ADCCLK frequency. See the following table for more details on how to properly configure the CLKFREQ register.

Table 9-2. CLKFREQ Register Configuration

CLKFREQ.FRANGE Values	ADCCLK Frequency Range (MHz)
0	>1 to 4
1	>4 to 8
2	>8 to 16

Table 9-2. CLKFREQ Register Configuration (continued)

CLKFREQ.FRANGE Values	ADCCLK Frequency Range (MHz)
3	>16 to 20
4	>20 to 24

When the internal voltage reference is used for ADC operation, the ADCCLK is divided by 2 and then used as the conversion clock (CONVCLK), it means CONVCLK frequency is not to exceed 12MHz.

9.2.6 Common ADC Use Cases

There are many ADC use-cases from an operating mode and clocking standpoint but the majority of them fit into one of the items below:

- Triggers in RUN or SLEEP mode
  - If ADC is triggered to start a conversion (software or Event), and the device is in RUN0 or RUN1 or SLEEP0 or SLEEP1 mode, then:
    - Sample clock can be ULPCLK, HFCLK, or SYSOSC in this mode
    - SYSOSC is the clock source of the conversion. If the internal reference is used, the conversion clock frequency CONVCLK frequency is not to exceed 12MHz.
  - If ADC is triggered to start a conversion (software or Event), and the device is in RUN2 or SLEEP2 mode (SYSOSC is disabled, MCLK = LFCLK = 32kHz), then:
    - Sample clock can be ULPCLK or SYSOSC in this mode
    - SYSCTL interprets the ADC CLK REQ as an asynchronous fast clock request, enabling SYSOSC at 24MHz and forcing MCLK or ULPCLK to 24MHz MHz until the ADC de-asserts the request
    - CCONRUN must be cleared in this use case
    - CCONSTOP must be cleared in this use case
- Triggers in STOP mode
  - Sample clock can be ULPCLK or SYSOSC in this mode
  - If ADC is triggered to start a conversion (Event), and the device is in STOP0 mode (ULPCLK = 4MHz), then:
    - SYSOSC is the clock source of the conversion. If the internal reference is used, the conversion clock frequency CONVCLK frequency is not to exceed 12MHz.
  - If ADC is triggered to start a conversion (Event), and the device is in STOP2 mode (SYSOSC disabled), then:
    - The trigger event propagates through the event fabric at 32kHz, the ADC receives the trigger and assert the ADC CLK REQ (CPCLK REQ) to SYSCTL, and SYSCTL receives the ADC CLK REQ as an asynchronous fast clock request, suspending STOP, enabling SYSOSC at 24MHz , and forcing MCLK or ULPCLK to 24MHz until the ADC de-asserts the ADC CLK REQ
    - CCONRUN must be cleared
    - CCONSTOP must be cleared
- Triggers in STANDBY mode
  - Sample clock can be ULPCLK or SYSOSC in this mode
  - If ADC is triggered to start a conversion (Event), and the device is in STANDBY0 mode (SYSOSC is disabled but ULPCLK is running), then:
    - The trigger event propagates through event fabric at 32kHz, the ADC receives the trigger and asserts the ADC CLK REQ (CPCLK REQ) to SYSCTL, and SYSCTL interprets the ADC CLK REQ as an asynchronous fast clock request, suspending STANDBY, enabling SYSOSC at 24MHz , and forcing MCLK/ULPCLK to 24MHz until the ADC de-asserts the ADC CLK REQ
    - CCONRUN must be cleared
    - CCONSTOP must be cleared
  - If ADC is triggered to start a conversion (Event-TIMG8), and the device is in STANDBY1 (ULPCLK is gated with STOPCLKSTBY set), then:
    - The TIMG0 or TIMG1 event triggers an asynchronous fast clock request to suspend STANDBY mode, start SYSOSC at 24MHz , and force MCLK or ULPCLK to 24MHz z; there are then 41 SYSOSC cycles

for the TIMG0 or TIMG1 event to proceed through the event fabric and for the ADC to capture the timer event and assert the ADC CLK REQ to hold the SYSOSC enabled to run the conversion
- When the ADC de-asserts the ADC CLK REQ, ULPCLK runs for 41 additional cycles to allow any ADC event (DMA request or IRQ) to propagate, after which SYSCTL resumes STANDBY with STOPCLKSTBY (STANDBY1)
- CCONRUN must be cleared
- CCONSTOP must be cleared

9.2.7 Power Down Behavior

The ENABLE bit in the PWREN register enables or disables the ADC peripheral. The ADC should be disabled when it is not in use to save power. The PWRDN bit in the CTL0 register selects the ADC power down policy between AUTO and MANUAL. This takes effect when the ADC operates in RUN, SLEEP, and STOP MCU power modes.

PWRDN should be configured based on the max ADC sampling rate required and the operational needs in different MCU power modes. ADC hardware does not force the power down policy to AUTO during operation in STOP mode. It follows the user setting irrespective of the device power modes.

The reset value of PWRDN is 0, which has the default behavior of automatic power down of the ADC peripheral at the end of a conversion and when the next sample signal is not required to be asserted immediately. When the PWRDN bit is set to ‘1’ it selects manual power down behavior. In this setting, the ADC is not powered down at the end of a conversion and remains enabled. This means that the ADC peripheral would only be powered down using the PWREN register.

Note

Refer to the device-specific data sheet for specifications on the ADC wakeup and enable time.

9.2.8 Sampling Trigger Sources and Sampling Modes

Sample Triggers

There are two sampling trigger sources available which can be selected through the TRIGSRC bit in the CTL1 register; one is a software trigger and the other is an event trigger.

When the software trigger is selected as the source, the application software can set the SC bit in the CTL1 register to initiate the sample phase. When the event trigger is selected as the source, a rising edge on the selected event from the event manager will initiate the sample phase. An event is always edge triggered.

Sampling Modes

There are two sampling modes available, AUTO and MANUAL, which can be selected through the SAMPMODE bit in the CTL1 register.

9.2.8.1 AUTO Sampling Mode

In AUTO mode, the sample signal is generated synchronous to the sampling clock (SAMPCLK) and can be programmed using an internal sampling timer to determine the duration of the sampling window. The sample timer is 10-bits wide and there are two sample time compare registers (SCOMPx) available to account for various source impedances to measure signals from. One of these two SCOMP registers can be selected using the STIME bit in the MEMCTL register.

There is a 2-3 cycle latency from when the sampling is triggered and when the sampling period starts. This latency can be bypassed by setting ULPCLK as the source for ADCCLK. This synchronization bypass feature is very useful for deterministic sampling

Figure 9-2 shows the ADC sample and conversion timing diagram when the ADC is configured in AUTO sampling mode.

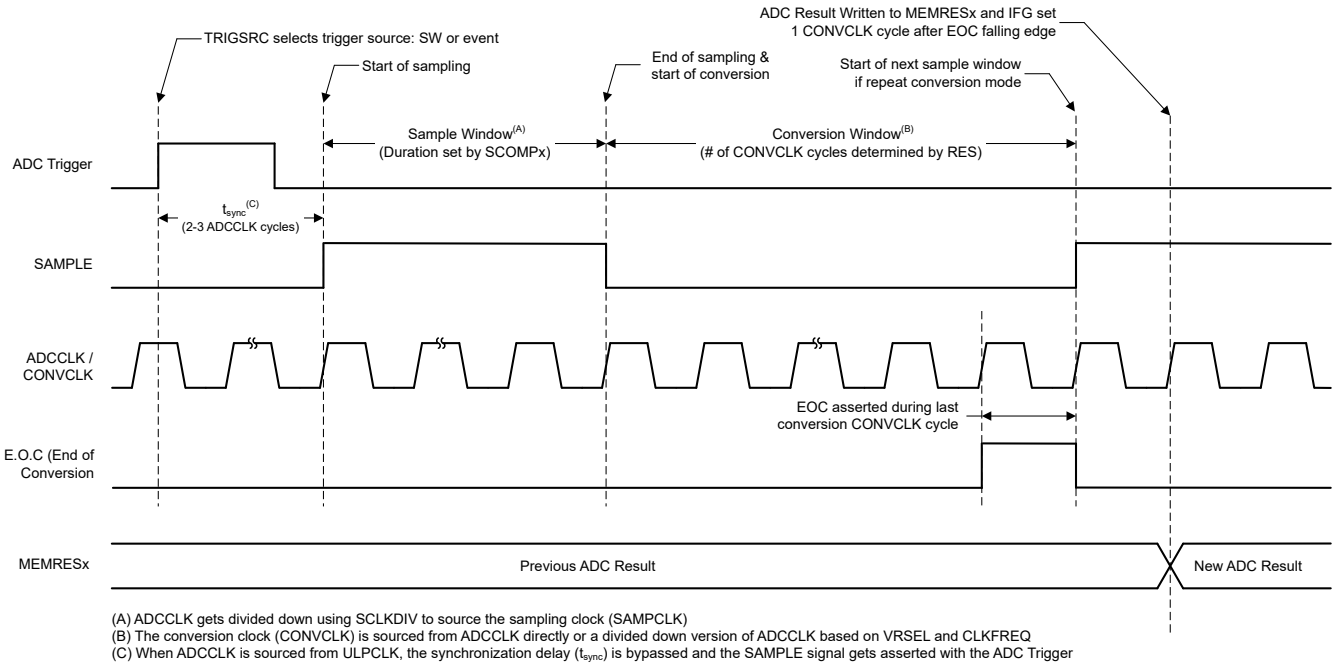


Figure 9-2. AUTO Sampling Mode ADC Sample and Conversion Timing Diagram

Note

When the reset value of PWRDN is set as '0' which has the default behavior of automatic power down, ADC wakeup/enable time needs to be considered in each sample window. Refer to the device-specific data sheet for specifications on the ADC wakeup/enable time. For example, if the maximum ADC wakeup/enable time is 5uS, it means the duration set by SCOMPx should be > (5uS + Duration for sample window).

9.2.8.2 MANUAL Sampling Mode

In MANUAL mode, the sample signal is generated when the SC bit is set which can be asynchronous to the sampling clock. The duration of the sampling window is controlled by software by holding the SC bit high. Because an event is always edge triggered, manual mode with event trigger is not supported for any of the conversion modes. Software trigger with manual sampling mode is supported only for single channel single conversion mode and it is not supported for any of the other three conversion modes.

There is a 2-3 cycle synchronization latency from when the sample window ends to when the conversion window begins.

Figure 9-3 shows the ADC sample and conversion timing diagram when the ADC is configured in MANUAL sampling mode:

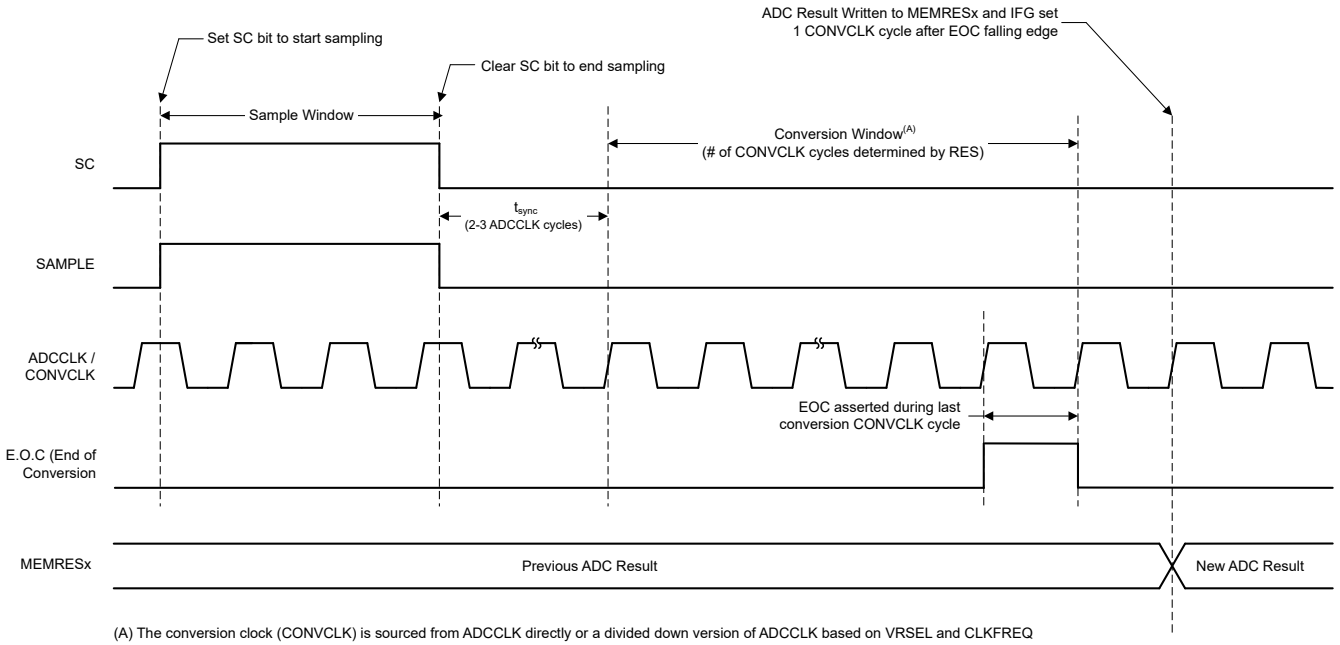


Figure 9-3. MANUAL Sampling Mode ADC Sample and Conversion Timing Diagram

Note

1. In MANUAL sampling mode the CCONRUN bit in the CLKCFG register must be set to 1.
2. When the reset value of PWRDN is set as '0' which has the default behavior of automatic power down, ADC wakeup/enable time needs to be considered before sample window. It means that after setting ENABLE bit in the PWREN register to enable ADC, then application software should set up a delay to wait the ADC wakeup/enable time before starting sampling. Refer to the device-specific data sheet for specifications on the ADC wakeup/enable time.
3. Once SC bit set by software, the SC bit is automatically cleared by hardware after the sample duration, which is the duration during which the analog signal is sampled. If the software attempts to set the SC bit before the sample duration + 2-3 cycles synchronization latency, TOVIFG flag will not be set.

9.2.9 Sampling Period

The sampling clock source is selected in the SYSCTL module using the SAMPCLK bits in the CLKCFG register. The desired sampling period for ADC operation can be generated using the internal clock divider and/or the sample timer, which applies to AUTO sampling mode. The internal clock divider is configured using the SCLKDIV bits in the CTL0 register and has divide options of 1, 2, 4, 8, 16, 24, 32, and 48.

The duration of the sampling period can be programmed to one of two user-defined values set by the SCOMP0 and SCOMP1 sample timer registers. The value in SCOMPx configures the sampling period by defining the number of sample time clocks to set the sample window to. The default SCOMPx sample timer value translates to 1 cycle wide sample pulse which allows the sampling period to be solely based on the sample clock and SCLKDIV. In general, there are three knobs that can be leveraged to control the sample period: SCOMPx, SCLKDIV, and the source of the sample clock.

When AUTO power down mode is selected using PWRDN=0, the module enable signal to the ADC peripheral is generated one sampling clock cycle after the sample signal is asserted. This should be considered by the user in the sample window calculation in addition to the ADC power time or settling time needs of other analog modules such as the Temperature Sensor, VREF, etc.

9.2.10 Conversion Modes

There are four conversion modes available in the ADC:

1. Single channel single conversion
   - The channel can be selected using MEMCTL
   - The selected channel is sampled and converted only once
   - Multiple conversions are performed when HW averaging is enabled
2. Repeat single channel conversion
   - The channel can be selected using MEMCTL
   - The selected channel is repeatedly sampled and converted until ENC is cleared by software
     - If the TRIG bit is set, a trigger is needed to move to the next conversion
   - Multiple conversions are performed when HW averaging is enabled
3. Sequence of channels conversion
   - Groups of channels can be formed using STARTADD, ENDADD, and MEMCTL registers
   - Each of the channels in the group is sampled and converted only once
   - Multiple conversions are performed on a channel during the sequence when HW averaging is enabled
   - The sequence will complete even if ENC is cleared in the middle of the sequence
4. Repeat sequence of channels conversion
   - Groups of channels can be formed using STARTADD, ENDADD, and MEMCTL registers
   - The group of channels are sampled and converted repeatedly until ENC is cleared by software
     - If the TRIG bit is set, a trigger is needed to move to the next conversion
   - When ENC is cleared the operation stops at the end of the last conversion
   - Multiple conversions are performed on a channel during a sequence when averaging is enabled

The following steps outline the recommended process for configuring the ADC for a desired conversion mode:

1. Use the CONSEQ bits in the CTL1 register to select the desired ADC conversion mode
2. Use the STARTADD bits in the CTL2 register to select which MEMCTLx is used for single conversion or as first MEMCTL for a sequence mode
3. If using a sequence mode, use the ENDADD bits in the CTL2 register to select which MEMCTLx is used for the last conversion of the sequence
4. Assign an ADC input channel to the appropriate MEMCTLx register using the CHANSEL bits
   - For sequence modes, you must assign an ADC input channel for each MEMCTLx that is part of the configured sequence
5. Select HW or SW trigger using the TRIGSRC bit in the CTL1 register
6. Select AUTO or MANUAL sampling mode using the SAMPMODE bit in the CTL1 register
   - If using AUTO mode, program the desired sample timer value in the SCOMPx register and use the STIME bits in the MEMCTLx register to select the appropriate sample timer source (SCOMP0 or SCOMP1)
7. If using repeat single channel or sequence conversion modes, program the TRIG bit in each MEMCTLx register to indicate if a trigger will be needed to step to the next MEMCTL in the sequence
8. Set the ENC bit in the CTL1 register to enable ADC conversions
9. The following table matrix depicts the next step of ADC configuration and usage based on the selected trigger and sampling modes:

Table 9-3. Trigger and Sample Mode ADC Usage Matrix

	Trigger Mode	
	SW Trigger	**Event Trigger**

Table 9-3. Trigger and Sample Mode ADC Usage Matrix (continued)

Sampling Mode	**AUTO**	• Set SC bit to start the sample phase (duration determined by sample timer) • Conversion starts once sample phase is over • In single channel single conversion, ENC is cleared when conversion is over • SC bit is automatically cleared once the trigger is captured • For repeat and sequence modes, if TRIG is set in MEMCTL, the SC bit needs to be set for the next conversion to proceed	• HW trigger starts the sample phase (duration determined by sample timer) • Conversion starts once sample phase is over • In single channel single conversion, ENC is cleared when conversion is over • For repeat and sequence modes, ADC waits for HW trigger or automatically starts the next conversion based on TRIG setting
	MANUAL	• Set SC bit to start the sample phase (SC bit is not automatically reset) • Clear the SC bit to end the sample phase and start the conversion • In single channel single conversion, ENC bit is cleared when conversion is over • Repeated/sequential conversion modes and HW averaging are NOT supported in this configuration	ADC operation is NOT supported in this configuration

10. The ADC results are stored in the MEMRES register of the associated MEMCTL (for example, the MEMCTL0 result is stored in MEMRES0).
    - For repeat conversion modes, the result in MEMRES is updated after every associated MEMCTL conversion
11. For repeated conversion modes, clear the ENC bit to stop ADC operation

9.2.11 Data Format

The ADC supports two data formats – unsigned binary and 2's complement signed binary. Unsigned binary results are stored right-justified in the MEMRES register or FIFO. Signed binary results are stored left justified in the MEMRES register or FIFO.

Table 9-4. ADC Data Formats

Data Format	Resolution	Result Range (decimal)	Result Range (hex)
Unsigned	8-bit	0 to 255	0000h to 00FFh
	10-bit	0 to 1023	0000h to 03FFh
	12-bit	0 to 4095	0000h 0FFFh
Signed	8-bit	-128 to 127	8000h to 7F00h
	10-bit	-512 to 511	8000h to 7FC0h
	12-bit	-2048 to 2047	8000h to 7FF0h

9.2.12 Advanced Features

The following sections describe the additional features and benefits provided with the ADC peripheral and how to leverage them in an application.

9.2.12.1 Window Comparator

There is one window comparator unit available in the ADC which can be used to check if the input signal is within predefined threshold values set by software. The ADC result that goes into MEMRES or FIFO is what gets checked against the threshold values of the window comparator.

Based on the comparison it can generate 3 interrupt conditions:

1. LOWIFG – Conversion result is below the Low threshold (WCLOW)

2. HIGHIFG – Conversion result is above the High threshold (WCHIGH)
3. INIFG – Conversion result is in between or equal to the Low and High thresholds

The window comparator low and high threshold values are global for all channels and the window comparison feature can be enabled for each channel as needed using the WINCOMP bit in the MEMCTL register.

When the ADC result data format (CTL2.DF) or resolution (CTL2.RES) configuration is changed, the window comparator threshold values are not reset by hardware and are retained as is. The software application is expected to reconfigure the threshold values as appropriate after changing the data format and/or resolution configuration.

9.2.12.2 DMA and FIFO Operation

The ADC has a dedicated interface for communicating to and from the DMA. This interface is useful to offload work from the CPU by using the DMA to store ADC results to memory automatically. Figure 9-4 shows the signals that make up this interface:

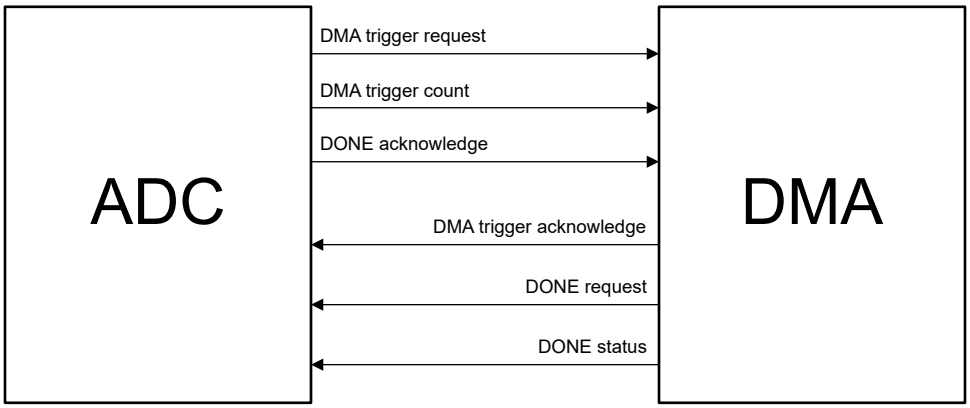


Figure 9-4. Internal ADC-DMA Interface

Note

The “DMA trigger count” signal indicates the number of samples that can be transferred by the DMA upon one trigger request. The “DONE status” signal is used by the ADC to generate the DMA DONE interrupt and it indicates if the DMA data transfer of programmed block size is completed.

The DMAEN bit in the CTL2 register is used to enable the DMA for ADC data transfer. The DMAEN bit is cleared by ADC hardware when the DMA “DONE status” signal is asserted. Software is expected to re-enable the DMA using DMAEN to arm the ADC to generate the next DMA trigger.

The ADC also incorporates an optional First-In-First-Out buffer to provide a way for ADC results to be stored for future use, such as transferring to memory by the DMA. Either the CPU or the DMA can be used to move data from the ADC regardless of whether the FIFO is enabled or disabled. The memory result flags in the RIS register of the third event publisher serve as the FIFO threshold and can be unmasked to generate the DMA trigger.

The following sections explain the details of using the ADC+DMA/CPU in various conversion modes and with the FIFO enabled or disabled

ADC-DMA/CPU Operation in Non-FIFO Mode (FIFOEN=0)

ADC-DMA/CPU Operation in **Non-FIFO Mode (FIFOEN=0)**

- Single Conversion and Repeat Single Conversion
  - Configure STARTADD bits to select the desired MEMCTLx register
    - MEMCTLx **is correlated to** MEMRESx
    - MEMRESx **is correlated to** MEMRESIFGx
  - Configure MEMCTL CHANSEL bits to select the desired ADC channel

  - Conversion data is available in MEMRESx
  - MEMRESIFGx can be set to generate a CPU interrupt or the DMA trigger
  - SAMPCNT **must be programmed to 1** by SW for DMA operation
  - The conversion overflow flag OVIFG is set when the ADC updates MEMRESx before the previous sample is read by the CPU or DMA
  - The conversion underflow flag UVIFG is set when the CPU or DMA reads the MEMRESx register before the next conversion result is available
- Sequence Conversion and Repeat Sequence Conversion
  - Configure STARTADD bits to select the first MEMCTL in the sequence
  - Configure ENDADD bits to select the last MEMCTL in the sequence
    - MEMCTLx **is correlated to** MEMRESx
    - MEMRESx **is correlated to** MEMRESIFGx
  - Configure each MEMCTLx CHANSEL bits to select the desired ADC channels
  - Conversion data is available in MEMRESx
  - MEMRESIFGx can be set to generate a CPU interrupt or the DMA trigger
  - SAMPCNT must be programmed by SW to a **suitable value based on threshold setting** by SW for DMA operation
  - The conversion overflow flag OVIFG is set when the ADC updates MEMRESx before the previous sample is read by the CPU or DMA
  - The conversion underflow flag UVIFG is set when the CPU or DMA reads the MEMRESx register before the next conversion result is available

Note

For DMA based operation, the MEMCTL start address should be smaller than the end address for single sequence conversion as DMA source does not roll back. Repeat sequence conversion mode does not support DMA based data transfer because the DMA does not support circular addressing mode.

ADC-DMA/CPU Operation in FIFO Mode (FIFOEN=1)

- Single Conversion and Repeat Single Conversion
  - Configure STARTADD bits to select the desired MEMCTLx register
    - MEMCTLx **is NOT correlated to** MEMRESx
    - MEMRESx **is correlated to** MEMRESIFGx
  - Configure MEMCTL CHANSEL bits to select the desired ADC channel
  - Conversion data is loaded sequentially into MEMRES0,1,2,….N (organized as a FIFO)
  - The CPU or DMA must read ADC samples from the dedicated FIFODAT register and not from MEMRES registers directly
    - Data in the FIFO is always compacted with two samples and provided as 32-bit data upon a FIFODAT read by CPU or DMA
  - MEMRESIFGx can be used as a threshold condition to generate a CPU interrupt or DMA trigger
    - For full use of the FIFO, the last MEMRESIFG can be used
  - SAMPCNT must be programmed by SW to a **suitable value based on threshold setting** for DMA operation
  - The conversion overflow flag OVIFG is set when the ADC updates MEMRESx before the previous sample is read by the CPU or DMA
  - The conversion underflow flag UVIFG is set when the CPU or DMA reads the FIFODAT register before the conversion result is available in the MEMRESx registers.

Note

Single conversion mode with FIFO enabled is not recommended for CPU or DMA based operation. It will lead to underflow condition and unwanted 16-bit data will have to be discarded in software.

- Sequence Conversion and Repeat Sequence Conversion
  - Configure STARTADD bits to select the first MEMCTL in the sequence

- Configure ENDADD bits to select the last MEMCTL in the sequence
  - MEMCTLx **is NOT correlated to** MEMRESx
  - MEMRESx **is correlated to** MEMRESIFGx
- Configure each MEMCTLx CHANSEL bits to select the desired ADC channels
- Conversion data is loaded sequentially into MEMRES0,1,2,….N (organized as a FIFO)
- The CPU or DMA must read ADC samples from the dedicated FIFODAT register and not from MEMRES registers directly
  - Data in the FIFO is always compacted with two samples and provided as 32-bit data upon a FIFODAT read by CPU or DMA
- MEMRESIFGx can be used as a threshold condition to generate a CPU interrupt or DMA trigger
  - For full use of the FIFO, the last MEMRESIFG can be used
- SAMPCNT must be programmed by SW to a **suitable value based on threshold setting** for DMA operation
- The conversion overflow flag OVIFG is set when the ADC updates MEMRESx before the previous sample is read by the CPU or DMA
- The conversion underflow flag UVIFG is set when the CPU or DMA reads the FIFODAT register before the conversion result is available in the MEMRESx registers

Note

- The data in FIFODAT register won't be cleared automatically after CPU or DMA reads. New conversion data overwrites the previous data in FIFODAT register.
- To ensure synchronized reading of bytes from the 32-bit FIFO, which stores 16-bit samples, specific DMA triggers can be used. In particular, selecting MEMRES1 and MEMRES3 will synchronize the reading of bytes from the FIFO with the corresponding MEMRESx bytes.
- If the ADC is disabled during either the repeat sequence mode or normal repeat mode, it's worth noting that an additional conversion may occur before the ADC completely stops.

Table 9-5. ADC-DMA/CPU Operation Summary Matrix

Conversion Mode	FIFO Disabled (FIFOEN=0) Samples not compacted Read from MEMRESx registers directly		FIFO Enabled (FIFOEN=1) Samples always compacted Read from FIFODAT register only	
	CPU Read/Write	DMA Read/Write	CPU Read/Write	DMA Read/Write
Single	Supported	Supported SAMPCNT=1 Sample in 16 bits	Not recommended Underflow flag will set Unwanted 16-bits should be ignored	Not recommended Underflow flag will set Unwanted 16-bits should be ignored
Repeat Single	Supported	Supported SAMPCNT=1 Sample in 16 bits	Supported MEMRESIFG=CPU interrupt FIFODAT read in 32-bits	Supported MEMRESIFG=DMA trigger SAMPCNT=Samples in 32-bits
Sequence	Supported	Supported SAMPCNT=Sample in 16 bits STARTADD<ENDADD	Supported MEMRESIFG=CPU interrupt FIFODAT read in 32-bits	Supported MEMRESIFG=DMA trigger SAMPCNT=Samples in 32-bits
Repeat Sequence	Supported	Not Supported	Supported MEMRESIFG=CPU interrupt FIFODAT read in 32-bits	Supported MEMRESIFG=DMA trigger SAMPCNT=Samples in 32-bits

9.2.12.3 Analog Peripheral Interconnection

The MSPM0 platform of MCUs provides a rich set of high-performance analog peripherals which can interact with each other to perform various analog signal chain functions. The items below describe how the ADC interacts with the other on-board analog peripherals:

ADC with Internal Reference Module (VREF)

The ADC has a dedicated enable request and ready interface with the internal voltage reference module. VREF enable is asserted upon sample trigger while internal reference buffer is selected for ADC operation. The ready response from VREF is captured in the ADC status register (REFBUFRDY).

The ready response from VREF **does not** gate the sample phase in the ADC. The ADC sample window is started upon the sample trigger and the settling time of the internal reference buffer needs to be considered in the sample period as appropriate. Software can enable the VREF module using the software enable bit so that it is already settled by the time the ADC starts sampling the input channel. In this case, the sampling time can be smaller and does not have to account for the enable time of VREF.

ADC with Temperature Sensor

The temperature sensor module enable signal is generated by the ADC when the temp sense channel is selected. The settling time of the temperature sensor should be accounted for in the sample period as there is no ready response from the temperature sensor.

9.2.13 Status Register

The ADC status register, STATUS, contains two bits – REFBUFRDY and BUSY.

- REFBUFRDY is set when the ADC receives the ready signal from the internal reference buffer (VREF/ REFBUF) after asserting the enable request
- BUSY equaling ‘1’ indicates that the ADC is busy performing a sample or convert operation
  - For **single channel single conversion**, it signals that a trigger has been received and sample or conversion is ongoing. BUSY will be cleared when the conversion completes
  - For **repeat single conversion**, it signals that repeat single operation has begun and has not ended. BUSY will be cleared when ENC is written ‘0’ and the last conversion completes
  - For **sequence of channels conversion**, it signals that the sequence of channels conversion has started. BUSY will be cleared at the end of the sequence
  - For **repeat sequence of channels conversion**, it signals the repeat sequence is ongoing. BUSY will be cleared when ENC is written ‘0’ and the last conversion completes

9.2.14 ADC Events

The ADC peripheral contains three event publishers and one event subscriber.

One event publisher (CPU_INT) manages ADC interrupt requests (IRQs) to the CPU subsystem through a static event route. The second event publisher (GEN_EVENT) can be used to publish ADC events to a subscriber through a generic event route channel. The third event publisher (DMA_TRIG) can be used as an ADC-to- DMA trigger to send ADC events directly to the DMA through a DMA event route.

The event subscriber (FSUB_0) can be used to subscribe to events which are published to the event fabric through a generic event route channel.

The ADC events are summarized in Table 9-6.

Table 9-6. ADC Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt event	Publisher	ADC	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from ADC to CPU
Generic publisher event	Publisher	ADC	Generic event channel	Generic route (FPUB_0)	GEN_EVENT registers, FPUB_0 register	Trigger generic event channel from ADC
DMA trigger event	Publisher	ADC	DMA	DMA route	DMA_TRIG registers	Fixed trigger route from ADC to DMA

Table 9-6. ADC Events (continued)

Event	Type	Source	Destination	Route	Configuration	Functionality
Generic subscriber event	Subscriber	Other peripherals	ADC	Generic route(FSUB_0)	FSUB_0	ADC subscription to generic event channel

9.2.14.1 CPU Interrupt Event Publisher (CPU_INT)

The ADC peripheral provides many interrupt sources which can be configured to source a CPU interrupt event. In order of decreasing interrupt priority, the CPU interrupt events from the ADC are given in Table 9-7.

Table 9-7. ADC CPU Interrupt Event Conditions (CPU_INT)

Index (IIDX)	Name	Description
0x0	NO_INTR	No bit set (IIDX.STAT = 0) means there is no pending interrupt request
0x1	OVIFG	Conversion overflow interrupt flag is set when the ADC updates MEMRESx before the previous sample is read by the CPU or DMA
0x2	TOVIFG	Sequence conversion time overflow interrupt flag is set when the ADC receives a new sampling trigger while the previous sample+conversion is still in progress
0x3	HIGHIFG	High threshold compare interrupt flag is set when the MEMRESx result register is higher than the WCHIGH threshold of the window comparator
0x4	LOWIFG	Low threshold compare interrupt flag is set when the MEMRESx result register is lower than the WCLOW threshold of the window comparator
0x5	INIFG	In-range comparator interrupt flag is set when the MEMRESx result register is within the range of WCLOW and WCHIGH of the window comparator
0x6	DMADONE	DMA done interrupt flag is set when the DMA data transfer of programmed block size is completed
0x7	UVIFG	Conversion underflow interrupt flag, the UVIFG flag is set when the CPU or DMA reads the MEMRESx register before the next conversion result is available
0x9 up to 0x20	MEMRESIFG[0 up to 24](1)	Memory register interrupt flag is set when MEMRESx is loaded with a new conversion result

(1) Refer to the device-specific data sheet to see how many conversion-result storage registers (MEMRES) your device supports.

The CPU interrupt event configuration is managed with the CPU_INT event management registers. Interrupt (RIS) flags are cleared upon software reading the IIDX register or writing to the respective ICLR register bits. See Section 6.2.5 for guidance on configuring the Event registers for CPU interrupts.

9.2.14.2 Generic Event Publisher (GEN_EVENT)

The ADC peripheral provides 4 interrupt sources, one of which can be configured to publish an event to a generic event route channel. Table 9-8 lists these interrupt sources.

Table 9-8. ADC Generic Event Publisher Conditions (GEN_EVENT)

Index	Name	Description
0x0	NO_INTR	No bit set means there is no pending interrupt request
0x3	HIGHIFG	High threshold compare interrupt flag is set when the MEMRESx result register is higher than the WCHIGH threshold of the window comparator
0x4	LOWIFG	Low threshold compare interrupt flag is set when the MEMRESx result register is lower than the WCLOW threshold of the window comparator
0x5	INIFG	In-range comparator interrupt flag is set when the MEMRESx result register is within the range of WCLOW and WCHIGH of the window comparator
0x9	MEMRESIFG0	Memory register interrupt flag is set when MEMRES0 is loaded with a new conversion result

The generic event publisher configuration is managed with the GEN_EVENT event management registers. Interrupt (RIS) flags are cleared based on acknowledgment (ACK) signal from the subscriber module received over the event fabric. See Section 6.2.5 for guidance on configuring the Event registers for generic event publishers.

The generic event channel which GEN_EVENT is to publish to must be selected by writing the target generic channel ID to the **FPUB_0** register in the ADC. See Section 6.1.3.3 for guidance on configuring generic event routes.

If this publisher is not used in an application, the FPUB_0 register can be left in a disconnected state (set equal to zero) and no events should be unmasked through the MIS register in the ADC GEN_EVENT register set.

9.2.14.3 DMA Trigger Event Publisher (DMA_TRIG)

The ADC module provides many interrupt sources which can be configured to source the DMA trigger. In order of decreasing interrupt priority, the DMA trigger events from the ADC are given in Table 9-9. When the DMA channel is needed by the ADC, the DMA trigger should be unmasked in the IMASK register of DMA_TRIG and the DMA should be configured as needed to support the ADC operation.

Table 9-9. ADC DMA Trigger Event Conditions (DMA_TRIG)

Index (IIDX)	Name	Description
0x0	NO_INTR	No bit set (IIDX.STAT = 0) means there is no pending interrupt request
0x9 up to 0x20	MEMRESIFG[0 up to 24](1)	Memory register interrupt flag is set when MEMRESx is loaded with a new conversion result

(1) Check the device-specific data sheet to see how many conversion-result storage registers (MEMRES) your device supports.

The DMA trigger event configuration is managed with the DMA_TRIG event management registers. The interrupt (RIS) flags are cleared based on ACK from DMA. See Section 6.2.5 for guidance on configuring the Event registers for DMA triggers.

9.2.14.4 Generic Event Subscriber (FSUB_0)

The ADC peripheral supports receiving events routed through a generic channel from other peripherals. Refer to Section 6.1.3.3 and Section 6.2.3 for how generic event route works.

Once the channel to be used is determined, and both the publisher and subscriber ports for the peripherals being connected are known, use the steps below to establish the event connection. In this example, a GPIO triggered ADC application will be configured, using GPIO Port A to publish an event to generic channel 1, with ADC0 subscribing to generic channel 1 as a start-of-conversion trigger.

1. Configure the GEN_EVENT registers of GPIO Port A to set the event request based on the appropriate event (for example, a DIN rise event).
2. Store 0x1 into the FPUB_0 register of GPIO Port A to publish the GPIO event selected by the GEN_EVENT registers to generic route channel 1. Channel 1 must not be in use by another peripheral.
3. Store 0x1 the FSUB_0 register of ADC0 so that ADC0 is listening for events published by the timer to channel 1.
4. Configure ADC0 to trigger from the subscriber port according to the configuration instructions in Section 9.2.8
5. Configure and enable the appropriate GPIO pin to monitor input voltage events

9.3 ADC0 Registers

Table 9-10 lists the memory-mapped registers for the ADC0 registers. All register offset addresses not listed in Table 9-10 should be considered as reserved locations and the register contents should not be modified.

Table 9-10. ADC0 Registers

Offset	Acronym	Register Name	Section
400h	FSUB_0	Subscriber Configuration Register.	Section 9.3.1
444h	FPUB_1	Publisher Configuration Register.	Section 9.3.2
800h	PWREN	Power enable	Section 9.3.3
804h	RSTCTL	Reset Control	Section 9.3.4
808h	CLKCFG	ADC clock configuration Register	Section 9.3.5
814h	STAT	Status Register	Section 9.3.6
1028h	IMASK	Interrupt mask	Section 9.3.7
1030h	RIS	Raw interrupt status	Section 9.3.8
1038h	MIS	Masked interrupt status	Section 9.3.9
1040h	ISET	Interrupt set	Section 9.3.10
1048h	ICLR	Interrupt clear	Section 9.3.11
1058h	IMASK	Interrupt mask	Section 9.3.12
1060h	RIS	Raw interrupt status	Section 9.3.13
1068h	MIS	Masked interrupt status	Section 9.3.14
1070h	ISET	Interrupt set	Section 9.3.15
1078h	ICLR	Interrupt clear	Section 9.3.16
1088h	IMASK	Interrupt mask extension	Section 9.3.17
1090h	RIS	Raw interrupt status extension	Section 9.3.18
1098h	MIS	Masked interrupt status extension	Section 9.3.19
10A0h	ISET	Interrupt set extension	Section 9.3.20
10A8h	ICLR	Interrupt clear extension	Section 9.3.21
1100h	CTL0	Control Register 0	Section 9.3.22
1104h	CTL1	Control Register 1	Section 9.3.23
1108h	CTL2	Control Register 2	Section 9.3.24
110Ch	CTL3	Control Register 3	Section 9.3.25
1114h	SCOMP0	Sample Time Compare 0 Register	Section 9.3.26
1118h	SCOMP1	Sample Time Compare 1 Register	Section 9.3.27
111Ch	REFCFG	Reference Buffer Configuration Register	Section 9.3.28
1148h	WCLOW	Window Comparator Low Threshold Register	Section 9.3.29
1150h	WCHIGH	Window Comparator High Threshold Register	Section 9.3.30
1160h	FIFODATA	FIFO Data Register	Section 9.3.31
1170h	ASCRES	ASC Result Register	Section 9.3.32
1180h + formula	MEMCTL_y	Conversion Memory Control Register	Section 9.3.33
1280h + formula	MEMRES_y	Memory Result Register	Section 9.3.34
1340h	STATUS	Status Register	Section 9.3.35

Complex bit access types are encoded to fit into small table cells. Table 9-11 shows the codes that are used for access types in this section.

Table 9-11. ADC0 Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RH	R H	Read Set or cleared by hardware
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

9.3.1 FSUB_0 Register (Offset = 400h) [Reset = 00000000h]

FSUB_0 is shown in Figure 9-5 and described in Table 9-12.

Return to the Table 9-10.

Subscriber port

Figure 9-5. FSUB_0 Register

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

Table 9-12. FSUB_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 255.

9.3.2 FPUB_1 Register (Offset = 444h) [Reset = 00000000h]

FPUB_1 is shown in Figure 9-6 and described in Table 9-13.

Return to the Table 9-10.

Publisher port

Figure 9-6. FPUB_1 Register

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

Table 9-13. FPUB_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 255.

9.3.3 PWREN Register (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 9-7 and described in Table 9-14.

Return to the Table 9-10.

Register to control the power state

Figure 9-7. PWREN Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/WK-0h

Table 9-14. PWREN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R	0h	

0	ENABLE	R/WK	0h	Enable the power
				#none# must be set to 26h to write to this bit.
				0h = Disable Power
				1h = Enable Power

9.3.4 RSTCTL Register (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 9-8 and described in Table 9-15.

Return to the Table 9-10.

Register to control reset assertion and de-assertion

Figure 9-8. RSTCTL Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
R-0h						WK-0h	WK-0h

Table 9-15. RSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	R	0h	

1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register
				#none# must be set to B1h to write to this bit.
				0h = Writing 0 has no effect
				1h = Clear reset sticky bit

0	RESETASSERT	WK	0h	Assert reset to the peripheral
				#none# must be set to B1h to write to this bit.
				0h = Writing 0 has no effect
				1h = Assert reset

9.3.5 CLKCFG Register (Offset = 808h) [Reset = 00000000h]

CLKCFG is shown in Figure 9-9 and described in Table 9-16.

Return to the Table 9-10.

ADC clock configuration

Figure 9-9. CLKCFG Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CCONSTOP	CCONRUN	RESERVED		SAMPCLK	
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	

Table 9-16. CLKCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key A9h = KEY to allow write access to this register
23-6	RESERVED	R	0h	
5	CCONSTOP	R/W	0h	CCONSTOP: Forces SYSOSC to run at base frequency when device is in STOP mode which can be used as ADC sample or conversion clock source. 0h = ADC conversion clock source is not kept continuously on during STOP mode. 1h = ADC conversion clock source kept continuously on during STOP mode.
4	CCONRUN	R/W	0h	CCONRUN: Forces SYSOSC to run at base frequency when device is in RUN mode which can be used as ADC sample or conversion clock source. 0h = ADC conversion clock source is not kept continuously on during RUN mode. 1h = ADC conversion clock source kept continuously on during RUN mode.
3-2	RESERVED	R	0h	
1-0	SAMPCLK	R/W	0h	ADC sample clock source selection. 0h = ULPCLK is the source of ADC sample clock. 1h = SYSOSC is the source of ADC sample clock. 2h = HFCLK clock is the source of ADC sample clock. Note : HFCLK may not be available on all the devices.

9.3.6 STAT Register (Offset = 814h) [Reset = 000X0000h]

STAT is shown in Figure 9-10 and described in Table 9-17.

Return to the Table 9-10.

peripheral enable and reset status

Figure 9-10. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							RESETSTKY
R-0h							R-X
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 9-17. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	X	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

9.3.7 IMASK Register (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 9-11 and described in Table 9-18.

Return to the Table 9-10.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 9-11. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ASCDONE	UVIFG	DMADONE	INIFG	LOWIFG	HIGHIFG	TOVIFG	OVIFG
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-18. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7	ASCDONE	R	0h	Mask for ASC done raw interrupt flag 0h = Interrupt is not pending. 1h = Interrupt is pending.

Table 9-18. IMASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	UVIFG	R/W	0h	Raw interrupt flag for MEMRESx underflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
5	DMADONE	R	0h	Raw interrupt flag for DMADONE. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1	TOVIFG	R/W	0h	Raw interrupt flag for sequence conversion timeout overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
0	OVIFG	R/W	0h	Raw interrupt flag for MEMRESx overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.

9.3.8 RIS Register (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 9-12 and described in Table 9-19.

Return to the Table 9-10.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 9-12. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ASCDONE	UVIFG	DMADONE	INIFG	LOWIFG	HIGHIFG	TOVIFG	OVIFG
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-19. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7	ASCDONE	R	0h	Raw interrupt flag for ASC done 0h = Interrupt is not pending. 1h = Interrupt is pending.

Table 9-19. RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	UVIFG	R/W	0h	Raw interrupt flag for MEMRESx underflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
5	DMADONE	R	0h	Raw interrupt flag for DMADONE. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1	TOVIFG	R/W	0h	Raw interrupt flag for sequence conversion trigger overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
0	OVIFG	R/W	0h	Raw interrupt flag for MEMRESx overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.

9.3.9 MIS Register (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 9-13 and described in Table 9-20.

Return to the Table 9-10.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 9-13. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ASCDONE	UVIFG	DMADONE	INIFG	LOWIFG	HIGHIFG	TOVIFG	OVIFG
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-20. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7	ASCDONE	R	0h	Masked interrupt status for ASC done 0h = Interrupt is not pending. 1h = Interrupt is pending.

Table 9-20. MIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	UVIFG	R/W	0h	Raw interrupt flag for MEMRESx underflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
5	DMADONE	R	0h	Raw interrupt flag for DMADONE. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1	TOVIFG	R/W	0h	Raw interrupt flag for sequence conversion timeout overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
0	OVIFG	R/W	0h	Raw interrupt flag for MEMRESx overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.

9.3.10 ISET Register (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 9-14 and described in Table 9-21.

Return to the Table 9-10.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 9-14. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ASCDONE	UVIFG	DMADONE	INIFG	LOWIFG	HIGHIFG	TOVIFG	OVIFG
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-21. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7	ASCDONE	R	0h	Set ASC done flag in RIS 0h = Interrupt is not pending. 1h = Interrupt is pending.

Table 9-21. ISET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	UVIFG	R/W	0h	Raw interrupt flag for MEMRESx underflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
5	DMADONE	R	0h	Raw interrupt flag for DMADONE. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1	TOVIFG	R/W	0h	Raw interrupt flag for sequence conversion timeout overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
0	OVIFG	R/W	0h	Raw interrupt flag for MEMRESx overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.

9.3.11 ICLR Register (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 9-15 and described in Table 9-22.

Return to the Table 9-10.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 9-15. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ASCDONE	UVIFG	DMADONE	INIFG	LOWIFG	HIGHIFG	TOVIFG	OVIFG
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-22. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7	ASCDONE	R	0h	Clear ASC done flag in RIS 0h = Interrupt is not pending. 1h = Interrupt is pending.

Table 9-22. ICLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	UVIFG	R/W	0h	Raw interrupt flag for MEMRESx underflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
5	DMADONE	R/W	0h	Raw interrupt flag for DMADONE. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1	TOVIFG	R/W	0h	Raw interrupt flag for sequence conversion timeout overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
0	OVIFG	R/W	0h	Raw interrupt flag for MEMRESx overflow. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.

9.3.12 IMASK Register (Offset = 1058h) [Reset = 00000000h]

IMASK is shown in Figure 9-16 and described in Table 9-23.

Return to the Table 9-10.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 9-16. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MEMRESIFG0
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			INIFG	LOWIFG	HIGHIFG	RESERVED	
R-0h			R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 9-23. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-5	RESERVED	R	0h	
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1-0	RESERVED	R	0h	

9.3.13 RIS Register (Offset = 1060h) [Reset = 00000000h]

RIS is shown in Figure 9-17 and described in Table 9-24.

Return to the Table 9-10.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 9-17. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MEMRESIFG0
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			INIFG	LOWIFG	HIGHIFG	RESERVED	
R-0h			R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 9-24. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-5	RESERVED	R	0h	
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1-0	RESERVED	R	0h	

9.3.14 MIS Register (Offset = 1068h) [Reset = 00000000h]

MIS is shown in Figure 9-18 and described in Table 9-25.

Return to the Table 9-10.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 9-18. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MEMRESIFG0
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			INIFG	LOWIFG	HIGHIFG	RESERVED	
R-0h			R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 9-25. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-5	RESERVED	R	0h	
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1-0	RESERVED	R	0h	

9.3.15 ISET Register (Offset = 1070h) [Reset = 00000000h]

ISET is shown in Figure 9-19 and described in Table 9-26.

Return to the Table 9-10.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 9-19. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MEMRESIFG0
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			INIFG	LOWIFG	HIGHIFG	RESERVED	
R-0h			R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 9-26. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-5	RESERVED	R	0h	
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1-0	RESERVED	R	0h	

9.3.16 ICLR Register (Offset = 1078h) [Reset = 00000000h]

ICLR is shown in Figure 9-20 and described in Table 9-27.

Return to the Table 9-10.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 9-20. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MEMRESIFG0
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED			INIFG	LOWIFG	HIGHIFG	RESERVED	
R-0h			R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 9-27. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-5	RESERVED	R	0h	
4	INIFG	R/W	0h	Mask INIFG in MIS_EX register. 0h = Interrupt is not pending. 1h = Interrupt is pending.
3	LOWIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being below than the WCLOWx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
2	HIGHIFG	R/W	0h	Raw interrupt flag for the MEMRESx result register being higher than the WCHIGHx threshold of the window comparator. This bit is reset to 0 by IIDX read or when corresponding bit in ICLR_EX is set to 1. 0h = Interrupt is not pending. 1h = Interrupt is pending.
1-0	RESERVED	R	0h	

9.3.17 IMASK Register (Offset = 1088h) [Reset = 00000000h]

IMASK is shown in Figure 9-21 and described in Table 9-28.

Return to the Table 9-10.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 9-21. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 9-28. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-0	RESERVED	R	0h	

9.3.18 RIS Register (Offset = 1090h) [Reset = 00000000h]

RIS is shown in Figure 9-22 and described in Table 9-29.

Return to the Table 9-10.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 9-22. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 9-29. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-0	RESERVED	R	0h	

9.3.19 MIS Register (Offset = 1098h) [Reset = 00000000h]

MIS is shown in Figure 9-23 and described in Table 9-30.

Return to the Table 9-10.

Extension of Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 9-23. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 9-30. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-0	RESERVED	R	0h	

9.3.20 ISET Register (Offset = 10A0h) [Reset = 00000000h]

ISET is shown in Figure 9-24 and described in Table 9-31.

Return to the Table 9-10.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 9-24. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 9-31. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-0	RESERVED	R	0h	

9.3.21 ICLR Register (Offset = 10A8h) [Reset = 00000000h]

ICLR is shown in Figure 9-25 and described in Table 9-32.

Return to the Table 9-10.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 9-25. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				MEMRESIFG3	MEMRESIFG2	MEMRESIFG1	MEMRESIFG0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 9-32. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	MEMRESIFG3	R/W	0h	Raw interrupt status for MEMRES3. This bit is set to 1 when MEMRES3 is loaded with a new conversion result. Reading MEMRES3 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
10	MEMRESIFG2	R/W	0h	Raw interrupt status for MEMRES2. This bit is set to 1 when MEMRES2 is loaded with a new conversion result. Reading MEMRES2 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
9	MEMRESIFG1	R/W	0h	Raw interrupt status for MEMRES1. This bit is set to 1 when MEMRES1 is loaded with a new conversion result. Reading MEMRES1 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
8	MEMRESIFG0	R/W	0h	Raw interrupt status for MEMRES0. This bit is set to 1 when MEMRES0 is loaded with a new conversion result. Reading MEMRES0 register will clear this bit, or when the corresponding bit in ICLR is set to 1 0h = No new data ready. 1h = A new data is ready to be read.
7-0	RESERVED	R	0h	

9.3.22 CTL0 Register (Offset = 1100h) [Reset = 00000000h]

CTL0 is shown in Figure 9-26 and described in Table 9-33.

Return to the Table 9-10.

Control Register 0

Figure 9-26. CTL0 Register

31	30	29	28	27	26	25	24
RESERVED					SCLKDIV		
R-0h					R/W-0h		
23	22	21	20	19	18	17	16
RESERVED							PWRDN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENC
R-0h							RH/W-0h

Table 9-33. CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	
26-24	SCLKDIV	R/W	0h	Sample clock divider 0h = Do not divide clock source 1h = Divide clock source by 2 2h = Divide clock source by 4 3h = Divide clock source by 8 4h = Divide clock source by 16 5h = Divide clock source by 24 6h = Divide clock source by 32 7h = Divide clock source by 48
23-17	RESERVED	R	0h	
16	PWRDN	R/W	0h	Power down policy 0h = ADC is powered down on completion of a conversion if there is no pending trigger 1h = ADC remains powered on as long as it is enabled through software.
15-1	RESERVED	R	0h	
0	ENC	RH/W	0h	Enable conversion 0h = Conversion disabled. ENC change from ON to OFF will abort single or repeat sequence on a MEMCTLx boundary. The current conversion will finish and result stored in corresponding MEMRESx. 1h = Conversion enabled. ADC sequencer waits for valid trigger (software or hardware).

9.3.23 CTL1 Register (Offset = 1104h) [Reset = 00000000h]

CTL1 is shown in Figure 9-27 and described in Table 9-34.

Return to the Table 9-10.

Control Register 1

Figure 9-27. CTL1 Register

31	30	29	28	27	26	25	24
RESERVED	AVGD			RESERVED	AVGN		
R-0h	R/W-0h			R-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED			SAMPMODE	RESERVED		CONSEQ	
R-0h			R/W-0h	R-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED							SC
R-0h							RH/W-0h
7	6	5	4	3	2	1	0
RESERVED							TRIGSRC
R-0h							R/W-0h

Table 9-34. CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	
30-28	AVGD	R/W	0h	Hardware averager denominator. The number to divide the accumulated value by (this is a shift). Note result register is maximum of 16-bits long so if not shifted appropriately result will be truncated. 0h (R/W) = No shift 1h (R/W) = 1 bit shift 2h (R/W) = 2 bit shift 3h (R/W) = 3 bit shift 4h (R/W) = 4 bit shift 5h (R/W) = 5 bit shift 6h (R/W) = 6 bit shift 7h (R/W) = 7 bit shift
27	RESERVED	R	0h	
26-24	AVGN	R/W	0h	Hardware averager numerator. Selects number of conversions to accumulate for current MEMCTLx and then it is divided by AVGD. Result will be stored in MEMRESx. 0h (R/W) = Disables averager 1h (R/W) = Averages 2 conversions before storing in MEMRESx register 2h (R/W) = Averages 4 conversions before storing in MEMRESx register 3h (R/W) = Averages 8 conversions before storing in MEMRESx register 4h (R/W) = Averages 16 conversions before storing in MEMRESx register 5h (R/W) = Averages 32 conversions before storing in MEMRESx register 6h (R/W) = Averages 64 conversions before storing in MEMRESx register 7h (R/W) = Averages 128 conversions before storing in MEMRESx register
23-21	RESERVED	R	0h	

Table 9-34. CTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	SAMPMODE	R/W	0h	Sample mode. This bit selects the source of the sampling signal. MANUAL option is not valid when TRIGSRC is selected as hardware event trigger. 0h = AUTO 1h = MANUAL
19-18	RESERVED	R	0h	
17-16	CONSEQ	R/W	0h	Conversion sequence mode 0h = ADC channel in MEMCTLx pointed by STARTADD will be converted once 1h = ADC channel sequence pointed by STARTADD and ENDADD will be converted once 2h = ADC channel in MEMCTLx pointed by STARTADD will be converted repeatedly 3h = ADC channel sequence pointed by STARTADD and ENDADD will be converted repeatedly
15-9	RESERVED	R	0h	
8	SC	RH/W	0h	Start of conversion 0h = When SAMPMODE is set to MANUAL, clearing this bit will end the sample phase and the conversion phase will start. When SAMPMODE is set to AUTO, writing 0 has no effect. 1h = When SAMPMODE is set to MANUAL, setting this bit will start the sample phase. Sample phase will last as long as this bit is set. When SAMPMODE is set to AUTO, setting this bit will trigger the timer based sample time.
7-1	RESERVED	R	0h	
0	TRIGSRC	R/W	0h	Sample trigger source 0h = Software trigger 1h = Hardware event trigger

9.3.24 CTL2 Register (Offset = 1108h) [Reset = 00000000h]

CTL2 is shown in Figure 9-28 and described in Table 9-35.

Return to the Table 9-10.

Control Register 2

Figure 9-28. CTL2 Register

31	30	29	28	27	26	25	24
RESERVED					ENDADD		
R-0h					R/W-0h		
23	22	21	20	19	18	17	16
RESERVED					STARTADD		
R-0h					R/W-0h		
15	14	13	12	11	10	9	8
SAMPCNT					FIFOEN	RESERVED	DMAEN
R/W-0h					R/W-0h	R-0h	RH/W-0h
7	6	5	4	3	2	1	0
RESERVED					RES		DF
R-0h					R/W-0h		R/W-0h

Table 9-35. CTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28-24	ENDADD	R/W	0h	Sequence end address. These bits select which MEMCTLx is the last one for the sequence mode. The value of ENDADD is 0x00 to 0x17, corresponding to MEMRES0 to MEMRES23. 00h = MEMCTL0 is selected as end address of sequence. 01h = MEMCTL1 is selected as end address of sequence. 02h = MEMCTL2 is selected as end address of sequence. 03h = MEMCTL3 is selected as end address of sequence.
23-21	RESERVED	R	0h	
20-16	STARTADD	R/W	0h	Sequencer start address. These bits select which MEMCTLx is used for single conversion or as first MEMCTL for sequence mode. The value of STARTADD is 0x00 to 0x17, corresponding to MEMRES0 to MEMRES23. 00h = MEMCTL0 is selected as start address of a sequence or for a single conversion. 01h = MEMCTL1 is selected as start address of a sequence or for a single conversion. 02h = MEMCTL2 is selected as start address of a sequence or for a single conversion. 03h = MEMCTL3 is selected as start address of a sequence or for a single conversion.
15-11	SAMPCNT	R/W	0h	Number of ADC converted samples to be transferred on a DMA trigger 0h = Minimum value 18h = Maximum value
10	FIFOEN	R/W	0h	Enable FIFO based operation 0h = Disable 1h = Enable
9	RESERVED	R	0h	

Table 9-35. CTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	DMAEN	RH/W	0h	Enable DMA trigger for data transfer. Note: DMAEN bit is cleared by hardware based on DMA done signal at the end of data transfer. Software has to re-enable DMAEN bit for ADC to generate DMA triggers. 0h (R/W) = DMA trigger not enabled 1h (R/W) = DMA trigger enabled
7-3	RESERVED	R	0h	
2-1	RES	R/W	0h	Resolution. These bits define the resolution of ADC conversion result. Note : A value of 3 defaults to 12-bits resolution. 0h = 12-bits resolution 1h = 10-bits resolution 2h = 8-bits resolution
0	DF	R/W	0h	Data read-back format. Data is always stored in binary unsigned format. 0h = Digital result reads as Binary Unsigned. 1h = Digital result reads Signed Binary. (2s complement), left aligned.

9.3.25 CTL3 Register (Offset = 110Ch) [Reset = 00000000h]

CTL3 is shown in Figure 9-29 and described in Table 9-36.

Return to the Table 9-10.

Control Register 3. This register is used to configure ADC for ad-hoc single conversion.

Figure 9-29. CTL3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		ASCVRSEL		RESERVED			ASCSTIME
R-0h		R/W-0h		R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			ASCCHSEL				
R-0h			R/W-0h				

Table 9-36. CTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	
13-12	ASCVRSEL	R/W	0h	Selects voltage reference for ASC operation. VEREFM must be connected to on-board ground when external reference option is selected. Note: Writing value 0x3 defaults to INTREF. 0h = VDDA reference. 1h = EXTREF pin reference. 2h = Internal reference.
11-9	RESERVED	R	0h	
8	ASCSTIME	R/W	0h	ASC sample time compare value select. This is used to select between SCOMP0 and SCOMP1 registers for ASC operation. 0h = Select SCOMP0 1h = Select SCOMP1
7-5	RESERVED	R	0h	

Table 9-36. CTL3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	ASCCHSEL	R/W	0h	ASC channel select 00h = Selects channel 0 01h = Selects channel 1 02h = Selects channel 2 03h = Selects channel 3 04h = Selects channel 4 05h = Selects channel 5 06h = Selects channel 6 07h = Selects channel 7 08h = Selects channel 8 09h = Selects channel 9 0Ah = Selects channel 10 0Bh = Selects channel 11 0Ch = Selects channel 12 0Dh = Selects channel 13 0Eh = Selects channel 14 0Fh = Selects channel 15 10h = Selects channel 16 11h = Selects channel 17 12h = Selects channel 18 13h = Selects channel 19 14h = Selects channel 20 15h = Selects channel 21 16h = Selects channel 22 17h = Selects channel 23 18h = Selects channel 24 19h = Selects channel 25 1Ah = Selects channel 26 1Bh = Selects channel 27 1Ch = Selects channel 28 1Dh = Selects channel 29 1Eh = Selects channel 30 1Fh = Selects channel 31

9.3.26 SCOMP0 Register (Offset = 1114h) [Reset = 00000000h]

SCOMP0 is shown in Figure 9-30 and described in Table 9-37.

Return to the Table 9-10.

Sample time compare 0 register. Specifies the sample time, in number of ADC sample clock cycles. CTL0.ENC must be 0 to write to this register.

Figure 9-30. SCOMP0 Register

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
RESERVED																						VAL									
R-0h																						R/W-0h									

Table 9-37. SCOMP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9-0	VAL	R/W	0h	Specifies the number of sample clocks. When VAL = 0 or 1, number of sample clocks = Sample clock divide value. When VAL > 1, number of sample clocks = VAL x Sample clock divide value. Note: Sample clock divide value is not the value written to SCLKDIV but the actual divide value (SCLKDIV = 2 implies divide value is 4). Example: VAL = 4, SCLKDIV = 3 implies 32 sample clock cycles.

9.3.27 SCOMP1 Register (Offset = 1118h) [Reset = 00000000h]

SCOMP1 is shown in Figure 9-31 and described in Table 9-38.

Return to the Table 9-10.

Sample time compare 1 register. Specifies the sample time, in number of ADC sample clock cycles. CTL0.ENC must be 0 to write to this register.

Figure 9-31. SCOMP1 Register

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
RESERVED																						VAL									
R-0h																						R/W-0h									

Table 9-38. SCOMP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9-0	VAL	R/W	0h	Specifies the number of sample clocks. When VAL = 0 or 1, number of sample clocks = Sample clock divide value. When VAL > 1, number of sample clocks = VAL x Sample clock divide value. Note: Sample clock divide value is not the value written to SCLKDIV but the actual divide value (SCLKDIV = 2 implies divide value is 4). Example: VAL = 4, SCLKDIV = 3 implies 32 sample clock cycles.

9.3.28 REFCFG Register (Offset = 111Ch) [Reset = 00000000h]

REFCFG is shown in Figure 9-32 and described in Table 9-39.

Return to the Table 9-10.

Reference buffer configuration register

Figure 9-32. REFCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			IBPROG		RESERVED	REFVSEL	REFEN
R-0h			R/W-0h		R-0h	R/W-0h	R/W-0h

Table 9-39. REFCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4-3	IBPROG	R/W	0h	Configures reference buffer bias current output value 0h = 1uA 1h = 0.5uA 2h = 2uA 3h = 0.67uA
2	RESERVED	R	0h	
1	REFVSEL	R/W	0h	Configures reference buffer output voltage 0h = Reference buffer generates 2.5V output 1h = Reference buffer generates 1.4V output
0	REFEN	R/W	0h	Reference buffer enable 0h = Disable 1h = Enable

9.3.29 WCLOW Register (Offset = 1148h) [Reset = 00000000h]

WCLOW is shown in Figure 9-33 and described in Table 9-40.

Return to the Table 9-10.

Window Comparator Low Threshold Register.
The data format that is used to write and read WCLOW depends on the value of DF bit in CTL2 register.
CTL0.ENC must be 0 to write to this register.
Note: Change in ADC data format or resolution does not reset WCLOW.

Figure 9-33. WCLOW Register

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
RESERVED																DATA															
R-0h																R/W-0h															

Table 9-40. WCLOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	DATA	R/W	0h	If DF = 0, unsigned binary format has to be used. The value based on the resolution has to be right aligned with the MSB on the left. For 10-bits and 8-bits resolution, unused bits have to be 0s. If DF = 1, 2s-complement format has to be used. The value based on the resolution has to be left aligned with the LSB on the right. For 10-bits and 8-bits resolution, unused bits have to be 0s.

9.3.30 WCHIGH Register (Offset = 1150h) [Reset = 00000000h]

WCHIGH is shown in Figure 9-34 and described in Table 9-41.

Return to the Table 9-10.

Window Comparator High Threshold Register.
The data format that is used to write and read WCHIGH depends on the value of DF bit in CTL2 register.
CTL0.ENC must be 0 to write to this register.
Note: Change in ADC data format or resolution does not reset WCHIGH.

Figure 9-34. WCHIGH Register

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
RESERVED																DATA															
R-0h																R/W-0h															

Table 9-41. WCHIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	DATA	R/W	0h	If DF = 0, unsigned binary format has to be used. The threshold value has to be right aligned, with the MSB on the left. For 10-bits and 8-bits resolution, unused bit have to be 0s. If DF = 1, 2s-complement format has to be used. The value based on the resolution has to be left aligned with the LSB on the right. For 10-bits and 8-bits resolution, unused bit have to be 0s.

9.3.31 FIFODATA Register (Offset = 1160h) [Reset = 00000000h]

FIFODATA is shown in Figure 9-35 and described in Table 9-42.

Return to the Table 9-10.

FIFO data register. This is a virtual register used to do read from FIFO.

Figure 9-35. FIFODATA Register

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
DATA																															
R/W-0h																															

Table 9-42. FIFODATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	Read from this data field returns the ADC sample from FIFO.

9.3.32 ASCRES Register (Offset = 1170h) [Reset = 00000000h]

ASCRES is shown in Figure 9-36 and described in Table 9-43.

Return to the Table 9-10.

ASC result register

Figure 9-36. ASCRES Register

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
RESERVED																DATA															
R-0h																R-0h															

Table 9-43. ASCRES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	DATA	R	0h	Result of ADC ad-hoc single conversion. If DF = 0, unsigned binary: The conversion result is right aligned. In 10 and 8 bit modes, the unused MSB bits are forced to 0. If DF = 1, 2s-complement format: The conversion result is left aligned. In 10 and 8 bit modes, the unused LSB bits are forced to 0. The data is stored in the right-justified format and is converted to the left-justified 2s-complement format during read back.

9.3.33 MEMCTL_y Register (Offset = 1180h + formula) [Reset = 00000000h]

MEMCTL_y is shown in Figure 9-37 and described in Table 9-44.

Return to the Table 9-10.

Conversion Memory Control Register.
CTL0.ENC must be 0 to write to this register.

Offset = 1180h + (y * 4h); where y = 0h to 17h

Figure 9-37. MEMCTL_y Register

31	30	29	28	27	26	25	24
RESERVED			WINCOMP	RESERVED			TRIG
R-0h			R/W-0h	R-0h			R/W-0h
23	22	21	20	19	18	17	16
RESERVED			BCSEN	RESERVED			AVGEN
R-0h			R/W-0h	R-0h			R/W-0h
15	14	13	12	11	10	9	8
RESERVED			STIME	RESERVED		VRSEL	
R-0h			R/W-0h	R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED			CHANSEL				
R-0h			R/W-0h				

Table 9-44. MEMCTL_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	WINCOMP	R/W	0h	Enable window comparator. 0h = Disable 1h = Enable
27-25	RESERVED	R	0h	
24	TRIG	R/W	0h	Trigger policy. Indicates if a trigger will be needed to step to the next MEMCTL in the sequence or to perform next conversion in the case of repeat single channel conversions. 0h = Next conversion is automatic 1h = Next conversion requires a trigger
23-21	RESERVED	R	0h	
20	BCSEN	R/W	0h	Enable burn out current source. 0h = Disable 1h = Enable
19-17	RESERVED	R	0h	
16	AVGEN	R/W	0h	Enable hardware averaging. 0h (R/W) = Averaging disabled. 1h = Averaging enabled.
15-13	RESERVED	R	0h	
12	STIME	R/W	0h	Selects the source of sample timer period between SCOMP0 and SCOMP1. 0h = Select SCOMP0 1h = Select SCOMP1
11-10	RESERVED	R	0h	

Table 9-44. MEMCTL_y Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	VRSEL	R/W	0h	Voltage reference selection. VEREFM must be connected to on-board ground when external reference option is selected. Note: Writing value 0x3 defaults to INTREF. 0h = VDDA reference 1h = External reference from pin 2h = Internal reference
7-5	RESERVED	R	0h	
4-0	CHANSEL	R/W	0h	Input channel select. 00h = Selects channel 0 01h = Selects channel 1 02h = Selects channel 2 03h = Selects channel 3 04h = Selects channel 4 05h = Selects channel 5 06h = Selects channel 6 07h = Selects channel 7 08h = Selects channel 8 09h = Selects channel 9 0Ah = Selects channel 10 0Bh = Selects channel 11 0Ch = Selects channel 12 0Dh = Selects channel 13 0Eh = Selects channel 14 0Fh = Selects channel 15 10h = Selects channel 16 11h = Selects channel 17 12h = Selects channel 18 13h = Selects channel 19 14h = Selects channel 20 15h = Selects channel 21 16h = Selects channel 22 17h = Selects channel 23 18h = Selects channel 24 19h = Selects channel 25 1Ah = Selects channel 26 1Bh = Selects channel 27 1Ch = Selects channel 28 1Dh = Selects channel 29 1Eh = Selects channel 30 1Fh = Selects channel 31

9.3.34 MEMRES_y Register (Offset = 1280h + formula) [Reset = 00000000h]

MEMRES_y is shown in Figure 9-38 and described in Table 9-45.

Return to the Table 9-10.

Memory Result Register

Offset = 1280h + (y * 4h); where y = 0h to 17h

Figure 9-38. MEMRES_y Register

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
RESERVED																DATA															
R-0h																R-0h															

Table 9-45. MEMRES_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	DATA	R	0h	MEMRES result register. If DF = 0, unsigned binary: The conversion results are right aligned. In 10 and 8 bit modes, the unused MSB bits are forced to 0. If DF = 1, 2s-complement format: The conversion results are left aligned. In 10 and 8 bit modes, the unused LSB bits are forced to 0. The data is stored in the right-justified format and is converted to the left-justified 2s-complement format during read back.

9.3.35 STATUS Register (Offset = 1340h) [Reset = 00000000h]

STATUS is shown in Figure 9-39 and described in Table 9-46.

Return to the Table 9-10.

Status Register

Figure 9-39. STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					ASCACT	REFBUFRDY	BUSY
R-0h					R-0h	R-0h	R-0h

Table 9-46. STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	
2	ASCACT	R	0h	ASC active 0h = Idle or done 1h = ASC active
1	REFBUFRDY	R	0h	Indicates reference buffer is powered up and ready. 0h = Not ready 1h = Ready
0	BUSY	R	0h	Busy. This bit indicates that an active ADC sample or conversion operation is in progress. 0h = No ADC sampling or conversion in progress. 1h = ADC sampling or conversion is in progress.

Chapter 10
VREF

The VREF module contains a configurable voltage reference buffer which allows users to supply a stable internal reference to on-board analog peripherals. It also supports bringing in an external reference for applications where higher accuracy is required. This chapter describes the features and operation of the VREF module.

10.1 VREF Overview

VREF allows users to choose between using an internally generated reference voltage or using an externally provided reference voltage from outside the MCU.

The VREF module features include:

- 1.4V and 2.5V user-selectable internal references
- Sample and hold mode supports VREF operation down to STANDBY operating mode
- Internal reference supports ADC operation up to 1 Msps

shows the block diagram of the VREF module.

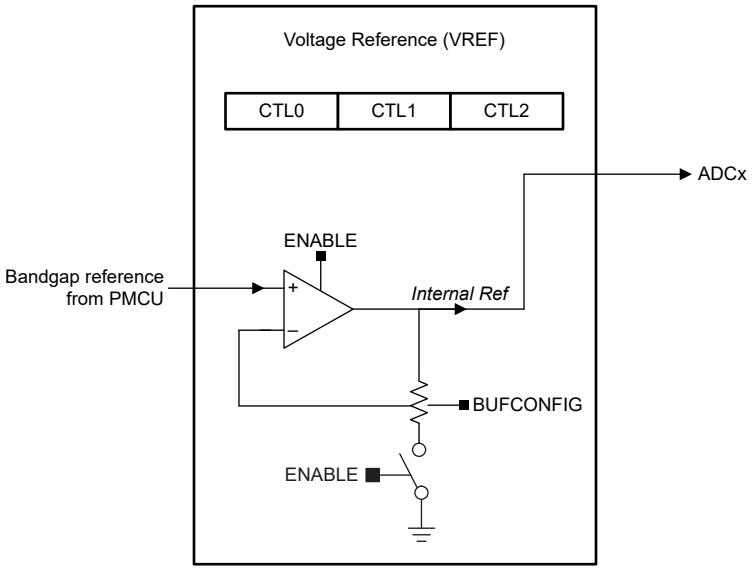


Figure 10-1. VREF Block Diagram

10.2 VREF Operation

The VREF module is configured with user software. The setup and operation of VREF is discussed in the following sections.

10.2.1 Internal Reference Generation

To use VREF to generate an internal voltage reference, the user must first enable the power to the module using the ENABLE control bit in the PWREN register and then enable the reference buffer using the ENABLE control bit in the CTL0 register. The VREF module generates voltage references based on the factory trimmed bandgap coming from the PMU. The bandgap reference is buffered through a noninverting amplifier to generate one of two internal reference voltages (1.4V or 2.5V). Only one voltage can be selected at a time using the BUFCONFIG control bit in CTL0.

After it is enabled and settled, the internal reference can be used as an accurate and stable voltage reference for the on-board ADC. Refer to the the ADC chapter for more info on how can leverage this reference voltage.

The VREF provides a READY indication bit in the CTL1 register. The first time the VREF is enabled, the READY bit will remain cleared until the VREF is started and settled, after which the READY bit will be set by hardware. If the VREF is disabled, the READY bit will be cleared back to zero by hardware. If the VREF is re-enabled later, the READY bit will not be set, and application software must manage the VREF startup time.

10.3 VREF Registers

Table 10-1 lists the memory-mapped registers for the VREF registers. All register offset addresses not listed in Table 10-1 should be considered as reserved locations and the register contents should not be modified.

Table 10-1. VREF Registers

Offset	Acronym	Register Name	Section
800h	PWREN	Power enable	Section 10.3.1
804h	RSTCTL	Reset Control	Section 10.3.2
814h	STAT	Status Register	Section 10.3.3
1000h	CLKDIV	Clock Divider	Section 10.3.4
1008h	CLKSEL	Clock Selection	Section 10.3.5
1100h	CTL0	Control 0	Section 10.3.6
1104h	CTL1	Control 1	Section 10.3.7
1108h	CTL2	Control 2	Section 10.3.8

Complex bit access types are encoded to fit into small table cells. Table 10-2 shows the codes that are used for access types in this section.

Table 10-2. VREF Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
K	K	Write protected by a key
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

10.3.1 PWREN Register (Offset = 800h) [Reset = XX00000Xh]

PWREN is shown in Figure 10-2 and described in Table 10-3.

Return to the Table 10-1.

Register to control the power state

Figure 10-2. PWREN Register

31	30	29	28	27	26	25	24
KEY							
W-X							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							K-X

Table 10-3. PWREN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	X	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R	0h	

0	ENABLE	K	X	Enable the power
				#none# must be set to 26h to write to this bit.
				0h = Disable Power
				1h = Enable Power

10.3.2 RSTCTL Register (Offset = 804h) [Reset = XX00000Xh]

RSTCTL is shown in Figure 10-3 and described in Table 10-4.

Return to the Table 10-1.

Register to control reset assertion and de-assertion

Figure 10-3. RSTCTL Register

31	30	29	28	27	26	25	24
KEY							
W-X							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
R-0h						WK-X	WK-X

Table 10-4. RSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	X	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	R	0h	

1	RESETSTKYCLR	WK	X	Clear the RESETSTKY bit in the STAT register
				#none# must be set to B1h to write to this bit.
				0h = Writing 0 has no effect
				1h = Clear reset sticky bit

0	RESETASSERT	WK	X	Assert reset to the peripheral
				#none# must be set to B1h to write to this bit.
				0h = Writing 0 has no effect
				1h = Assert reset

10.3.3 STAT Register (Offset = 814h) [Reset = 000X0000h]

STAT is shown in Figure 10-4 and described in Table 10-5.

Return to the Table 10-1.

peripheral enable and reset status

Figure 10-4. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							RESETSTKY
R-0h							R-X
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 10-5. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	X	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

10.3.4 CLKDIV Register (Offset = 1000h) [Reset = 00000000h]

CLKDIV is shown in Figure 10-5 and described in Table 10-6.

Return to the Table 10-1.

Clock Divider

Figure 10-5. CLKDIV Register

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

Table 10-6. CLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	
2-0	RATIO	R/W	0h	Selects divide ratio of module clock to be used in sample and hold logic

10.3.5 CLKSEL Register (Offset = 1008h) [Reset = 00000000h]

CLKSEL is shown in Figure 10-6 and described in Table 10-7.

Return to the Table 10-1.

Clock Selection

Figure 10-6. CLKSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUSCLK_SEL	MFCLK_SEL	LFCLK_SEL	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R-0h

Table 10-7. CLKSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	BUSCLK_SEL	R/W	0h	Selects BUSCLK as clock source if enabled
2	MFCLK_SEL	R/W	0h	Selects MFCLK as clock source if enabled
1	LFCLK_SEL	R/W	0h	Selects LFCLK as clock source if enabled
0	RESERVED	R	0h	

10.3.6 CTL0 Register (Offset = 1100h) [Reset = 00000000h]

CTL0 is shown in Figure 10-7 and described in Table 10-8.

Return to the Table 10-1.

Control 0 register.

Figure 10-7. CTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							SHMODE
R-0h							R/W-0h
7	6	5	4	3	2	1	0
BUFCONFIG	RESERVED						ENABLE
R/W-0h	R-0h						R/W-0h

Table 10-8. CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	SHMODE	R/W	0h	This bit enable sample and hold mode 0h = Sample and hold mode is disable 1h = Sample and hold mode is enable
7	BUFCONFIG	R/W	0h	These bits configure output buffer. 0h = output 2p5v : Configure Output Buffer to 2.5v 1h = output 1p4v : Configure Output Buffer to 1.4v
6-1	RESERVED	R	0h	
0	ENABLE	R/W	0h	This bit enables the VREF module. 0h = VREF is disabled 1h = VREF is enabled

10.3.7 CTL1 Register (Offset = 1104h) [Reset = 00000000h]

CTL1 is shown in Figure 10-8 and described in Table 10-9.

Return to the Table 10-1.

Control 1 register.

Figure 10-8. CTL1 Register

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

Table 10-9. CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	VREFLOSEL	R/W	0h	This bit select VREFLO pin
0	READY	R	0h	These bits defines status of VREF 0h = VREF output is not ready 1h = VREF output is ready

10.3.8 CTL2 Register (Offset = 1108h) [Reset = 00000000h]

CTL2 is shown in Figure 10-9 and described in Table 10-10.

Return to the Table 10-1.

Control 2 register.

Figure 10-9. CTL2 Register

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
HCYCLE																SHCYCLE															
R/W-0h																R/W-0h															

Table 10-10. CTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	HCYCLE	R/W	0h	Hold cycle count Total cycles of module clock for hold phase when VREF is working in sample and hold mode in STANDBY to save power. Please refer VREF section of data sheet for recommended values of sample and hold times. 0h = smallest hold cycle FFFFh = largest hold cycle
15-0	SHCYCLE	R/W	0h	Sample and Hold cycle count Total cycles of module clock for sample and hold phase when VREF is working in sample and hold mode in STANDBY to save power. This field should be greater than HCYCLE field. The difference between this field and HCYCLE gives the number of cycles of sample phase. Please refer VREF section of data sheet for recommended values of sample and hold times. 0h = smallest sample and hold cycle count FFFFh = largest sample and hold cycle

Chapter 11

UART

The universal asynchronous receiver-transmitter (UART) module provides an interface for serial communication protocols.

11.1 UART Overview

11.1.1 Purpose of the Peripheral

This interface can be used transfer data between a MSPM0 device and another device with an asynchronous serial communication protocol like LIN (local interconnection network), ISO7816 (Smart card protocol), IrDA (infrared data association), hardware flow control (CTS/RTS) and multiprocessor communications are supported.

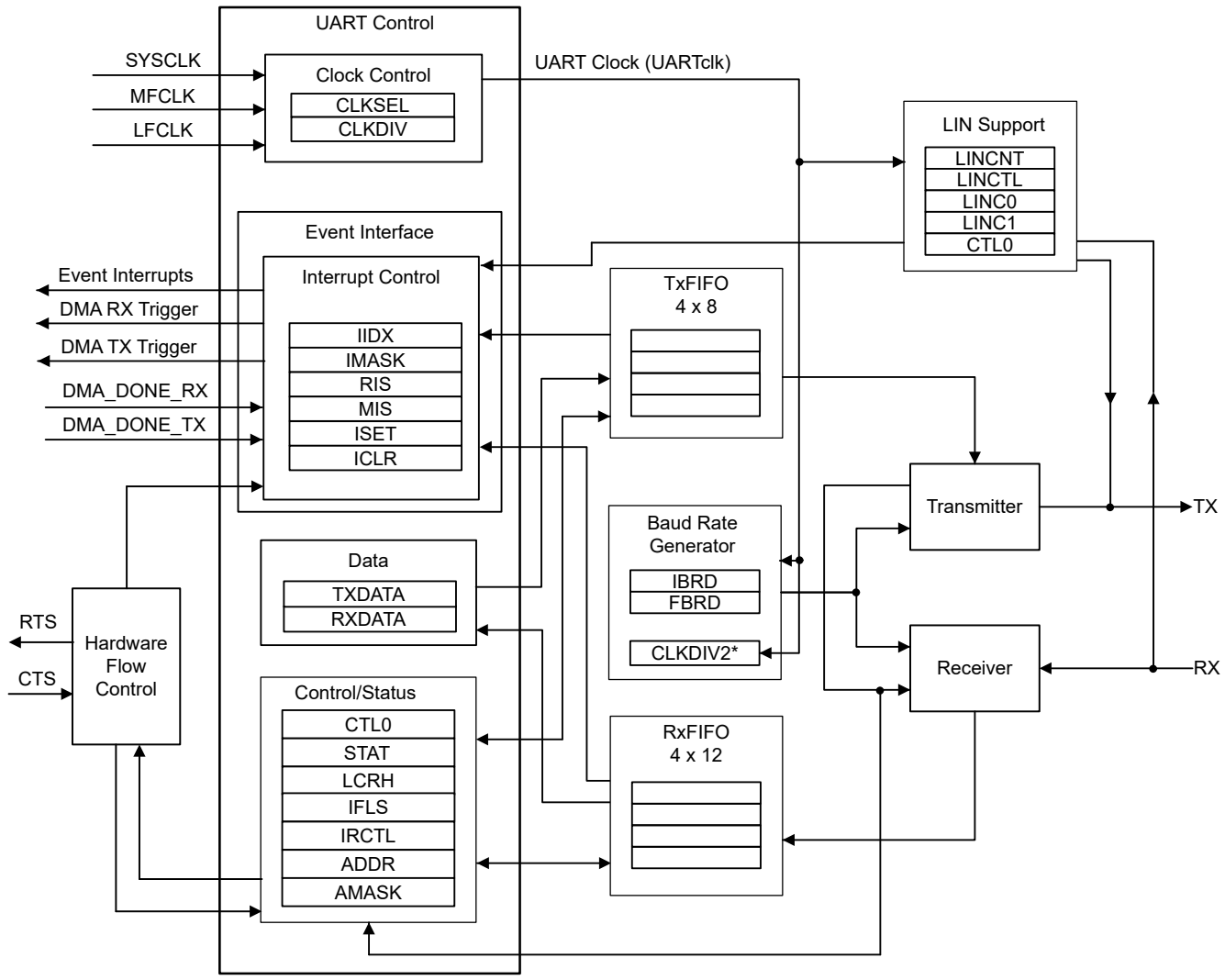
11.1.2 Features

The UART controller includes the following features:

- Fully programmable serial interface
  - 5, 6, 7 or 8 data bits
  - Even, odd, stick, or no-parity bit generation and detection
  - 1 or 2 stop bit generation
  - LSB-first or MSB-first data transmit and receive
  - Line-break detection
  - Glitch filter on the input signals
  - Programmable baud-rate generation with oversampling by 16, 8 or 3
- Separated transmit and receive 4 depth FIFOs reduce CPU interrupt service loading
- Supports DMA data transfer
- Standard FIFO-level and End-of-Transmission interrupts
- Active in all low-power mode including stop and standby mode
- Support for waking up SYSOSC via an asynchronous fast clock request upon start bit detection when operating in low power modes (supports up to 19200B rates when using SYSOSC in FCL mode (1% accuracy)
- Support loopback mode operation
- Support hardware flow control
- Support 9-bit multi-drop configuration
- Protocols supported:
  - Local Interconnect Network (LIN) support
  - DALI
  - IrDA
  - ISO7816 Smart card
  - RS485
  - Manchester coding
  - Idle-Line Multiprocessor

There are two types of UART instances: Extend and Main. The following table compares the features of UART Extend and UART Main.

11.1.3 Functional Block Diagram



*CLKDIV2 is for IrDA mode only

Figure 11-1. UART Functional Block Diagram

11.2 UART Operation

This section describes the operation of the UART peripheral.

11.2.1 Clock Control

The UART internal functional clock is selected and divided from the functional clock of the IP.

- Use UARTx.CLKSEL register to select the source of the UART functional clock.
  - BUSSCLK: the current bus clock is selected as the source for UART. The current bus clock depends on power domain. If the UART instance is in power domain 1 (PD1) refer to MCLK, if the UART instance is in power domain 0 (PD0) refer to ULPCLK.
  - MFCLK: MFCLK is selected as the source for UART, refer to MFCLK.
  - LFCLK: LFCLK is selected as the source for UART, refer to LFCLK
- Use UARTx.CLKDIV register to select the divide ratio of the UART function clock,options are from divide by 1 to 8. For UART Extend, there is a CLKDIV2 register to further divide the UART function clock to support the IrDA mode. When using IrDA mode, CLKDIV2.RATIO must be set to 1h for proper IrDA clocking.

The selected source clock is always available and the frequency depends on the power mode, for more information, see the Clock Module (CKM) section. After enabling the UART module by setting the ENABLE bit, the module will be ready to start receiving and transmitting data.

11.2.2 Signal Descriptions

UART communications require two pins: Receive Data (RX) and Transmit Data (TX):

- RX (Receive Data): RX is the serial data input. Glitch filter and oversampling techniques are used on the receive signal to ensure accurate incoming data.
- TX (Transmit Data): TX is the serial data output. When the transmitter is enabled and no data needs to be transmitted, the TX pin will be held high. In some bidirectional protocols like ISO7816 Smart card, this pin is also used to receive data.

In hardware flow control mode, the following pins are also used:

- CTS (Clear To Send): when driven high by external signal, this signal blocks the data transmission at the end of the current transfer.
- RTS (Request To Send): when low, this signal indicates that the UART is ready to receive data.

11.2.3 General Architecture and Protocol

UART transmits and receives characters at a bit rate that is asynchronous to another device. Timing for each character is based on the selected baud-rate. The transmit and receive functions use the same baud-rate frequency.

In general, control registers should only be programmed when the UART is disabled (ENABLE bit in the UARTx.CTL0 register is cleared). If the UART is operating and gets disabled during transmit or receive operation, the current transaction gets completed before the UART stops. The baud-rate divisor registers (UARTx.IBRD and UARTx.FBRD) can be modified without disabling the UART.

11.2.3.1 Transmit Receive Logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. The control logic outputs the serial bit stream beginning with a start bit and followed by the data bits (LSB first), parity bit, and the stop bits according to the programmed configuration.

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Overrun, parity, frame error checking, and line-break detection are also performed, and their status accompanies the data that is written to the receive FIFO, this is the reason for 12-bit receive FIFO.

UARTx.CTL0.ENABLE bit is used to enable and disable the UART module, UARTx.CTL0.TXE and RXE bits are used to enable the transmit and receive mode, UARTx.LCRH.WLEN bit is used to configure the number of data bits transmitted or received in a frame, UARTx.LCRH.PEN is used to enable parity and UARTx.LCRH.STP2 is used to send two stop bits.

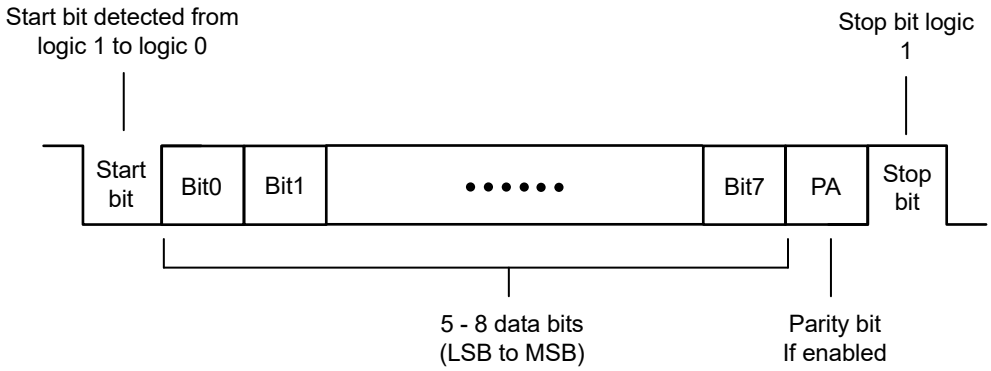


Figure 11-2. UART Character Frame

11.2.3.2 Bit Sampling

By default, UARTx.CTL0.HSE is set to 0 and 16 oversampling is selected and receiving bits are expected to have the length of 16 UART clock UARTclk cycles and is sampled on the 8th UARTclk cycle.

Setting the UARTx.CTL0.HSE bit to 1 selects the 8 oversampling where the receiving bits are expected to have the length of 8 UART clock UARTclk cycles and is sampled on the 4th UARTclk cycle.

Setting the UARTx.CTL0.HSE bit to 2 selects the 3 oversampling, the receiving bits are expected to have the length of 3 UART clock UARTclk cycles and are sampled on the 2nd UARTclk cycle.

The aforementioned scenarios assume an IBRD =1 and FBRD =0.

Depending on the application:

- Select oversampling by 3 or 8 to achieve higher speed with UARTclk/8 or UARTclk/3. In this case the receiver tolerance to clock deviation is reduced.
- Select oversampling by 16 to increase the tolerance of the receiver to clock deviations. The maximum speed is limited to UARTclk/16.

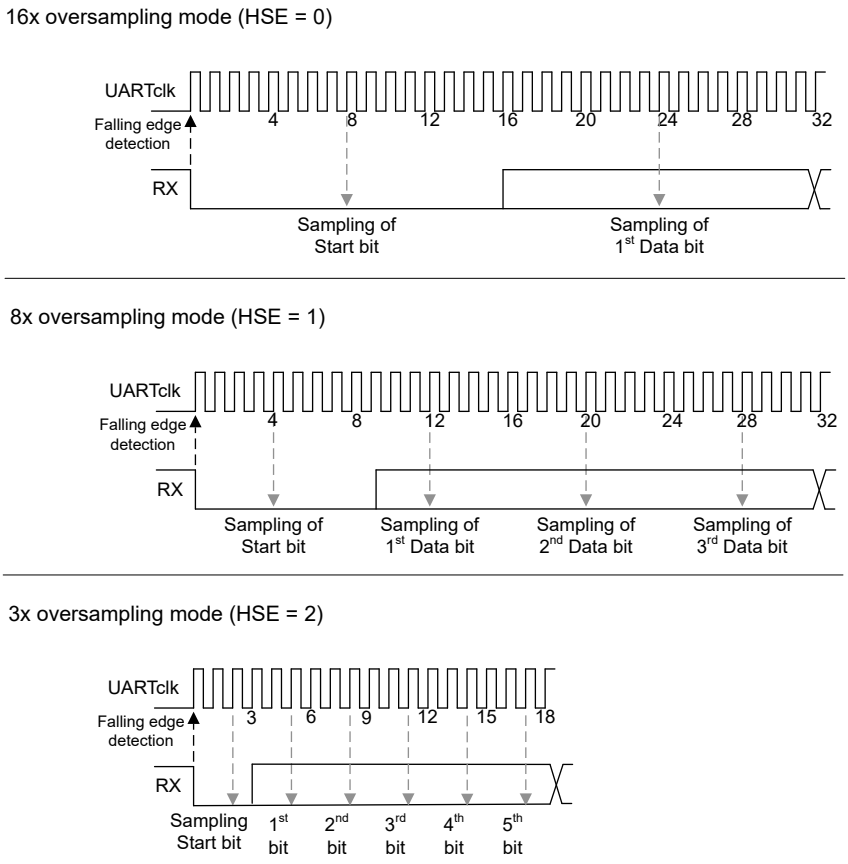


Figure 11-3. UART Oversampling mode

11.2.3.3 Majority Voting Feature

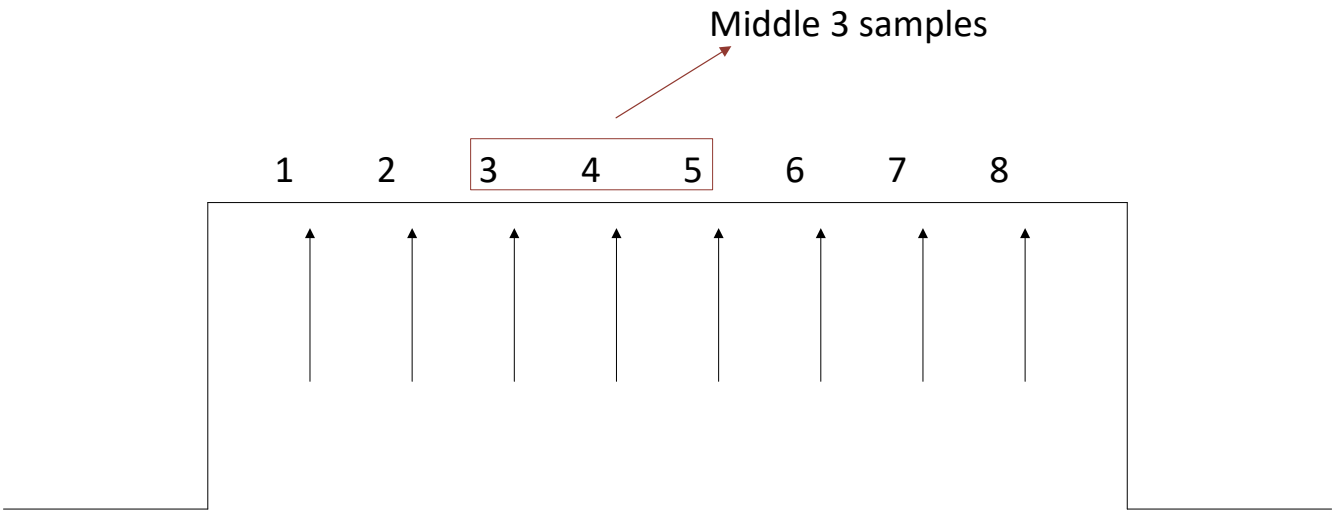


Figure 11-4. Majority voting for 8 oversampling

The majority voting feature of the UART provides noise immunity by sampling each bit 3 times in the center of the bit period. For example, in the case when the UARTx.CTL0.HSE is set to 0 with 16 oversampling; the 7th, 8th and 9th bit are sampled and the majority value is considered as final value to be sampled. When the UARTx.CTL0.HSE is set to 1 with 8 oversampling; then the 3rd ,4th and 5th bits are sampled and majority value is considered as final value to be sampled. The oversampling is only applicable for 16 and 8 oversampling. When the 3 samples used for majority vote are not equal; the RIS.NE (noise error bit) is set. The received data is transferred despite the noise error. The NE bit will get appended to the received data before storing it in the RXDATA register at bit position 12. Please note that the majority voting feature is implemented only for data bits. One can select majority voting or single sample using the MAJVOTE control bit in the CTL0 register.

Note

Even though UART instances have noise filters, this feature provides an extra layer of noise immunity for longer glitches

11.2.3.4 Baud Rate Generation

The baud-rate divisor is a 22-bit number consisting of a 16-bit integer and a 6-bit fractional part. The number formed by these two values is used by the baud-rate generator to determine the bit sample period. Having a fractional baud-rate divisor allows the UART to generate all of the standard baud-rates very accurately

The 16-bit integer is loaded through the UART Integer Baud-Rate Divisor UARTx.IBRD register and the 6-bit fractional part is loaded with the UART Fractional Baud-Rate Divisor UARTx.FBRD register.

The baud-rate divisor can be calculated by using the following formula:

$$BRD = UART\ Clock / (Oversampling \times Baud\ rate) \quad (10)$$

UART clock is the clock output of the UART clock control logic, configured by CLKSEL and CLKDIV. Oversampling is selected by the HSE bit in the UARTx.CTL0 register and it can be 16, 8 or 3.

- UARTx.IBRD = INT(BRD), integer part of the BRD
- UARTx.FBRD = BRD % 64 , fractional part

The integer part of BRD is loaded into UARTx.IBRD register. The 6-bit fractional number must be loaded into the UARTx.FBRD register.

Note

When IBRD = 0, FBRD is ignored and no data gets transferred by the UART. Similarly, when IBRD = 65535 (that is 0xFFFF), then FBRD must not be greater than zero. If this is exceeded it results in an aborted transmission or reception.

The following example shows a simple method to calculate IBRD.DIVINT and FBRD.DIVFRAC for a baud rate of 19200 bit/s:

UART Clock = 40 MHz
Oversampling = 16
Baudrate = 19200 bit/s

$$BRD = \frac{UARTclk}{OVS \times Baudrate} = \frac{40MHz}{16 \times 19200\ bit/s} = 130.2083333$$

→ UARTx.IBRD.DIVINT= 130 (=82h)

UARTx.FBRRD.DIFRAC
= INT((.2083333 x 64) + 0.5)
= INT(13.833333)
= 13d (= Dh)

Note: The adder '+0.5' ensures rounding to the closest integer value to keep the rounding error as small as possible

Figure 11-5. Baud Rate Configuration

When updating the baud-rate divisor (UARTx.IBRD or UARTx.IFRD), the UART.LCRH register must also be written, so any changes to the baud-rate divisor must be followed by a write to the LCRH register for the changes to take effect. The contents of the UART.IBRD and UART.FBRD registers are not updated until transmission or reception of the current character is complete.

11.2.3.5 Data Transmission

Data received or transmitted is stored in two FIFOs, though the receive FIFO has an extra four bits per character for status information.

Transmit data:

For transmission, data is written into the transmit FIFO. If the UART is enabled, it causes a data frame to start transmitting with the parameters indicated in the UARTx.LCRH register. Data continues to be transmitted until there is no data left in the transmit FIFO. The BUSY bit in the UARTx.STAT register is asserted as soon as data is written to the transmit FIFO (that is, if the FIFO is nonempty) and remains asserted while data is being transmitted. The BUSY bit is negated only when the transmit FIFO is empty, and the last character has been transmitted to the shift register, including the stop bits. The UART can indicate that it is busy even though the UART can no longer be enabled. BUSY also is set during the generation of a BREAK signal.

Receive data:

When the receiver is idle (the RX signal is continuously 1), and the data input goes low (a start bit has been received), the receive counter begins running and data is sampled on the different cycle based on the oversampling setting of the HSE bit in UARTx.CTL0 register .The start bit is valid and recognized if the RX signal is still low after certain number for cycles based on the oversampling setting. After a valid start bit is detected, successive data bits are sampled according to the programmed length of the data characters. The parity bit is then checked if parity mode is enabled. Data length and parity are defined in the UART.LCRH register. Oversampling is explained in Section 11.2.3.2.

Lastly, a valid stop bit is confirmed if the RXD signal is high, otherwise a framing error has occurred. When a full word is received, the data is stored in the receive FIFO along with any error bits associated with that word.

11.2.3.6 Error and Status

For received data, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO. The error and status can be can be retrieved by reading UARTx.RXDATA register as showing in Table 11-1.

Table 11-1. UART Error and Conditions

Error Condition(1)	Bit Field	Description
Framing error	FRMERR	A framing error occurs when a low stop bit is detected. When two stop bits are used, both stop bits are checked for framing error. When a framing error is detected, the FRMERR bit is set.
Parity error	PARERR	A parity error is a mismatch between the number of 1s in a character and the value of the parity bit. When an address bit is included in the character, it is included in the parity calculation. When a parity error is detected, the PARERR bit is set.
Receive overrun	OVRERR	An overrun error occurs when a character is loaded into RXDATA/FIFO before the prior character has been read. When an overrun occurs, the OVRERR bit is set.
Break condition	BRKERR	A break is detected when all received data, parity, and stop bits are 0. When a break condition is detected, the BRKERR bit is set. A break condition can also set the interrupt flag RXINT if the break interrupt enable IMASK.BRKERR bit is set.

(1) Framing error and break condition are not set when LIN mode is enabled, this pattern is used to signal a Sync frame for LIN. Break detection is not available for IRDA, Manchester and Dali mode.

UART module flag status can also be checked by reading UARTx.STAT register as showing in Table 11-2.

Table 11-2. UART Flag Status

Bit Field	Description
BUSY	This bit is set as soon as the transmit FIFO becomes nonempty (regardless of whether UART is enabled). In IDLE_Line mode the busy signal also stays set during the idle time generation.
RXFE	This bit is set when receive FIFO is empty. If the FIFO is disabled (FEN is 0), the receive holding register is full. If the FIFO is enabled (FEN is 1), the receive FIFO is full.
RXFF	This bit is set when receive FIFO is full. If the FIFO is disabled (FEN is 0), the receive holding register is empty. If the FIFO is enabled (FEN is 1), the receive FIFO is empty.
TXFE	This bit is set when transmit FIFO is empty. If the FIFO is disabled (FEN is 0), the transmit holding register is full. If the FIFO is enabled (FEN is 1), the transmit FIFO is full.
TXFF	This bit is set when transmit FIFO is full. If the FIFO is disabled (FEN is 0), the transmit holding register is empty. If the FIFO is enabled (FEN is 1), the transmit FIFO is empty.
CTS	This bit is set when CTS signal is asserted (low) and cleared when CTS signal is not asserted (high).
IDLE	IDLE mode has been detected in idle line multiprocessor mode. The IDLE bit is used as an address tag for each block of characters. In idle line multiprocessor format, this bit is set when a received character is an address.

11.2.3.7 Local Interconnect Network (LIN) Support

This section is only relevant for UART Extend, which supports LIN mode. Refer to the device data sheet for the device-specific configuration of UART extend and UART main.

For supporting local interconnect network (LIN) protocol, the following hardware enhancements are implemented in the UART module:

- 16 bit up-counter (LINCNT) clocked by the UART clock.
- Interrupt capability on counter overflow (CPU_INT.IMASK.LINOVF).
- 16 bit capture register (LINC0) with two configurable modes
  - Capture of LINCNT value on RXD falling edge. Interrupt capability on capture.
  - Compare of LINCNT with interrupt capability on match.
- 16 bit capture register (LINC1) can be configured:
  - Capture LINCNT value on RXD rising edge. Interrupt capability on capture.

LIN transmission

Sending the break signal can be done by setting the BRK bit in UARTx.LCRH register. This bit needs to be set before the data is written into the FIFO or transmit data register TXDATA.

To generate LIN responder signals such as wake signals, the TX pin can be configured by TXD_OUT and TXD_CTL_EN bits in register UARTx.CTL0 to be software controlled. By setting TXD_CTL_EN bit to 1, the TX output pin can be controlled by the TXD_OUT bit if the UART transmit is disabled (CTL0.TXE is cleared).

LIN reception

LIN commander issues a break field and sync field at the start of every frame. Hardware must be added such that the LIN responder software driver can reasonably detect BREAK-SYNC and measure the necessary timing parameters to adjust the baud rate or determine an error.

For LIN reception, break field detection and compare mode are needed. To configure these features:

1. Initialize LIN counter to 0 (UARTx.LINCNT = 0)
2. Enable counter compare match mode (UARTx.LINCTL.LINC0_MATCH = 1)
3. Load UARTx.LINC0 (counter capture 0 register) with counter value corresponding to 9.5 x T_{bit} (refer to LIN Specification for details on this timing).
4. Enable LINC0 match interrupt (CPU_INT.IMASK.LINC0 = 1)
5. Setup LIN count control (UARTx.LINCTL):
   - Enable count while low signal on RXD (LINCTL.CNTRXLOW = 1)
   - Enable LIN counter clearing on RXD falling edge (LINCTL.ZERONE = 1)
   - Enable LIN counter (LINCTL.CTRENA = 1)

   Optional:
   - User can enable the rising edge on UART TX signal interrupt (CPU_INT.IMASK.RXPE = 1), when the RXPE interrupt fires, the software can read the LINCTR directly to see the BREAK field timing.
   - User can enable the LIN counter overflow interrupt (CPU_INT.IMASK.LINOVF = 1) to detect the BREAK field is too long and overflows 16bit counter. The timeout can be calculated as $t_{Timeout}$= 2^{16}/ UART clock.

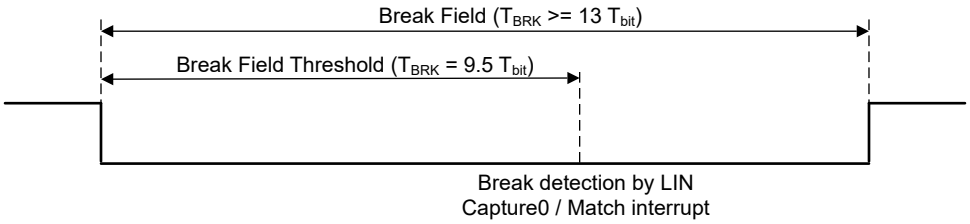


Figure 11-6. LIN Break Field Detection

Sync field validation is required to ensure accuracy of this LIN header part and to calculate the commander baud rate. The synch field consists of the data 0x55 inside a byte field (see Figure 11-7). After software detects a valid BREAK, it can then set the counter to measure the SYNC field. Both capture registers in LIN counter are used so that software sees fewer interrupts. Figure 11-7 shows the SYNC byte format. The LINCTR should be set to 0 on the start bit falling edge and count continuously. The LINC0 capture or RX falling edge interrupts fire at the falling edges of the RX line. During the interrupt processing, software can measure the individual HIGH-LOW times of the bits themselves using the values in the LINC0 and LINC1 registers to make sure all of the timings are valid.

The following flow describes a possible LIN sync field validation procedure:

1. Initialize LIN counter to 0 (UARTx.LINCNT = 0) after detecting a valid break field.
2. Enable interrupt on RX falling edge (CPU_INT.IMASK.RXNE = 1)
3. Setup LIN count control (LINCTL):
   - Enable LIN counter capture on raising RX edge (LINCTL.LINC1CAP = 1)
   - Enable LIN counter capture on falling RX edge (LINCTL.LINC0CAP = 1)
   - Enable LIN counter clearing on RX falling edge (LINCTL.ZERONE = 1)
   - Enable LIN counter (LINCTL.CTRENA = 1)

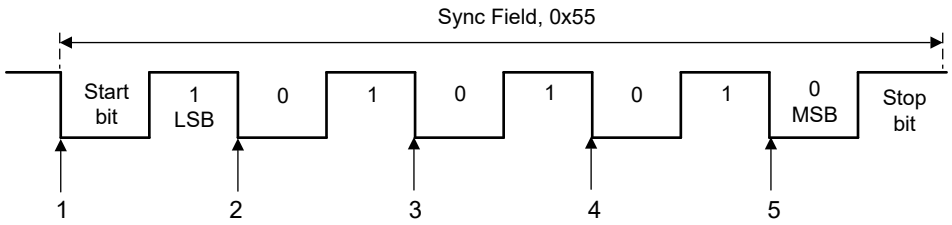


Figure 11-7. LIN SYNC Field Detection

Actions at each falling edge of the RX line for the sync field as showing below:

1. LIN counter is set to 0 and start counting on the falling RX edge. (LINCTL.ZERONE = 1)
2. RX falling edge interrupt trigger (RXNE):
   - Read capture register LINC0 (falling edge) and LINC1 (rising edge) values
   - Verify bit times
3. RX falling edge interrupt trigger (RXNE):
   - Read capture register LINC0 (falling edge) and LINC1 (rising edge) values
   - Verify bit times
4. RX falling edge interrupt trigger (RXNE):
   - Read capture register LINC0 (falling edge) and LINC1 (rising edge) values
   - Verify bit times
5. RX falling edge interrupt trigger (RXNE):
   - Read capture register LINC0 (falling edge) and LINC1 (rising edge) values
   - Verify bit times
   - Calculate the proper baud rate to set. Software must set the baud rate before the start bit of the PID field after sync field.

On each interrupt occurrence, the capture registers must be read and the bit times need to be validated by the application software. In case of a bit time verification error, the sync field analysis process must be aborted and the application software must switch back to break detection.

In case of errors like a breaking commander communication during sync field detection, a timeout interrupt can be generated by enabling the LIN counter overflow (IMASK.LINOVF = 1). When the counter overflows, the interrupt handler can abort the sync field analysis and switch back to break detection. The time the counter overflow interrupt occurs can be calculated as $t_{Timeout}= 2^{16}$/ UART clock.

Note

The sync field is automatically stored in the RX FIFO and can be misread as the PID. Therefore, flush the RX FIFO after the sync field is received and before the PID is received.

11.2.3.7.1 LIN Responder Transmission Delay

The interrupt RXINT for starting the transmission on the responder line always occurs at the mid-point of the commander STOP bit. Depending on the BUSCLK and baud-rate; there might not be enough response space between the STOP bit of commander and START bit of the responder; and the data transmission can start before end of STOP bit.

To overcome this, a delay of half the STOP bit period can be added as response space time to make sure that there is sufficient delay time between STOP and START bit, before start of responder data transmission.

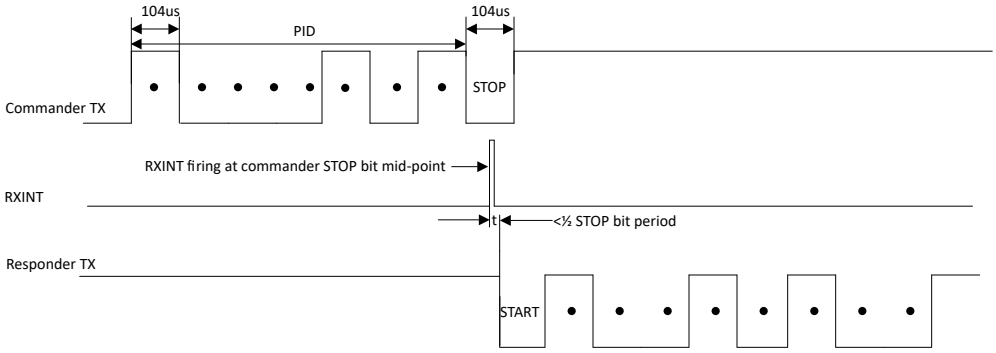


Figure 11-8. LIN Responder Transmission Delay

11.2.3.8 Flow Control

Flow control can be accomplished by hardware and the following sections describe the implementation method.

In UART mode (CTL0.MODE set to 0), hardware flow control between two devices is accomplished by connecting the RTS output to the CTS input on the receiving device, and connecting the RTS output on the receiving device to the CTS input. The RTS output signal is low active, the CTS input expects a low signal on a send request as shown in Figure 11-9.

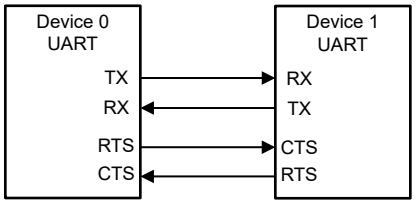


Figure 11-9. Flow Control

The CTS input controls the transmitter, the Device 0 and Device 1 transmitter can only transmit data when their CTS input is asserted low. When RTS flow control is enabled, the RTS output signal indicates the state of the receive FIFO. For example, the CTS of the Device 1 remains asserted low until the preprogrammed RX FIFO level of Device 0 is reached, indicating that the receive FIFO of Device 0 has no space to store additional characters.

The CTSEN and RTSEN bits in the UART.CTL0 register specify the flow control mode as shown in Table 11-3.

Table 11-3. Flow Control Enable

CTSEN	RTSEN	Description
1	1	RTS and CTS flow control enabled
1	0	Only CTS flow control enabled
0	1	Only RTS flow control enabled
0	0	Both RTS and CTS flow control disabled

When RTSEN is set to 1, the value of the CTL0.RTS bit is ignored and the RTS output signal is generated by the hardware trigger levels as described below. When RTSEN bit is cleared, the RTS signal output is controlled by the CTL0.RTS bit for SW control.

RTS flow control:

The RTS flow control logic is linked to the programmable receive FIFO watermark levels, it can be configured using UARTx.IFLS register. When RTS flow control is enabled, the RTS is asserted (low) until the receive FIFO is filled up to the watermark level. When the receive FIFO watermark level is reached, the RTS signal is de-asserted (high), indicating that there is no more room to receive any more data. The transmission of data is

expected to cease after the current character has been transmitted. The RTS signal is reasserted (low) when data has been read out of the receive FIFO so that it is filled to less than the watermark level. If RTS flow control is disabled and the UART is still enabled, then data is received until the receive FIFO is full, or no more data is transmitted to it.

As the RTS signal is de-asserted when the FIFO watermark level is reached by putting the last received character into the FIFO. This means on a back to back transmit another character transfer could already been started by the sender. Therefore, in such cases the watermark level should be set to one level lower to ensure all data can be received and put into the FIFO.

CTS flow control:

If CTS flow control is enabled, then the transmitter checks the CTS signal before transmitting the next byte. If the CTS signal is asserted (low), it transmits the byte otherwise transmission does not occur. The data continues to be transmitted while CTS is asserted (low), and the transmit FIFO is not empty. If the transmit FIFO is empty and the CTS signal is asserted (low) no data is transmitted. If the CTS signal is de-asserted (high) and CTS flow control is enabled, then the current character transmission is completed before stopping. If CTS flow control is disabled and the UART is enabled, then the data continues to be transmitted until the transmit FIFO is empty.

11.2.3.9 Idle-Line Multiprocessor

When IDLELINE is set in the CTL0.MODE register bits, the idle-line multiprocessor format is selected. Blocks of data are separated by an idle time on the transmit or receive lines (Figure 11-10). An idle receive line is detected when ten or more continuous ones (marks) are received after the one or two stop bits of a character. The baud-rate generator is switched off after reception of an idle line until the next start edge is detected. When an idle line is detected, the IDLE bit in UARTx.STAT is set. In Idle-Line mode the UART receiver operates in no parity mode and the UART word length (UARTx.LCRH.WLEN) must be set to 8bit.

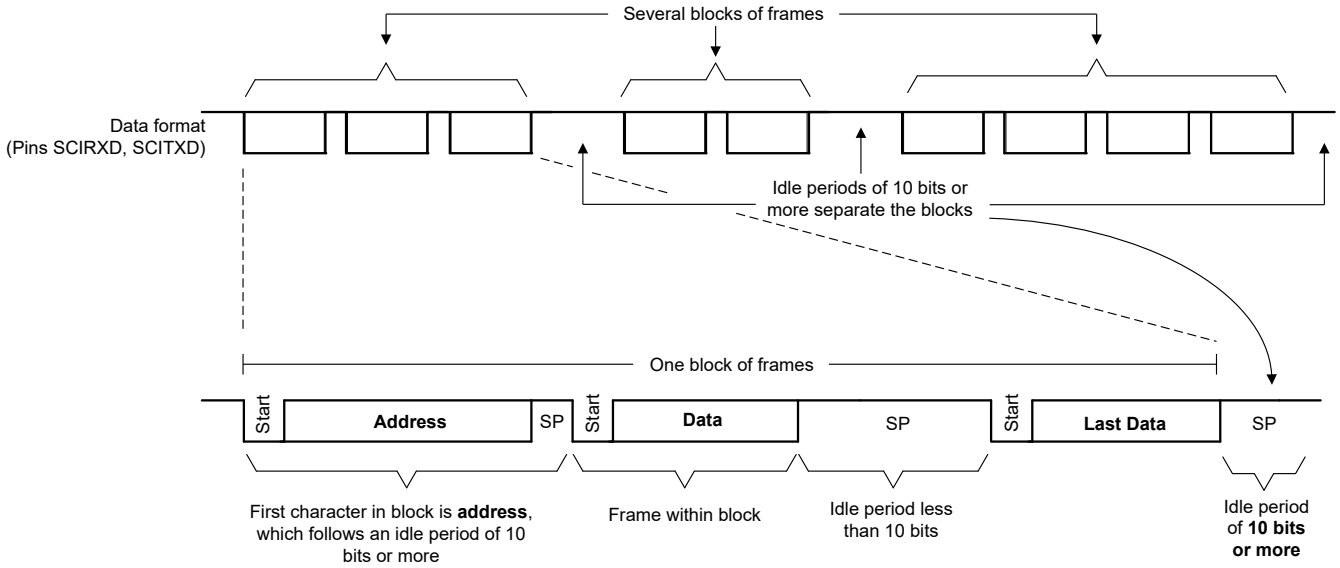


Figure 11-10. Idle-Line Multiprocessor

The first character received after an idle period is an address character. The IDLE bit in UARTx.STAT register is used as an address tag for each block of characters. In idle-line multiprocessor format, this bit is set when a received character is an address.

If an address character is received it is compared against the ADDR register with the AMASK applied. If the received character matches, the address character and all following received characters are transferred into the RXDATA buffer and interrupts/flags are generated until the next address without a match is received. The IDLE bit in UARTx.STAT register is automatically cleared when the address does match; otherwise the IDLE bit is set until the address is matched.

When the SENDIDLE bit in UARTx.LCRH register is set the UART inserts an idle period of 11 bit times on the bus, an ongoing transfer is finished first. The next transfer will be delayed till this idle period has finished. Then the next transfer can start with an address character.

The following procedure sends out an idle frame to indicate an address character followed by associated data:

1. Set SENDIDLE then write the address character to TXDATA. TXDATA must be ready for new data (TXINT interrupt must be 1). This generates an idle period of exactly 11 bits followed by the address character. SENDIDLE is reset automatically when the address character has been transferred (all bits are sent out of shift register).
2. Write desired data characters to TXDATA. TXDATA must be ready for new data (TXINT interrupt must be 1). The data written to TXDATA is transferred to the shift register and transmitted as soon as the shift register is ready for new data.

The idle-line time must not be exceeded between address and data transmission or between data transmissions. Otherwise, the transmitted data is misinterpreted as an address.

BUSY bit in IDLELINE mode:

- TX: BUSY is set during the generation of an IDLELINE signal and while the address and data bytes are sent
- RX: The BUSY signal set during the receive of data and till first 10 idle bits are received.

11.2.3.10 9-Bit UART Mode

9-bit mode is enabled by setting MODE bit to ADDR9BIT in the UARTx.CTL0 register. This feature is useful in a multi-drop configuration of the UART where a single controller connected to multiple peripherals can communicate with a particular peripheral through its address or set of addresses along with a qualifier for an address byte. In 9-bit UART mode, the parity enable/mode bits are ignored and the UART word length (UARTx.LCRH.WLEN) must be set to 8 bit.

Receive Transaction:

In 9-bit mode, a peripheral checks for the address qualifier at the location of the parity bit. If set, the received byte is compared with the preprogrammed address in UARTx.ADDR register:

- If the address matches, a 9-bit mode address match interrupt (ADDR_MATCH) is generated, if enabled and further data get received.
- If the address does not match, the address byte and the subsequent data bytes get dropped. .

The address can be predefined in the UART.ADDR register to match with the received byte. The matching can be extended to a set of addresses using the address mask in the UART.AMASK register. By default, the UART.AMASK is 0xFF, meaning that only the specified address must match

Transmit Transaction:

All the send transactions in 9-bit UART mode are interpreted as:

- Address bytes, if the 9th bit is set
- Data bytes, if the 9th bit is cleared

In 9-bit mode, the 9th bit can be controlled by software. The EPS bit setting of the LCRH register reflects the 9th bit for transmit transactions. To indicate an address byte, the software must set the EPS bit before the byte transmission. For data byte transmissions, the EPS bit must be cleared before the byte transmission. For a complete transmit transaction, the address byte must be transmitted as a single byte transaction with EPS bit set, followed by a data byte burst with EPS bit cleared.

9^{th} bit handling:

Table 11-4. 9th Bit Handling

PEN	SPS	EPS	9^{th} Bit (Transmitted or Verified)
0	X	X	Not transmitted or verified
1	1	0	0 (= Data)

Table 11-4. 9th Bit Handling (continued)

PEN	SPS	EPS	9th Bit (Transmitted or Verified)
1	1	1	1 (= Address)

11.2.3.11 RS485 Support

RS485 is a standard used in serial communications systems. This standard can be used effectively over long distances and in electrically noisy environments. Multiple receivers can be connected to such a network. These characteristics make RS-485 useful in industrial control systems and similar applications.

With the RS485 direction signal an external RS485 PHY can be controlled. The RTS I/O is used in this mode for the direction signal. The signal is set automatically to high once a data transmit is started. It will stay set between bytes if they are sent back to back. If a data receive is ongoing a new transmit should be delayed till this data has been received and the direction signal has been set to transmit.

Data exchange sequence as show in Figure 11-11:

- Wait till an ongoing receive has been finished.
- Activate direction signal to transmit on RTS Pin
- Send data (one or more Bytes)
- Wait till an ongoing receive has been finished.
- Deactivate direction signal to transmit on RTS Pin

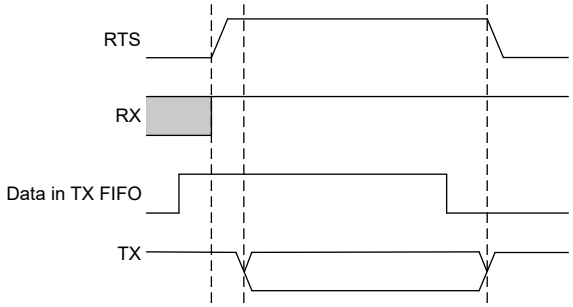


Figure 11-11. RS485 Data Exchange

Two bit fields to the LCRH register define the setup and hold time of the external driver direction control:

- EXTDIR_SETUP bits defines the number of UART clock ticks the signal to control the external driver for the RS485 will be set before the START bit is send. The generated setup time will be between EXDIR_SETUP value and EXDIR_SETUP + one baud rate cycle
- EXTDIR_HOLD bits defines the number of UART clock ticks the signal to control the external driver for the RS485 will be reset after the beginning of the stop bit. (If 2 STOP bits are enabled the beginning of the 2nd STOP bit.)

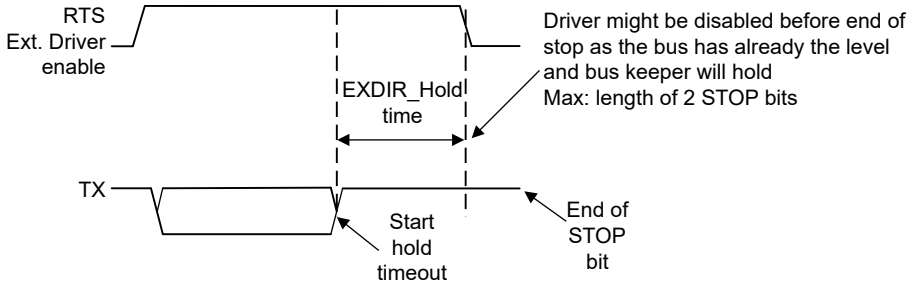


Figure 11-12. RS485 External Control

11.2.3.12 DALI Protocol

DALI stands for Digital Addressable Lighting Interface. It is an International Standard (IEC 62386) lighting control system, providing a single interface for all electronic control gear (light sources) and electronic control devices (lighting controllers).

The UART module supports the low level DALI Protocol sending and receiving the bit streams for forward and backward frames. The timing between any forward and backward frame sequence needs to be handled and checked by software.

Transmitting a forward or backward frame:

When transmitting a forward frame user needs to ensure to write second byte to buffer before first byte has been shifted out. The hardware will then send the two bytes without inserting the stop bits in-between. Otherwise the stop bits will be sent and the data will be handled like a backward frame.

Receiving data:

The UART module in DALI mode will check the 9th bit after the start bit to detect a Forward frame or backward frame. If this bit does have a change of the phase (= no stop bit) a forward frame is detected and:

- Address compare is done in software or hardware depending on MASK
- Address and data are always transferred into the RX Buffer

Otherwise a backward frame is detected and the data is transferred into the RX Buffer without setting the ADDR_MATCH interrupt.

The AMASK register can be used as group assignment used during multicast operation with the MSB to indicate if the device is part of a DALI group. To enable the UART in DALI mode to respond on all ADDRESS the AMASK register needs to be cleared.

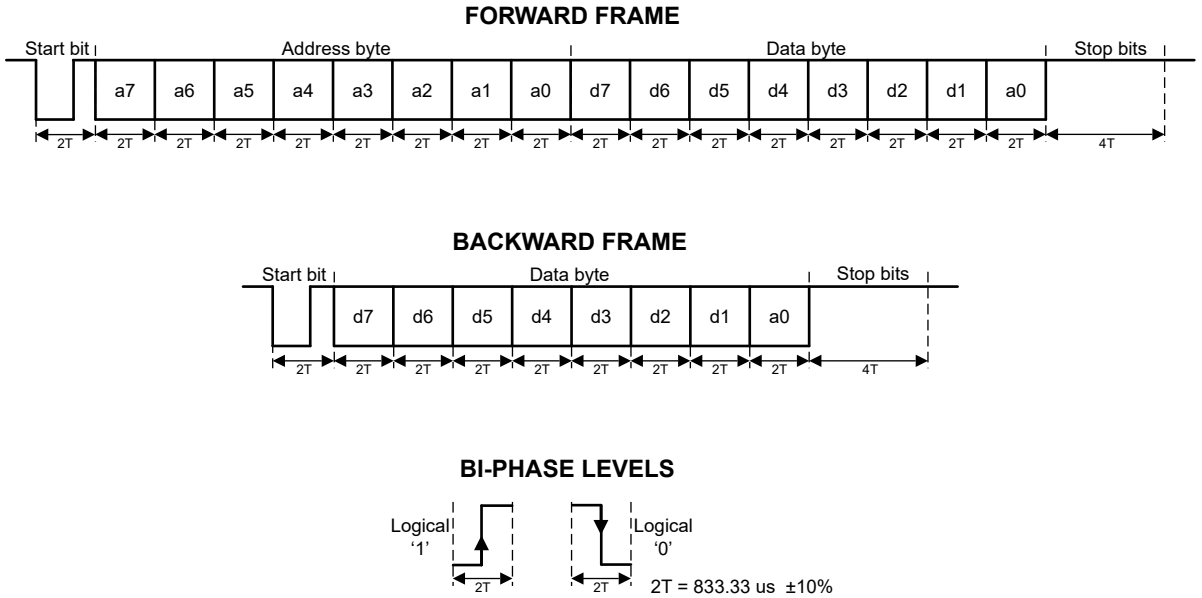


Figure 11-13. DALI Protocol

The limits for the times T shall be: 334 µs < T < 500 µs according to standard IEC 62386-102. DALI mode requires Manchester encoding to be enabled. When using DALI mode (CTL0.MODE set to DALI), the CTL0 and LCRH register must be set to:

- 8-bit word length (WLEN bit)
- No parity / 2 Stop Bit
- Manchester encoding enabled
- Baud-rate configuration set to match 2T = 833.33us
- DALI mode requires the FIFO to be enabled

11.2.3.13 Manchester Encoding and Decoding

UART provides option to receive and transmit Manchester encoded data. The function is enabled by the MENC bit in CTL0 register to generate the IEEE 802.3 compliant waveform. With the invert function in GPIO control module the output signals can be inverted to generate the G. E. Thomas compliant waveform.

The output signal is generated by XORing the data with UART clock signal. The UART clock needs to be double the speed of the baud-rate. So for the data transmit there is an edge at the beginning and the middle of each data bit. For the receive signal the edge in the middle of the bit is detected to decode the RX data.

11.2.3.14 IrDA Encoding and Decoding

When IREN bit in UARTx.IRCTL register is set, the IrDA encoder and decoder are enabled and provide hardware bit shaping for IrDA communication. IrDA en/decoding should only be used with UART mode (UARTx.CTL0.MODE is 0)

IrDA Encoding

The encoder sends a pulse for every zero bit in the transmit bit stream coming from the UART (see Figure 11-14). The pulse duration is defined by IRTXPL bits specifying the number of one-half clock periods of the clock selected by IRTXCLK bit.

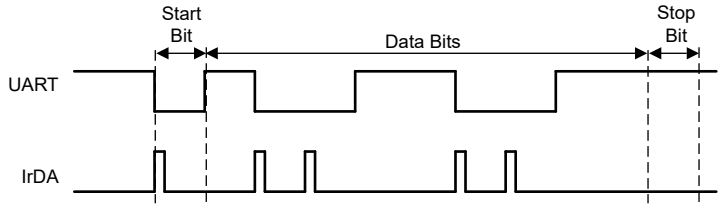


Figure 11-14. IrDA Protocol

(more information will be added in next revision)

IrDA Decoding

The decoder detects high pulses when IRRXPL = 0. Otherwise, it detects low pulses.

A programmable digital filter stage can be enabled by setting UARTx.GFCTL.DGFSEL > 0. When IRCTL.IREN is set, also the digital glitch filter should be set so that only pulses longer than the programmed filter length are passed and shorter pulses are discarded. (See the Glitch Suppression chapter on how to set the filter.)

11.2.3.15 ISO7816 Smart Card Support

The UART offers basic support to allow communication with an ISO7816 smart card. When configuring the MODE bits to smart card (0x4) of the UARTx.CTL0 register, the TXD signal is used as a bit clock, and the RXD signal is used as the half-duplex communication line connected to the smart card. Further smart card signals are not supported by the UART.

The clock rate of the UART clock in ISO7816 mode must be in the range of 1MHz to 5MHz when using ISO7816 mode, the UARTx.LCRH register must be set to:

- 8-bit word length (WLEN bits configured to 0x3)
- Even parity (PEN set and EPS set)
- No stick parity (SPS cleared)

In ISO7816 mode, the UART automatically uses 2 stop bits; therefore the STP2 bit of the LCRH register is ignored.

If a parity error is detected during a transmission, RXD is pulled low during the second stop bit. In this case, the UART aborts the transmission, it flushes the transmit FIFO and discards any data it contains. Additionally it raises a parity error interrupt, allowing the software to detect the problem and initiate retransmission of the affected data, as the UART does not support automatic retransmission in this case. The UART does not

support automatic retransmission on parity errors. If a parity error is detected on transmission, all further transmit operations are aborted and software must handle retransmission of the affected byte or message.

In Smartcard Mode, the receiver in case of parity error will drive the line low and a parity interrupt flag is asserted. The transmitter responds based on the value of this bit.

11.2.3.16 Address Detection

The UARTx.ADDR register is used to set the specific address that should be matched with receiving address byte. This register is used in conjunction with UARTx.AMASK register to form a match for address-byte received. Only bits where the AMASK is set to '1' are considered. So for full address the AMASK register is set to 0xFF. This feature is used in DALI, UART 9-Bit or Idle-Line mode.

Table 11-5. Address Detection

Condition	DALI Mode	Idle Line Mode	9-Bit Mode
Address match	Address and Data is moved to RXDATA	Address and Data is moved to RXDATA	Address and Data is moved to RXDATA
Address mismatch	Address and Data will be dropped	Address and Data will be dropped	Address and Data will be dropped

11.2.3.17 FIFO Operation

The UART has two FIFOs with a depth of 4 entries, one for transmit and one for receive. The FIFOs are accessed through the UART Data (TXDATA/RXDATA) registers. Read operations of the RXDATA register return a 12-bit value consisting of 8 data bits and 4 error flags. Write operations to TXDATA place 8-bit data in the transmit FIFO.

Out of reset, both FIFOs are disabled and act as 1-byte-deep holding registers. The FIFOs are enabled by setting the FEN bit in UARTx.CTL0. FIFO status can be monitored through the UARTx.STAT register and the interrupt events.

Hardware monitors empty, full and overrun conditions

The UARTx.STAT register contains empty and full flags (TXFE, TXFF, RXFE, and RXFF bits), and the CPU_INT.RIS register shows overrun status of the receive FIFO through the OVRERR bit. There is no indicator for a transmit FIFO overrun. A write is just lost, in case it overruns the transmit FIFO. If the FIFOs are disabled, the empty and full flags are set according to the status of the 1-byte-deep holding registers. When receiving more data than the FIFO can capture the oldest data will be overwritten with the received data.

The trigger point at which the FIFOs generate interrupts is controlled through the UARTx.IFLS register. Both FIFOs can be individually configured to trigger interrupts at different levels. Available configurations for transmit FIFO include 3/4, 1/2, 1/4 and empty, for receive FIFO 1/4, 1/2, 3/4 and full. For example, if the 3/4 option is selected for the receive FIFO, the UART generates a receive interrupt after 3 data bytes are received. Out of reset, both FIFOs are configured to trigger an interrupt at the 1/2 mark. The FIFO integrity is indeterminate under the following conditions:

- After a UART Send Break has been initiated (LCRH.BRK = 1)
- If the software disables the UART in the middle of a transmission with data in the FIFO, and then re-enables it.

11.2.3.18 Loopback Operation

The UART can be placed into an internal loopback mode for diagnostic or debug work by setting the LBE bit in the UART.CTL0 register. In loopback mode, the UART operates with the following behavior:

- Data transmitted on the TX output is received on the RX input
- Data transmitted on the TX output is not propagated to the TX IO pin
- Data received on the RX IO pin is ignored

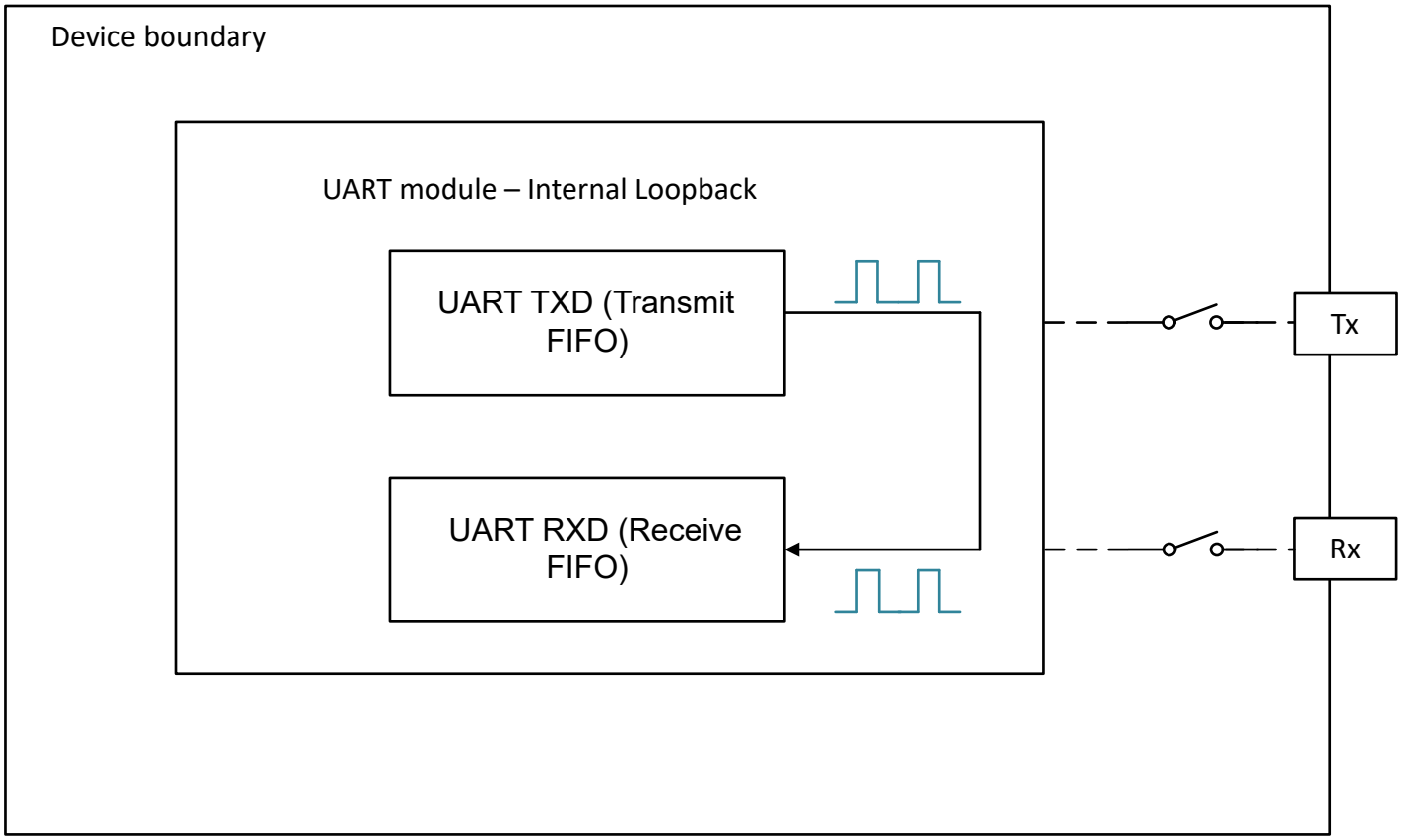


Figure 11-15. UART Loopback mode

11.2.3.19 Glitch Suppression

Digital filter

Digital filter is based on the UART functional clock. The DGFSEL bits in the UARTx.GFCTL register can be programmed to provide glitch suppression on the RX line and assure proper signal values. The glitch suppression value is in terms of functional clocks. All signals are delayed internally when glitch suppression is nonzero. For example, if DGFSEL is set to 0x5, 5 clocks should be added onto the calculation for the expected transaction time. The DGFSEL need to be configured for the glitch suppression pulse width to be shorter than 1/3 of a normal data pulse, to avoid a normal pulse is filtered unexpectedly.

Analog filter

The analog glitch suppression on the RX line is based on the analog glitch filter and it can be selected with the AGFSEL bits in the UARTx.GFCTL register. See data sheet for the select-able glitch filter values. The analog glitch filter is enabled with the AGFEN, if not set the input signals will be passed through to the UART module without filtering.

11.2.4 Low Power Operation

(More information will be added in next revision)

11.2.5 Reset Considerations

Software Reset Considerations

A software reset can be executed with setting the RESETASSERT together with the KEY in the RSTCTL register. An ongoing transfer will be terminated immediately and can leave the software in an undefined state. Therefore, before requesting a reset an ongoing transfer should be terminated.

Hardware Reset Considerations

A hardware reset also initializes the IO configuration. This sets the IOs to a high impedance state and the data lines can float. If this is critical for the application or connected devices on the UART interface external pull up or down resistors might be required.

11.2.6 Initialization

Before the UART is setup or configuration changes, the ENABLE bit should be cleared to avoid unpredictable behavior during the updates or for the first data receive or transmitted afterward.

To enable and initialize the UART, use the following steps:

1. Configure RX and TX pin functions by using the IOMUX registers.
2. Reset the peripheral using UARTx.RSTCTL register
3. Enable the power to UART peripheral using the UARTx.PWREN register
4. Select the UART function clock source and divide options using UART.CLKSEL and UART.CLKDIV registers.
5. Disable the UART by clearing the UART.CTL0.ENABLE bit.
6. Use the baud-rate equation in Section 11.2.3.4 to calculate the UARTx.IBRD and UARTx.FBRD registers.
7. Write the integer portion of the BRD to the UART.IBRD register.
8. Write the fractional portion of the BRD to the UART.FBRD register.
9. Write the desired oversampling and FIFO configuration to the UART.CTL0 register
10. Write the desired serial parameters to the UART.LCRH register.
11. Enable the UART by setting the UART.CTL0.ENABLE bit.

11.2.7 Interrupt and Events Support

The UART module contains three event publishers and no event subscribers. One event publisher (CPU_INT) manages UART interrupt requests (IRQs) to the CPU subsystem through a static event route. The second and third event publishers (DMA_TRIG_RX, DMA_TRIG_TX) are used to setup the trigger signaling for the DMA through DMA event route.

The UART events are summarized in Table 11-6.

Table 11-6. UART Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt	Publisher	UART	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from UART to CPU
DMA trigger	Publisher	UART	DMA	DMA event route	DMA_TRIG_RX registers	Fixed interrupt route from UART to DMA
DMA trigger	Publisher	UART	DMA	DMA event route	DMA_TRIG_TX registers	Fixed interrupt route from UART to DMA

11.2.7.1 CPU Interrupt Event Publisher (CPU_INT)

The UART module provides 18 interrupt sources which can be configured to source a CPU interrupt event. In order of decreasing interrupt priority, the CPU interrupt events from the UART are:

Table 11-7. UART CPU Interrupt Event Conditions (CPU_INT)

IIDX STAT	Name	Description
0x01	RTOUT	UART receive timeout interrupt, This interrupt is asserted when the receive FIFO is not empty, and no further data is received specified time in the UARTx.IFLS.RXTOSEL bits. More information provided below.
0x02	FRMERR	UART framing error interrupt, see Section 11.2.3.6 for more information.
0x03	PARERR	UART parity error interrupt, see Section 11.2.3.6 for more information.

Table 11-7. UART CPU Interrupt Event Conditions (CPU_INT) (continued)

IIDX STAT	Name	Description
0x04	BRKERR	UART break error interrupt, see Section 11.2.3.6 for more information.
0x05	OVRERR	UART receive overrun error interrupt, see Section 11.2.3.6 for more information.
0x06	RXNE	Falling edge on RX interrupt, this interrupt triggers when there is a falling edge on RX line.
0x07	RXPE	Rising edge on RX interrupt, this interrupt triggers when there is a rising edge on RX line.
0x08	LINC0	LIN capture 0 match interrupt, this interrupt triggers when the defined capture 0 value is reached in LIN counter.
0x09	LINC1	LIN capture 1 match interrupt, this interrupt triggers when the defined capture 1 value is reached in LIN counter.
0x0A	LINOVF	LIN counter overflow interrupt, this interrupt triggers when the 16bit LIN counter overflows.
0x0B	RXINT	UART receive interrupt. More information provided below.
0x0C	TXINT	UART transmit interrupt. More information provided below.
0x0D	EOT	UART end of transmission interrupt, it indicates that the last bit of all transmitted data and status has left the serializer and without any further data in the TX FIFO.
0x0E	ADDR_MATCH	Address match interrupt, used in protocols with address to indicate address match happened.
0x0F	CTS	UART clear to send interrupt, indicate the CTS signal status.
0x10	DMA_DONE_RX	This interrupt is set if the RX DMA channel sends the DONE signal.
0x11	DMA_DONE_TX	This interrupt is set if the TX DMA channel sends the DONE signal.

The CPU interrupt event configuration is managed with the CPU_INT event management registers. See Section 6.2.5 for guidance on configuring the Event registers for CPU interrupts.

The receive timeout interrupt is asserted when the receive FIFO is not empty, and no further data is received specified time in the IFLS.RXTOSEL bits. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), by reading the interrupt index from IIDX or when a 1 is written to the corresponding bit in the ICLR register.

The receive interrupt (RXINT, 0x0B) changes state when one of the following events occurs:

- If the FIFOs are enabled and the receive FIFO reaches the programmed trigger level, the RXINT bit is set. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level, by reading the interrupt index from IIDX or by writing a 1 to the RXINT bit in ICLR.
- If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the RXINT bit is set. The receive interrupt is cleared by performing a single read of the receive FIFO, by reading the interrupt index from IIDX or by writing a 1 to the RXINT bit in ICLR.

The transmit interrupt (TXINT, 0x0C) changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO progresses through the programmed trigger level, the TXINT bit is set. The transmit interrupt is based on a transition through level, therefore the FIFO must be written past the programmed trigger level otherwise no further transmit interrupts will be generated. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level, by reading the interrupt index from IIDX or by writing a 1 to the TXINT bit in ICLR.
- If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the TXINT bit is set. It is cleared by performing a single write to the transmit FIFO, by reading the interrupt index from IIDX or by writing a 1 to the TXINT bit in ICLR.

11.2.7.2 DMA Trigger Publisher (DMA_TRIG_RX, DMA_TRIG_TX)

DMA_TRIG_RX and DMA_TRIG_TX registers are used to setup the trigger signaling for the DMA. This can be setup in a flexible way to trigger the DMA for receive or transmit events with the trigger conditions in Table 11-8and Table 11-9.

DMA_TRIG_RX is used for triggering the DMA to do a receive data transfer and DMA_TRIG_TX is used for triggering the DMA to do a transmit data transfer.

Table 11-8. UART DMA Trigger Condition (DMA_TRIG_RX)

IIDX STAT	Name	Description
0x01	RTOUT	UART receive timeout interrupt, This interrupt is asserted when the receive FIFO is not empty, and no further data is received specified time in the UARTx.IFLS.RXTOSEL bits. More information provided below.
0x0B	RXINT	UART receive interrupt. More information provided below.

The receive timeout interrupt is asserted when the receive FIFO is not empty, and no further data is received specified time in the IFLS.RXTOSEL bits. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), by reading the interrupt index from IIDX or when a 1 is written to the corresponding bit in the ICLR register.

The receive interrupt (RXINT, 0x0B) changes state when one of the following events occurs:

- If the FIFOs are enabled and the receive FIFO reaches the programmed trigger level, the RXINT bit is set. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level, by reading the interrupt index from IIDX or by writing a 1 to the RXINT bit in ICLR.
- If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the RXINT bit is set. The receive interrupt is cleared by performing a single read of the receive FIFO, by reading the interrupt index from IIDX or by writing a 1 to the RXINT bit in ICLR.

Table 11-9. UART DMA Trigger Condition (DMA_TRIG_TX)

IIDX STAT	Name	Description
0x0C	TXINT	UART transmit interrupt. More information provided below.

The transmit interrupt (TXINT, 0x0C) changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO progresses through the programmed trigger level, the TXINT bit is set. The transmit interrupt is based on a transition through level, therefore the FIFO must be written past the programmed trigger level otherwise no further transmit interrupts will be generated. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level, by reading the interrupt index from IIDX or by writing a 1 to the TXINT bit in ICLR.
- If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the TXINT bit is set. It is cleared by performing a single write to the transmit FIFO, by reading the interrupt index from IIDX or by writing a 1 to the TXINT bit in ICLR.

The DMA trigger event configuration is managed with the DMA_TRIG_RX and DMA_TRIG_TX event management registers. See Section 6.2.5 for guidance on configuring the Event registers and Section 6.1.3.2 for on how DMA trigger event works.

11.2.8 Emulation Modes

The module behavior while the device is in debug mode is controlled by the FREE and SOFT bits in PDBGCTL register.

When the device is in debug mode and set into halt mode below behavior can be configured.

Table 11-10. Debug Mode Peripheral Behavior

PDBGCTL.FREE	PDBGCTL.SOFT	Function
1	**x**	Modules continues operation
0	**0**	Module stops immediately
0	**1**	Module stops after the next transfer has been finished

11.3 UART0 Registers

Table 11-11 lists the memory-mapped registers for the UART0 registers. All register offset addresses not listed in Table 11-11 should be considered as reserved locations and the register contents should not be modified.

Table 11-11. UART0 Registers

Offset	Acronym	Register Name	Section
800h	PWREN	Power enable	Go
804h	RSTCTL	Reset Control	Go
808h	CLKCFG	Peripheral Clock Configuration Register	Go
814h	STAT	Status Register	Go
1000h	CLKDIV	Clock Divider	Go
1008h	CLKSEL	Clock Select for Ultra Low Power peripherals	Go
1018h	PDBGCTL	Peripheral Debug Control	Go
1020h	IIDX	Interrupt index	Go
1028h	IMASK	Interrupt mask	Go
1030h	RIS	Raw interrupt status	Go
1038h	MIS	Masked interrupt status	Go
1040h	ISET	Interrupt set	Go
1048h	ICLR	Interrupt clear	Go
10E0h	EVT_MODE	Event Mode	Go
10E4h	INTCTL	Interrupt control register	Go
1100h	CTL0	UART Control Register 0	Go
1104h	LCRH	UART Line Control Register	Go
1108h	STAT	UART Status Register	Go
110Ch	IFLS	UART Interrupt FIFO Level Select Register	Go
1110h	IBRD	UART Integer Baud-Rate Divisor Register	Go
1114h	FBRD	UART Fractional Baud-Rate Divisor Register	Go
1118h	GFCTL	Glitch Filter Control	Go
1120h	TXDATA	UART Transmit Data Register	Go
1124h	RXDATA	UART Receive Data Register	Go
1148h	AMASK	Self Address Mask Register	Go
114Ch	ADDR	Self Address Register	Go

Complex bit access types are encoded to fit into small table cells. Table 11-12 shows the codes that are used for access types in this section.

Table 11-12. UART0 Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

11.3.1 PWREN Register (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 11-16 and described in Table 11-13.

Return to the Summary Table.

Register to control the power state

Figure 11-16. PWREN Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/WK-0h

Table 11-13. PWREN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key 26h = KEY to allow write access to this register
23-1	RESERVED	R	0h	
0	ENABLE	R/WK	0h	Enable the power KEY must be set to 26h to write to this bit. 0h = Disable Power 1h = Enable Power

11.3.2 RSTCTL Register (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 11-17 and described in Table 11-14.

Return to the Summary Table.

Register to control reset assertion and de-assertion

Figure 11-17. RSTCTL Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
R-0h						WK-0h	WK-0h

Table 11-14. RSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	R	0h	
1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Clear reset sticky bit
0	RESETASSERT	WK	0h	Assert reset to the peripheral KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Assert reset

11.3.3 CLKCFG Register (Offset = 808h) [Reset = 00000000h]

CLKCFG is shown in Figure 11-18 and described in Table 11-15.

Return to the Summary Table.

Peripheral Clock Configuration Register

Figure 11-18. CLKCFG Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							BLOCKASYNC
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 11-15. CLKCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key A9h = KEY to allow write access to this register
23-9	RESERVED	R	0h	
8	BLOCKASYNC	R/W	0h	Async Clock Request is blocked from starting SYSOSC or forcing bus clock to 24MHz 0h = Disable 1h = Enable
7-0	RESERVED	R	0h	

11.3.4 STAT Register (Offset = 814h) [Reset = 000X0000h]

STAT is shown in Figure 11-19 and described in Table 11-16.

Return to the Summary Table.

Reset status register

Figure 11-19. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							RESETSTKY
R-0h							R-X
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 11-16. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	X	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

11.3.5 CLKDIV Register (Offset = 1000h) [Reset = 00000000h]

CLKDIV is shown in Figure 11-20 and described in Table 11-17.

Return to the Summary Table.

This register is used to specify module-specific divide ratio of the functional clock

Figure 11-20. CLKDIV Register

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

Table 11-17. CLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	
2-0	RATIO	R/W	0h	Selects divide ratio of module clock 0h = Do not divide clock source 1h = Divide clock source by 2 2h = Divide clock source by 3 3h = Divide clock source by 4 4h = Divide clock source by 5 5h = Divide clock source by 6 6h = Divide clock source by 7 7h = Divide clock source by 8

11.3.6 CLKSEL Register (Offset = 1008h) [Reset = 00000000h]

CLKSEL is shown in Figure 11-21 and described in Table 11-18.

Return to the Summary Table.

Clock source selection for peripherals

Figure 11-21. CLKSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUSCLK_SEL	MFCLK_SEL	LFCLK_SEL	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R-0h

Table 11-18. CLKSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	BUSCLK_SEL	R/W	0h	Selects BUS CLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
2	MFCLK_SEL	R/W	0h	Selects MFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
1	LFCLK_SEL	R/W	0h	Selects LFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
0	RESERVED	R	0h	

11.3.7 PDBGCTL Register (Offset = 1018h) [Reset = 00000000h]

PDBGCTL is shown in Figure 11-22 and described in Table 11-19.

Return to the Summary Table.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 11-22. PDBGCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R-0h						R/W-0h	R/W-0h

Table 11-19. PDBGCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	SOFT	R/W	0h	Soft halt boundary control. This function is only available, if FREE is set to 'STOP' 0h = The peripheral will halt immediately, even if the resultant state will result in corruption if the system is restarted 1h = The peripheral blocks the debug freeze until it has reached a boundary where it can resume without corruption
0	FREE	R/W	0h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

11.3.8 IIDX Register (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 11-23 and described in Table 11-20.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 11-23. IIDX Register

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 11-20. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	UART Module Interrupt Vector Value. This register provides the highest priority interrupt index. A read clears the corresponding interrupt flag in RIS and MIS registers. 15h-1Fh = Reserved 00h = No interrupt pending 01h = UART receive time-out interrupt; Interrupt Flag: RT; Interrupt Priority: Highest 02h = UART framing error interrupt; Interrupt Flag: FE 03h = UART parity error interrupt; Interrupt Flag: PE 04h = UART break error interrupt; Interrupt Flag: BE 05h = UART receive overrun error interrupt; Interrupt Flag: OE 06h = Negative edge on UARTxRXD interrupt; Interrupt Flag: RXNE 07h = Positive edge on UARTxRXD interrupt; Interrupt Flag: RXPE 08h = LIN capture 0 / match interrupt; Interrupt Flag: LINC0 09h = LIN capture 1 interrupt; Interrupt Flag: LINC1 0Ah = LIN hardware counter overflow interrupt; Interrupt Flag: LINOVF 0Bh = UART receive interrupt; Interrupt Flag: RX 0Ch = UART transmit interrupt; Interrupt Flag: TX 0Dh = UART end of transmission interrupt (transmit serializer empty); Interrupt Flag: EOT 0Eh = 9-bit mode address match interrupt; Interrupt Flag: MODE_9B Fh = UART Clear to Send Modem interrupt; Interrupt Flag: CTS 10h = DMA DONE on RX 11h = DMA DONE on TX 12h = Noise Error Event

11.3.9 IMASK Register (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 11-24 and described in Table 11-21.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 11-24. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						NERR	DMA_DONE_TX
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DMA_DONE_RX	CTS	ADDR_MATCH	EOT	TXINT	RXINT	LINOVF	LINC1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
LINC0	RXPE	RXNE	OVRERR	BRKERR	PARERR	FRMERR	RTOUT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-21. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	
17	NERR	R/W	0h	Noise Error on triple voting. Asserted when the 3 samples of majority voting are not equal 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
16	DMA_DONE_TX	R/W	0h	Enable DMA Done on TX Event Channel Interrupt 0h = Interrupt disabled 1h = Set Interrupt Mask
15	DMA_DONE_RX	R/W	0h	Enable DMA Done on RX Event Channel Interrupt 0h = Interrupt disabled 1h = Set Interrupt Mask
14	CTS	R/W	0h	Enable UART Clear to Send Modem Interrupt. 0h = Interrupt disabled 1h = Set Interrupt Mask
13	ADDR_MATCH	R/W	0h	Enable Address Match Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
12	EOT	R/W	0h	Enable UART End of Transmission Interrupt Indicates that the last bit of all transmitted data and flags has left the serializer and without any further Data in the TX FIFO or Buffer. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
11	TXINT	R/W	0h	Enable UART Transmit Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
10	RXINT	R/W	0h	Enable UART Receive Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 11-21. IMASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	LINOVF	R/W	0h	Enable LIN Hardware Counter Overflow Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
8	LINC1	R/W	0h	Enable LIN Capture 1 Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
7	LINC0	R/W	0h	Enable LIN Capture 0 / Match Interrupt . 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
6	RXPE	R/W	0h	Enable Positive Edge on UARTxRXD Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
5	RXNE	R/W	0h	Enable Negative Edge on UARTxRXD Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
4	OVRERR	R/W	0h	Enable UART Receive Overrun Error Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3	BRKERR	R/W	0h	Enable UART Break Error Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	PARERR	R/W	0h	Enable UART Parity Error Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	FRMERR	R/W	0h	Enable UART Framing Error Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	RTOUT	R/W	0h	Enable UARTOUT Receive Time-Out Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

11.3.10 RIS Register (Offset = 1030h) [Reset = 000XXXXXh]

RIS is shown in Figure 11-25 and described in Table 11-22.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 11-25. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						NERR	DMA_DONE_TX
R-0h						R/W-X	R/W-X
15	14	13	12	11	10	9	8
DMA_DONE_RX	CTS	ADDR_MATCH	EOT	TXINT	RXINT	LINOVF	LINC1
R/W-X	R/W-X	R/W-X	R/W-X	R/W-0h	R/W-X	R/W-X	R/W-X
7	6	5	4	3	2	1	0
LINC0	RXPE	RXNE	OVRERR	BRKERR	PARERR	FRMERR	RTOUT
R/W-X	R/W-X	R/W-X	R/W-X	R/W-X	R/W-X	R/W-X	R/W-X

Table 11-22. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	
17	NERR	R/W	X	Noise Error on triple voting. Asserted when the 3 samples of majority voting are not equal 0h = Interrupt did not occur 1h = Interrupt occurred
16	DMA_DONE_TX	R/W	X	DMA Done on TX Event Channel Interrupt 0h = Interrupt disabled 1h = Interrupt occurred
15	DMA_DONE_RX	R/W	X	DMA Done on RX Event Channel Interrupt 0h = Interrupt disabled 1h = Interrupt occurred
14	CTS	R/W	X	UART Clear to Send Modem Interrupt. 0h = Interrupt disabled 1h = Interrupt occurred
13	ADDR_MATCH	R/W	X	Address Match Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
12	EOT	R/W	X	UART End of Transmission Interrupt Indicates that the last bit of all transmitted data and flags has left the serializer and without any further Data in the TX FIFO or Buffer. 0h = Interrupt did not occur 1h = Interrupt occurred
11	TXINT	R/W	0h	UART Transmit Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
10	RXINT	R/W	X	UART Receive Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred

Table 11-22. RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	LINOVF	R/W	X	LIN Hardware Counter Overflow Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
8	LINC1	R/W	X	LIN Capture 1 Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
7	LINC0	R/W	X	LIN Capture 0 / Match Interrupt . 0h = Interrupt did not occur 1h = Interrupt occurred
6	RXPE	R/W	X	Positive Edge on UARTxRXD Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
5	RXNE	R/W	X	Negative Edge on UARTxRXD Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
4	OVRERR	R/W	X	UART Receive Overrun Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
3	BRKERR	R/W	X	UART Break Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
2	PARERR	R/W	X	UART Parity Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
1	FRMERR	R/W	X	UART Framing Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
0	RTOUT	R/W	X	UARTOUT Receive Time-Out Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred

11.3.11 MIS Register (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 11-26 and described in Table 11-23.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 11-26. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						NERR	DMA_DONE_TX
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DMA_DONE_RX	CTS	ADDR_MATCH	EOT	TXINT	RXINT	LINOVF	LINC1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
LINC0	RXPE	RXNE	OVRERR	BRKERR	PARERR	FRMERR	RTOUT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-23. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	
17	NERR	R/W	0h	Noise Error on triple voting. Asserted when the 3 samples of majority voting are not equal 0h = Interrupt did not occur 1h = Interrupt occurred
16	DMA_DONE_TX	R/W	0h	Masked DMA Done on TX Event Channel Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
15	DMA_DONE_RX	R/W	0h	Masked DMA Done on RX Event Channel Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
14	CTS	R/W	0h	Masked UART Clear to Send Modem Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
13	ADDR_MATCH	R/W	0h	Masked Address Match Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
12	EOT	R/W	0h	UART End of Transmission Interrupt Indicates that the last bit of all transmitted data and flags has left the serializer and without any further Data in the TX FIFO or Buffer. 0h = Interrupt did not occur 1h = Interrupt occurred
11	TXINT	R/W	0h	Masked UART Transmit Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
10	RXINT	R/W	0h	Masked UART Receive Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred

Table 11-23. MIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	LINOVF	R/W	0h	Masked LIN Hardware Counter Overflow Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
8	LINC1	R/W	0h	Masked LIN Capture 1 Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
7	LINC0	R/W	0h	Masked LIN Capture 0 / Match Interrupt . 0h = Interrupt did not occur 1h = Interrupt occurred
6	RXPE	R/W	0h	Masked Positive Edge on UARTxRXD Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
5	RXNE	R/W	0h	Masked Negative Edge on UARTxRXD Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
4	OVRERR	R/W	0h	Masked UART Receive Overrun Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
3	BRKERR	R/W	0h	Masked UART Break Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
2	PARERR	R/W	0h	Masked UART Parity Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
1	FRMERR	R/W	0h	Masked UART Framing Error Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
0	RTOUT	R/W	0h	Masked UARTOUT Receive Time-Out Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred

11.3.12 ISET Register (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 11-27 and described in Table 11-24.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 11-27. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						NERR	DMA_DONE_TX
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DMA_DONE_RX	CTS	ADDR_MATCH	EOT	TXINT	RXINT	LINOVF	LINC1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
LINC0	RXPE	RXNE	OVRERR	BRKERR	PARERR	FRMERR	RTOUT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-24. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	
17	NERR	R/W	0h	Noise Error on triple voting. Asserted when the 3 samples of majority voting are not equal 0h = Writing this has no effect 1h = Set the interrupt
16	DMA_DONE_TX	R/W	0h	Set DMA Done on TX Event Channel Interrupt 0h = Interrupt disabled 1h = Set Interrupt
15	DMA_DONE_RX	R/W	0h	Set DMA Done on RX Event Channel Interrupt 0h = Interrupt disabled 1h = Set Interrupt
14	CTS	R/W	0h	Set UART Clear to Send Modem Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
13	ADDR_MATCH	R/W	0h	Set Address Match Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
12	EOT	R/W	0h	Set UART End of Transmission Interrupt Indicates that the last bit of all transmitted data and flags has left the serializer and without any further Data in the TX FIFO or Buffer. 0h = Writing 0 has no effect 1h = Set Interrupt
11	TXINT	R/W	0h	Set UART Transmit Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
10	RXINT	R/W	0h	Set UART Receive Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt

Table 11-24. ISET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	LINOVF	R/W	0h	Set LIN Hardware Counter Overflow Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
8	LINC1	R/W	0h	Set LIN Capture 1 Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
7	LINC0	R/W	0h	Set LIN Capture 0 / Match Interrupt . 0h = Writing 0 has no effect 1h = Set Interrupt
6	RXPE	R/W	0h	Set Positive Edge on UARTxRXD Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
5	RXNE	R/W	0h	Set Negative Edge on UARTxRXD Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
4	OVRERR	R/W	0h	Set UART Receive Overrun Error Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
3	BRKERR	R/W	0h	Set UART Break Error Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
2	PARERR	R/W	0h	Set UART Parity Error Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
1	FRMERR	R/W	0h	Set UART Framing Error Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt
0	RTOUT	R/W	0h	Set UARTOUT Receive Time-Out Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt

11.3.13 ICLR Register (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 11-28 and described in Table 11-25.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 11-28. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						NERR	DMA_DONE_TX
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
DMA_DONE_RX	CTS	ADDR_MATCH	EOT	TXINT	RXINT	LINOVF	LINC1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
LINC0	RXPE	RXNE	OVRERR	BRKERR	PARERR	FRMERR	RTOUT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-25. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	
17	NERR	R/W	0h	Noise Error on triple voting. Asserted when the 3 samples of majority voting are not equal 0h = Writing 0 has no effect 1h = Clear Interrupt
16	DMA_DONE_TX	R/W	0h	Clear DMA Done on TX Event Channel Interrupt 0h = Interrupt disabled 1h = Clear Interrupt
15	DMA_DONE_RX	R/W	0h	Clear DMA Done on RX Event Channel Interrupt 0h = Interrupt disabled 1h = Clear Interrupt
14	CTS	R/W	0h	Clear UART Clear to Send Modem Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
13	ADDR_MATCH	R/W	0h	Clear Address Match Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
12	EOT	R/W	0h	Clear UART End of Transmission Interrupt Indicates that the last bit of all transmitted data and flags has left the serializer and without any further Data in the TX FIFO or Buffer. 0h = Writing 0 has no effect 1h = Clear Interrupt
11	TXINT	R/W	0h	Clear UART Transmit Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
10	RXINT	R/W	0h	Clear UART Receive Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt

Table 11-25. ICLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	LINOVF	R/W	0h	Clear LIN Hardware Counter Overflow Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
8	LINC1	R/W	0h	Clear LIN Capture 1 Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
7	LINC0	R/W	0h	Clear LIN Capture 0 / Match Interrupt . 0h = Writing 0 has no effect 1h = Clear Interrupt
6	RXPE	R/W	0h	Clear Positive Edge on UARTxRXD Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
5	RXNE	R/W	0h	Clear Negative Edge on UARTxRXD Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
4	OVRERR	R/W	0h	Clear UART Receive Overrun Error Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
3	BRKERR	R/W	0h	Clear UART Break Error Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
2	PARERR	R/W	0h	Clear UART Parity Error Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
1	FRMERR	R/W	0h	Clear UART Framing Error Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt
0	RTOUT	R/W	0h	Clear UARTOUT Receive Time-Out Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt

11.3.14 EVT_MODE Register (Offset = 10E0h) [Reset = 000000XXh]

EVT_MODE is shown in Figure 11-29 and described in Table 11-26.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 11-29. EVT_MODE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		INT2_CFG		INT1_CFG		INT0_CFG	
R-0h		R-X		R-X		R-X	

Table 11-26. EVT_MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5-4	INT2_CFG	R	X	Event line mode select for event corresponding to none.INT_EVENT2 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
3-2	INT1_CFG	R	X	Event line mode select for event corresponding to none.INT_EVENT1 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
1-0	INT0_CFG	R	X	Event line mode select for event corresponding to none.INT_EVENT0 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

11.3.15 INTCTL Register (Offset = 10E4h) [Reset = 00000000h]

INTCTL is shown in Figure 11-30 and described in Table 11-27.

Return to the Summary Table.

Interrupt control register

Figure 11-30. INTCTL Register

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

Table 11-27. INTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	INTEVAL	W	0h	Writing a 1 to this field re-evaluates the interrupt sources. 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS.

11.3.16 CTL0 Register (Offset = 1100h) [Reset = 00000038h]

CTL0 is shown in Figure 11-31 and described in Table 11-28.

Return to the Summary Table.

UART Control Register
The CTL0 register is the control register. All the bits are cleared on reset except for the Transmit Enable (TXE) and Receive Enable (RXE) bits, which are set. To enable the UART module, the UARTEN bit must be set. If software requires a configuration change in the module, the UARTEN bit must be cleared before the configuration changes are written. If the UART is disabled during a transmit or receive operation, the current transaction is completed prior to the UART stopping. NOTE: The CTL0 register should not be changed while the UART is enabled or else the results are unpredictable. The following sequence is recommended for making changes to the CTL0 register.
1. Disable the UART.
2. Wait for the end of transmission or reception of the current character.
3. Flush the transmit FIFO by clearing bit FEN in the UART control register CTL0.
4. Reprogram the control register.
5. Enable the UART.

Figure 11-31. CTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				MSBFIRST	MAJVOTE	FEN	HSE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
HSE	CTSEN	RTSEN	RTS	RESERVED	MODE		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h		
7	6	5	4	3	2	1	0
MENC	TXD_OUT	TXD_OUT_EN	TXE	RXE	LBE	RESERVED	ENABLE
R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-1h	R/W-0h	R-0h	R/W-0h

Table 11-28. CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	
19	MSBFIRST	R/W	0h	Most Significant Bit First This bit has effect both on the way protocol byte is transmitted and received. Notes: User needs to match the protocol to the correct value of this bit to send MSb or LSb first. The hardware engine will send the byte entirely based on this bit. 0h = Least significant bit is sent first in the protocol packet 1h = Most significant bit is sent first in the protocol packet
18	MAJVOTE	R/W	0h	Majority Vote Enable When Majority Voting is enabled, the three center bits are used to determine received sample value. In case of error (all 3 bits are not the same), noise error is detected and bits RIS.NERR and register RXDATA.NERR are set. Oversampling of 16 : bits 7, 8, 9 are used Oversampling of 8 : bits 3, 4, 5 are used Disabled : Single sample value (center value) used 0h = Majority voting is disabled 1h = Majority voting is enabled

Table 11-28. CTL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	FEN	R/W	0h	UART Enable FIFOs 0h = The FIFOs are disabled (Character mode). The FIFOs become 1-byte-deep holding registers. 1h = The transmit and receive FIFO buffers are enabled (FIFO mode).
16-15	HSE	R/W	0h	High-Speed Bit Oversampling Enable **NOTE:** The bit oversampling influences the UART baud-rate configuration. The state of this bit has no effect on clock generation in ISO7816 smart card mode (the SMART bit is set). 0h = 16x oversampling. 1h = 8x oversampling. 2h = 3x oversampling. IrDA, Manchester and DALI not supported when 3x oversampling is enabled.
14	CTSEN	R/W	0h	Enable Clear To Send 0h = CTS hardware flow control is disabled. 1h = CTS hardware flow control is enabled. Data is only transmitted when the UARTxCTS signal is asserted.
13	RTSEN	R/W	0h	Enable hardware controlled Request to Send 0h = RTS hardware flow control is disabled. 1h = RTS hardware flow control is enabled. Data is only requested (by asserting UARTxRTS) when the receive FIFO has available entries.
12	RTS	R/W	0h	Request to Send If RTSEN is set the RTS output signals is controlled by the hardware logic using the FIFO fill level or TXDATA buffer. If RTSEN is cleared the RTS output is controlled by the RTS bit. The bit is the complement of the UART request to send, RTS modem status output. 0h = Signal not RTS 1h = Signal RTS
11	RESERVED	R	0h	
10-8	MODE	R/W	0h	Set the communication mode and protocol used. (Not defined settings uses the default setting: 0) 0h = Normal operation 1h = RS485 mode: UART needs to be IDLE with receiving data for the in EXTDIR_HOLD set time. EXTDIR_SETUP defines the time the RTS line is set to high before sending. When the buffer is empty the RTS line is set low again. A transmit will be delayed as long the UART is receiving data. 2h = The UART operates in IDLE Line Mode 3h = The UART operates in 9 Bit Address mode 4h = ISO7816 Smart Card Support The application must ensure that it sets 8-bit word length (WLEN set to 3h) and even parity (PEN set to 1, EPS set to 1, SPS set to 0) in UARTLCRH when using ISO7816 mode. The value of the STP2 bit in UARTLCRH is ignored and the number of stop bits is forced to 2. 5h = DALI Mode:
7	MENC	R/W	0h	Manchester Encode enable 0h = Disable Manchester Encoding 1h = Enable Manchester Encoding
6	TXD_OUT	R/W	0h	TXD Pin Control Controls the TXD pin when TXD_OUT_EN = 1 and TXE = 0. 0h = TXD pin is low 1h = TXD pin is high
5	TXD_OUT_EN	R/W	1h	TXD Pin Control Enable. When the transmit section of the UART is disabled (TXE = 0), the TXD pin can be controlled by the TXD_OUT bit. 0h = TXD pin can not be controlled by TXD_OUT 1h = TXD pin can be controlled by TXD_OUT

Table 11-28. CTL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TXE	R/W	1h	UART Transmit Enable If the UART is disabled in the middle of a transmission, it completes the current character before stopping. **NOTE:** To enable transmission, the UARTEN bit must be set. 0h = The transmit section of the UART is disabled. The UARTxTXD pin of the UART can be controlled by the TXD_CTL bit when enabled. 1h = The transmit section of the UART is enabled.
3	RXE	R/W	1h	UART Receive Enable If the UART is disabled in the middle of a receive, it completes the current character before stopping. **NOTE:** To enable reception, the UARTEN bit must be set. 0h = The receive section of the UART is disabled. 1h = The receive section of the UART is enabled.
2	LBE	R/W	0h	UART Loop Back Enable 0h = Normal operation. 1h = The UARTxTX path is fed through the UARTxRX path internally.
1	RESERVED	R	0h	
0	ENABLE	R/W	0h	UART Module Enable. If the UART is disabled in the middle of transmission or reception, it completes the current character before stopping. If the ENABLE bit is not set, all registers can still be accessed and updated. It is recommended to setup and change the UART operation mode with having the ENABLE bit cleared to avoid unpredictable behavior during the setup or update. If disabled the UART module will not send or receive any data and the logic is held in reset state. 0h = Disable Module 1h = Enable module

11.3.17 LCRH Register (Offset = 1104h) [Reset = 00000000h]

LCRH is shown in Figure 11-32 and described in Table 11-29.

Return to the Summary Table.

UART Line Control Register The LCRH register is the line control register. Serial parameters such as data length, parity, and stop bit selection are implemented in this register. When updating the baud-rate divisor (UARTIBRD or UARTIFRD), the LCRH register must also be written. The write strobe for the baud-rate divisor registers is tied to the LCRH register.

Figure 11-32. LCRH Register

31	30	29	28	27	26	25	24
RESERVED						EXTDIR_HOLD	
R-0h						R/W-0h	
23	22	21	20	19	18	17	16
EXTDIR_HOLD				EXTDIR_SETUP			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SENDIDLE	SPS	WLEN		STP2	EPS	PEN	BRK
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-29. LCRH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	
25-21	EXTDIR_HOLD	R/W	0h	Defines the number of UARTclk ticks the signal to control the external driver for the RS485 will be reset after the beginning of the stop bit. (If 2 STOP bits are enabled the beginning of the 2nd STOP bit.) 0h = Smallest value 1Fh = Highest possible value
20-16	EXTDIR_SETUP	R/W	0h	Defines the number of UARTclk ticks the signal to control the external driver for the RS485 will be set before the START bit is send 0h = Smallest value 1Fh = Highest possible value
15-8	RESERVED	R	0h	
7	SENDIDLE	R/W	0h	UART send IDLE pattern. When this bit is set an SENDIDLE period of 11 bit times will be sent on the TX line. The bit is cleared by hardware afterward. 0h = Disable Send Idle Pattern 1h = Enable Send Idle Pattern
6	SPS	R/W	0h	UART Stick Parity Select The Stick Parity Select (SPS) bit is used to set either a permanent '1' or a permanent '0' as parity when transmitting or receiving data. Its purpose is to typically indicate the first byte of a package or to mark an address byte, for example in a multi-drop RS-485 network. When bits PEN, EPS, and SPS of UARTLCRH are set, the parity bit is transmitted and checked as a 0. When bits PEN and SPS are set and EPS is cleared, the parity bit is transmitted and checked as a 1. 0h = Disable Stick Parity 1h = Enable Stick Parity

Table 11-29. LCRH Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	WLEN	R/W	0h	UART Word Length The bits indicate the number of data bits transmitted or received in a frame as follows: 0h = 5 bits (default) 1h = 6 bits 2h = 7 bits 3h = 8 bits
3	STP2	R/W	0h	UART Two Stop Bits Select When in 7816 smart card mode (the SMART bit is set in the UARTCTL register), the number of stop bits is forced to 2. 0h = One stop bit is transmitted at the end of a frame. 1h = Two stop bits are transmitted at the end of a frame. The receive logic checks for two stop bits being received and provide Frame Error if either is invalid.
2	EPS	R/W	0h	UART Even Parity Select This bit has no effect when parity is disabled by the PEN bit. For 9-Bit UART Mode transmissions, this bit controls the address byte and data byte indication (9th bit). 0 = The transferred byte is a data byte 1 = The transferred byte is an address byte 0h = Odd parity is performed, which checks for an odd number of 1s. 1h = Even parity generation and checking is performed during transmission and reception, which checks for an even number of 1s in data and parity bits.
1	PEN	R/W	0h	UART Parity Enable 0h = Parity is disabled and no parity bit is added to the data frame. 1h = Parity checking and generation is enabled.
0	BRK	R/W	0h	UART Send Break 0h = Normal use. 1h = A low level is continually output on the UARTxTXD signal, after completing transmission of the current character. For the proper execution of the break command, software must set this bit for at least two frames (character periods).

11.3.18 STAT Register (Offset = 1108h) [Reset = 00000XXXh]

STAT is shown in Figure 11-33 and described in Table 11-30.

Return to the Summary Table.

UART Status Register

Figure 11-33. STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						IDLE	CTS
R-0h						R-X	R-X
7	6	5	4	3	2	1	0
TXFF	TXFE	RESERVED		RXFF	RXFE	RESERVED	BUSY
R-X	R-X	R-0h		R-0h	R-X	R-0h	R-X

Table 11-30. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9	IDLE	R	X	IDLE mode has been detected in Idleline-Multiprocessor-Mode. The IDLE bit is used as an address tag for each block of characters. In idle-line multiprocessor format, this bit is set when a received character is an address. 0h = IDLE has not been detected before last received character. (In idle-line multiprocessor mode). 1h = IDLE has been detected before last received character. (In idle-line multiprocessor mode).
8	CTS	R	X	Clear To Send 0h = The CTS signal is not asserted (high). 1h = The CTS signal is asserted (low).
7	TXFF	R	X	UART Transmit FIFO Full The meaning of this bit depends on the state of the FEN bit in the CTL0 register. 0h = The transmitter is not full. 1h = If the FIFO is disabled (FEN is 0), the transmit holding register is full. If the FIFO is enabled (FEN is 1), the transmit FIFO is full.
6	TXFE	R	X	UART Transmit FIFO Empty The meaning of this bit depends on the state of the FEN bit in the CTL0 register. 0h = The transmitter has data to transmit. 1h = If the FIFO is disabled (FEN is 0), the transmit holding register is empty. If the FIFO is enabled (FEN is 1), the transmit FIFO is empty.
5-4	RESERVED	R	0h	
3	RXFF	R	0h	UART Receive FIFO Full The meaning of this bit depends on the state of the FEN bit in the CTL0 register. 0h = The receiver can receive data. 1h = If the FIFO is disabled (FEN is 0), the receive holding register is full. If the FIFO is enabled (FEN is 1), the receive FIFO is full.

Table 11-30. STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RXFE	R	X	UART Receive FIFO Empty The meaning of this bit depends on the state of the FEN bit in the CTL0 register. 0h = The receiver is not empty. 1h = If the FIFO is disabled (FEN is 0), the receive holding register is empty. If the FIFO is enabled (FEN is 1), the receive FIFO is empty.
1	RESERVED	R	0h	
0	BUSY	R	X	UART Busy This bit is set as soon as the transmit FIFO or TXDATA register becomes nonempty (regardless of whether UART is enabled) or if a receive data is currently ongoing (after the start edge have been detected until a complete byte, including all stop bits, has been received by the shift register). In IDLE_Line mode the Busy signal also stays set during the idle time generation. 0h = The UART is not busy. 1h = The UART is busy transmitting data. This bit remains set until the complete byte, including all stop bits, has been sent/received from/into the shift register.

11.3.19 IFLS Register (Offset = 110Ch) [Reset = 00000022h]

IFLS is shown in Figure 11-34 and described in Table 11-31.

Return to the Summary Table.

The IFLS register is the interrupt FIFO level select register. You can use this register to define the levels at which the TX, RX and timeout interrupt flags are triggered. The interrupts are generated based on a transition through a level rather than being based on the level. That is, the interrupts are generated when the fill level progresses through the trigger level. For example, if the receive trigger level is set to the half-way mark, the interrupt is triggered when the receive FIFO is filled with two or more characters. Out of reset, the TXIFLSEL and RXIFLSEL bits are configured so that the FIFOs trigger an interrupt at the half-way mark.

Figure 11-34. IFLS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RXTOSEL			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED	RXIFLSEL			RESERVED	TXIFLSEL		
R-0h	R/W-2h			R-0h	R/W-2h		

Table 11-31. IFLS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11-8	RXTOSEL	R/W	0h	UART Receive Interrupt Timeout Select. When receiving no start edge for an additional character within the set bit times a RX interrupt is set even if the FIFO level is not reached. A value of 0 disables this function. 0h = Smallest value Fh = Highest possible value
7	RESERVED	R	0h	
6-4	RXIFLSEL	R/W	2h	UART Receive Interrupt FIFO Level Select The trigger points for the receive interrupt are as follows: Note: In ULP domain the trigger levels are used for: 0: LVL_1_4 4: LVL_FULL For undefined settings the default configuration is used. 0h = RX FIFO >= 1/4 full Note: For ULP Domain 1h = RX FIFO >= 1/4 full 2h = RX FIFO >= 1/2 full (default) 3h = RX FIFO >= 3/4 full 4h = RX FIFO is full Note: For ULP Domain 5h = RX FIFO is full 7h = RX FIFO >= 1 entry available Note: esp. required for DMA Trigger
3	RESERVED	R	0h	

Table 11-31. IFLS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	TXIFLSEL	R/W	2h	UART Transmit Interrupt FIFO Level Select The trigger points for the transmit interrupt are as follows: Note: for undefined settings the default configuration is used. 1h = TX FIFO <= 3/4 empty 2h = TX FIFO <= 1/2 empty (default) 3h = TX FIFO <= 1/4 empty 5h = TX FIFO is empty 7h = TX FIFO >= 1 entry free Note: esp. required for DMA Trigger

11.3.20 IBRD Register (Offset = 1110h) [Reset = 00000000h]

IBRD is shown in Figure 11-35 and described in Table 11-32.

Return to the Summary Table.

When changing the IBRD register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register. See Baud-Rate Generation chapter for configuration details.

Figure 11-35. IBRD Register

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
RESERVED																DIVINT															
R-0h																R/W-0h															

Table 11-32. IBRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	DIVINT	R/W	0h	Integer Baud-Rate Divisor 0h = Smallest value FFFFh = Highest possible value

11.3.21 FBRD Register (Offset = 1114h) [Reset = 00000000h]

FBRD is shown in Figure 11-36 and described in Table 11-33.

Return to the Summary Table.

UART Fractional Baud-Rate Divisor Register The FBRD register is the fractional part of the baud-rate divisor value. All the bits are cleared on reset. When changing the FBRD register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the UARTLCRH register. See Baud-Rate Generation chapter for configuration details.

Figure 11-36. FBRD Register

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

Table 11-33. FBRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5-0	DIVFRAC	R/W	0h	Fractional Baud-Rate Divisor 0h = Smallest value 3Fh = Highest possible value

11.3.22 GFCTL Register (Offset = 1118h) [Reset = 0000000Xh]

GFCTL is shown in Figure 11-37 and described in Table 11-34.

Return to the Summary Table.

This register control the glitch filter on the RX input.

Figure 11-37. GFCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CHAIN	AGFSEL		AGFEN
R-0h				R/W-0h	R/W-0h		R/W-0h
7	6	5	4	3	2	1	0
RESERVED		DGFSEL					
R-0h		R/W-X					

Table 11-34. GFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	CHAIN	R/W	0h	Analog and digital noise filters chaining enable. 0 DISABLE: When 0, chaining is disabled and only digital filter output is available to IP logic for sampling 1 ENABLE: When 1, analog and digital glitch filters are chained and the output of the combination is made available to IP logic for sampling 0h = Disabled 1h = Enabled
10-9	AGFSEL	R/W	0h	Analog Glitch Suppression Pulse Width This field controls the pulse width select for the analog glitch suppression on the RX line. See device data sheet for exact values. 0h = Pulses shorter then 5ns length are filtered. 1h = Pulses shorter then 10ns length are filtered. 2h = Pulses shorter then 25ns length are filtered. 3h = Pulses shorter then 50ns length are filtered.
8	AGFEN	R/W	0h	Analog Glitch Suppression Enable 0h = Analog Glitch Filter disable 1h = Analog Glitch Filter enable
7-6	RESERVED	R	0h	
5-0	DGFSEL	R/W	X	Glitch Suppression Pulse Width This field controls the pulse width select for glitch suppression on the RX line. The value programmed in this field gives the number of cycles of functional clock up to which the glitch has to be suppressed on the RX line. In IRDA mode: The minimum pulse length for receive is given by: t(MIN) = (DGFSEL) / f(IRTXCLK) 0h = Bypass GF 3Fh = Highest Possible Value

11.3.23 TXDATA Register (Offset = 1120h) [Reset = 00000000h]

TXDATA is shown in Figure 11-38 and described in Table 11-35.

Return to the Summary Table.

UART Transmit Data Register. This register is the transmit data register (the interface to the FIFOs). For transmitted data, if the FIFO is enabled, data written to this location is pushed onto the transmit FIFO. If the FIFO is disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). A write to this register initiates a transmission from the UART.

Figure 11-38. TXDATA Register

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
RESERVED																								DATA							
R-0h																								R/W-0h							

Table 11-35. TXDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	DATA	R/W	0h	Data Transmitted or Received Data that is to be transmitted via the UART is written to this field. When read, this field contains the data that was received by the UART. 0h = Smallest value FFh = Highest possible value

11.3.24 RXDATA Register (Offset = 1124h) [Reset = 00000000h]

RXDATA is shown in Figure 11-39 and described in Table 11-36.

Return to the Summary Table.

UART Receive Data Register. This register is the data receive register (the interface to the FIFOs). For received data, if the FIFO is enabled, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO. If the FIFO is disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register.

Figure 11-39. RXDATA Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			NERR	OVRERR	BRKERR	PARERR	FRMERR
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DATA							
R-0h							

Table 11-36. RXDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	
12	NERR	R	0h	Noise Error. Writing to this bit has no effect. The flag is cleared by writing 1 to the NERR bit in the UART EVENT ICLR register. 0h = No noise error occurred 1h = Noise error occurred during majority voting
11	OVRERR	R	0h	UART Receive Overrun Error Writing to this bit has no effect. The flag is cleared by writing 1 to the OVRERR bit in the UART EVENT ICLR register. In case of a receive FIFO overflow, the FIFO contents remain valid because no further data is written when the FIFO is full. Only the contents of the shift register are overwritten. The CPU must read the data to empty the FIFO. 0h = No data has been lost due to a receive overrun. 1h = New data was received but could not be stored, because the previous data was not read (resulting in data loss).
10	BRKERR	R	0h	UART Break Error Writing to this bit has no effect. The flag is cleared by writing 1 to the BRKERR bit in the UART EVENT ICLR register. This error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state) and the next valid start bit is received. 0h = No break condition has occurred 1h = A break condition has been detected, indicating that the receive data input was held low for longer than a full-word transmission time (defined as start, data, parity, and stop bits).
9	PARERR	R	0h	UART Parity Error Writing to this bit has no effect. The flag is cleared by writing 1 to the PARERR bit in the UART EVENT ICLR register. 0h = No parity error has occurred 1h = The parity of the received data character does not match the parity defined by bits 2 and 7 of the UARTLCRH register.

Table 11-36. RXDATA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	FRMERR	R	0h	UART Framing Error Writing to this bit has no effect. The flag is cleared by writing 1 to the FRMERR bit in the UART EVENT ICLR register. This error is associated with the character at the top of the FIFO. 0h = No framing error has occurred 1h = The received character does not have a valid stop bit sequence, which is one or two stop bits depending on the UARTLCRH.STP2 setting (a valid stop bit is 1).
7-0	DATA	R	0h	Received Data. When read, this field contains the data that was received by the UART. 0h = Smallest value FFh = Highest possible value

11.3.25 AMASK Register (Offset = 1148h) [Reset = 000000FFh]

AMASK is shown in Figure 11-40 and described in Table 11-37.

Return to the Summary Table.

Self Address Mask Register The AMASK register is used to enable the address mask for 9-bit or Idle-Line mode. The address bits are masked to create a set of addresses to be matched with the received address byte. Used in DALI, UART 9-Bit or Idle-Line mode.

Figure 11-40. AMASK Register

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
RESERVED																								VALUE							
R-0h																								R/W-FFh							

Table 11-37. AMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	VALUE	R/W	FFh	Self Address Mask for 9-Bit Mode This field contains the address mask that creates a set of addresses that should be matched. A 0 bit in the MSK bit field configures, that the corresponding bit in the ADDR bit field of the UARTxADDR register is don't care. A 1 bit in the MSK bit field configures, that the corresponding bit in the ADDR bit field of the UARTxADDR register must match. 0h = Smallest value FFh = Highest possible value

11.3.26 ADDR Register (Offset = 114Ch) [Reset = 00000000h]

ADDR is shown in Figure 11-41 and described in Table 11-38.

Return to the Summary Table.

Self Address Register The ADDR register is used to write the specific address that should be matched with the receiving byte when the Address Mask (AMASK) is set to FFh. This register is used in conjunction with AMASK to form a match for address-byte received.
Used in DALI, UART 9-Bit or Idle-Line mode.

Figure 11-41. ADDR Register

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
RESERVED																								VALUE							
R-0h																								R/W-0h							

Table 11-38. ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	VALUE	R/W	0h	Self Address for 9-Bit Mode This field contains the address that should be matched when UARTxAMASK is FFh. 0h = Smallest value FFh = Highest possible value

Chapter 12

SPI

The serial peripheral interface (SPI) module provides a standardized serial interface to transfer data between MSPM0 devices and other external devices with SPI interface.

12.1 SPI Overview

The SPI module provides a standardized serial interface to transfer data between MSPM0 devices and other external devices (such as a Sensors, Memory, ADCs, or DACs) using SPI protocols

12.1.1 Purpose of the Peripheral

The SPI module acts as a controller or peripheral interface for synchronous serial communication with peripheral devices and other controllers. The transmit and receive paths are buffered with internal, independent FIFO memories allowing up to 4 entries with 16-bit width. A DMA interface is also provided to allow the data exchange with the transmit and receive FIFOs.

12.1.2 Features

The SPI modules have the following features:

- Configurable as a controller or a peripheral
- Programmable clock bit rate and prescaler
- Separate transmit (TX) and receive (RX) first-in first-out buffers (FIFOs)
- Programmable data frame size from 4-bits to 16-bits (Controller Mode)
- Programmable data frame size from 7-bits to 16-bits (Peripheral Mode)
- Supports PACKEN feature that allows the packing of two 16 bit FIFO entries into a 32-bit value to improve CPU performance. However, not all devices support packing. Please check device specific data sheet.
- Interrupts for transmit and receive FIFOs, overrun and timeout interrupts, and DMA done
- Programmable SPI mode support Motorola SPI, MICROWIRE, or Texas Instruments format
- Single bit parity will be supported in both transmit and receive paths using the CTL1.PTEN and CTL1.PREN bits
- Direct memory access controller interface (DMA):
  - Separate channels for transmit and receive
  - Transfer complete interrupt

12.1.3 Functional Block Diagram

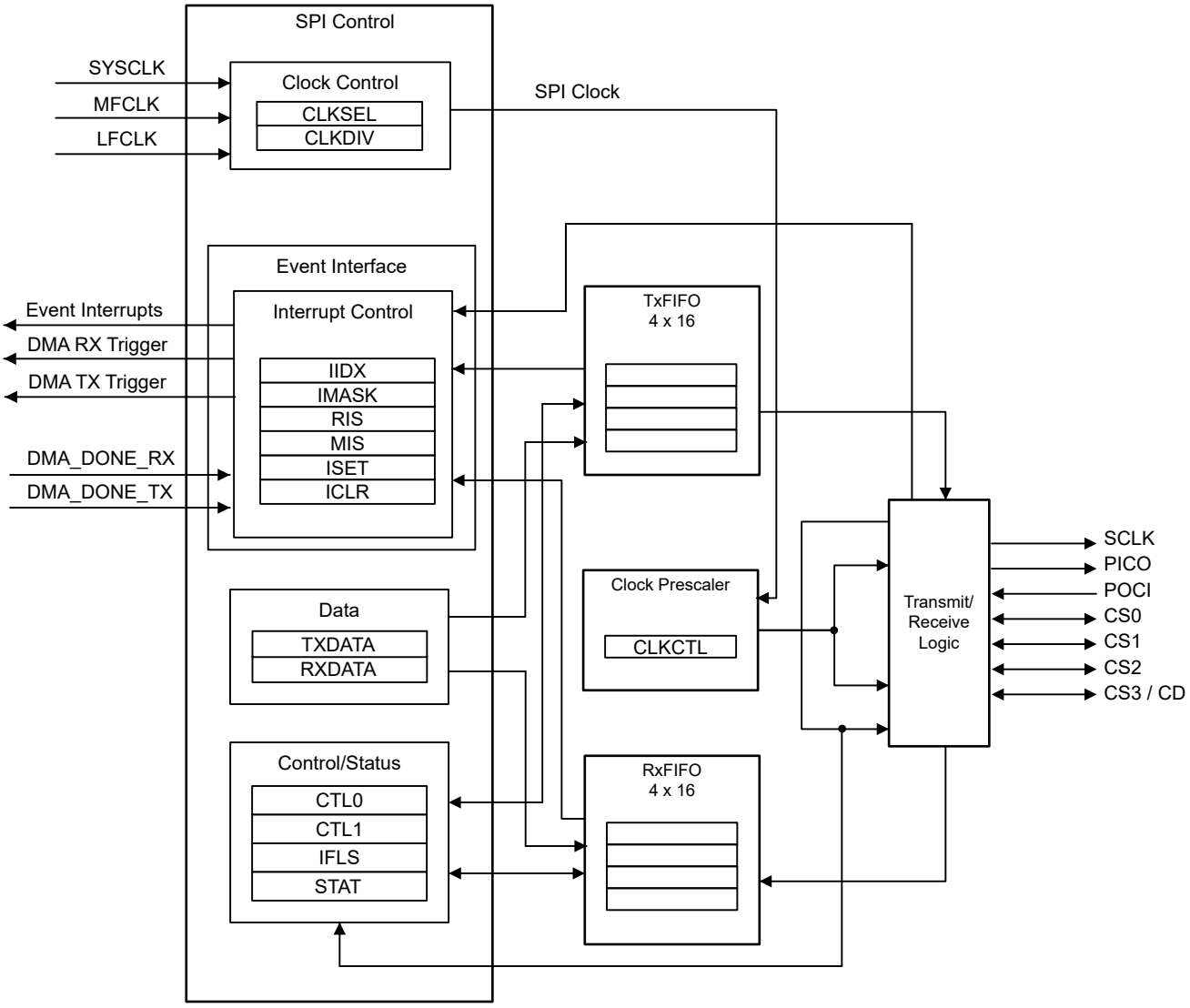


Figure 12-1. SPI Functional Block Diagram

12.1.4 External Connections and Signal Descriptions

Figure 12-2 and Table 12-1 show an overview of the pin functions for different operation modes of the SPI module.

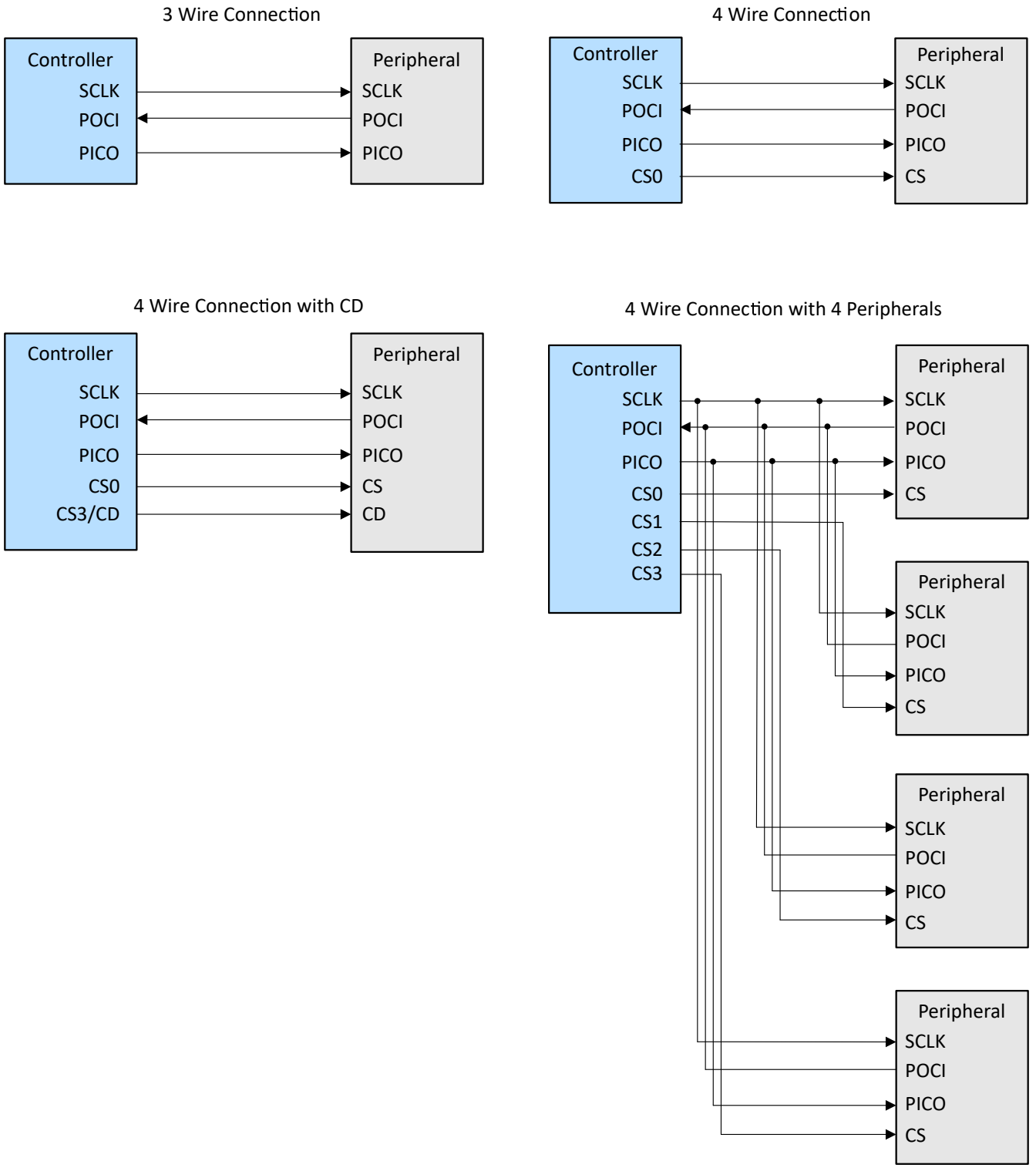


Figure 12-2. External Connections for Different SPI Configurations

Table 12-1. Pin Function Overview

Standard SPI	Feature
SCLK	**SPI clock** Controller mode: SCLK is an output Peripheral mode: SCLK is an input
PICO	**Controller out, peripheral in** Controller mode: PICO is the data output line Peripheral mode: PICO is the data input line
POCI	**Controller in, peripheral out** Controller mode: POCI is the data input line Peripheral mode: POCI is the data output line
CS0	Chip select signal 0, used in 4-pin mode
CS1	Chip select signal 1, used in 4-pin mode
CS2	Chip select signal 2, used in 4-pin mode
CS3/CD	Chip select signal 3 or Command, used in 4-pin mode

12.2 SPI Operation

12.2.1 Clock Control

The SPI internal functional clock is selected and divided from the clock sourced to this module.

- Use SPIx.CLKSEL register to select the source of the SPI functional clock.
  - BUSSCLK: the current bus clock is selected as the source for SPI. The current bus clock depends on power domain. If the SPI instance is in power domain 1 (PD1) please refer to MCLK, if the SPI instance is in power domain 0 (PD0) refer to ULPCLK.
  - MFCLK: MFCLK is selected as the source for SPI, refer to MFCLK.
  - LFCLK: LFCLK is selected as the source for SPI, refer to LFCLK
- Use SPIx.CLKDIV register to select the divide ratio of the SPI function clock. Options are from divide by 1 to 8.

The SPI module must be enabled before being configured for use by using the ENABLE bit in SPIx.PWREN register (see peripheral power enable). When the SPI will be setup or the configuration should be changed the ENABLE bit should be cleared to avoid unpredictable behavior during the updates or for the first data receive or transmitted afterward.

The maximum SPI frequency supported with controller and peripheral mode depends on the device clock option and IO option. Please refer to specific device data sheet spec for more information.

12.2.2 General Architecture

12.2.2.1 Chip Select and Command Handling

12.2.2.1.1 Chip Select Control

One can configure the SPI to be controller mode by setting the CTL1.MS bit to 1, and in peripheral mode by clearing the CTL1.MS bit.

The CTL0.CSSEL bit selects which connected peripheral is addressed by the up to 4 CS signals. The bits are controlled by the SPI module in controller or target/peripheral mode. The selected signal is controlled during the transfers.

The chip select signal needs to be provided by the controller in four-wire mode and the chip select polarity can be inverted by configuring the PINCM.CSx.INV register.

In peripheral mode, the clock is provided by the controller and used by the peripheral to capture the data. The peripheral has the option to operate in 3-wire or 4-wire mode. 4-wire mode only accepts data transfers if the CS is activated.

When the CTL0.CSCLR bit is set, the transmit/receive shift register counter is cleared automatically when the CS goes to the inactive state. When using the Motorola 4-wire or National Microwire mode, follow these constraints:

- The CS disable period must be longer than 2 functional clock cycles before the CS pin is re-asserted
- The CS lead time (CS active to the first bit clock edge) must be at least 2 SPI functional clock cycles

Following these constraints helps the peripheral to synchronize again on the controller in case of a disturbance or glitch on the clock line or during initialization. This bit is relevant only in the peripheral mode.

- CTL0.CSCLR = 0: The transmit/receive bit counter state is retained when the CS signal disables the peripheral.
- CTL0.CSCLR = 1: The transmit/receive bit counter is cleared when the CS signal disables the peripheral.

Note

The CSCLR function requires the CS disable pulse to be longer than 2 SPI function clock cycles to proper detect and clear the bit counter in the SPI. The CS lead time (CS active to the first bit clock edge) also needs to be at least 2 SPI function clock cycles.

12.2.2.1.2 Command Data Control

When using the Motorola frame format, the CDMODE bit can be set to use the CS3/CD line as signal to distinguish between Command and Data information. This is often used for LCD or data storage devices.

- CD level low: command function
- CD level high: data function

The CTL1.CDMODE can be written with a value of 1-14 to specify the number of bytes and the CD line will go low for the given numbers of bytes which are sent by the SPI, starting with the next value to be transmitted. After the number of bytes are transmitted the CD will go high automatically. If a value of 0xF is set the C/D stays low permanently, a value of 0 set the CD line to high immediately after the current character has been transmitted.

This option is only available in controller mode. CTL1.CDENABLE can only be updated when the SPI module is disabled, CTL1.CDMODE can be updated between the different data packages. The counter will be reset with CDENABLE or SPI ENABLE set to disabled. Before setting a new value in CTL1.CDMODE the status of the FIFO should be checked to be empty and the SPI should be in Idle mode.

When writing a new value into CTL1.CDMODE, the internal counter will be reset and the new value will be used for counting. If the counter did count down to 0 and another command package should be sent the CDMODE needs to be set first again, otherwise the next data is send as data with the CD pin signaling data mode.

12.2.2.2 Data Format

The control bit CTL1.MSB defines the direction of the data input and output with most-significant-bit (MSB) or least-significant-bit (LSB) first. If the parity is enabled the parity bits is always received as last bit.

With the control register bits CTL0.DSS the bit length per transfer will be defined between 4-16 bits for Controller mode and 7-16 bits for Peripheral mode.

A transfer will be triggered with writing to the TX buffer register. The data write needs to have at least the number of bits of the transfer. For example, if only a byte is written to the TX buffer but the length of the transfer is > 8 the missing bits will be filled with 0s. On the receive path the received data will be moved to the RXFIFO or RX buffer after the number of bit defined in CTL0.DSS register have been received.

The RX and TX buffer shall be accessed with at least the bits covering one transfer.

- 4-8 bits : byte access (Peripheral mode: 7-8 Bits)
- 9-16 bits : 16 bit access

Clock polarity (CTL0.SPO) is used to control the clock polarity when data is not being transferred and it is only used in the Motorola SPI frame mode.

- 0h = peripheral produces a steady state LOW value on the CLKOUT pin when data is not being transferred.
- 1h = peripheral produces a steady state HIGH value on the CLKOUT pin when data is not being transferred.

Clock phase (CTL0.SPH) bit selects the clock edge that captures data and enables it to change state. It has the most impact on the first bit transmitted by either permitting or not permitting a clock transition before the first data capture edge. Please refer to Motorola SPI frame mode section to check the diagrams.

- 0h = Data is captured on the first clock edge transition.
- 1h = Data is captured on the second clock edge transition.

The SPI can be configured to work in Peripheral mode with CTL1.MS bit = 0. In Peripheral mode the clock is provided by the controller and available for the peripheral on the CLK pins which needs to be configured for input. The Clock Select and divider control bits are not used. The CS input signal is used to select/enable the data receive path of the peripheral in 4 wire mode.

The SPI can be configured to work as Controller with CTL1.MS bit = 1. In Controller mode the clock needs to be generated by selecting the available clock sources with the clock select bits. It also needs to control the CS signal depending on the selected protocol.

When setting the CTL1.PEN bit the last bit will be used as parity to evaluate the integrity of the previous bits. The CTL1.PES bit selects the parity mode as even or odd. When detecting a fault, the interrupt flag RIS.PER is set to mark the data as invalid. Parity checking is a feature to improve the robustness of the communication.

12.2.2.3 Delayed data sampling

In circumstances when the input data arrives at the POCI pin with some delay due to runtime conditions and the following input data sampling stage, the previous data would be sampled at the sampling clock edge. To compensate for such condition, a delayed sampling can be set with the CLKCTL.DSAMPLE bits. The delayed sampling is only available in controller mode. The delay can be adjusted in steps of SPI input clock steps with setting the control register bits CLKCTL.DSAMPLE. The maximum allowed delay should not exceed the length of one data frame.

12.2.2.4 Clock Generation

The SPI includes a programmable bit rate clock divider and prescaler to generate the serial output clock (SCLK).

Bit rates supported are up to, the input clocks divided by 2. The input clock selection depends on the specific device, refer to the device data sheet and Clock Control section.

The SPI functionality can work with any of these selected inputs : SYSCLK, MFCLK and LFCLK

"SPI Clock" is the output after clock division performed according to ratio selected by the CLKDIV register. SPI clock = Selected input clock / (1 + CLKDIV)

SPI Sampling Clock (SCLK) is the output after dividing the SPI Clock by the Prescalar value. SCLK = SPI Clock / ((1 + SCR)*2)

If the factor of two (*2) is set by CLKDIV the input clock must be at least 2 times faster than SPI clock.

12.2.2.5 FIFO Operation

Transmit FIFO

The TX FIFO is a 16-bit-wide, 4-location-deep, first-in first-out memory buffer. The CPU writes data to the FIFO by writing the SPI Data Register TXDATA.DATA, and data is stored in the FIFO until it is read out by the transmission logic.

When configured as a controller or a peripheral, parallel data is written into the TX FIFO before serial conversion and transmission to the attached peripheral or controller, respectively, through the PICO or POCI pin.

In peripheral mode, the SPI transmits data each time the controller initiates a transaction. If the TX FIFO is empty and the controller initiates a transfer, the peripheral transmits the most-recent value written to the transmit

FIFO. User or software is responsible to make valid data available to the FIFO as needed. The SPI can be configured to generate an interrupt or a DMA request when the FIFO is empty. The transmit FIFO has a TXFIFO_UNF interrupt to indicate a FIFO underflow condition.

Receive FIFO

The RX FIFO is a 16-bit-wide, 4-location-deep, first-in-first-out memory buffer. Received data from the serial interface is stored in the buffer until read out by the CPU or DMA, which accesses the read FIFO by reading the SPIx.RXDATA register.

When configured as a controller or peripheral, serial data is received through the POCI or PICO pin. As the access pointer for the FIFO will be updated with each access, the data needs to be accessed by single transfers.

With the FIFO fill level trigger signals located in the STAT register (TFE, TNF, RFE, RNF) the FIFO buffer allows an application to continuously stream serial data in one buffer while the application moves or process the data from the other buffer. If the FIFO is full and new data is written into the FIFO without reading data the RXFIFO overflow event is set. The receive FIFO has a RXFULL interrupt to indicate a FIFO full condition.

12.2.2.6 Loopback mode

The SPI module can be placed into an internal loopback mode for diagnostic or debug work by setting the LBM bit in the CTL1 register. In loopback mode, the data from the TX FIFO can serially transmitted into the RX FIFO. The data from the RX FIFO can be read to check whether correct transmission has occurred or not. The external toggling of the IOs has no effect when the module is set in the internal loopback mode.

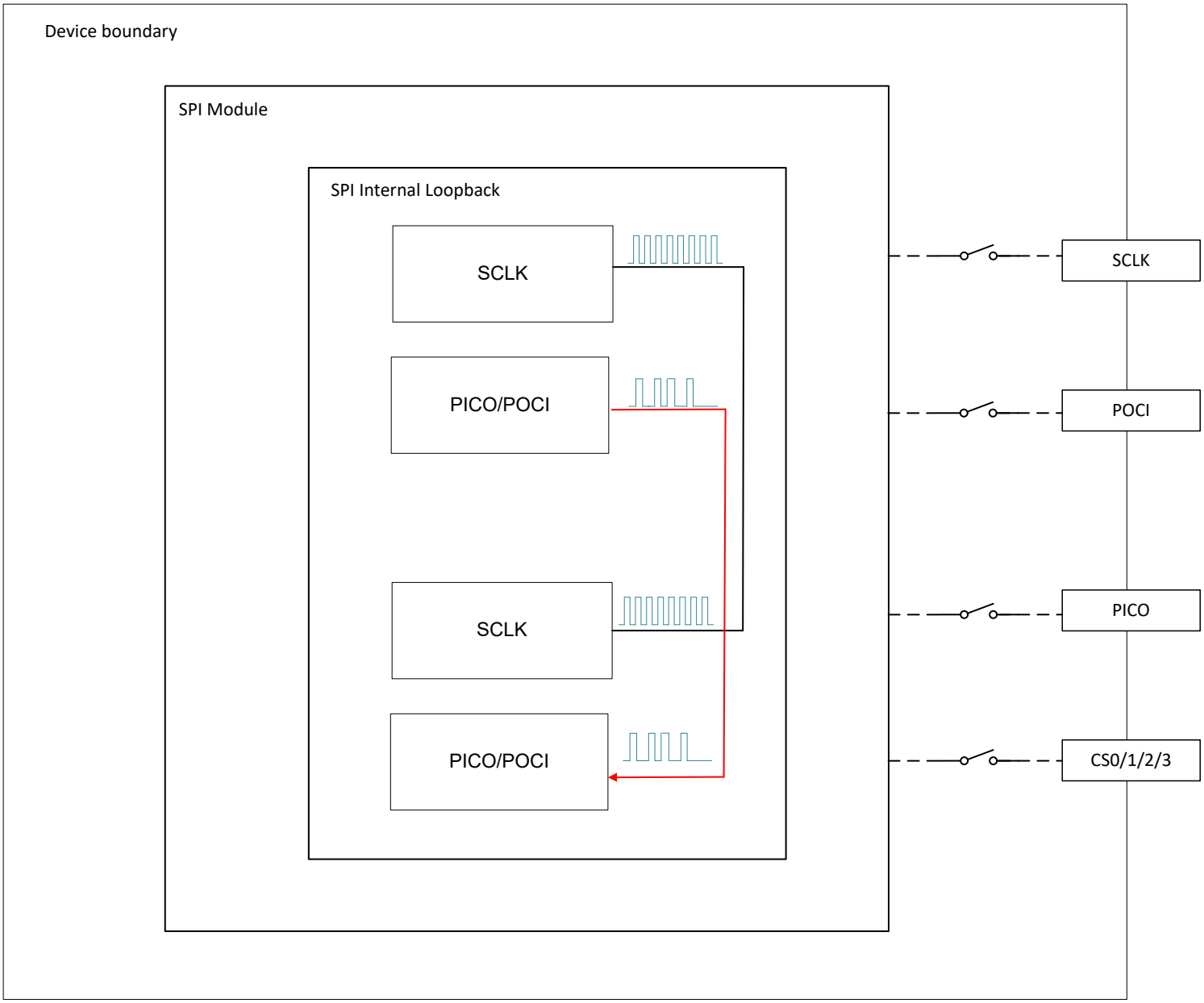


Figure 12-3. Internal Loop Back Mode

12.2.2.7 DMA Operation

The SPI provides an interface to the DMA controller with separate channels for transmit and receive. The DMA operation of the SPI is enabled through the SPI Event and DMA register. When DMA operation is enabled, the SPI asserts a DMA request on the receive or transmit channel when the associated FIFO can transfer data.

For the receive channel a transfer request is asserted whenever the amount of data in the receive FIFO is at or above the FIFO trigger level configured using the RXIFLSEL bit in IFLS register or the receive timeout has triggered, in this case the amount of data received so far will be transmitted.

For the transmit channel a transfer request is asserted whenever the transmit FIFO contains fewer characters than the FIFO trigger level configured using the TXIFLSEL bit in IFLS register. The DMA transfer requests are handled automatically by the DMA controller depending on how the DMA channel is configured.

The DMA transfers can be configured and aligned between the data width of the SPI transfers and the bus accesses width of 8/16 bits to make an efficient usage of the bus. The trigger and transfers are independent for receive and transmit.

See more information about the interrupt and event in Section 12.2.6.2 section.

12.2.2.8 Repeat Transfer mode

With the CTL1.REPEATTX bits the last character transmit will be repeated as defined by the register bits. A value of 0 in CTL1.REPEATTX bits will disable this mode and this function is only available in Controller mode. The transfer will be started with writing a data into the TX Buffer. Then the data will be repeatedly sent with the given value. The behavior is identical as if the data would be written into the TX Buffer that many times as defined by the value here. It can be used to clean a transfer or to pull a certain amount of data from a peripheral.

When REPEATTX is used it needs to be aligned with the data in the FIFO. So the below shown sequence should be used:

- Wait and check till FIFO is empty
- Setup REPEATTX
- Write to TXDATA / FIFO
- Wait till requested data has been received

12.2.2.9 Low Power Mode

The SPI module is located in power domain 1 (PD1) and as such is only active in RUN and SLEEP modes. If the SPI module is enabled by application software, an entry into STOP or STANDBY low-power mode forces the SPI module to be temporarily disabled while the device is in STOP or STANDBY mode.

12.2.3 Protocol Descriptions

The protocol format mode can be selected by using CTL0.FRF register. The supported options include Motorola 3 wire, Motorola 4 wire, Texas Instruments Synchronous and MICROWIRE.

12.2.3.1 Motorola SPI Frame Format

The Motorola SPI interface is a 4-wire interface where the CS signal behaves as a peripheral select. In the 3-wire mode the CS signals is not required and the module behaves as if always selected. The main feature of the Motorola SPI format is that the inactive state and phase of the SCLK signal can be programmed through the SPO and SPH bits in the SPIx.CTL0 control register.

SPO Clock Polarity Bit

If the CTL0.SPO clock polarity control bit is clear, the bit produces a steady-state low value on the SCLK pin when data is not being transferred. If the CTL0.SPO bit is set, the bit places a steady-state high value on the SCLK pin when data is not being transferred.

SPH Phase-Control Bit

The CTL0.SPH phase-control bit selects the clock edge that captures data, and allows it to change state. The state of this bit has the most impact on the first bit transmitted, by either allowing or not allowing a clock transition before the first data capture edge. If the CTL0.SPH phase-control bit is clear, data is captured on the first clock edge transition. If the SPH bit is set, data is captured on the second clock edge transition.

Motorola SPI Frame Format with SPO = 0 and SPH = 0

Figure 12-4shows signal sequences for Motorola SPI format with SPO = 0 and SPH = 0.

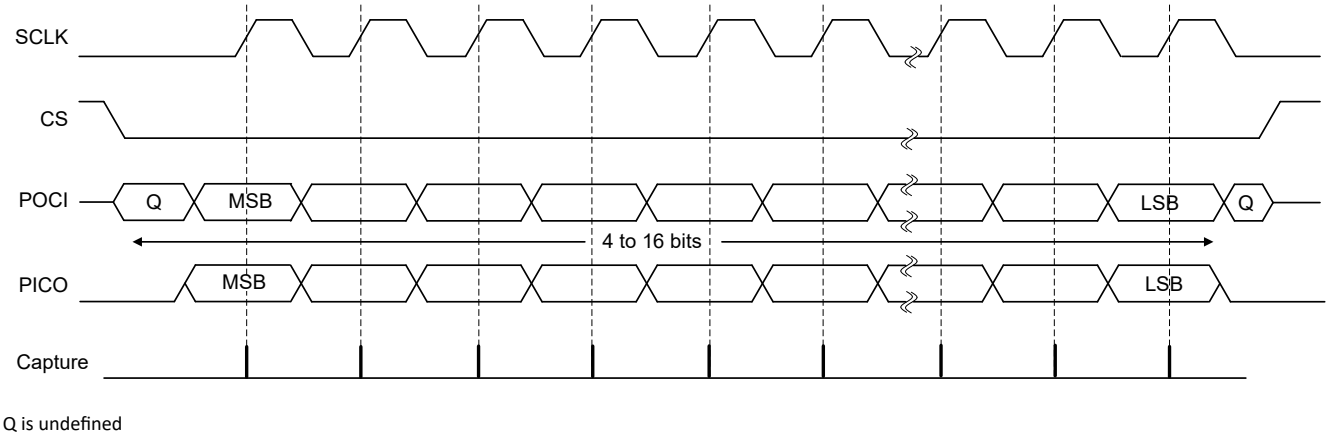


Figure 12-4. Motorola SPI Format With SPO = 0 and SPH = 0

In this configuration, the following occurs during idle periods:

- SCLK is forced low
- CS is forced high
- The transmit data line PICO is forced low
- When the SPI is configured as a controller, it enables the SCLK pin
- When the SPI is configured as a peripheral, it disables the SCLK pin

If the SPI is enabled and valid data is in the TX FIFO, the CS controller signal is driven low at the start of transmission which causes enabling of peripheral data onto the POCI input line of the controller. The controller PICO output pin is enabled.

One-half SCLK period later, valid controller data is transferred to the PICO pin. Once both the controller and peripheral data are set, the SCLK controller clock pin goes high after an additional one-half SCLK period. The data is now captured on the rising edges and propagated on the falling edges of the SCLK signal.

For a single-word transmission after all bits of the data word are transferred, the CS line is returned to its IDLE high state one SCLK period after the last bit is captured. For continuous back-to-back transmissions, the CS signal must pulse high between each data word transfer because the peripheral-select pin freezes the data in its serial peripheral register and does not allow altering of the data if the SPH bit is clear. The controller device must raise the CS pin of the peripheral device between each data transfer to enable the serial peripheral data write. When the continuous transfer completes, the CS pin is returned to its IDLE state one SCLK period after the last bit is captured.

Motorola SPI Frame Format with SPO = 0 and SPH = 1

Figure 12-5 shows the signal sequence for Motorola SPI format with SPO = 0 and SPH = 1.

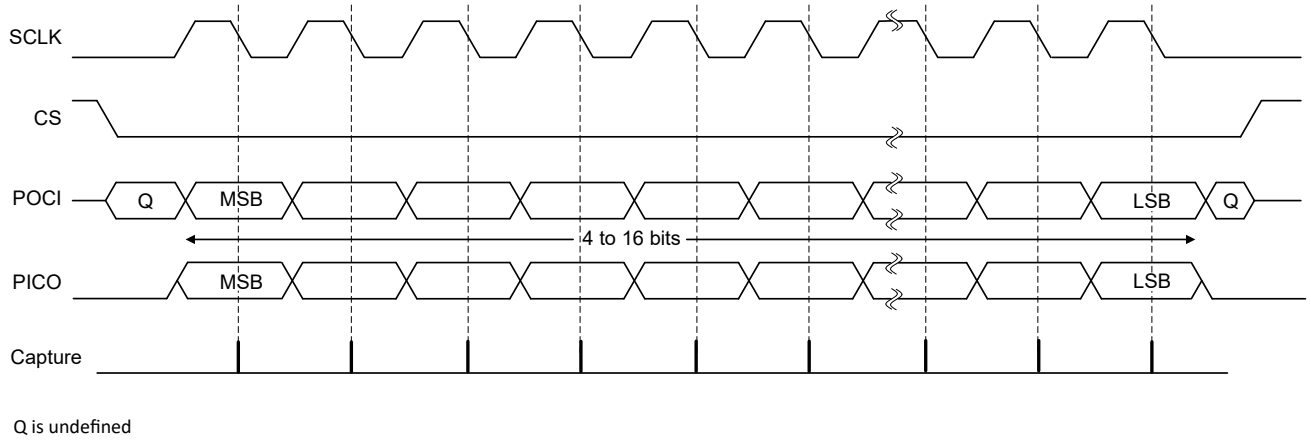


Figure 12-5. Motorola SPI Frame Format With SPO = 0 and SPH = 1

In this configuration, the following occurs during idle periods:

- SCLK is forced low
- CS is forced high
- The transmit data line PICO is forced low
- When the SPI is configured as a controller, it enables the SCLK pin
- When the SPI is configured as a peripheral, it disables the SCLK pin

If the SPI is enabled and valid data is in the TX FIFO, the CS controller signal goes low at the start of transmission. The controller PICO output is enabled. After an additional one-half SCLK period, both controller and peripheral valid data are enabled onto their respective transmission lines. At the same time, SCLK is enabled with a rising-edge transition. Data is then captured on the falling edges and propagated on the rising edges of the SCLK signal.

For a single-word transfer, after all bits are transferred, the CS line is returned to its IDLE high state one SCLK period after the last bit is captured. For continuous back-to-back transfers, the CS pin is held low between successive data words and terminates like a single-word transfer.

Motorola SPI Frame Format with SPO = 1 and SPH = 0

Figure 12-6 shows signal sequences for Motorola SPI format with SPO = 1 and SPH = 0.

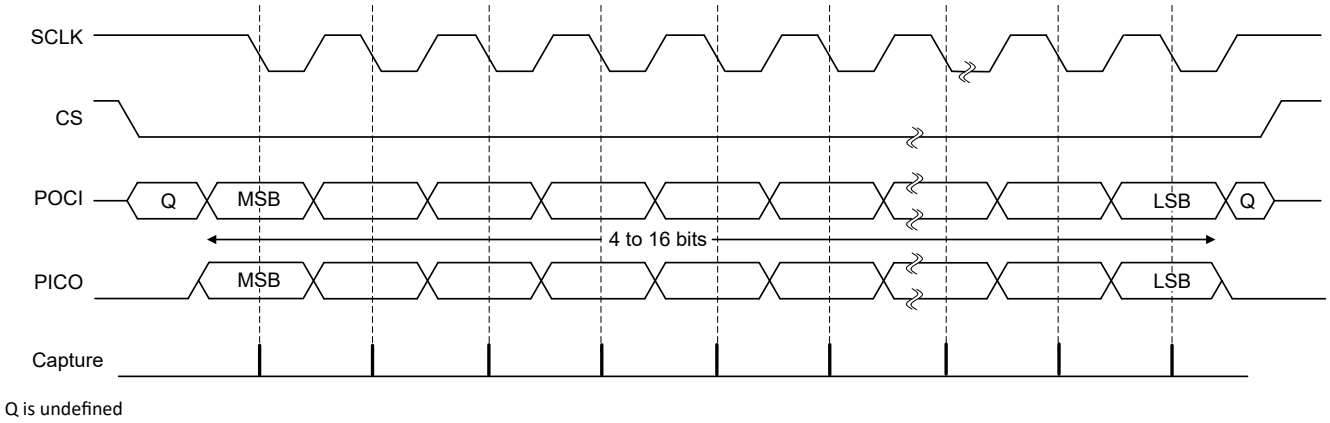


Figure 12-6. Motorola SPI Frame Format With SPO = 1 and SPH = 0

In this configuration, the following occurs during idle periods:

- SCLK is forced high
- CS is forced high

- The transmit data line PICO is arbitrarily forced low
- When the SPI is configured as a controller, it enables the SCLK pin
- When the SPI is configured as a peripheral, it disables the SCLK pin

If the SPI is enabled and valid data is in the TX FIFO, the SPI CS controller signal goes low at the start of transmission and transfers peripheral data onto the POCI line of the controller immediately. The controller PICO output pin is enabled.

One-half SCLK period later, valid controller data is transferred to the PICO line. When both the controller and peripheral data have been set, the SCLK controller clock pin becomes low after one additional half SCLK period. Data is captured on the falling edges and propagated on the rising edges of the SCLK signal.

For a single-word transmission after all bits of the data word are transferred, the CS line is returned to its IDLE high state one SCLK period after the last bit is captured. For continuous back-to-back transmissions, the CS signal must pulse high between each data word transfer as the peripheral-select pin freezes the data in its serial peripheral register and keeps it from being altered if the SPH bit is clear. The controller device must raise the CS pin of the peripheral device between each data transfer to enable the serial peripheral data write. When the continuous transfer completes, the CS pin returns to its IDLE state one SCLK period after the last bit is captured.

Motorola SPI Frame Format with SPO = 1 and SPH = 1

Figure 12-7shows the signal sequence for Motorola SPI format with SPO = 1 and SPH = 1.

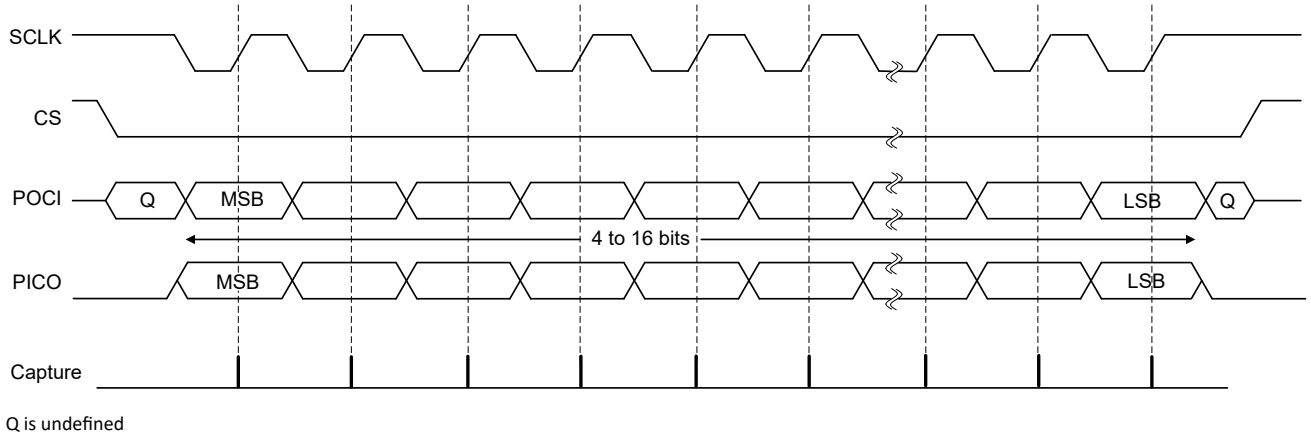


Figure 12-7. Motorola SPI Frame Format With SPO = 1 and SPH = 1

In this configuration, the following occurs during idle periods:

- SCLK is forced high
- CS is forced high
- The transmit data line PICO is arbitrarily forced low
- When the SPI is configured as a controller, it enables the SCLK pin
- When the SPI is configured as a peripheral, it disables the SCLK pin

If the SPI is enabled and valid data is in the TX FIFO, the start of transmission is signified by the CS controller signal going low. The controller PICO output pin is enabled. After an additional one-half SCLK period, both controller and peripheral data are enabled onto their respective transmission lines. At the same time, SCLK is enabled with a falling-edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SCLK signal.

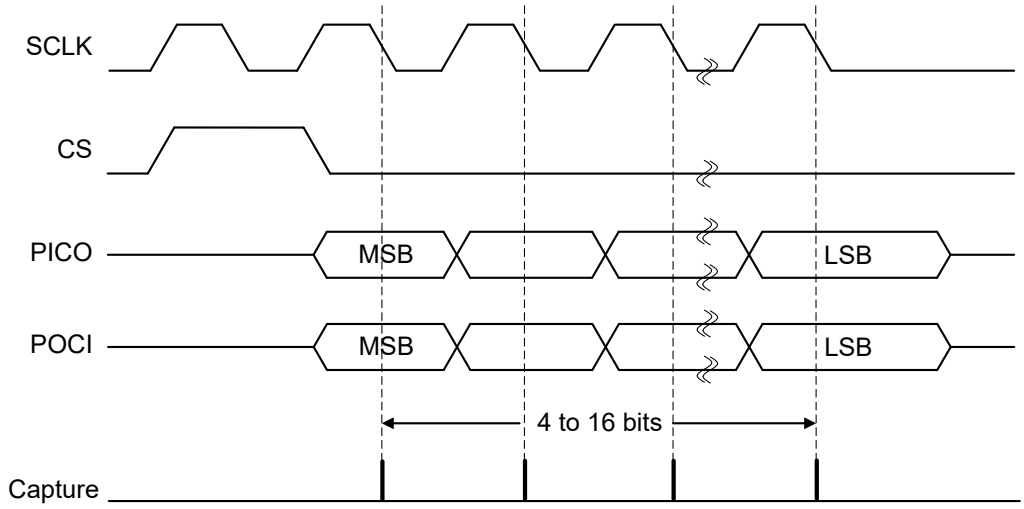
For a single word transmission, after all bits are transferred, the CS line returns to its IDLE high state one SCLK period after the last bit is captured. For continuous back-to-back transmissions, the CS pin remains in its active low state until the final bit of the last word is captured and then returns to its IDLE state. For continuous back-to-back transfers, the CS pin is held low between successive data words and terminates like a single-word transfer.

The serial clock (SCLK) is held inactive while the SPI is idle and SCLK transitions at the programmed frequency only during active transmission or reception of data. The IDLE state of SCLK provides a receive timeout indication that occurs when the RX FIFO still contains data after a timeout period.

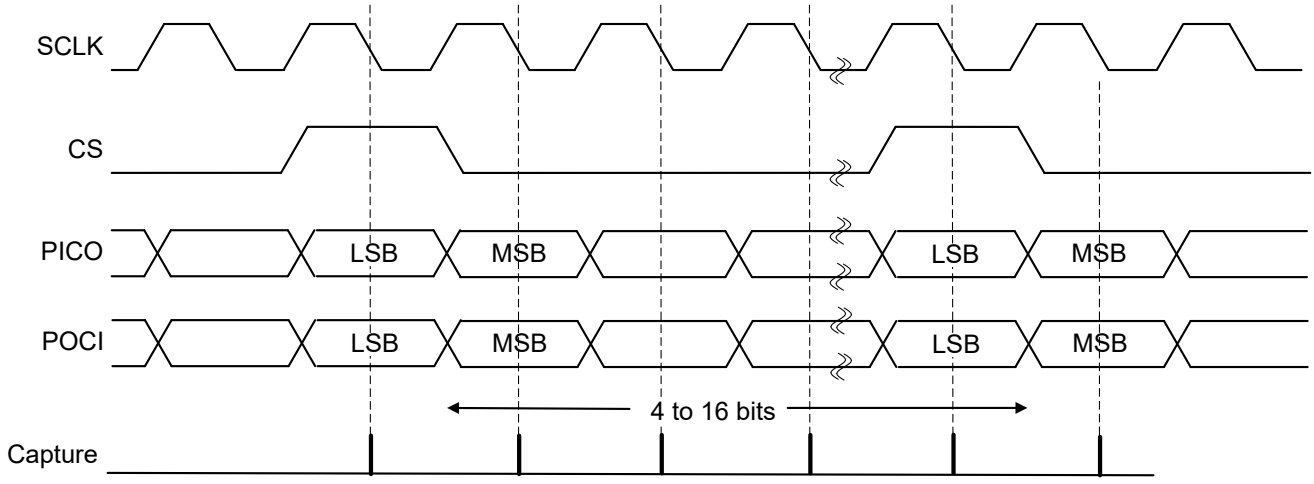
12.2.3.2 Texas Instruments Synchronous Serial Frame Format

The SPI peripheral is compatible with Texas Instruments Synchronous Serial frame format.

Figure 12-8 shows the TI synchronous serial frame format for a single and continuous transmitted frame.



Single Transmission Signal Sequences



Continuous Transmission Signal Sequences

Figure 12-8. TI Synchronous Serial Frame Format

SCLK and CS are forced low and the transmit data line PICO is put in tristate whenever the SPI is idle. When the bottom entry of the TX FIFO contains data, CS is pulsed high for one SCLK period. The transmitted value is also transferred from the TX FIFO to the serial shift register of the transmit logic. On the next rising edge of SCLK, the MSB of the 4- to 16-bit data frame is shifted out on the PICO pin. Likewise, the MSB of the received data is shifted onto the POCI pin by the off-chip serial peripheral device. Both the SPI and the off-chip serial peripheral device then clock each data bit into their serial shifter on each falling edge of SCLK. The received

data is transferred from the serial shifter to the RX FIFO on the first rising edge of SCLK after the least significant bit (LSB) is latched.

The serial clock (SCLK) is held inactive while the SPI is idle and SCLK transitions at the programmed frequency only during active transmission or reception of data. The IDLE state of SCLK provides a receive time-out indication that occurs when the RX FIFO still contains data after a time-out period.

12.2.4 Reset Considerations

Software Reset Considerations

A Software reset can be executed with setting the RESETASSERT together with the KEY in the RSTCTL register. An ongoing transfer will be terminated immediately and can leave the software in an undefined state. Therefore, before requesting a Reset an ongoing Transfer should be terminated.

Hardware Reset Considerations

A hardware reset also initializes the IO configuration. This sets the IOs to a high impedance state and the data lines can float. If this is critical for the application or connected devices on the SPI interface external pull up or down resistors might be required.

12.2.5 Initialization

To enable and initialize the SPI, the following steps are necessary:

1. Configure the IOMUX with the appropriate GPIO pins for which the SPI signals are multiplexed to

Note

Pull-ups can be used to avoid unnecessary toggles on the SPI pins, which can take the peripheral to a wrong state. In addition, if the SCLK signal is programmed to steady state High through the SPO bit in the CTL0 register, then software must also configure the GPIO port pin corresponding to the SCLK signal as a pull-up.

For each of the frame formats, the SPI is configured using the following steps:

1. Ensure that the ENABLE bit in the CTL1 register is clear before making any configuration changes.
2. Select and configure the clock prescale divisor by writing the CLKSEL and CLKDIV register.
3. Select whether the SPI is a controller or peripheral:
   - For controller operations, set the MS bit in the CTL1 register.
   - For peripheral mode, clear the MS bit in the CTL1 register.
4. Configure the clock divisor by writing the CLKCTL register.
5. Please note that a SPI Software reset (See section 15.3.4) is required when switching SPI protocol format.
6. Configure the CTL0 and CTL1 register with based on the desired protocol, data width and other special configurations.
7. Optionally configure DMA
8. Enable the SPI by setting the ENABLE bit in the CTL1 register.

12.2.6 Interrupt and Events Support

The SPI module contains three event publishers and no event subscribers. One event publisher (CPU_INT) manages SPI interrupt requests (IRQs) to the CPU subsystem through a static event route. The second and third event publishers (DMA_TRIG_RX, DMA_TRIG_TX) are used to setup the trigger signaling for the DMA through DMA event route.

The SPI events are summarized in Table 12-2.

Table 12-2. SPI Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt	Publisher	SPI	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from SPI to CPU

Table 12-2. SPI Events (continued)

Event	Type	Source	Destination	Route	Configuration	Functionality
DMA trigger	Publisher	SPI	DMA	DMA event route	DMA_TRIG_RX registers	Fixed interrupt route from SPI RX to DMA
DMA trigger	Publisher	SPI	DMA	DMA event route	DMA_TRIG_TX registers	Fixed interrupt route from SPI TX to DMA

12.2.6.1 CPU Interrupt Event Publisher (CPU_INT)

The SPI module provides 9 interrupt sources that can source a CPU interrupt event. Table 12-3 lists the CPU interrupt events from the SPI in order of decreasing priority.

Table 12-3. SPI CPU_INT Trigger Condition

IIDX STAT	Name	Description
0x01	RXFIFO_OVF	RXFIFO overflow event. This interrupt is set if an RX FIFO overflow has been detected.
0x02	PER	Parity error event. This bit if a Parity error has been detected.
0x03	RTOUT	Peripheral receive timeout event. When in peripheral mode and not receiving data for the CTL1.RXTIMEOUT selected number of functional clock cycles.
0x04	RX	Receive FIFO event. This interrupt is set if the selected receive FIFO level has been reached.
0x05	TX	Transmit FIFO event. This interrupt is set if the selected transmit FIFO level has been reached.
0x06	TXEMPTY	Transmit FIFO empty interrupt. This is set if all data in the transmit FIFO have been shifted out.
0x07	IDLE	SPI Idle. SPI has done finished transfers and changed into IDLE mode. This bit is set when STAT.BUSY bit goes low.
0x08	DMA_DONE1_RX	This interrupt is set if the RX DMA channel sends the DONE signal.
0x09	DMA_DONE1_TX	This interrupt is set if the TX DMA channel sends the DONE signal.

The CPU interrupt event configuration is managed with the CPU_INT event management registers. See Section 6.2.5 for guidance on configuring the Event registers.

12.2.6.2 DMA Trigger Publisher (DMA_TRIG_RX, DMA_TRIG_TX)

DMA_TRIG_RX and DMA_TRIG_TX registers are used to setup the trigger signaling for the DMA. This can be setup in a flexible way to trigger the DMA for receive or transmit events with the trigger conditions in Table 12-4 and Table 12-5.

DMA_TRIG_RX is used for triggering the DMA to do a receive data transfer and DMA_TRIG_TX is used for triggering the DMA to do a transmit data transfer.

Table 12-4. SPI DMA_TRIG_RX DMA Trigger Condition

IIDX STAT	Name	Description
0x03	RTOUT	Peripheral receive timeout event. When in peripheral mode and not receiving data for the CTL1.RXTIMEOUT selected number of functional clock cycles.
0x04	RX	Receive FIFO event. This interrupt is set if the selected receive FIFO level has been reached.

Table 12-5. SPI DMA_TRIG_TX DMA Trigger Condition

IIDX STAT	Name	Description
0x05	TX	Transmit FIFO event. This interrupt is set if the selected transmit FIFO level has been reached.

The DMA trigger event configuration is managed with the DMA_TRIG_RX and DMA_TRIG_TX event management registers. See Section 6.2.5 for guidance on configuring the Event registers and Section 6.1.3.2 for on how DMA trigger event works.

12.2.7 Emulation Modes

The module behavior while the device is in debug mode is controlled by the FREE and SOFT bits in PDBGCTL register.

When the device is in debug mode and set into halt mode below behavior can be configured.

Table 12-6. Debug Mode Peripheral Behavior

PDBGCTL.FREE	PDBGCTL.SOFT	Function
1	**x**	Modules continues operation
0	**0**	Module stops immediately
0	**1**	Module stops after the next transfer has been finished

12.3 SPI Registers

Table 12-7 lists the memory-mapped registers for the SPI registers. All register offset addresses not listed in Table 12-7 should be considered as reserved locations and the register contents should not be modified.

Table 12-7. SPI Registers

Offset	Acronym	Register Name	Group	Section
4h	SCLK	SCLK		Go
8h	PICO	PICO		Go
Ch	POCI	POCI		Go
18h	CS0	SPI Chip Select 0		Go
1Ch	CS1_POCI1	SPI Chip Select 1		Go
20h	CS2_POCI2	SPI Chip Select 2		Go
24h	CS3_CD_POCI3	SPI Chip Select 3		Go
204h	SCLK	FUPDATE version of SCLK		Go
208h	PICO	FUPDATE version of PICO		Go
20Ch	POCI	FUPDATE version of POCI		Go
218h	CS0	FUPDATE version of CS0		Go
21Ch	CS1_POCI1	FUPDATE version of CS1		Go
220h	CS2_POCI2	FUPDATE version of CS2		Go
224h	CS3_CD_POCI3	FUPDATE version of CS3		Go
480h	CPU_CONNECT_0	CPU Connect		Go
504h	DMA_MAP_RX	DMA Map		Go
505h	DMA_TRIG_RX	DMA Trigger		Go
506h	DMA_ENTRY_RX	DMA Entry		Go
508h	DMA_MAP_TX	DMA Map		Go
509h	DMA_TRIG_TX	DMA Trigger		Go
50Ah	DMA_ENTRY_TX	DMA Entry		Go
800h	PWREN	Power enable		Go
804h	RSTCTL	Reset Control		Go
808h	CLKCFG	Peripheral Clock Configuration Register		Go
814h	STAT	Status Register		Go
1000h	CLKDIV	Clock Divider		Go
1004h	CLKSEL	Clock Select for Ultra Low Power peripherals		Go
1018h	PDBGCTL	Peripheral Debug Control		Go
1020h	IIDX	Interrupt Index Register	CPU_INT	Go
1028h	IMASK	Interrupt mask	CPU_INT	Go
1030h	RIS	Raw interrupt status	CPU_INT	Go
1038h	MIS	Masked interrupt status	CPU_INT	Go
1040h	ISET	Interrupt set	CPU_INT	Go
1048h	ICLR	Interrupt clear	CPU_INT	Go
1050h	IIDX	Interrupt Index Register	DMA_TRIG_RX	Go
1058h	IMASK	Interrupt mask	DMA_TRIG_RX	Go
1060h	RIS	Raw interrupt status	DMA_TRIG_RX	Go
1068h	MIS	Masked interrupt status	DMA_TRIG_RX	Go

Table 12-7. SPI Registers (continued)

Offset	Acronym	Register Name	Group	Section
1070h	ISET	Interrupt set	DMA_TRIG_RX	Go
1078h	ICLR	Interrupt clear	DMA_TRIG_RX	Go
1080h	IIDX	Interrupt Index Register	DMA_TRIG_TX	Go
1088h	IMASK	Interrupt mask	DMA_TRIG_TX	Go
1090h	RIS	Raw interrupt status	DMA_TRIG_TX	Go
1098h	MIS	Masked interrupt status	DMA_TRIG_TX	Go
10A0h	ISET	Interrupt set	DMA_TRIG_TX	Go
10A8h	ICLR	Interrupt clear	DMA_TRIG_TX	Go
10E0h	EVT_MODE	Event Mode		Go
10E4h	INTCTL	Interrupt control register		Go
1100h	CTL0	SPI control register 0		Go
1104h	CTL1	SPI control register 1		Go
1108h	CLKCTL	Clock prescaler and divider register.		Go
110Ch	IFLS	Interrupt FIFO Level Select Register		Go
1110h	STAT	Status Register		Go
1130h	RXDATA	RXDATA Register		Go
1140h	TXDATA	TXDATA Register		Go
1E00h	TEST0	Test 0 Register		Go

Complex bit access types are encoded to fit into small table cells. Table 12-8 shows the codes that are used for access types in this section.

Table 12-8. SPI Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

12.3.1 SCLK (Offset = 4h) [Reset = 00000000h]

SCLK is shown in Figure 12-9 and described in Table 12-9.

Return to the Summary Table.

SCLK Signal Controller : Clock Output Peripheral: Clock Input

Figure 12-9. SCLK

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-9. SCLK Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-9. SCLK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.2 PICO (Offset = 8h) [Reset = 00000000h]

PICO is shown in Figure 12-10 and described in Table 12-10.

Return to the Summary Table.

PICO Signal Controller : Data Output Peripheral: Data Input

Figure 12-10. PICO

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-10. PICO Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-10. PICO Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.3 POCI (Offset = Ch) [Reset = 00000000h]

POCI is shown in Figure 12-11 and described in Table 12-11.

Return to the Summary Table.

POCI Signal Controller : Data Input Peripheral: Data Output

Figure 12-11. POCI

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-11. POCI Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-11. POCI Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.4 CS0 (Offset = 18h) [Reset = 00000000h]

CS0 is shown in Figure 12-12 and described in Table 12-12.

Return to the Summary Table.

SPI Chip Select 0: Controller : Output Peripheral: Input

Figure 12-12. CS0

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-12. CS0 Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-12. CS0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.5 CS1_POCI1 (Offset = 1Ch) [Reset = 00000000h]

CS1_POCI1 is shown in Figure 12-13 and described in Table 12-13.

Return to the Summary Table.

SPI Chip Select 1 / POCI1 Controller : Output / Input Peripheral: - / Output

Figure 12-13. CS1_POCI1

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-13. CS1_POCI1 Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-13. CS1_POCI1 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.6 CS2_POCI2 (Offset = 20h) [Reset = 00000000h]

CS2_POCI2 is shown in Figure 12-14 and described in Table 12-14.

Return to the Summary Table.

SPI Chip Select 2 / POCI2 Controller : Output / Input Peripheral: - / Output

Figure 12-14. CS2_POCI2

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-14. CS2_POCI2 Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-14. CS2_POCI2 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.7 CS3_CD_POCI3 (Offset = 24h) [Reset = 00000000h]

CS3_CD_POCI3 is shown in Figure 12-15 and described in Table 12-15.

Return to the Summary Table.

SPI Chip Select 3 / Command Data / POCI3 Controller : Output / Output / Input Peripheral: - / - / Output

Figure 12-15. CS3_CD_POCI3

31	30	29	28	27	26	25	24
RESERVED	GFLT	SLEW	WCOMP	WUEN	INV	HIGHZ1	HIGHZ0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	DRV			HYSTEN	INENA	PIPU	PIPD
R/W-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GSTATE		RESERVED					
R/W-0h		R/W-0h					
7	6	5	4	3	2	1	0
PSTATE		RESERVED					
R/W-0h		R/W-0h					

Table 12-15. CS3_CD_POCI3 Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	
30	GFLT	R/W	0h	Glitch Filter Enable 0h = No internal glitch filter 1h = Use internal glitch filter
29	SLEW	R/W	0h	Reserved Slew Rate Control 0h = No Slew Rate Control 1h = Use Slew Rate Control
28	WCOMP	R/W	0h	Wake up compare value 0h = Match 0 will wake 1h = Match 1 will wake
27	WUEN	R/W	0h	Wake up enable 0h = Wake up not enabled 1h = Wake up enabled
26	INV	R/W	0h	Invert digital input/output relative to peripheral/GPIO 0h = Input and output are noninverted 1h = Input and output are inverted
25	HIGHZ1	R/W	0h	High-Z instead of high output 0h = Pin can be driven high 1h = Pin is tri-stated instead of driven high
24	HIGHZ0	R/W	0h	High-Z instead of low output 0h = Pin can be driven low 1h = Pin is tri-stated instead of driven low
23	RESERVED	R/W	0h	
22-20	DRV	R/W	0h	Drive strength options 0h = Lowest drive strength 1h = Drive strength 2/8 2h = Drive strength 3/8 3h = Drive strength 4/8 4h = Drive strength 5/8 5h = Drive strength 6/8 6h = Drive strength 7/8 7h = Highest drive strength

Table 12-15. CS3_CD_POCI3 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	HYSTEN	R/W	0h	Hysteresis enable 0h = No hysteresis 1h = Hysteresis on
18	INENA	R/W	0h	Input enable 0h = Inputs 0 to connected core 1h = Inputs IO pad value to connected core
17	PIPU	R/W	0h	Pull up enable 0h = No pull up 1h = Pull up
16	PIPD	R/W	0h	Pull down enable 0h = No pull down 1h = Pull down
15-14	GSTATE	R/W	0h	GPIO Channel State 0h = G-Channel is in Unassigned State 1h = G-Channel is in Handover State 2h = G-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = G-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
13-8	RESERVED	R/W	0h	
7-6	PSTATE	R/W	0h	Peripheral-Analog Channel State 0h = P-Channel is in Unassigned State 1h = P-Channel is in Handover State 2h = P-Channel is in Connected State and not Locked (That is F field is allowed to change without going back through Unassigned state) 3h = P-Channel is in Connected State and Locked (That is F field is not allowed to change to a different nonzero value until both G and P channels go to Unassigned)
5-0	RESERVED	R/W	0h	

12.3.8 SCLK (Offset = 204h) [Reset = 00000000h]

SCLK is shown in Figure 12-16 and described in Table 12-16.

Return to the Summary Table.

FUPDATE version of SCLK

Figure 12-16. SCLK

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-16. SCLK Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.9 PICO (Offset = 208h) [Reset = 00000000h]

PICO is shown in Figure 12-17 and described in Table 12-17.

Return to the Summary Table.

FUPDATE version of PICO

Figure 12-17. PICO

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-17. PICO Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.10 POCI (Offset = 20Ch) [Reset = 00000000h]

POCI is shown in Figure 12-18 and described in Table 12-18.

Return to the Summary Table.

FUPDATE version of POCI

Figure 12-18. POCI

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-18. POCI Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.11 CS0 (Offset = 218h) [Reset = 00000000h]

CS0 is shown in Figure 12-19 and described in Table 12-19.

Return to the Summary Table.

FUPDATE version of CS0

Figure 12-19. CS0

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-19. CS0 Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.12 CS1_POCI1 (Offset = 21Ch) [Reset = 00000000h]

CS1_POCI1 is shown in Figure 12-20 and described in Table 12-20.

Return to the Summary Table.

FUPDATE version of CS1_POCI1

Figure 12-20. CS1_POCI1

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-20. CS1_POCI1 Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.13 CS2_POCI2 (Offset = 220h) [Reset = 00000000h]

CS2_POCI2 is shown in Figure 12-21 and described in Table 12-21.

Return to the Summary Table.

FUPDATE version of CS2_POCI2

Figure 12-21. CS2_POCI2

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-21. CS2_POCI2 Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.14 CS3_CD_POCI3 (Offset = 224h) [Reset = 00000000h]

CS3_CD_POCI3 is shown in Figure 12-22 and described in Table 12-22.

Return to the Summary Table.

FUPDATE version of CS3_CD_POCI3

Figure 12-22. CS3_CD_POCI3

31	30	29	28	27	26	25	24
RESERVED				IOADDR			
W-0h				W-0h			
23	22	21	20	19	18	17	16
IOADDR							
W-0h							
15	14	13	12	11	10	9	8
IOADDR							
W-0h							
7	6	5	4	3	2	1	0
IOADDR						LOCK	GSEL
W-0h						W-0h	W-0h

Table 12-22. CS3_CD_POCI3 Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	W	0h	
27-2	IOADDR	W	0h	IO Address. This is the address that corresponds to the SOC address[27:2] of the module IP instance specific IO signal in the “Full Write” subregion of the pinmux subregion.
1	LOCK	W	0h	Sets lock bit 0h = Writing this value has no effect 1h = Set channel lock bit
0	GSEL	W	0h	GPIO channel Select 0: Select the P-Channel for the F update 1: Select the G-Channel for the F update 0h = Select the P-Channel for the F update 1h = Select the G-Channel for the F update

12.3.15 CPU_CONNECT_0 (Offset = 480h) [Reset = 00000000h]

CPU_CONNECT_0 is shown in Figure 12-23 and described in Table 12-23.

Return to the Summary Table.

Directly connect peripheral publisher port to application processor

Figure 12-23. CPU_CONNECT_0

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						CPUSS0_CONN	RESERVED
R/W-0h						R/W-0h	R/W-0h

Table 12-23. CPU_CONNECT_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1	CPUSS0_CONN	R/W	0h	CPUSS0 connect bit. 0h = The CPU is not connected. 1h = The CPU is connected.
0	RESERVED	R/W	0h	

12.3.16 DMA_MAP_RX (Offset = 504h) [Reset = 00h]

DMA_MAP_RX is shown in Figure 12-24 and described in Table 12-24.

Return to the Summary Table.

Trigger port ID in the DMA for this peripheral trigger

Figure 12-24. DMA_MAP_RX

7	6	5	4	3	2	1	0
RESERVED	TRIG_ID						
R-0h	R-0h						

Table 12-24. DMA_MAP_RX Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	
6-0	TRIG_ID	R	0h	The trigger port ID in the DMA for this peripheral trigger 0h = No trigger selected 1h = Trigger 1 selected 7Fh = Trigger 127 selected

12.3.17 DMA_TRIG_RX (Offset = 505h) [Reset = 00h]

DMA_TRIG_RX is shown in Figure 12-25 and described in Table 12-25.

Return to the Summary Table.

Trigger control and status register for this peripheral trigger

Figure 12-25. DMA_TRIG_RX

7	6	5	4	3	2	1	0
RESERVED	THRHLD			TRIGLOST	STATECLR	STATE	
R/W-0h	R-0h			R-0h	R-0/W-0h	R-0h	

Table 12-25. DMA_TRIG_RX Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R/W	0h	
6-4	THRHLD	R	0h	The threshold for the DMA to accept the trigger request. 0h = Lowest threshold possible 6h = Highest threshold possible
3	TRIGLOST	R	0h	Sticky flag that is set whenever a trigger request is received while the trigger port is in TRIGGER_PEND or TRIGGERED. Cleared by writing to STATECLR. 0h = Trigger not lost 1h = Trigger was lost
2	STATECLR	R-0/W	0h	Clear trigger state. Writing 1 to this register clears any pending DMA trigger on this port and transitions the port to Untriggered state. 0h = Writing 0 has no effect 1h = Clear DMA trigger
1-0	STATE	R	0h	Returns the current state of the DMA tx trigger port 0h = Channel was not triggered 1h = Channel trigger is pending 2h = Channel was triggered

12.3.18 DMA_ENTRY_RX (Offset = 506h) [Reset = 0FFFh]

DMA_ENTRY_RX is shown in Figure 12-26 and described in Table 12-26.

Return to the Summary Table.

Descriptor connect to peripheral DMA trigger

Figure 12-26. DMA_ENTRY_RX

15	14	13	12	11	10	9	8
RESERVED				ENTRY_ID			
R/W-				R/W-FFFh			
7	6	5	4	3	2	1	0
ENTRY_ID							
R/W-FFFh							

Table 12-26. DMA_ENTRY_RX Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R/W	0h	
11-0	ENTRY_ID	R/W	FFFh	The ID of the DMA descriptor that this trigger is routed to. This allows to ensure that another DMA channel could not listen or influence the DMA channel responsible for handling the data of this peripheral. 0h = DCLB index i=0-15. This can only be used with dedicated DCLBs. Fh = DCLB index i=0-15. This can only be used with dedicated DCLBs. 10h = DMA entry in RACE memory at index i=16-4094. This can only be used if limitless DMA is enabled in the system. FFEh = DMA entry in RACE memory at index i=16-4094. This can only be used if limitless DMA is enabled in the system. FFFh = Trigger not enabled

12.3.19 DMA_MAP_TX (Offset = 508h) [Reset = 00h]

DMA_MAP_TX is shown in Figure 12-27 and described in Table 12-27.

Return to the Summary Table.

Trigger port ID in the DMA for this peripheral trigger

Figure 12-27. DMA_MAP_TX

7	6	5	4	3	2	1	0
RESERVED	TRIG_ID						
R-0h	R-0h						

Table 12-27. DMA_MAP_TX Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R	0h	
6-0	TRIG_ID	R	0h	The trigger port ID in the DMA for this peripheral trigger 0h = No trigger selected 1h = Trigger 1 selected 7Fh = Trigger 127 selected

12.3.20 DMA_TRIG_TX (Offset = 509h) [Reset = 00h]

DMA_TRIG_TX is shown in Figure 12-28 and described in Table 12-28.

Return to the Summary Table.

Trigger control and status register for this peripheral trigger

Figure 12-28. DMA_TRIG_TX

7	6	5	4	3	2	1	0
RESERVED	THRHLD			TRIGLOST	STATECLR	STATE	
R/W-0h	R-0h			R-0h	R-0/W-0h	R-0h	

Table 12-28. DMA_TRIG_TX Field Descriptions

Bit	Field	Type	Reset	Description
7	RESERVED	R/W	0h	
6-4	THRHLD	R	0h	The threshold for the DMA to accept the trigger request. 0h = Lowest threshold possible 6h = Highest threshold possible
3	TRIGLOST	R	0h	Sticky flag that is set whenever a trigger request is received while the trigger port is in TRIGGER_PEND or TRIGGERED. Cleared by writing to STATECLR. 0h = Trigger not lost 1h = Trigger was lost
2	STATECLR	R-0/W	0h	Clear trigger state. Writing 1 to this register clears any pending DMA trigger on this port and transitions the port to Untriggered state. 0h = Writing 0 has no effect 1h = Clear DMA trigger
1-0	STATE	R	0h	Returns the current state of the DMA tx trigger port 0h = Channel was not triggered 1h = Channel trigger is pending 2h = Channel was triggered

12.3.21 DMA_ENTRY_TX (Offset = 50Ah) [Reset = 0FFFh]

DMA_ENTRY_TX is shown in Figure 12-29 and described in Table 12-29.

Return to the Summary Table.

Descriptor connect to peripheral DMA trigger

Figure 12-29. DMA_ENTRY_TX

15	14	13	12	11	10	9	8
RESERVED				ENTRY_ID			
R/W-				R/W-FFFh			
7	6	5	4	3	2	1	0
ENTRY_ID							
R/W-FFFh							

Table 12-29. DMA_ENTRY_TX Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R/W	0h	
11-0	ENTRY_ID	R/W	FFFh	The ID of the DMA descriptor that this trigger is routed to. This allows to ensure that another DMA channel could not listen or influence the DMA channel responsible for handling the data of this peripheral. 0h = DCLB index i=0-15. This can only be used with dedicated DCLBs. Fh = DCLB index i=0-15. This can only be used with dedicated DCLBs. 10h = DMA entry in RACE memory at index i=16-4094. This can only be used if limitless DMA is enabled in the system. FFEh = DMA entry in RACE memory at index i=16-4094. This can only be used if limitless DMA is enabled in the system. FFFh = Trigger not enabled

12.3.22 PWREN (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 12-30 and described in Table 12-30.

Return to the Summary Table.

Register to control the power state

Figure 12-30. PWREN

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R/W-0h							R/WK-0h

Table 12-30. PWREN Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R/W	0h	
0	ENABLE	R/WK	0h	Enable the power KEY must be set to 26h to write to this bit. 0h = Disable Power 1h = Enable Power

12.3.23 RSTCTL (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 12-31 and described in Table 12-31.

Return to the Summary Table.

Register to control reset assertion and de-assertion

Figure 12-31. RSTCTL

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
W-0h						WK-0h	WK-0h

Table 12-31. RSTCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	W	0h	
1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Clear reset sticky bit
0	RESETASSERT	WK	0h	Assert reset to the peripheral KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Assert reset

12.3.24 CLKCFG (Offset = 808h) [Reset = 00000000h]

CLKCFG is shown in Figure 12-32 and described in Table 12-32.

Return to the Summary Table.

Peripheral Clock Configuration Register

Figure 12-32. CLKCFG

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							BLOCKASYNC
R/W-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R/W-0h							

Table 12-32. CLKCFG Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to Allow State Change -- 0xA9 A9h = key value to allow change field of GPRCM
23-9	RESERVED	R/W	0h	
8	BLOCKASYNC	R/W	0h	Async Clock Request is blocked from starting SYSOSC or forcing bus clock to 32MHz 0h = Not block async clock request 1h = Block async clock request
7-0	RESERVED	R/W	0h	

12.3.25 STAT (Offset = 814h) [Reset = 00000000h]

STAT is shown in Figure 12-33 and described in Table 12-33.

Return to the Summary Table.

peripheral enable and reset status

Figure 12-33. STAT

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

Table 12-33. STAT Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	0h	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

12.3.26 CLKDIV (Offset = 1000h) [Reset = 00000000h]

CLKDIV is shown in Figure 12-34 and described in Table 12-34.

Return to the Summary Table.

This register is used to specify module-specific divide ratio of the functional clock

Figure 12-34. CLKDIV

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

Table 12-34. CLKDIV Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R/W	0h	
2-0	RATIO	R/W	0h	Selects divide ratio of module clock 0h = Do not divide clock source 1h = Divide clock source by 2 2h = Divide clock source by 3 3h = Divide clock source by 4 4h = Divide clock source by 5 5h = Divide clock source by 6 6h = Divide clock source by 7 7h = Divide clock source by 8

12.3.27 CLKSEL (Offset = 1004h) [Reset = 00000000h]

CLKSEL is shown in Figure 12-35 and described in Table 12-35.

Return to the Summary Table.

Clock source selection for peripherals

Figure 12-35. CLKSEL

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				SYSCLK_SEL	MFCLK_SEL	LFCLK_SEL	RESERVED
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 12-35. CLKSEL Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3	SYSCLK_SEL	R/W	0h	Selects SYSCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
2	MFCLK_SEL	R/W	0h	Selects MFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
1	LFCLK_SEL	R/W	0h	Selects LFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
0	RESERVED	R/W	0h	

12.3.28 PDBGCTL (Offset = 1018h) [Reset = 00000003h]

PDBGCTL is shown in Figure 12-36 and described in Table 12-36.

Return to the Summary Table.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 12-36. PDBGCTL

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R/W-						R/W-1h	R/W-1h

Table 12-36. PDBGCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1	SOFT	R/W	1h	Soft halt boundary control. This function is only available, if FREE is set to 'STOP' 0h = The peripheral will halt immediately, even if the resultant state will result in corruption if the system is restarted 1h = The peripheral blocks the debug freeze until it has reached a boundary where it can resume without corruption
0	FREE	R/W	1h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

12.3.29 IIDX (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 12-37 and described in Table 12-37.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 12-37. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 12-37. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No interrupt pending 1h = RX FIFO Overflow Event/interrupt pending 2h = Transmit Parity Event/interrupt pending 3h = SPI receive time-out interrupt 4h = Receive Event/interrupt pending 5h = Transmit Event/interrupt pending 6h = Transmit Buffer Empty Event/interrupt pending 7h = End of Transmit Event/interrupt pending 8h = DMA Done for Receive Event/interrupt pending 9h = DMA Done for Transmit Event/interrupt pending Ah = TX FIFO underflow interrupt Bh = RX FIFO Full Interrupt

12.3.30 IMASK (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 12-38 and described in Table 12-38.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 12-38. IMASK

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED					RXFULL	TXFIFO_UNF	DMA_DONE_TX
R/W-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DMA_DONE_RX	IDLE	TXEMPTY	TX	RX	RTOUT	PER	RXFIFO_OVF
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 12-38. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R/W	0h	
10	RXFULL	R/W	0h	RX FIFO Full Interrupt Mask 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
9	TXFIFO_UNF	R/W	0h	TX FIFO underflow interrupt mask 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
8	DMA_DONE_TX	R/W	0h	DMA Done 1 event for TX event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
7	DMA_DONE_RX	R/W	0h	DMA Done 1 event for RX event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
6	IDLE	R/W	0h	SPI Idle event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
5	TXEMPTY	R/W	0h	Transmit FIFO Empty event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
4	TX	R/W	0h	Transmit FIFO event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3	RX	R/W	0h	Receive FIFO event. This interrupt is set if the selected Receive FIFO level has been reached 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	RTOUT	R/W	0h	Enable SPI Receive Time-Out event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 12-38. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PER	R/W	0h	Parity error event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	RXFIFO_OVF	R/W	0h	RXFIFO overflow event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

12.3.31 RIS (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 12-39 and described in Table 12-39.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 12-39. RIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RXFULL	TXFIFO_UNF	DMA_DONE_TX
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DMA_DONE_RX	IDLE	TXEMPTY	TX	RX	RTOUT	PER	RXFIFO_OVF
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 12-39. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	
10	RXFULL	R	0h	RX FIFO Full Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
9	TXFIFO_UNF	R	0h	TX FIFO Underflow Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
8	DMA_DONE_TX	R	0h	DMA Done 1 event for TX. This interrupt is set if the TX DMA channel sends the DONE signal. This allows the handling of the DMA event inside the mapped peripheral. 0h = Interrupt did not occur 1h = Interrupt occurred
7	DMA_DONE_RX	R	0h	DMA Done 1 event for RX. This interrupt is set if the RX DMA channel sends the DONE signal. This allows the handling of the DMA event inside the mapped peripheral. 0h = Interrupt did not occur 1h = Interrupt occurred
6	IDLE	R	0h	SPI has done finished transfers and changed into IDLE mode. This bit is set when BUSY goes low. 0h = Interrupt did not occur 1h = Interrupt occurred
5	TXEMPTY	R	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been move to the shift register. 0h = Interrupt did not occur 1h = Interrupt occurred
4	TX	R	0h	Transmit FIFO event. This interrupt is set if the selected Transmit FIFO level has been reached. 0h = Interrupt did not occur 1h = Interrupt occurred

Table 12-39. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RX	R	0h	Receive FIFO event. This interrupt is set if the selected Receive FIFO level has been reached 0h = Interrupt did not occur 1h = Interrupt occurred
2	RTOUT	R	0h	SPI Receive Time-Out event. 0h = Interrupt did not occur 1h = Interrupt occurred
1	PER	R	0h	Parity error event: this bit is set if a Parity error has been detected 0h = Interrupt did not occur 1h = Interrupt occurred
0	RXFIFO_OVF	R	0h	RXFIFO overflow event. This interrupt is set if an RX FIFO overflow has been detected. 0h = Interrupt did not occur 1h = Interrupt occurred

12.3.32 MIS (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 12-40 and described in Table 12-40.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 12-40. MIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RXFULL	TXFIFO_UNF	DMA_DONE_TX
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
DMA_DONE_RX	IDLE	TXEMPTY	TX	RX	RTOUT	PER	RXFIFO_OVF
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 12-40. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	
10	RXFULL	R	0h	RX FIFO Full Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
9	TXFIFO_UNF	R	0h	TX FIFO underflow interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
8	DMA_DONE_TX	R	0h	Masked DMA Done 1 event for TX. 0h = Interrupt did not occur 1h = Interrupt occurred
7	DMA_DONE_RX	R	0h	Masked DMA Done 1 event for RX. 0h = Interrupt did not occur 1h = Interrupt occurred
6	IDLE	R	0h	Masked SPI IDLE mode event. 0h = Interrupt did not occur 1h = Interrupt occurred
5	TXEMPTY	R	0h	Masked Transmit FIFO Empty event. 0h = Interrupt did not occur 1h = Interrupt occurred
4	TX	R	0h	Masked Transmit FIFO event. This interrupt is set if the selected Transmit FIFO level has been reached. 0h = Interrupt did not occur 1h = Interrupt occurred
3	RX	R	0h	Masked receive FIFO event. This interrupt is set if the selected Receive FIFO level has been reached 0h = Interrupt did not occur 1h = Interrupt occurred
2	RTOUT	R	0h	Masked SPI Receive Time-Out Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 12-40. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PER	R	0h	Masked Parity error event: this bit if a Parity error has been detected 0h = Interrupt did not occur 1h = Interrupt occurred
0	RXFIFO_OVF	R	0h	Masked RXFIFO overflow event. This interrupt is set if an RX FIFO overflow has been detected. 0h = Interrupt did not occur 1h = Interrupt occurred

12.3.33 ISET (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 12-41 and described in Table 12-41.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 12-41. ISET

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED					RXFULL	TXFIFO_UNF	DMA_DONE_TX
W-0h					W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DMA_DONE_RX	IDLE	TXEMPTY	TX	RX	RTOUT	PER	RXFIFO_OVF
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 12-41. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	W	0h	
10	RXFULL	W	0h	Set RX FIFO Full Event 0h = Writing has no effect 1h = Set Interrupt
9	TXFIFO_UNF	W	0h	Set TX FIFO Underflow Event 0h = Writing has no effect 1h = Set interrupt
8	DMA_DONE_TX	W	0h	Set DMA Done 1 event for TX. 0h = Writing 0 has no effect 1h = Set Interrupt
7	DMA_DONE_RX	W	0h	Set DMA Done 1 event for RX. 0h = Writing 0 has no effect 1h = Set Interrupt
6	IDLE	W	0h	Set SPI IDLE mode event. 0h = Writing 0 has no effect 1h = Set Interrupt
5	TXEMPTY	W	0h	Set Transmit FIFO Empty event. 0h = Writing 0 has no effect 1h = Set Interrupt
4	TX	W	0h	Set Transmit FIFO event. 0h = Writing 0 has no effect 1h = Set Interrupt
3	RX	W	0h	Set Receive FIFO event. 0h = Writing 0 has no effect 1h = Set Interrupt
2	RTOUT	W	0h	Set SPI Receive Time-Out Event. 0h = Writing 0 has no effect 1h = Set Interrupt Mask

Table 12-41. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PER	W	0h	Set Parity error event. 0h = Writing 0 has no effect 1h = Set Interrupt
0	RXFIFO_OVF	W	0h	Set RXFIFO overflow event. 0h = Writing 0 has no effect 1h = Set Interrupt

12.3.34 ICLR (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 12-42 and described in Table 12-42.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 12-42. ICLR

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED					RXFULL	TXFIFO_UNF	DMA_DONE_TX
W-0h					W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
DMA_DONE_RX	IDLE	TXEMPTY	TX	RX	RTOUT	PER	RXFIFO_OVF
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 12-42. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	W	0h	
10	RXFULL	W	0h	Clear RX FIFO underflow event 0h = Writing has no effect 1h = Clear interrupt
9	TXFIFO_UNF	W	0h	Clear TXFIFO underflow event 0h = Writing has no effect 1h = Clear interrupt
8	DMA_DONE_TX	W	0h	Clear DMA Done 1 event for TX. 0h = Writing 0 has no effect 1h = Clear Interrupt
7	DMA_DONE_RX	W	0h	Clear DMA Done 1 event for RX. 0h = Writing 0 has no effect 1h = Clear Interrupt
6	IDLE	W	0h	Clear SPI IDLE mode event. 0h = Writing 0 has no effect 1h = Clear Interrupt
5	TXEMPTY	W	0h	Clear Transmit FIFO Empty event. 0h = Writing 0 has no effect 1h = Clear Interrupt
4	TX	W	0h	Clear Transmit FIFO event. 0h = Writing 0 has no effect 1h = Clear Interrupt
3	RX	W	0h	Clear Receive FIFO event. 0h = Writing 0 has no effect 1h = Clear Interrupt
2	RTOUT	W	0h	Clear SPI Receive Time-Out Event. 0h = Writing 0 has no effect 1h = Set Interrupt Mask

Table 12-42. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PER	W	0h	Clear Parity error event. 0h = Writing 0 has no effect 1h = Clear Interrupt
0	RXFIFO_OVF	W	0h	Clear RXFIFO overflow event. 0h = Writing 0 has no effect 1h = Clear Interrupt

12.3.35 IIDX (Offset = 1050h) [Reset = 00000000h]

IIDX is shown in Figure 12-43 and described in Table 12-43.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 12-43. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 12-43. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No interrupt pending 3h = SPI receive time-out interrupt 4h = Receive Event/interrupt pending

12.3.36 IMASK (Offset = 1058h) [Reset = 00000000h]

IMASK is shown in Figure 12-44 and described in Table 12-44.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 12-44. IMASK

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				RX	RTOUT	RESERVED	
R/W-0h				R/W-0h	R/W-0h	R/W-0h	

Table 12-44. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3	RX	R/W	0h	Receive FIFO event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	RTOUT	R/W	0h	SPI Receive Time-Out event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1-0	RESERVED	R/W	0h	

12.3.37 RIS (Offset = 1060h) [Reset = 00000000h]

RIS is shown in Figure 12-45 and described in Table 12-45.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 12-45. RIS

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED				RX	RTOUT	RESERVED	
R-				R-0h	R-0h	R-	

Table 12-45. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	RX	R	0h	Receive FIFO event. This interrupt is set if the selected Receive FIFO level has been reached 0h = Interrupt did not occur 1h = Interrupt occurred
2	RTOUT	R	0h	SPI Receive Time-Out Event. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1-0	RESERVED	R	0h	

12.3.38 MIS (Offset = 1068h) [Reset = 00000000h]

MIS is shown in Figure 12-46 and described in Table 12-46.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 12-46. MIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RX	RTOUT	RESERVED	
R-0h				R-0h	R-0h	R-0h	

Table 12-46. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	RX	R	0h	Receive FIFO event mask. 0h = Interrupt did not occur 1h = Interrupt occurred
2	RTOUT	R	0h	SPI Receive Time-Out event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1-0	RESERVED	R	0h	

12.3.39 ISET (Offset = 1070h) [Reset = 00000000h]

ISET is shown in Figure 12-47 and described in Table 12-47.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 12-47. ISET

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED				RX	RTOUT	RESERVED	
W-0h				W-0h	W-0h	W-0h	

Table 12-47. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	W	0h	
3	RX	W	0h	Set Receive FIFO event. 0h = Writing 0 has no effect 1h = Set Interrupt
2	RTOUT	W	0h	Set SPI Receive Time-Out event. 0h = Writing 0 has no effect 1h = Set Interrupt Mask
1-0	RESERVED	W	0h	

12.3.40 ICLR (Offset = 1078h) [Reset = 00000000h]

ICLR is shown in Figure 12-48 and described in Table 12-48.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 12-48. ICLR

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED				RX	RTOUT	RESERVED	
W-0h				W-0h	W-0h	W-0h	

Table 12-48. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	W	0h	
3	RX	W	0h	Clear Receive FIFO event. 0h = Writing 0 has no effect 1h = Clear Interrupt
2	RTOUT	W	0h	Clear SPI Receive Time-Out event. 0h = Writing 0 has no effect 1h = Set Interrupt Mask
1-0	RESERVED	W	0h	

12.3.41 IIDX (Offset = 1080h) [Reset = 00000000h]

IIDX is shown in Figure 12-49 and described in Table 12-49.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 12-49. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 12-49. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No interrupt pending 5h = Transmit Event/interrupt pending

12.3.42 IMASK (Offset = 1088h) [Reset = 00000000h]

IMASK is shown in Figure 12-50 and described in Table 12-50.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 12-50. IMASK

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

Table 12-50. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	
4	TX	R/W	0h	Transmit FIFO event mask. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3-0	RESERVED	R/W	0h	

12.3.43 RIS (Offset = 1090h) [Reset = 00000000h]

RIS is shown in Figure 12-51 and described in Table 12-51.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 12-51. RIS

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

Table 12-51. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4	TX	R	0h	Transmit FIFO event: A read returns the current mask for transmit FIFO interrupt. On a write of 1, the mask for transmit FIFO interrupt is set which means the interrupt state will be reflected in MIS.TXMIS. A write of 0 clears the mask which means MIS.TXMIS will not reflect the interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred
3-0	RESERVED	R	0h	

12.3.44 MIS (Offset = 1098h) [Reset = 00000000h]

MIS is shown in Figure 12-52 and described in Table 12-52.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 12-52. MIS

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

Table 12-52. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4	TX	R	0h	Masked Transmit FIFO event 0h = Interrupt did not occur 1h = Interrupt occurred
3-0	RESERVED	R	0h	

12.3.45 ISET (Offset = 10A0h) [Reset = 00000000h]

ISET is shown in Figure 12-53 and described in Table 12-53.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 12-53. ISET

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

Table 12-53. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	W	0h	
4	TX	W	0h	Set Transmit FIFO event. 0h = Writing 0 has no effect 1h = Set Interrupt
3-0	RESERVED	W	0h	

12.3.46 ICLR (Offset = 10A8h) [Reset = 00000000h]

ICLR is shown in Figure 12-54 and described in Table 12-54.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 12-54. ICLR

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

Table 12-54. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	W	0h	
4	TX	W	0h	Clear Transmit FIFO event. 0h = Writing 0 has no effect 1h = Clear Interrupt
3-0	RESERVED	W	0h	

12.3.47 EVT_MODE (Offset = 10E0h) [Reset = 00000029h]

EVT_MODE is shown in Figure 12-55 and described in Table 12-55.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 12-55. EVT_MODE

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED		INT2_CFG		INT1_CFG		INT0_CFG	
R/W-		R-2h		R-2h		R-1h	

Table 12-55. EVT_MODE Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	
5-4	INT2_CFG	R	2h	Event line mode select for event corresponding to none.INT_EVENT2 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
3-2	INT1_CFG	R	2h	Event line mode select for event corresponding to none.INT_EVENT1 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
1-0	INT0_CFG	R	1h	Event line mode select for event corresponding to none.INT_EVENT0 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

12.3.48 INTCTL (Offset = 10E4h) [Reset = 00000000h]

INTCTL is shown in Figure 12-56 and described in Table 12-56.

Return to the Summary Table.

Interrupt control register

Figure 12-56. INTCTL

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

Table 12-56. INTCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	INTEVAL	W	0h	Writing a 1 to this field re-evaluates the interrupt sources. 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS.

12.3.49 CTL0 (Offset = 1100h) [Reset = 00000000h]

CTL0 is shown in Figure 12-57 and described in Table 12-57.

Return to the Summary Table.

SPI control register 0

Figure 12-57. CTL0

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED	CSCLR	CSSEL		RESERVED		SPH	SPO
R/W-0h	R/W-0h	R/W-0h		R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PACKEN	FRF		DSS				
R/W-0h	R/W-0h		R/W-0h				

Table 12-57. CTL0 Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R/W	0h	
14	CSCLR	R/W	0h	Clear shift register counter on CS inactive This bit is relevant only in the peripheral, CTL1.CP=0. 0h = Disable automatic clear of shift register when CS goes to disable. 1h = Enable automatic clear of shift register when CS goes to disable.
13-12	CSSEL	R/W	0h	Select the CS line to control on data transfer This bit is applicable for both controller/target mode 0h (R/W) = CS line select: 0 1h (R/W) = CS line select: 1 2h (R/W) = CS line select: 2 3h (R/W) = CS line select: 3
11-10	RESERVED	R/W	0h	
9	SPH	R/W	0h	CLKOUT phase (Motorola SPI frame format only) This bit selects the clock edge that captures data and enables it to change state. It has the most impact on the first bit transmitted by either permitting or not permitting a clock transition before the first data capture edge. 0h = Data is captured on the first clock edge transition. 1h = Data is captured on the second clock edge transition.
8	SPO	R/W	0h	CLKOUT polarity (Motorola SPI frame format only) 0h = SPI produces a steady state LOW value on the CLKOUT 1h = SPI produces a steady state HIGH value on the CLKOUT
7	PACKEN	R/W	0h	Packing Enable. When 1, packing feature is enabled inside the IP When 0, packing feature is disabled inside the IP 0h = Packing feature disabled 1h = Packing feature enabled

Table 12-57. CTL0 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-5	FRF	R/W	0h	Frame format Select 0h = Motorola SPI frame format (3 wire mode) 1h = Motorola SPI frame format (4 wire mode) 2h = TI synchronous serial frame format 3h = National Microwire frame format
4-0	DSS	R/W	0h	Data Size Select. Values 0 - 2 are reserved and shall not be used. 3h = 4_BIT : 4-bit data SPI allows only values up to 16 Bit 3h (R/W) = Data Size Select bits: 4 4h (R/W) = Data Size Select bits: 5 5h (R/W) = Data Size Select bits: 6 6h (R/W) = Data Size Select bits: 7 7h (R/W) = Data Size Select bits: 8 8h (R/W) = Data Size Select bits: 9 9h (R/W) = Data Size Select bits: 10 Ah (R/W) = Data Size Select bits: 11 Bh (R/W) = Data Size Select bits: 12 Ch (R/W) = Data Size Select bits: 13 Dh (R/W) = Data Size Select bits: 14 Eh (R/W) = Data Size Select bits: 15 Fh (R/W) = Data Size Select bits: 16

12.3.50 CTL1 (Offset = 1104h) [Reset = 00000004h]

CTL1 is shown in Figure 12-58 and described in Table 12-58.

Return to the Summary Table.

SPI control register 1

Figure 12-58. CTL1

31	30	29	28	27	26	25	24
RESERVED		RXTIMEOUT					
R/W-0h		R/W-0h					
23	22	21	20	19	18	17	16
REPEATTX							
R/W-0h							
15	14	13	12	11	10	9	8
CDMODE				CDENABLE	RESERVED		PTEN
R/W-0h				R/W-0h	R/W-0h		R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PES	PREN	MSB	POD	CP	LBM	ENABLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h

Table 12-58. CTL1 Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	
29-24	RXTIMEOUT	R/W	0h	Receive Timeout (only for Peripheral mode) Defines the number of Clock Cycles before after which the Receive Timeout flag RTOUT is set. The time is calculated using the control register for the clock selection and divider in the Controller mode configuration. A value of 0 disables this function. 0h = Smallest value 3Fh = Highest possible value
23-16	REPEATTX	R/W	0h	Counter to repeat last transfer 0: repeat last transfer is disabled. x: repeat the last transfer with the given number. The transfer will be started with writing a data into the TX Buffer. Sending the data will be repeated with the given value, so the data will be transferred X+1 times in total. The behavior is identical as if the data would be written into the TX Buffer that many times as defined by the value here. It can be used to clean a transfer or to pull a certain amount of data by a peripheral. 0h = Smallest value FFh = Highest possible value
15-12	CDMODE	R/W	0h	Command/Data Mode Value When CTL1.CDENABLE is 1, CS3 line is used as C/D signal to distinguish between Command (C/D low) and Data (C/D high) information. When a value is written into the CTL1.CDMODE bits, the C/D (CS3) line will go low for the given numbers of byte which are sent by the SPI, starting with the next value to be transmitted after which, C/D line will go high automatically 0: Manual mode with C/D signal as High 1-14: C/D is low while this number of bytes are being sent after which, this field sets to 0 and C/D goes high. Reading this field at any time returns the remaining number of command bytes. 15: Manual mode with C/D signal as Low. 0h = Manual mode: Data 0h = Smallest value Fh = Manual mode: Command

Table 12-58. CTL1 Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CDENABLE	R/W	0h	Command/Data Mode enable 0h = CS3 is used for Chip Select 1h = CS3 is used as CD signal
10-9	RESERVED	R/W	0h	
8	PTEN	R/W	0h	Parity transmit enable If enabled, parity transmission will be done for both controller and peripheral modes. 0h = Parity transmission is disabled 1h = Parity transmission is enabled
7	RESERVED	R/W	0h	
6	PES	R/W	0h	Even Parity Select 0h = Odd Parity mode 1h = Even Parity mode
5	PREN	R/W	0h	Parity receive enable If enabled, parity reception check will be done for both controller and peripheral modes In case of a parity miss-match the parity error flag RIS.PER will be set. 0h = Disable Parity receive function 1h = Enable Parity receive function
4	MSB	R/W	0h	MSB first select. Controls the direction of the receive and transmit shift register. 0h = LSB first 1h = MSB first
3	POD	R/W	0h	Peripheral-mode: Data output disabled This bit is relevant only in Peripheral mode. In multiple-peripheral system topologies, SPI controller can broadcast a message to all peripherals, while only one peripheral drives the line. POD can be used by the SPI peripheral to disable driving data on the line. 0h = SPI can drive the POCI output in peripheral mode. 1h = SPI cannot drive the POCI output in peripheral mode.
2	CP	R/W	1h	Controller or peripheral mode select. This bit can be modified only when SPI is disabled, CTL1.ENABLE=0. 0h = Select Peripheral mode 1h = Select Controller Mode
1	LBM	R/W	0h	Loop back mode 0h = Disable loopback mode 1h = Enable loopback mode
0	ENABLE	R/W	0h	SPI enable 0h = Disable module function 1h = Enable module function

12.3.51 CLKCTL (Offset = 1108h) [Reset = 00000000h]

CLKCTL is shown in Figure 12-59 and described in Table 12-59.

Return to the Summary Table.

Clock prescaler and divider register. This register contains the settings for the Clock prescaler and divider settings.

Figure 12-59. CLKCTL

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DSAMPLE				RESERVED											
R/W-0h				R/W-0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						SCR									
R/W-0h						R/W-0h									

Table 12-59. CLKCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-28	DSAMPLE	R/W	0h	Delayed sampling value. In controller mode the data on the input pin will be delayed sampled by the defined clock cycles of internal functional clock hence relaxing the setup time of input data. This setting is useful in systems where the board delays and external peripheral delays are more than the input setup time of the controller. Please refer to the data sheet for values of controller input setup time and assess what DSAMPLE value meets the requirement of the system. Note: High values of DSAMPLE can cause HOLD time violations and must be factored in the calculations. 0h = Smallest value Fh = Highest possible value
27-10	RESERVED	R/W	0h	
9-0	SCR	R/W	0h	Serial clock divider: This is used to generate the transmit and receive bit rate of the SPI. The SPI bit rate is (SPI functional clock frequency)/((SCR+1)×2). SCR is a value from 0-1023. 0h = Smallest value 3FFh = Highest possible value

12.3.52 IFLS (Offset = 110Ch) [Reset = 00000012h]

IFLS is shown in Figure 12-60 and described in Table 12-60.

Return to the Summary Table.

The IFLS register is the interrupt FIFO level select register. You can use this register to define the levels at which the TX, RX and timeout interrupt flags are triggered. The interrupts are generated based on a transition through a level rather than being based on the level. That is, the interrupts are generated when the fill level progresses through the trigger level. For example, if the receive trigger level is set to the half-way mark, the interrupt is triggered when the receive FIFO is filled with two or more characters. Out of reset, the TXIFLSEL and RXIFLSEL bits are configured so that the FIFOs trigger an interrupt at the half-way mark.

Figure 12-60. IFLS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										RXIFLSEL			TXIFLSEL		
R/W-0h										R/W-2h			R/W-2h		

Table 12-60. IFLS Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	
5-3	RXIFLSEL	R/W	2h	SPI Receive Interrupt FIFO Level Select The trigger points for the receive interrupt are as follows: 0h = Reserved 1h = RX FIFO >= 1/4 full 2h = RX FIFO >= 1/2 full (default) 3h = RX FIFO >= 3/4 full 4h = Reserved 5h = RX FIFO is full 6h = Reserved 7h = Trigger when RX FIFO contains >= 1 frame
2-0	TXIFLSEL	R/W	2h	SPI Transmit Interrupt FIFO Level Select The trigger points for the transmit interrupt are as follows: 0h = Reserved 1h = TX FIFO <= 3/4 empty 2h = TX FIFO <= 1/2 empty (default) 3h = TX FIFO <= 1/4 empty 4h = Reserved 5h = TX FIFO is empty 6h = Reserved 7h = Trigger when TX FIFO has >= 1 frame free.

12.3.53 STAT (Offset = 1110h) [Reset = 0000000Fh]

STAT is shown in Figure 12-61 and described in Table 12-61.

Return to the Summary Table.

Status Register

Figure 12-61. STAT

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED			BUSY	RNF	RFE	TNF	TFE
R-			R-0h	R-1h	R-1h	R-1h	R-1h

Table 12-61. STAT Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4	BUSY	R	0h	Busy 0h = SPI is in idle mode. 1h = SPI is currently transmitting and/or receiving data, or transmit FIFO is not empty.
3	RNF	R	1h	Receive FIFO not full 0h = Receive FIFO is full. 1h = Receive FIFO is not full.
2	RFE	R	1h	Receive FIFO empty. 0h = Receive FIFO is not empty. 1h = Receive FIFO is empty.
1	TNF	R	1h	Transmit FIFO not full 0h = Transmit FIFO is full. 1h = Transmit FIFO is not full.
0	TFE	R	1h	Transmit FIFO empty. 0h = Transmit FIFO is not empty. 1h = Transmit FIFO is empty.

12.3.54 RXDATA (Offset = 1130h) [Reset = 00000000h]

RXDATA is shown in Figure 12-62 and described in Table 12-62.

Return to the Summary Table.

RXDATA Register
Reading this register returns value(s) of FIFO. If the FIFO is empty the last read value is returned.
Writing has not effect and is ignored.
When PACKEN=1,two entries of the FIFO are returned as a 32-bit value. When PACKEN=0, 1 entry of FIFO is returned as 16-bit value.

Figure 12-62. RXDATA

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
DATA																															
R-0h																															

Table 12-62. RXDATA Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	Received Data When PACKEN=1,two entries of the FIFO are returned as a 32-bit value. When PACKEN=0, 1 entry of FIFO is returned as 16-bit value. As data values are removed by the receive logic from the incoming data frame, they are placed into the entry in the receive FIFO, pointed to by the current FIFO write pointer. Received data less than 16 bits is automatically right justified in the receive buffer. 0h = Smallest value FFFFFFFFh = Highest possible value

12.3.55 TXDATA (Offset = 1140h) [Reset = 00000000h]

TXDATA is shown in Figure 12-63 and described in Table 12-63.

Return to the Summary Table.

TXDATA Register
Writing puts the data into the TX FIFO. Reading this register returns the last written value.
When PACKEN=0, only the lower 16-bits of data written into the register is transferred to one 16-bits wide TX FIFO entry
When PACKEN=1, upper and lower 16-bits of 32-bit write data are transferred to two16-bits wide TX FIFO entry

Figure 12-63. TXDATA

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
DATA																															
R/W-0h																															

Table 12-63. TXDATA Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	Transmit Data When read, last written value will be returned. If the last write to this field was a 32-bit write (with PACKEN=1), 32-bits will be returned and if the last write was a 16-bit write (PACKEN=0), those 16-bits will be returned. When written, one or two FIFO entries will be written depending on PACKEN value. Data values are removed from the transmit FIFO one value at a time by the transmit logic. It is loaded into the transmit serial shifter, then serially shifted out onto the TXD output pin at the programmed bit rate. When a data size of less than 16 bits is selected, the user must right-justify data written to the transmit FIFO. The transmit logic ignores the unused bits. 0h = Smallest value FFFFFFFFh = Highest possible value

12.3.56 TEST0 (Offset = 1E00h) [Reset = 00000000h]

TEST0 is shown in Figure 12-64 and described in Table 12-64.

Return to the Summary Table.

Test 0 register.

Figure 12-64. TEST0

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

Table 12-64. TEST0 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	DTB_MUX_SEL	R/W	0h	This bit field is used to select DTB mux digital output signals. 0h = Disables DTB MUX 1h = Selects test group 1 2h = Selects test group 2 3h = Selects test group 3 4h = Selects test group 4 5h = Selects test group 5 6h = Selects test group 6 7h = Selects test group 7 8h = Selects test group 8 9h = Selects test group 9

Chapter 13

I²C

The I²C module provides a standardized serial interface to transfer data between MSP devices and other external I²C devices (such as a sensors, memory, or DACs).

13.1 I²C Overview

The I²C module provides a standardized serial interface to transfer data between MSP devices and other external I²C devices (such as a sensors, memory, or DACs).

13.1.1 Purpose of the Peripheral

The I²C peripheral provides bidirectional data transfer through a two-wire serial bus consisting of a data (SDA) and clock (SCL) line. The I²C bus is widely used to interface with devices such as battery management ICs, sensors, other MCUs and so on. This I²C peripheral provides the ability to both transmit to and receive from other I²C devices on the bus.

13.1.2 Features

The controller includes I²C modules with the following features:

- Devices on the I²C bus can be designated as either a controller or a target with 7-bit addressing.
- Support four I²C modes
  - Controller transmit
  - Controller receive
  - Target transmit
  - Target receive
- Supported transmission speeds:
  - Standard-mode (Sm) with a bit rate up to 100 kbps
  - Fast-mode (Fm) with a bit rate up to 400 kbps
  - Fast-mode Plus (Fm+) with a bit rate up to 1 Mbps
- Independent 8-byte FIFOs for reception and transmission
- Dual target address capability
- Glitch suppression
- Independent controller and target interrupt generation
- Controller operation with arbitration, clock synchronization, multiple controller support
- Hardware support for SMBus and PMBus
  - Clock low timeout detection and interrupt
  - Quick command capability
- Hardware support for DMA with separate channels for transmit and receive

13.1.3 Functional Block Diagram

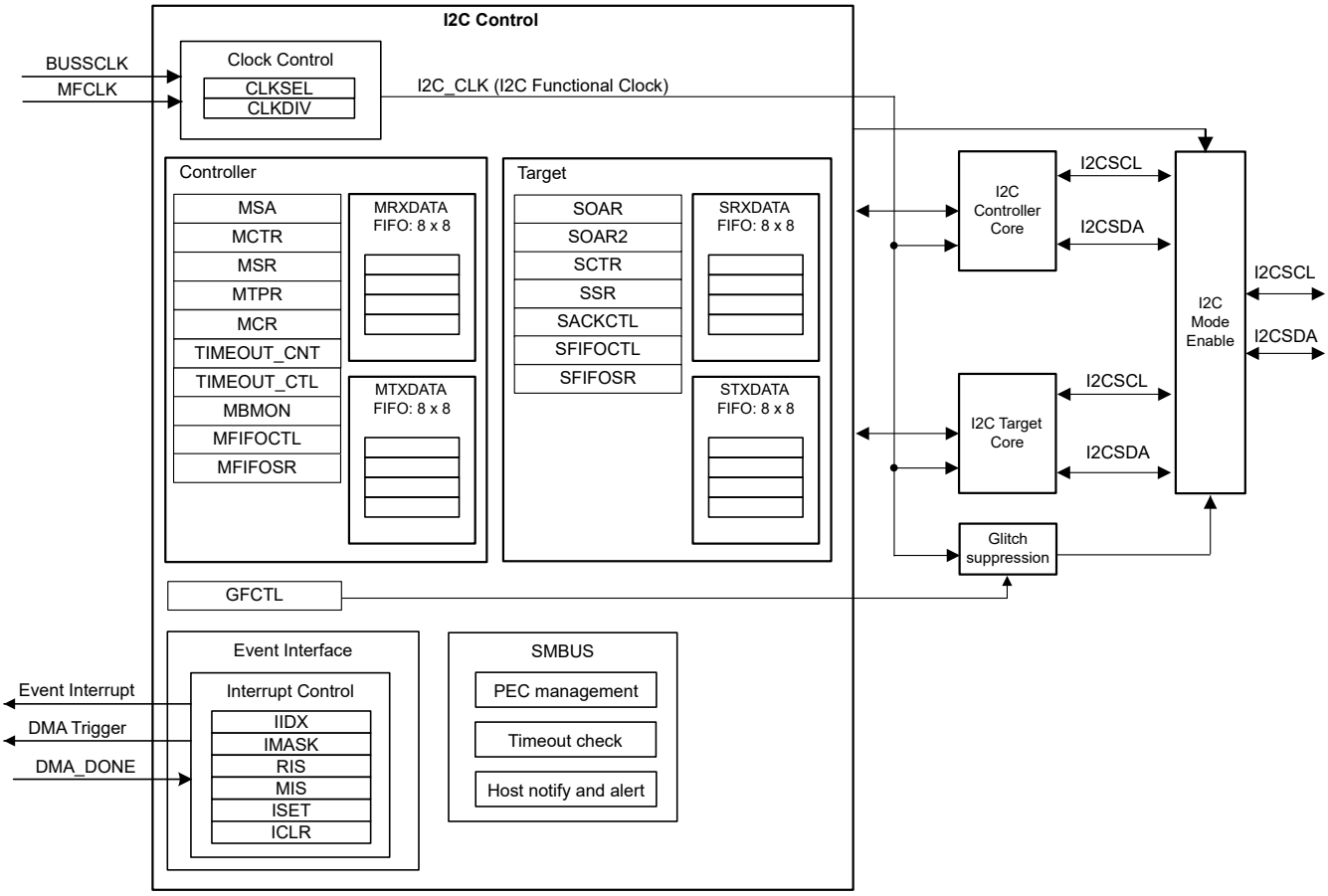


Figure 13-1. I2C Functional Block Diagram

13.1.4 Environment and External Connections

The I²C mode supports any target or controller I²C-compatible device. Figure 13-2 shows an example of an I²C bus. Each I²C peripheral instance is comprised of both controller and target functions where the target can be addressed with 2 independent user defined addresses. A device connected to the I²C bus can be considered as the controller or the target when performing data transfers. A controller initiates a data transfer and generates the clock signal SCL. Any device addressed by a controller is considered a target.

I²C data is communicated using the serial data (SDA) pin and the serial clock (SCL) pin. Both SDA and SCL are open-drain bidirectional and must be connected to a positive supply voltage using a pullup resistor.

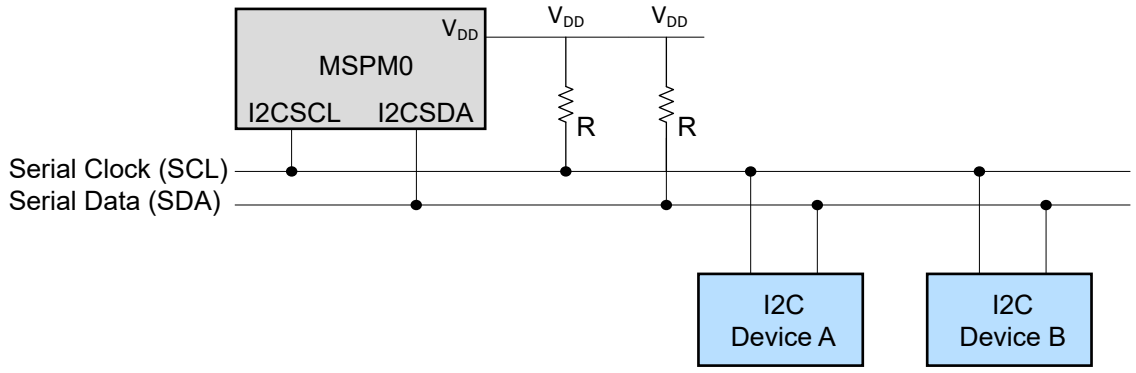


Figure 13-2. Bus Connection Diagram

13.2 I²C Operation

This section describes the operation of the I²C peripheral.

13.2.1 Clock Control

13.2.1.1 Clock Select and I²C Speed

Standard, Fast, and Fast Plus modes are selected using a value in the I²C Controller Timer Period (I2CMTPR) register that results in a maximum SCL frequency of:

- 100kbps for Standard mode
- 400kbps for Fast mode
- 1Mbps for Fast mode Plus

The I²C frequency I2C_FREQ is determined by the I2C_CLK frequency and bit fields TPR , SCL_LP , and SCL_HP where:

- I2C_CLK is the functional clock frequency to the I²C module. Note that the I²C internal functional clock is first divided from the source clock:
  - Use I2Cx.CLKSEL register to select the source of the I²C functional clock
    - BUSSCLK: the current bus clock is selected as the source for I²C. The current bus clock depends on power domain. If the I²C instance is in power domain 1 (PD1) refer to MCLK, if the I²C instance is in power domain 0 (PD0) refer to ULPCLK.
    - MFCLK: MFCLK is selected as the source for I²C, refer to MFCLK.
  - Use I2Cx.CLKDIV register to select divide ratio of I²C function clock, options are from divide by 1 to 8.
- SCL_LP is the low phase of SCL (which must be fixed at 6)
- SCL_HP is the high phase of SCL (which must be fixed at 4)
- TPR is the programmed value of the TPR bits in the I2Cx.MTPR register. This value is determined by replacing the known variables in the equation below and solving for TPR

The I²C frequency is calculated as follows:

$$I2C\_FREQ = I2C\_CLK / ((1+TPR) \times (SCL\_LP + SCL\_HP)) \quad (11)$$

For example, if the I²C functional clock frequency is 32MHz and target SCL frequency is 400kHz:

I2C_CLK = 32MHz

I2C_FREQ = 400kHz

SCL_LP = 6 , SCL_HP = 4

TPR = (I2C_CLK / (I2C_FREQ * (4 + 6))) - 1

TPR = 7 (0x07)

Table 13-1. Examples of Controller Clock Setting for Typical Clock Configurations

Functional Clock	TPR Bits Standard Mode 100kHz SCL	TPR Bits Fast Mode 400kHz SCL	TPR Bits Fast Mode Plus 1000kHz SCL
4MHz	0x03	-	-
8MHz	0x07	0x01	-
20MHz	0x13	0x04	0x01
32MHz	0x1F	0x07	0x02(1)
40MHz	0x27	0x09	0x03

(1) With 32MHz functional clock, TPR = 0x01 generates 1.6MHz SCL frequency, and TPR = 0x02 generates 1.067MHz SCL frequency.

I²C functional clock must be greater than or equal to 20 times the SCL frequency, I2C_CLK ≥ 20 × I2C_FREQ.

The following minimum functional clock frequencies are required when running certain I2C clock speeds:

- I2C_CLK ≥ 2MHz when working with I²C speed 0 to 100kHz
- I2C_CLK ≥ 8MHz when working with I²C speed 100 to 400kHz
- I2C_CLK ≥ 20MHz when working with I²C speed 400 kHz to 1MHz

13.2.1.2 Clock Startup

The selected clock source is always available and the frequency depends on the power mode, for more information refer to the PMU/Clock section. After enabling the I²C module by setting the I2C.PWREN.ENABLE bit, the module is ready to start receiving and transmitting data.

13.2.2 Signal Descriptions

The I²C bus consists of a clock signal and a data signal. The clock signal can be generated either internally (during controller operation) or externally (during target operation).

Table 13-2. I²C signal descriptions

Device Pin	Function
I2Cx_SCL	I²C clock signal
I2Cx_SDA	I²C data signal

13.2.3 General Architecture

13.2.3.1 I²C Bus Functional Overview

The I²C bus uses only two signals: SDA and SCL. SDA is the bidirectional serial data line and SCL is the bidirectional serial clock line. The bus is considered idle when both lines are in high state and no transfer is ongoing.

Every transaction on the I²C bus is 9-bits long, consisting of 8 data bits and 1 acknowledge bit. A transfer is defined as the time between a valid START and STOP condition—as described in Figure 13-3. The number of bytes per transfer is unrestricted; however, each data byte must be followed by an acknowledge bit and data must be transferred MSB first. When a receiver cannot receive another complete byte, it can hold the clock line SCL Low and force the transmitter into a wait state, this process is commonly known as clock stretching. The data transfer continues when the receiver releases the clock SCL.

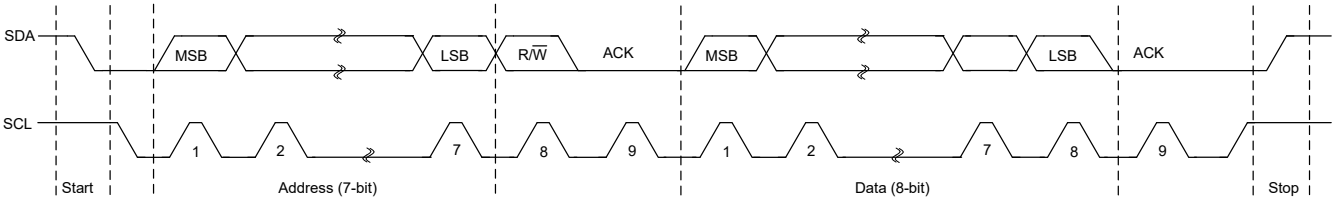


Figure 13-3. Module Data Transfer

The data on the SDA line must be stable during the high period of the clock, and the data line can only change when SCL is low (see Figure 13-4), otherwise START or STOP conditions are generated.

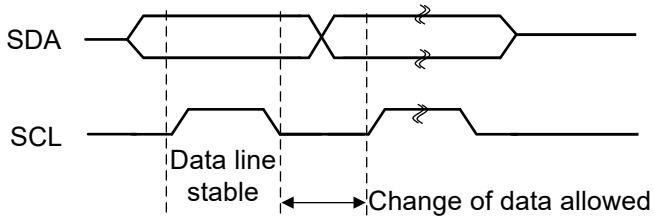


Figure 13-4. Data Change Period

13.2.3.2 START and STOP Conditions

The protocol of the I²C bus defines two states to begin and end a transaction: START and STOP. A high-to-low transition on the SDA line while the SCL is high is defined as a START condition, and a low-to-high transition on the SDA line while SCL is high is defined as a STOP condition. START and STOP conditions are always generated by the controller. The bus is considered busy after a START condition and free after a STOP condition.

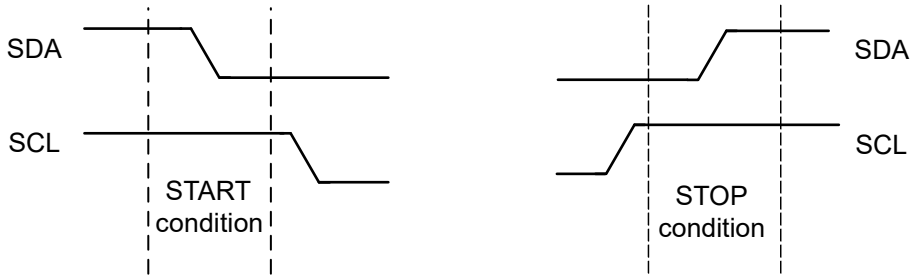


Figure 13-5. START and STOP Conditions

The STOP bit determines if the transaction stops at the end of the data cycle or continues on to a repeated START condition.

To generate a single transaction, the I²C controller target address I2Cx.MSA.SADDR register is written with the desired address, the I2Cx.MSA.DIR bit should be set to 1 to start a receive operation and 0 to start a transmit operation , and the control register (I2Cx.MCTR) is written with ACK = X (0 or 1), STOP = 1, START = 1, and RUN = 1 to perform the operation and generate a STOP at the end. When the operation is completed (or aborted due an error) the interrupt flags are set. When the I²C module operates in Controller receiver mode, the ACK bit is normally (in case of no error) set causing the I²C bus controller to transmit an acknowledge automatically after each byte. This bit must be cleared when the I²C bus controller requires no further data to be transmitted from the target transmitter. For more information on I²C controller mode configuration please refer to Section 13.2.4.1

When operating in target mode, the START and STOP bits in the CPU_INT.RIS register indicate detection of start and stop conditions on the bus and the CPU_INT.IMASK can be configured to allow START and STOP to be promoted to controller interrupts (when interrupts are enabled). For more information on I²C target mode configuration please refer to Section 13.2.4.2

Bus Status Flags:

- BUSBSY is set when a START is detected on the bus and sequentially cleared when a STOP condition is detected on the bus. It is also cleared after a clock timeout I2Cx.MSR.CLKTO is set once both SCL and SDA are pulled to high.
- BUSY is also set after a START/RESTART, and cleared after I2Cx.MCTR.MBLEN bytes of data is transferred. It is also cleared after data is NACK ‘d or a STOP.
- IDLE is set when the Controller I²C State machine is in the IDLE state indicating no ongoing transfer.

13.2.3.3 Data Format with 7-Bit Address

Data transfers follow the format shown in Figure 13-6. After the START condition, a target address is transmitted. This address is 7-bits long followed by an eighth bit, which is a data direction bit (this bit is only as Controller mode, DIR bit in the I2Cx.MSA register). If the I2Cx.MSA.DIR bit is 0, it indicates a transmit operation (send), and if it is set to 1, it indicates a request to receive data (receive). A data transfer is always terminated by a STOP condition generated by the controller; however, a controller can initiate communications with another device on the bus by generating a repeated START condition and addressing another target without first generating a STOP condition, see section Repeated Start. Various combinations of receive/transmit formats are then possible within a single transfer. The ninth bit is the Acknowledge bit, which is described in Section 13.2.3.4 .

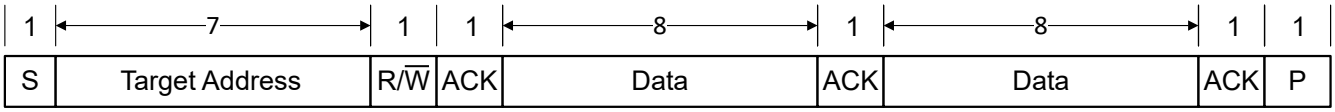


Figure 13-6. Data Format with 7-Bit Address

With the I2Cx.SCTR.GENCALL bit the I²C module can be enabled to respond on a General Call on the I²C bus. The General Call is identified by address of 0x00 and the R/W bit set to 0. The General Call interrupt can be enabled with the CPU_INT.IMASK.GENCALL bit.

13.2.3.4 Acknowledge

All bus transactions have a required acknowledge clock cycle that is generated by the controller. During the acknowledge cycle, the transmitter (which can be the controller or target) releases the SDA line. To acknowledge the transaction, the receiver must pull down SDA during the acknowledge clock cycle. The acknowledge cycle must comply with the data validity requirements.

When a target receiver does not acknowledge the target address, SDA must be left high by the target so that the controller can generate a STOP condition and abort the current transfer or generate a repeated START condition to start a new transfer. If the controller device is acting as a receiver during a transfer, it is responsible for acknowledging each transfer made by the target. Because the controller controls the number of bytes in the transfer, it signals the end of data to the target transmitter by not generating an acknowledge on the last data byte. The target transmitter must then release SDA to allow the controller to generate the STOP or a repeated START condition.

A target can generate ACK/NACK manually or automatically. When I2Cx.SACKCTL.ACKOEN=0, the target will send automatic ACK. Note that due to the FIFO, the device will receive and ACK all bytes automatically until the RX FIFO is full. Setting SACKCTL.ACKOEN =1 enables manual ACK. Manual ACK override can be used to evaluate each received byte or to slow down the communication when automatic FIFO reception is not desired. When manual ACK override operation is enabled, the I²C target module's clock is pulled low after the last data bit until this SACKCTL.ACKOVAL is written with the indicated response. The reception of new data is indicated by the SRXDONE interrupt flag.

If the controller receives a NACK while transmitting data the NACK and MTXDONE bit will be set in the RIS registers. If there is still data in the FIFO the TXEMPTY bit will not be set to inform software that a TX FIFO flush may be required.

13.2.3.5 Repeated Start

The direction of data flow on SDA can be changed by the controller, without first stopping a transfer, by issuing a repeated START condition. This is called a RESTART, see data format in Figure 13-7. After a RESTART is issued, the target address is again sent out with the new data direction specified by the R/W bit.

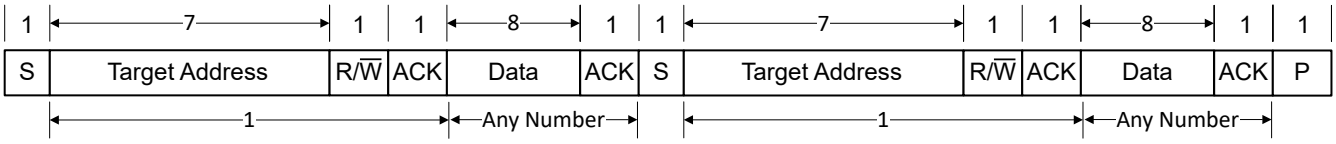


Figure 13-7. Data Format with Repeated Start

A repeated start sequence for a Controller transmit is as follows:

1. When the device is in the idle state, the Controller writes the target address to the I2Cx.MSA register and configures the DIR bit for the desired transfer type.
2. Data is written to the I2Cx.MTXDATA register.
3. When the BUSY bit in the MSR register is 0, the BURSTRUN and START bit in the I2Cx.MCTR register need to be set to initiate a transfer.
4. Wait till the BUSY bit in the I2Cx.MSR register gets 0.

5. The Controller does not generate a STOP condition but instead writes another target address to the I2Cx.MSA register and then sets the BURSTRUN and START bit again with a write operation to initiate the repeated START.

A repeated start sequence for a Controller receive is similar:

1. When the device is in idle, the Controller writes the target address to the I2Cx.MSA register and configures the DIR bit for the desired transfer type.
2. When the BUSY bit in the I2Cx.MSR register is 0, the BURSTRUN and START bit in the I2Cx.MCTR register need to be set to initiate a transfer.
3. The controller reads data from the I2Cx.MRXDATA register.
4. Wait till the BUSY bit in the I2Cx.MSR register gets 0.
5. The Controller does not generate a STOP condition but instead writes another target address to the I2Cx.MSA register and then sets the BURSTRUN and START bit again with a write operation to initiate the repeated START.

13.2.3.6 SCL Clock Low Timeout

The I²C target can extend the transaction by pulling the clock low to slow down communication. The I²C module has a 12-bit programmable counter that is used to track how long the clock has been held low. The upper 8 bits of the count value are software programmable through the I2CTIMEOUT_CTL register. The CNTL value programmed in the I2CTIMEOUT_CTL.TCNTLA register has to be greater than 0x01 to enable the timeout feature . Please note that the low timeout configuration needs to be set only during initialization and not during active state.

The application can program the eight most significant bits of the counter to reflect the acceptable cumulative low period in a transaction. Each count is equal to a timeout period of (1 + TPR) × 12 of functional clocks FCLK where the TPR is the programmable timer period. The Timeout counter A counts for the entire time SCL is held Low continuously. When SCL is high, the Timeout counter A is reloaded with the value in the I2CTIMEOUT_CTL.TCNTLA register bit, and begins counting down from this value at the falling edge of SCL.

The BUSSCLK clock generated for the timeout counter keeps running irrespective of the programmed I²C speed even if SCL is held low on the bus.

The I²C clock low timeout period is calculated as follows:

- Cumulative clock low period = Timeout counter * One timeout period
- One timeout period = BUSSCLK period * (1 + TPR) * 12
- Timeout counter = I2CTIMEOUT_CTL.TCNTLA register (this register contains the upper 8-bits of a 12-bit counter value, the lower 4-bits are set to 0)

As an example, if the BUSSCLK: clock is 20MHz and the I²C module was operating at 100kHz speed, the TPR would be equal to 19. See Section 13.2.1.1 on how TPR is calculated. One timeout period is equal to (1 / 20MHz) × (1 + 19) × 12 or 12µs. Programming the I2CTIMEOUT_CTL.TCNTLA register to 0xDA would translate to the value 0xDA0, because the lower 4-bits are set to 0x0. This would translate to a decimal value of 3488 clocks or a cumulative clock low period of 3488 × 12µs, or 41.856ms at 100kHz.

The TIMEOUTA bit in the I²C Controller Raw Interrupt Status (RIS) register is set when the clock timeout period is reached, allowing the controller to start corrective action to resolve the remote target state. In addition, the BUSBSY bit in the I²C Controller Status MSR register is set. This bit is cleared after I²C goes to idle or during the I²C controller reset. The status of the raw SDA and SCL signals are readable by software through the SDA and SCL bits in the I²C Controller Bus Monitor MBMON register to help determine the state of the remote target.

In the event of a timeout condition, application software must choose how it intends to attempt bus recovery. If a timeout is detected before the end of a transfer (receive or transmit), software should flush the FIFO before initializing the next transfer. The clock low timeout is needed for SMBus and PMBus implementation.

Note

The Controller clock low timeout counter register I2CTIMEOUT_CNT.TCNTA counts for the entire time SCL is held low continuously. When SCL is high, the counter is reloaded with the value in the I2CTIMEOUT_CTL.TCNTLA register, and begins counting down from this value at the falling edge of SCL.

13.2.3.7 Clock Stretching

In controller mode, the clock stretching can be disabled if no targets on the bus support it, allowing the controller to reach the maximum speed on the bus. Otherwise the clock can be slowed by a target keeping the clock low or due to the clock status detection delay within the I²C module.

To ensure compliance to the I²C specification, clock stretching needs to be enabled. Clock stretching is activated when either the RX FIFO full or TX FIFO empty is set. Clock stretching support can be enabled or disabled by configuring the CLKSTRETCH bit in I2Cx.MCR register.

In the target mode, clock stretching is enabled by default and it is signaled by the TREQ and RREQ bits of the I²C target status register I2Cx.SSR.

- When TREQ bit is set, it indicates the I²C controller has been addressed as a target transmitter and is using clock stretching to delay the controller until data has been written to the STXDATA FIFO (Target TX FIFO is empty).
- When RREQ bit is set, it indicates the I²C controller has outstanding receive data from the I2C controller and is using clock stretching to delay the controller until the data has been read from the SRXDATA FIFO (Target RX FIFO is full).

Note

Clock stretching in target mode may be used together with an asynchronous fast clock request to support bringing the device into a suspended low power mode state upon detection of an I2C start bit, enabling the I2C module to support 100kHz (standard mode) or 400kHz (fast mode) operation out of low power modes where the bus clock speed is below the minimum oversampling speed required by the respective mode. Refer to the clock selection and I²C speed section for the minimum frequency requirements. When clock stretching is used together with an asynchronous fast clock request, it is possible for the device to wait in STOP or STANDBY mode when the I²C is idle, and when an I²C bus edge is seen, the fast clock request will requests SYSOSC at base frequency and the bus clock will switch to SYSOSC at base frequency, allowing the I²C module to process the bus transaction and wake the processor if an interrupt is generated.

13.2.3.8 Dual Address

The I²C target interface supports dual address capability for the target. An additional programmable I²C Target Own Address Register I2CX. SOAR is provided and can be matched if enabled. When dual address disabled (I2Cx.SOAR2.OAR2EN=0) , the I²C target provides an ACK on the bus if the address matches the OAR field in the I2Cx.SOAR register. In dual address mode (I2Cx.SOAR2.OAR2EN=1), the I²C target provides an ACK on the bus if either the OAR field in the I2Cx.SOAR register or the OAR2 field in the I2Cx.SOAR2 register is matched.

The OAR2SEL bit in the I2Cx.SSR register indicates if the address that was ACKed is the alternate address or not. When this bit is clear, it indicates either the primary address match or no OAR2 address match.

13.2.3.9 Arbitration

A controller can start a transfer only if the bus is idle. It's possible for two or more controllers to generate a START condition within minimum hold time of the START condition. In these situations, an arbitration scheme takes place on the SDA line, while SCL is high (see Figure 13-8) . The first controller transmitter that generates a logic high is overruled by the opposing controller generating a logic low and the losing controller releases the bus until the bus is idle again.

Arbitration can take place over several bits. Its first stage is a comparison of address bits, and if both controllers are trying to address the same device, arbitration continues on to the comparison of data bits.

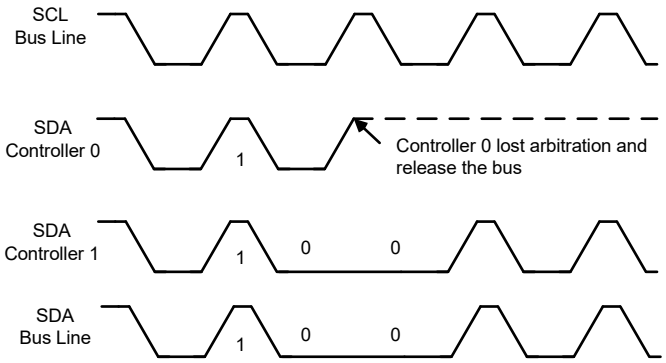


Figure 13-8. Arbitration Procedure Between Two Controller Transmitters

When an arbitration lost is detected the I2Cx.MSR.ARBLST flag is set. It will be reset by the hardware with the next STOP condition detected on the bus. Additionally the ARBLOST flags in CPU_INT.RIS registers are set.

If arbitration is lost when the I²C controller has initiated a transfer, the application should execute the following steps to correctly handle the arbitration loss:

- Flush TX FIFO
- Clear and mask the TX Empty interrupt by the TXEMPTY bit through the IMASK and ICLR register.
- Once the bus is IDLE, the TXFIFO can be filled and enabled, the TXEMPTY bit can be unmasked and a new transaction can be initiated.

13.2.3.10 Multiple Controller Mode

When operating in a multiple controller system, the I2Cx.MCR.MMST bit must be set.

During the arbitration procedure, the clocks from the different controllers must be synchronized. A device that first generates a low period on SCL overrules the other devices, forcing them to start their own low periods. SCL is then held low by the device with the longest low period. The other devices must wait for SCL to be released before starting their high periods

In multiple-controller configuration, the clock synchronization (Figure 13-9) during the arbitration is enabled, the SCL high time counts once the SCL line has been detected high. If not enabled the high time counts as soon as the SCL line has been set high by the I²C controller which allows the I²C to reach the maximum speed by the I2Cx.MTPR register specified speed.

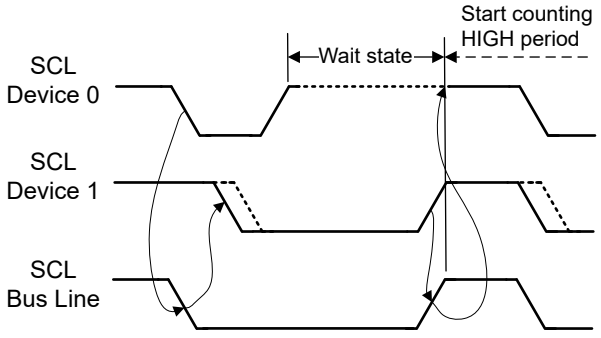


Figure 13-9. Clock Synchronization

13.2.3.11 Glitch Suppression

The I²C module supports glitch suppression on the SCL and SDA lines to meet the 50ns glitch suppression as specified in the I²C specification.

Analog glitch Filter

By default, an analog glitch filter is enabled and configured to suppress spikes with a pulse width up to 50ns. I²C spec advises to suppress noise spikes of less than 50ns. The user can disable this filter by clear the I2Cx.GFCTL.AGFEN bit and the suppression pulse width can also be configured by using I2Cx.GFCTL.AGFEN. The analog glitch filter can only be used to wake up the I²C in low-power mode.

Digital glitch Filter

The DGFSEL bits in the I2Cx.GFCTL register can be programmed to provide glitch suppression on the SCL and SDA lines and assure proper signal values. The glitch suppression value is in terms of the I²C functional clocks. All signals are delayed internally when glitch suppression is nonzero. For example, if DGFSEL bit is set to 0x7, 31 clocks should be added onto the calculation for the expected transaction time, for more information please see register description. Digital glitch filter can't be used to wake up the I2C from low-power mode.

Table 13-3. Glitch Suppression Filter

	Analog Glitch Filter	Digital Glitch Filter
Default	Default enabled with 50ns	Default bypassed
Pulse width of suppressed spikes	Configurable 5ns, 10ns, 25ns, 50ns	Programmable I²C clock cycle 1, 2, 3, 4, 8, 16, 31
Benefits	Available without needing clock	• Programmable length to provide extra filtering • Stable filtering length
Limitation	Variation with temperature, voltage, process	Does not work in low-power mode wakeup when there is no sufficient clock . Enabled only after 3 clock cycles after start of I²C packet

Note

The glitch filter configuration within the register I2Cx.GFCTL should only be modified while the I²C module for controller and target is not enabled.

13.2.3.12 FIFO operation

The receive data register I2Cx.MRXDATA (for controller mode) and I2Cx.SRXDATA (for target mode) are user accessible and contains the current character to be read from the RX FIFO stack. The last received character from the receive shift register will be push to the end of the FIFO Stack.

The transmit data register I2Cx.MTXDATA (for controller mode) and I2Cx.STXDATA (for target mode) are user accessible and holds the data last written data to the TX FIFO. The TX FIFO contains the data waiting to be moved into the transmit shift register and transmitted on SDA.

FIFOs are available for the controller and target receive and transmit. Each FIFO entry has a width of 8 bits and should be accessed in byte mode. Each FIFO has a programmable threshold point (configured by RXTRIG and TXTRIG bits in the I2Cx.MFIFOCTL register for controller mode and I2Cx.SFIFOCTL register for target mode) which indicates when the FIFO service interrupt should be generated. Additionally, a FIFO receive full and transmit empty interrupt can be enabled in the interrupt mask (IMASK) registers for the controller and target.

The content of the FIFO can be flushed with setting TXFLUSH or RXFLUSH bit to 1 in the I2Cx.FIFOCTL registers. When the I²C gets reset the content of the FIFO needs also to be cleared. FIFO clear should only be executed while the I²C is in IDLE mode. Before triggering the flush the FIFO interrupts should be disabled and after flush has completed the interrupt flags needs to be checked.

13.2.3.12.1 Flushing Stale Tx Data in Target Mode

In most use cases, the user does not want to transmit leftover data of the I2C peripheral Tx FIFO from previous frame in the next frame. The device provides a mechanism for the software to choose whether to flush stale data or not.

A status bit SSR.STALE_TXFIFO represents whether the data present inside I2C peripheral TX FIFO is stale or not. A control bit SCTR.TXWAIT_STALE_TXFIFO is used to enable modified empty indication to target logic - indicate empty to I2C peripheral FSM when Tx FIFO is empty OR stale data present in Tx FIFO. The SCTR.TXEMPTY_ON_TREQ control bit allows the RIS.STXEMPTY interrupt to be used to indicate the TREQ condition, the condition when SCL is being stretched waiting for transmit data from the I2C peripheral.

Recommended Sequence

1. When SCTR.TXWAIT_STALE_TXFIFO is set, on STOP, restart, or timeout, I2C peripheral FSM gets an empty indication even though stale data is present inside I2C peripheral Tx FIFO.
2. I2C module doesn't immediately generate an TX FIFO empty interrupt/DMA request. Instead, waits till the controller on the I2C bus attempts to read data from the I2C peripheral, at which point the I2C peripheral clock stretches.
3. The I2C peripheral issues the clock stretch (TREQ) interrupt to the CPU when SCTR.TXEMPTY_ON_TREQ is set.
4. Application software in the ISR checks for a SSR.STALE_TXFIFO flag and flushes the TX FIFO using SFIFOCTL.TXFLUSH, which also clears the status of SSR.STALE_TXFIFO.

13.2.3.13 Loopback mode

The I²C modules can be placed into an internal loopback mode for diagnostic or debug work by setting the LPBK bit in the I²C controller configuration I2Cx.MCR register. In loopback mode, the SDA and SCL signals from the controller part of the I²C are tied to the SDA and SCL signals of the target part of the I²C module to allow internal testing of the device without having to connect the I/Os. The SWUEN bit should be set to 0 for internal loopback to work correctly.

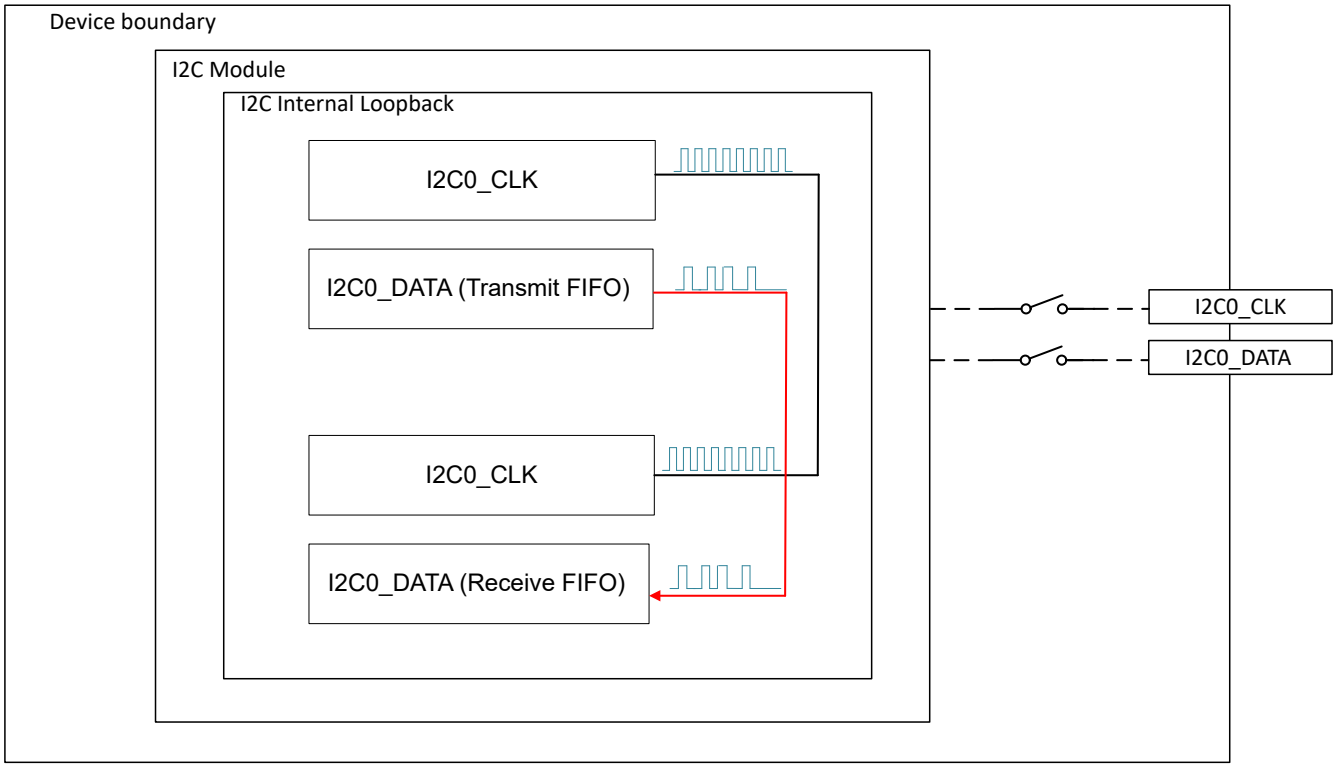


Figure 13-10. I2C Internal Loopback mode

13.2.3.14 Burst Mode

A burst mode is provided for the controller module which allows a sequence of data transfers using the DMA or software to handle the data in the FIFO. The burst mode is enabled by setting the MBLEN bits in the controller

control register I2Cx.MCTR to a value greater then '1'. This sets the number of bytes transferred by a burst. A copy of this value is automatically written to the MBCNT bits in the I²C controller status register I2Cx.MSR to be used as a down-counter during the burst transfer.

The bytes written to the I²C FIFO are transferred to the RX FIFO or TX FIFO depending on whether a transmit or receive is being executed. If data is not acknowledged (NACK) during a BURST and the STOP bit is set in the I2Cx.MCTR register, the transfer terminates. If the STOP bit is not set, software must issue a STOP or repeated START when a NACK interrupt is asserted. In the case of a NACK, the MBCNT bits in the I2Cx.MSR register can be used to determine the amount of data that was transferred prior to the burst termination. If the address is not acknowledged (NACK) during a transfer, then a STOP is issued.

13.2.3.15 DMA Operation

The I²C provides an interface to the DMA controller with two channels. The DMA operation of the I²C is enabled through the I²C event register and DMA peripheral registers. When the DMA functionality is enabled, the I²C asserts a DMA request on the selected channel when the associated FIFO can transfer or receive data.

For more information about I²C event and DMA, please refer to Interrupt and Events Support and DMA Trigger Publisher sections.

Note

Per DMA Channel only one event source should be enabled by the IMASK register at the same time. The DMA transaction descriptor needs to match for the selected trigger with a correct setup for either Controller or Target and RX or TX configuration. The DMA trigger should only be changed when no I²C transfer is ongoing and a previous triggered DMA transfer has been finished. If this cannot be ensured the I²C and DMA channel should be disabled first.

13.2.3.16 Low-Power Operation

I²C supports operation in low-power modes.

Supported power mode for controller mode:

- Controller mode with 100kHz speed can operate in RUN, SLEEP, STOP power mode
- Controller mode with 400kHz speed can operate in RUN, SLEEP power mode
- Controller mode with 1MHz speed can operate in RUN, SLEEP power mode

In low-power mode, the I²C interface automatically request the clock selected in the CLKSEL control register after detecting a start condition. The clock request is released after detection of the STOP condition.

Supported power mode and wakeup mode for target mode:

- Target mode with 100kHz speed can operate in RUN, SLEEP, STOP power mode
- Target mode with 100kHz speed can't operate in STANDBY mode because the maximum bus clock in STANDBY mode is 32kHz and support for 100kHz speed requires a minimum 400kHz clock. However, in STANDBY mode the I2C target can still wakeup from the start bit and perform the Asynchronous Fast Clock Requests to temporarily get a 24MHz/32MHz clock to receive the data until the FIFO or address match interrupt wakes up the CPU.
- Target mode with 400kHz speed in RUN, SLEEP power mode
- Target mode with 1MHz speed in RUN, SLEEP power mode

13.2.4 Protocol Descriptions

13.2.4.1 I²C Controller Mode

As showing in the function block diagram section, I²C peripheral has a set of specific controller registers to configure the operation when the module is configured as an I²C target mode.

13.2.4.1.1 Controller Configuration

The I²C Controller Control register I2Cx.MCTR and I²C Controller Target Address register I2Cx.MSA are used for controlling controller transmit and receive modes. The following settings can be modified to control the different transactions.

- Length indicates the number of bytes for the transaction and is configured by I2Cx.MCTR.MBLEN bit
- Direction (transmit or receive) is configured by I2Cx.MSA.DIR bit
- ACK generation is configured by I2Cx.MCTR.MBLEN bit
- STOP condition generation is configured by I2Cx.MCTR.STOP bit
- START or repeated START condition generation is configured by I2Cx.MCTR.START bit
- RUN enable the controller transaction and is configured by I2Cx.MCTR.BURSTRUN bit

Controller send data transactions

Table 13-4. Start Transmit From Idle Mode

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	0	x	0 or 1	1	1	START+ADDR+R/W+ DATA*n+(STOP)	Sending of STOP depends on STOP bit

Table 13-5. Continue Transmit When Last Transmission Finished Without stop

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	0	x	0 or 1	0	1	DATA*n+ (ACK/NACK)+ (STOP)	Sending of STOP depends on STOP bit

If there is a NACK response from the target, the controller will automatically send out a stop to finish the transmit. The Controller will be unable to send a RESTART after ADDR or DATA NACK.

Controller receive data transactions

Table 13-6. Start Receive From Idle Mode

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	1	0 or 1	0 or 1	1	1	START+ADDR +R/ W+DATA *n +(ACK/NACK) +(STOP)	The last data ACK or NACK depend on ACK bit; additional sending of STOP depends on STOP bit

Table 13-7. Continue Receive When Last Receive Finished Without STOP or NACK

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	1	0 or 1	0 or 1	0	1	DATA*n+ (ACK/NACK) + (STOP)	The last data followed by ACK or NACK depend on ACK bit; additional sending of STOP depends on STOP bit

This configuration is not allowed if last transaction ended with NACK, as NACK can only be followed by STOP or RESTART. The ACK and STOP bits should not be set to 1 at the same time, as the target needs to be informed to release bus lines before sending out STOP.

Controller repeated start transactions

If last transmit or receive finished without stop, a repeat start can be generated to initiate a new transaction

Table 13-8. Repeated Start Transmit

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	0	0 or 1	0 or 1	1	1	Restart+ADDR +R/ W+DATA*n +(STOP)	Additional sending of STOP depends on STOP bit

If there is a NACK response from the target, the controller will automatically send out a stop to finish the transmit. The Controller will be unable to send a RESTART after ADDR or DATA NACK.

Table 13-9. Repeated Start Receive

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	1	0 or 1	0 or 1	1	1	Restart+ADDR +R/ W+DATA*n +(ACK/NACK) +(STOP)	The last data followed by ACK or NACK depend on ACK bit; additional sending of STOP depends on STOP bit

The ACK and STOP bits should not be set to 1 at the same time, as the target needs to be informed to release bus lines before sending out STOP.

Controller send STOP only transaction

Table 13-10. Send STOP only

Length	Direction	ACK	STOP	START	RUN	Format	Comments
n (n>0)	X	X	1	0	1	STOP	STOP command

It is only allowed to send after previous transaction success finished, and STOP can't be sent without a NACK to the target if controller is currently in receive mode

Controller Quick Command transaction

The Quick command could only be sent at the transaction beginning, not following other transactions (without stop) or repeat start.

Table 13-11. Controller quick command

Length	Direction	ACK	STOP	START	RUN	Format	Comments
0	0/1	X	1	1	1	START+ADDR +R/ W+STOP	Quick command

13.2.4.1.2 Controller Mode Operation

I²C Controller Initialization

1. Configure SDA and SCL pin functions and select as input by using the IOMUX registers.
2. Reset the peripheral using I2Cx.RSTCTL register
3. Enable the power to peripheral using I2Cx.PWREN register
4. Select and configure the I²C clock using the CLKCTL and CLKDIV registers.
5. Set the desired SCL clock speed of by writing the TPR bit in I2Cx.MTPR register with the correct value. For more information about how to calculate TPR value refer to Clock Control section. For example, with 20MHz I²C clock to achieve 100 kbps SCL clock, TPR value will be 19 (0x13) so we can write the I2Cx.MTPR register with the value of 0x13.
6. Specify the target address and mode (transmit or receive) for the next operation by writing the I2Cx.MSA register. For example, if the target address is 0x3B (MSA.SADDR[7:1] = 0x3B) and we want to transmit (MSA.DIR[0] = 0x0) data then we can write the I2Cx.MSA register with a value of 0x76.
7. If controller is transmitting data, user can place data (byte) to be transmitted in the data register by writing the I2Cx.MTXDATA register with the desired data.

8. Configure the controller transmit or receive mode by writing the I2Cx.MCTR register. For more information on how to configure I2Cx.MCTR register for different mode please see Controller Configuration section above.
9. Enable desired interrupts and/or DMA event by using CPU_INT.IMASK register.

I²C Controller Status

User can read the I2Cx.MSR register to check the current state of the I²C controller.

Table 13-12. Controller Status Register

Bit Field	Description
BUSY	I2C controller busy. The BUSY bit is set during an ongoing transaction.
ERR	I2C error. The error can be from the target address not being acknowledged or the transmit data not being acknowledged.
ADRACK	Acknowledge address. This bit is set if the transmitted address was not acknowledged.
DATACK	Acknowledge data. This bit is set if the transmitted data was not acknowledged.
ARBLST	Arbitration lost. This bit is set if controller lost arbitration.
IDLE	I2C bus idle.
BUSBSY	I2C bus busy. The bit changes based on the START and STOP conditions, set it bus is busy.
CLKTO	Clock timeout error. This bit is set if the clock timeout error has occurred.
MBCNT	I2C controller transaction count. This field contains the current countdown value of the transaction.

I²C Controller Receiver Mode

For controller to start receive data out of the idle mode, user needs to set the START bit in I2Cx.MCTR register to generate the start condition. Then the controller automatically sends the START condition followed by the target address as soon as it detects that the bus is free. All the process below should be followed.

I2Cx.MSA.DIR is set to 1 to enable receive mode, I2Cx.MCTR.START is set to generate start condition, I2Cx.MBLEN can be programmed to indicate the number of bytes (n) for the receive operation. I2Cx.MCTR.ACK and I2Cx.MCTR.STOP bit can be set or clear based on user configuration. I2Cx.MCTR.BURSTRUN is set to start the operation. The packet format is START+ADDR+R+DATA*n +(ACK/NACK) + (STOP). The last data ACK/NACK depend on ACK bit, additional sending of STOP depends on STOP bit.

After last byte is received, the MRXDONE (0x01) interrupt in CPU_INT.IIDX register is set to indicate that controller receive transaction is completed. User can use the MRXFIFOTRG (0x03) interrupt in CPU_INT.IIDX register to read the data from the receive FIFO. This interrupt will trigger when controller RX FIFO contains >= defined bytes, the trigger level can be defined by using RXTRIG bit in I2Cx.MFIFOCTL register. The flow chart of controller receiver mode is shown in Figure 13-11.

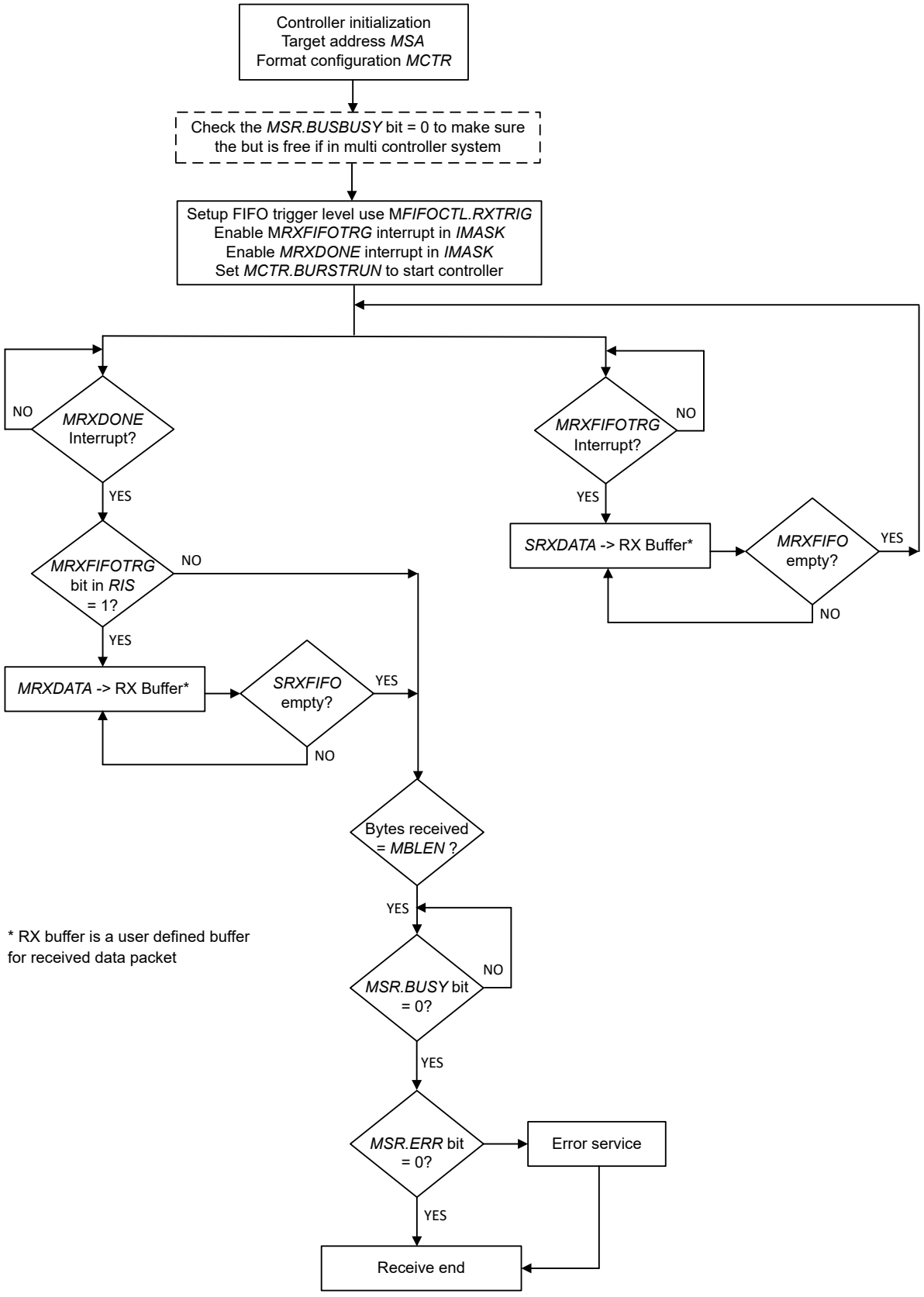


Figure 13-11. Controller Receiver Mode

I²C Controller transmitter Mode

For controller, to start transmit data out of the idle mode, user need to set the START bit in I2Cx.MCTR register to generate the start condition. Then the controller automatically sends the START condition followed by the target address as soon as it detects that the bus is free. The data written into the MTXFIFO is transmitted if arbitration is not lost during transmission of the target address. All the process below should be followed.

I2Cx.MSA.DIR is cleared to enable transmit mode, I2Cx.MCTR.START is set to generate start condition, I2Cx.MBLEN can be programmed to indicate the number of bytes (n) for the transmit operation. I2Cx.MCTR.STOP bit can be set or clear based on user configuration. I2Cx.MCTR.BURSTRUN is set to start the operation. The packet format is START+ADDR+W+DATA*n + (STOP), sending of STOP depends on STOP bit.

After last byte is transmitted, the MTXDONE (0x02) interrupt in CPU_INT.IIDX register is set to indicate that controller transmit transaction is completed. User can also set use the MTXEMPTY (0x06) interrupt in CPU_INT.IIDX register to see if the MTXFIFO is empty and ready to load more data. This interrupt will trigger if all data in the transmit FIFO have been shifted out. The flow chart of controller receiver mode is shown in Figure 13-12.

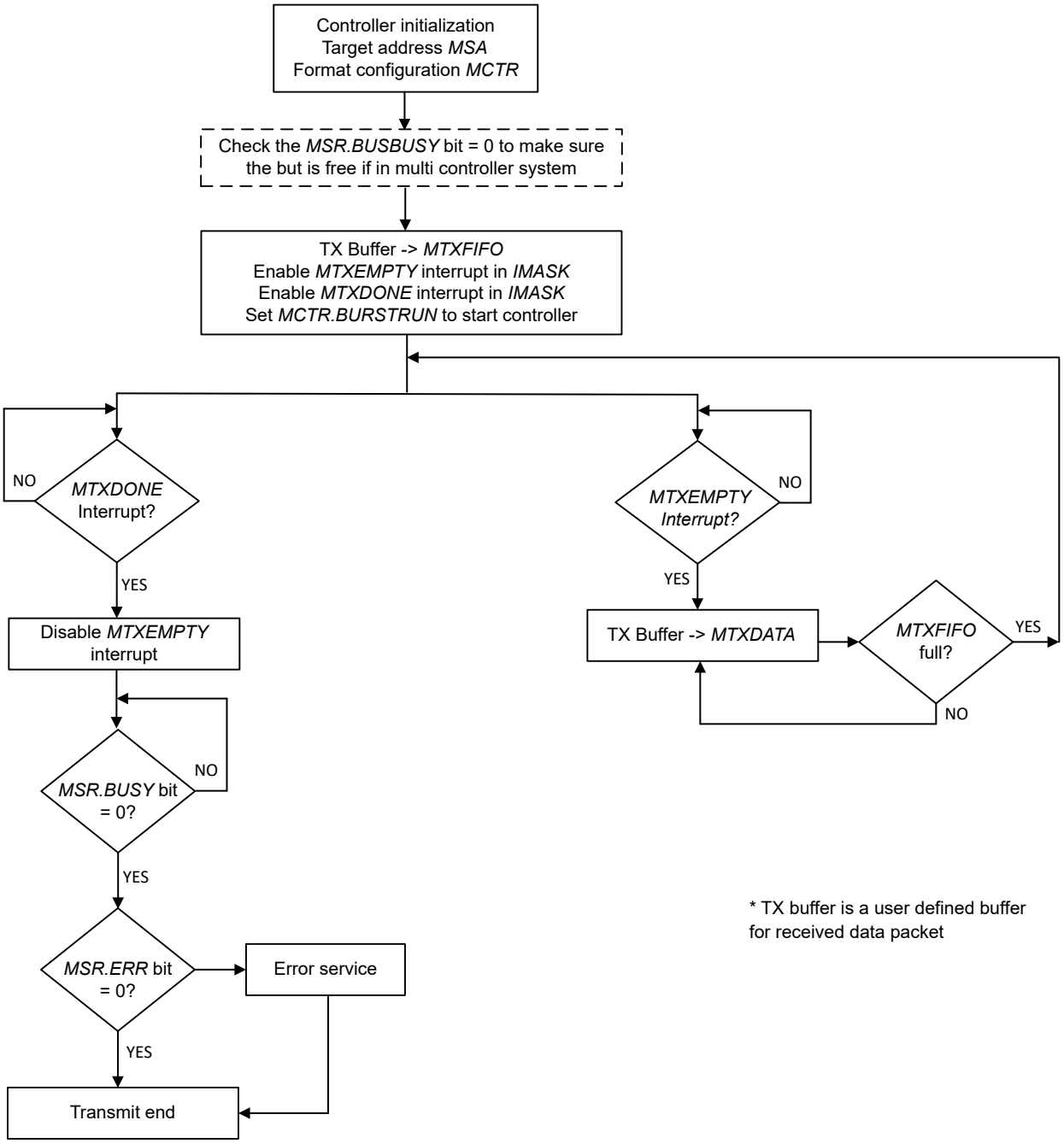


Figure 13-12. Controller Transmitter Mode

13.2.4.1.3 Read On TX Empty

A typical I2C transaction must first write the target to set the register number and then do a repeated start + read to read the data value. However, the RD_ON_EMPTY flag enables two I2C transactions with a single setup of the I2C controller. Software can set up both the write and read together, with the limitation being that the "write" phase cannot send more data than fits in the TX FIFO. This is an optimization to minimize interrupt processing requirements. The high-level set up is:

1. Set MSA to target address with RECEIVE = 1 (RD_ON_TXEMPTY must have I2C_MSR_DIR_RECEIVE set)
2. Set MCTR to initiate the transaction, in this use case write:

- MBLEN= size for READ phase
- RD_ON_TXEMPTY
- ACK_DISABLE (NACK+STOP after read)
- STOP_ENABLE
- START_ENABLE
- BURST_ENABLE

The expected behavior with this set up is that the controller does a START, transmits all TXFIFO data, and when TXFIFO is empty automatically does a repeated START, reads the MBLEN bytes, then STOP.

13.2.4.2 I²C Target Mode

13.2.4.2.1 Target Mode Operation

I²C Target Initialization

1. Configure SDA and SCL pin functions and select as input by using the IOMUX registers.
2. Reset the peripheral using I2Cx.RSTCTL register
3. Enable the power to peripheral using I2Cx.PWREN register
4. Select and configure the I²C clock using the CLKCTL and CLKDIV registers.
5. Configure at least one target address by writing the 7-bit address to I2Cx.SOAR register. The additional target address can be enabled and configured by using I2Cx.SOAR2 register.
6. Enable desired interrupts and/or DMA event by using CPU_INT.IMASK register.
7. The general call response can be enabled by setting the GENCALL bit in I2Cx.SCTR register.
8. Enable the I²C target mode by setting the ACTIVE bit in I2Cx.SCTR register.

I²C Target Status

User can read the I2Cx.SSR register to check the current state of the I²C target.

Table 13-13. Target Status Register

Bit Field	Description
RREQ	This bit is set if the I2C controller has outstanding receive data from the I2C controller and is using clock stretching to delay the controller until the data has been read from the SRXDATA FIFO (Target RX FIFO is full)
TREQ	This bit is set if the I2C controller has been addressed as a target transmitter and is using clock stretching to delay the controller until data has been written to the STXDATA FIFO (Target TX FIFO is empty).
OAR2SEL	This bit is set if SOAR2.OAR2 address matched and ACKed by the target.
QCMDST	Quick command status value. This bit is 0 if the last transaction was a normal transaction or a transaction has not occurred. This bit is set if the last transaction was a quick command transaction
QCMDRW	Quick command read / write status. This bit only has meaning when the QCMDST bit is set. This bit is 0 if quick command was a write. This bit is set if quick command was a read.

I²C Target Receiver Mode

Target receiver mode is entered when the target address transmitted by the controller matches its own address and a cleared R/W bit is received. In target receiver mode, serial data bits received on SDA are shifted in with the clock pulses that are generated by the controller device. The target device does not generate the clock, but it can hold SCL low if intervention of the CPU is required after a byte has been received.

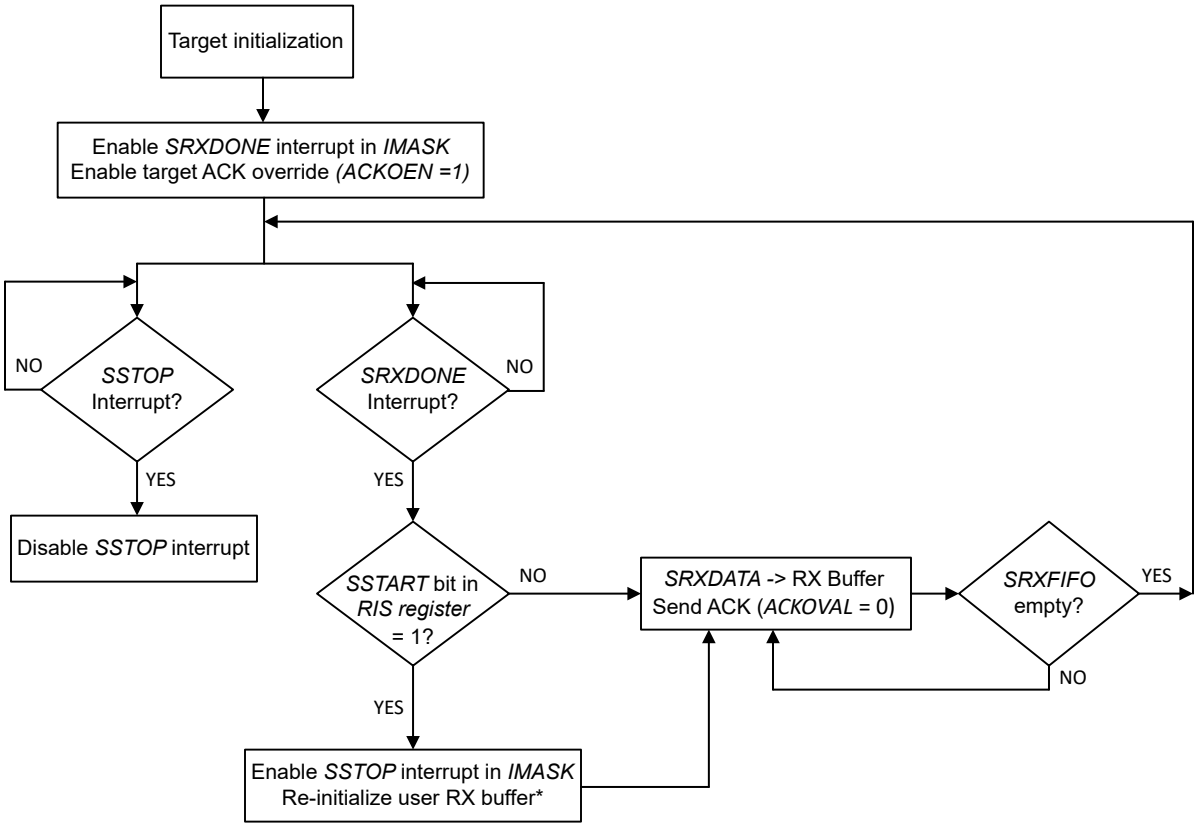
After the a data byte is received, the SRXDONE (0x11) interrupt in CPU_INT.IIDX register is set to indicate that a byte has been received. The I²C module automatically acknowledges the received data or user can choice to manually send acknowledge after each byte received by configuring the I2Cx.SACKCTL register.

When the controller generates a START condition, the SSTART (0x17) interrupt in CPU_INT.IIDX register is set. When the controller generates a STOP condition, the SSTOP (0x18) interrupt in CPU_INT.IIDX register is set.

User can also set use the SRXFIFOTRG (0x13) interrupt in CPU_INT.IIDX register to read the data from the receive FIFO. This interrupt will trigger when receive FIFO contains >= defined bytes, the trigger level can be defined by using RXTRIG bit in I2Cx.SFIFOCTL register.

The SRXDONE approach could be used if target wants to slow down communication to evaluate reception of every byte, while the SRXFIFOTRG approach could be used to maximize throughput and avoid clock stretching.

The flow chart of using SRXDONE and SRXFIFOTRG interrupt to read the receive data are shown in Figure 13-13 and Figure 13-14 respectively.



* RX buffer is a user defined buffer for received data packet

Figure 13-13. Target Receiver Mode using SRXDONE and ACK override

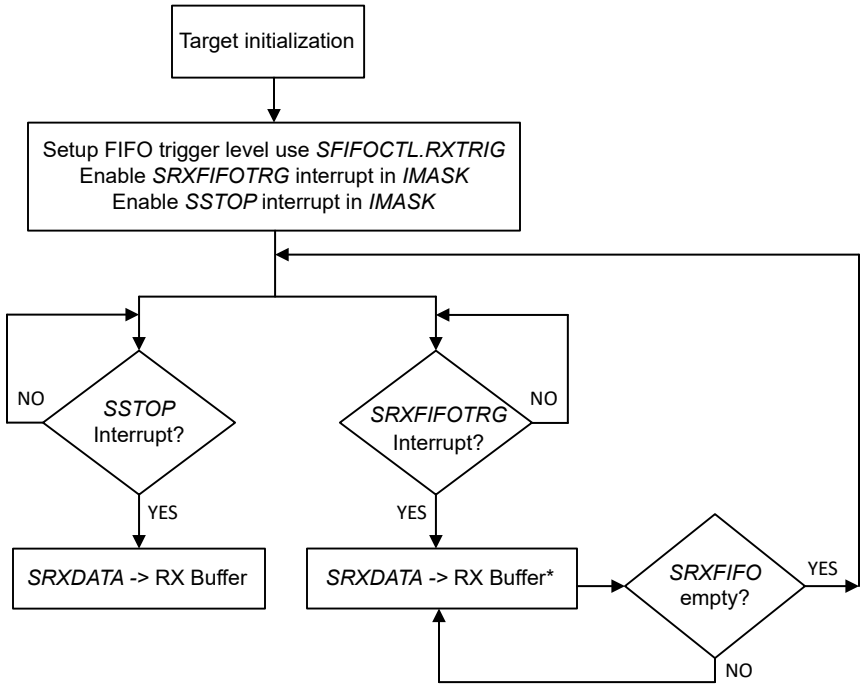


Figure 13-14. Target Receiver Mode using SRXFIFOTRG and automatic ACK

I²C Target Transmitter Mode

Target transmitter mode is entered when the target address transmitted by the controller is identical to its own address with a set R/W bit. The target transmitter shifts the serial data out on SDA with the clock pulses that are generated by the controller device. The target device does not generate the clock, but it can hold SCL low if intervention of the CPU is required after a byte has been transmitted.

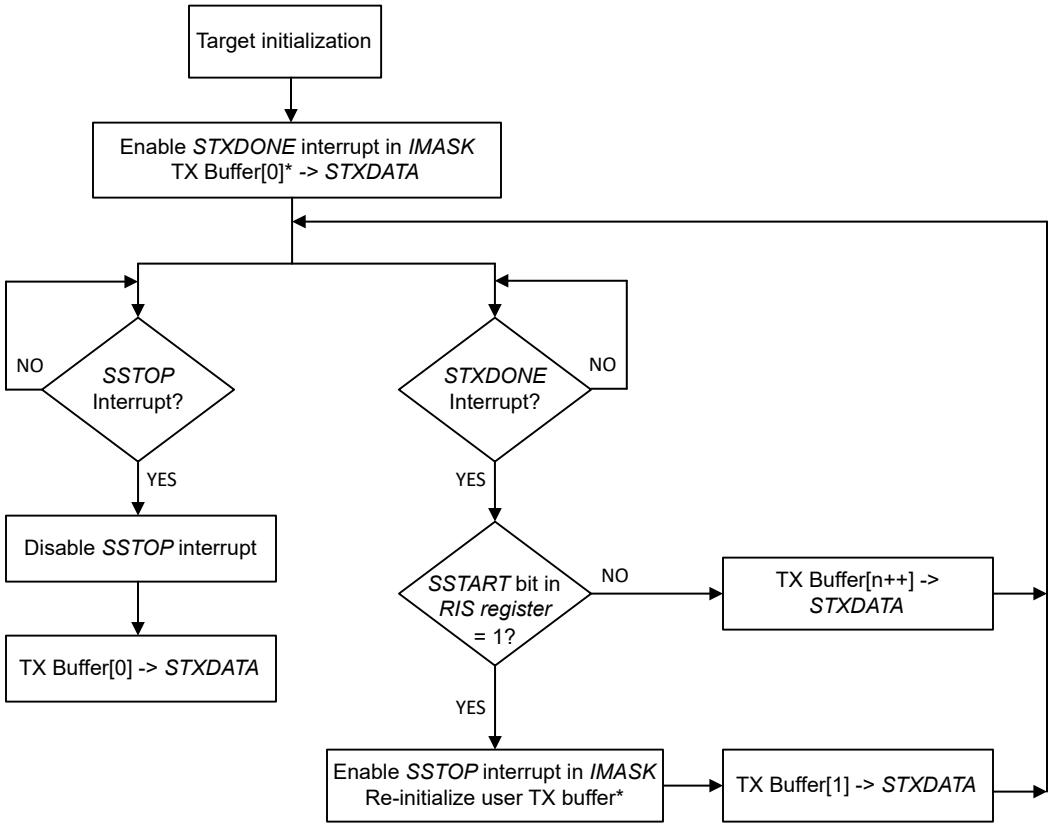
After a data byte is transmitted, the STXDONE (0x12) interrupt in CPU_INT.IIDX register is set to indicate that a byte has been transmitted.

When the controller generates a START condition, the SSTART (0x17) interrupt in CPU_INT.IIDX register is set. When the controller generates a STOP condition, the SSTOP (0x18) interrupt in CPU_INT.IIDX register is set.

User can also set use the STXFIFOTRG (0x14) interrupt in CPU_INT.IIDX register to load the data to the transmit FIFO. This interrupt will trigger when transmit FIFO contains <= defined bytes, the trigger level can be defined by using TXTRIG bit in I2Cx.SFIFOCTL register.

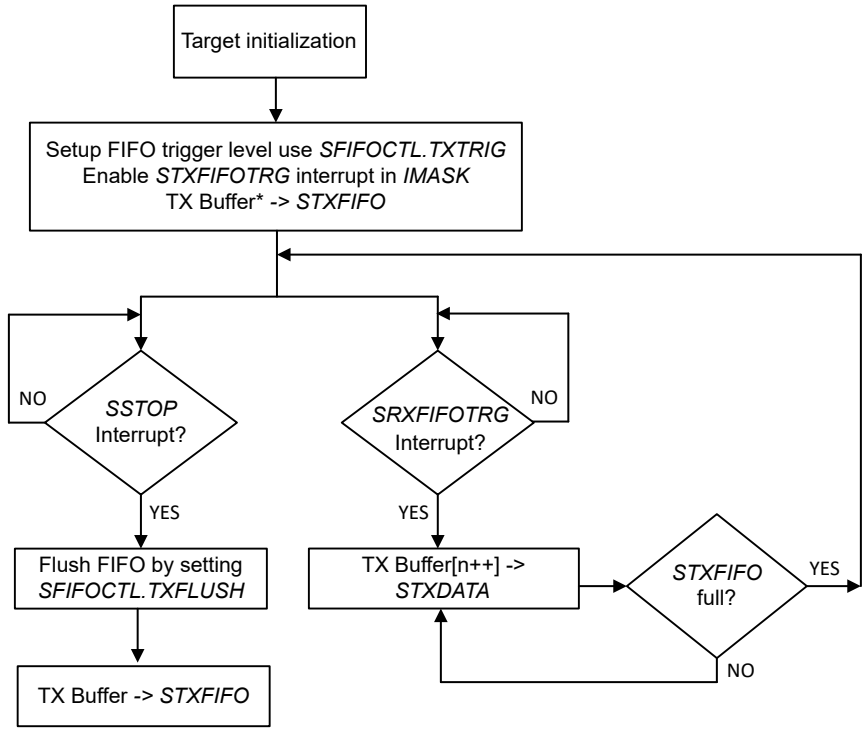
The STXDONE approach could be used if target wants to slow down communication to evaluate reception of every byte, while the STXFIFOTRG approach could be used to maximize throughput and avoid clock stretching.

The flow chat of using STXDONE and STXFIFOTRG interrupt to transmit data are shown in Figure 13-15 and Figure 13-16.



* TX buffer is a user defined buffer for transmit data packet

Figure 13-15. Target Transmitter Mode using STXDONE



* TX buffer is a user defined buffer for received data packet

Figure 13-16. Target Transmitter Mode using STXFIFOTRG

13.2.5 Reset Considerations

Software Reset Considerations

A Software reset can be executed by setting the RESETASSERT bit together with the KEY bit in the I2Cx.RSTCTL register. We recommend issues a reset only after terminating a transaction.

Hardware Reset Considerations

A hardware reset also initializes the IO configuration. This sets the IOs to a high-impedance state and with the external pullup resistors for I²C the lines pulled high.

Table 13-14 shows the behavior of status bits when Controller or Target gets disabled.

Table 13-14. Status Bits When Controller is Disabled

Register	Bit	Behavior When Controller Disabled	Behavior When Target Disabled	Behavior After Controller/Target Enabled
I2Cx.MSR	Busy	Reset State	Don't care	updates Start condition sending
	ERR	Reset State	Don't care	updates on next event detected
	ADRACK	Reset State	Don't care	updates on next event detected
	DATACK	Reset State	Don't care	updates on next event detected
	ARBLST	Reset State	Don't care	updates on next event detected
	IDLE	Reset State	Don't care	updates on next event detected
	BUSBSY	Reset State	Don't care	updates on next Start detected on bus (or SDA or SCL is low)
	CLKTO	Reset State	Don't care	updates on next event detected
	MBCNT	Reset State	Don't care	updates on next event detected
I2Cx.MCLKOCNT	CLKCNT	Reset State	Don't care	updates with Controller enable when SCL is high

Table 13-14. Status Bits When Controller is Disabled (continued)

Register	Bit	Behavior When Controller Disabled	Behavior When Target Disabled	Behavior After Controller/Target Enabled
I2Cx.MBMON	SCL	Reset State	Don't care	updates with Controller enable
	SDA	Reset State	Don't care	updates with Controller enable

Table 13-15. Status Bits When Target is Disabled

Register	Bit	Behavior When Controller Disabled	Behavior When Target Disabled	Behavior After Controller/Target Enabled
I2Cx.SSR	RREQ	Don't care	Reset State	updates on next event detected
	TREQ	Don't care	Reset State	updates on next event detected
	OAR2SEL	Don't care	Reset State	updates on next event detected
	QCMDST	Don't care	Reset State	updates on next event detected
	QCMDRW	Don't care	Reset State	updates on next event detected
I2Cx.SFIFOSR	RXFIFOCNT	Don't care	Unchanged	updates on FIFO access
	TXFIFOCNT	Don't care	Unchanged	updates on FIFO access

13.2.6 Initialization

Please see Section 13.2.4.1.2 for controller initialization and Section 13.2.4.2.1 for target initialization.

Note

The configuration registers of the I2C module (including the clock configuration, mode configuration, and timeout count) must not be modified by application software when an I2C data transfer is in progress.

13.2.7 Interrupt and Events Support

The I²C module contains three event publishers and no event subscribers. One event publisher (CPU_INT) manages I²C interrupt requests (IRQs) to the CPU subsystem through a static event route. The second and third event publishers (DMA_TRIG1, DMA_TRIG0) are used to setup the trigger signaling for the DMA through DMA event route.

The I²C events are summarized in Table 13-16.

Table 13-16. I2C Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt	Publisher	I²C	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from I²C to CPU
DMA trigger	Publisher	I²C	DMA	DMA event route	DMA_TRIG1 registers	Fixed interrupt route from I²C to DMA
DMA trigger	Publisher	I²C	DMA	DMA event route	DMA_TRIG0 registers	Fixed interrupt route from I²C to DMA

13.2.7.1 CPU Interrupt Event Publisher (CPU_INT)

The I²C module provides 24 interrupt sources which can be configured to source a CPU interrupt event. In order of decreasing interrupt priority, the CPU interrupt events from the I²C are:

Table 13-17. I²C CPU Interrupt Event Conditions for Controller (CPU_INT)

IIDX STAT	Name	Description
0x01	MRXDONE	Controller receive transaction completed interrupt
0x02	MTXDONE	Controller transmit transaction completed interrupt
0x03	MRXFIFOTRG	Controller receive FIFO trigger. Trigger when RX FIFO contains >= defined bytes
0x04	MTXFIFOTRG	Controller transmit FIFO trigger. Trigger when Transmit FIFO contains <= defined bytes
0x05	MRXFIFOFULL	Controller RXFIFO full event. This interrupt is set if an RX FIFO is full.
0x06	MTXEMPTY	Controller transmit FIFO empty interrupt. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode.
0x07	MCLKTO	Controller clock timeout interrupt
0x08	MNACK	Address/Data NACK interrupt
0x09	MSTART	Controller START detection interrupt
0x0A	MSTOP	Controller STOP detection interrupt
0x0B	MARBLOST	Controller arbitration lost interrupt
0x0C	MDMA_DONE_TX	Controller DMA TX done signal (see next section for more detail)
0x0D	MDMA_DONE_RX	Controller DMA RX done signal (see next section for more detail)

Table 13-18. I²C CPU Interrupt Event Conditions for Target (CPU_INT)

IIDX STAT	Name	Description
0x11	SRXDONE	Target receive transaction completed interrupt
0x12	STXDONE	Target transmit transaction completed interrupt
0x13	SRXFIFOTRG	Target receive FIFO trigger. It will trigger when receive FIFO contains >= defined bytes
0x14	STXFIFOTRG	Target transmit FIFO trigger. It will trigger when transmit FIFO contains <= defined bytes
0x15	SXFIFOFULL	Target RXFIFO full event. This interrupt is set if an RX FIFO is full.
0x16	STXEMPTY	Target transmit FIFO empty interrupt. This interrupt is set if all data in the Target Transmit FIFO have been shifted out and the transmit goes into idle mode.
0x17	SSTART	Target START detection interrupt
0x18	SSTOP	Target STOP detection interrupt
0x19	SGENCALL	General call interrupt
0x1A	SDMA_DONE_TX	Target DMA TX done signal (see next section for more detail)
0x1B	SDMA_DONE_RX	Target DMA RX done signal (see next section for more detail)

The CPU interrupt event configuration is managed with the CPU_INT event management registers. See Section 6.2.5 for guidance on configuring the Event registers for CPU interrupts.

13.2.7.2 DMA Trigger Publisher (DMA_TRIG1, DMA_TRIG0)

DMA_TRIG1 and DMA_TRIG0 registers are used to setup the trigger signaling for the DMA. This can be setup in a flexible way to trigger the DMA for Controller or Target and receive or transmit events with the following four trigger conditions:

Table 13-19. I²C DMA Trigger Condition (DMA_TRIG1 and DMA_TRIG0)

IIDX STAT	Name	Description
0x01	MRXFIFOTRG	Controller receive FIFO trigger. Trigger when RX FIFO contains >= defined bytes
0x02	MTXFIFOTRG	Controller transmit FIFO trigger. Trigger when Transmit FIFO contains <= defined bytes
0x03	SRXFIFOTRG	Target receive FIFO trigger. Trigger when RX FIFO contains >= defined bytes
0x04	STXFIFOTRG	Target transmit FIFO trigger. Trigger when Transmit FIFO contains <= defined bytes

The DMA trigger event configuration is managed with the DMA_TRIG1 and DMA_TRIG0 event management registers. See Section 6.2.5 for guidance on configuring the Event registers and Section 6.1.3.2 for on how DMA trigger event works. DMA_TRIG1 and DMA_TRIG0 are two event management registers that correspond to two DMA channels.

As shown in Figure 13-17, each DMA channel can be triggered by any of the conditions listed in Table 13-19 and it can generate either the controller DMA done signal or target DMA done signal.

For example, the user can configure the DMA_TRIG1 trigger using MTXFIFOTRG and the DMA_TRIG0 trigger using SRXFIFOTRG. When the Channel 1 DMA status changes to done, the MDMA_DONE_TX and MDMA_DONE_RX interrupts will set, and when the Channel 2 DMA status change to done, the SDMA_DONE_TX and SDMA_DONE_RX interrupts will set.

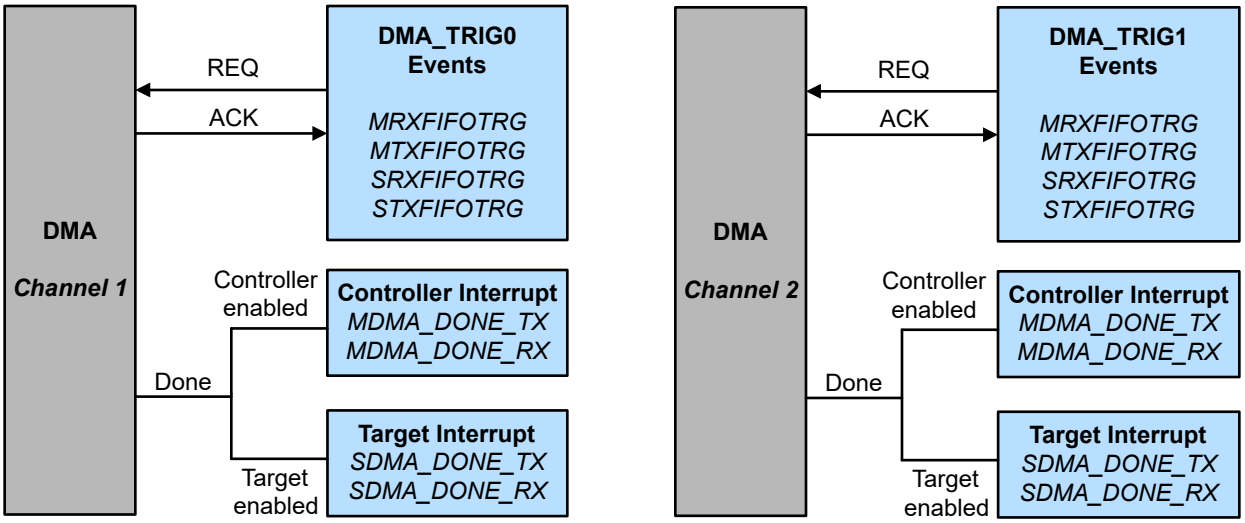


Figure 13-17. I²C DMA Trigger and Status

13.2.8 Emulation Modes

The module behavior while the device is in debug mode is controlled by the FREE and SOFT bits in PDBGCTL register.

When the device is in debug mode and set into halt mode below behavior can be configured.

Table 13-20. Debug Mode Peripheral Behavior

PDBGCTL.FREE	PDBGCTL.SOFT	Function
1	x	Modules continues operation
0	0	Module stops immediately
0	1	Module stops after the next transfer has been finished

13.3 I2C Registers

Table 13-21 lists the memory-mapped registers for the I2C registers. All register offset addresses not listed in Table 13-21 should be considered as reserved locations and the register contents should not be modified.

Table 13-21. I2C Registers

Offset	Acronym	Register Name	Section
800h	PWREN	Power enable	Go
804h	RSTCTL	Reset Control	Go
808h	CLKCFG	Peripheral Clock Configuration Register	Go
814h	STAT	Status Register	Go
1000h	CLKDIV	Clock Divider	Go
1004h	CLKSEL	Clock Select for Ultra Low Power peripherals	Go
1018h	PDBGCTL	Peripheral Debug Control	Go
1020h	IIDX	Interrupt index	Go
1028h	IMASK	Interrupt mask	Go
1030h	RIS	Raw interrupt status	Go
1038h	MIS	Masked interrupt status	Go
1040h	ISET	Interrupt set	Go
1048h	ICLR	Interrupt clear	Go
1050h	IIDX	Interrupt index	Go
1058h	IMASK	Interrupt mask	Go
1060h	RIS	Raw interrupt status	Go
1068h	MIS	Masked interrupt status	Go
1070h	ISET	Interrupt set	Go
1078h	ICLR	Interrupt clear	Go
1080h	IIDX	Interrupt index	Go
1088h	IMASK	Interrupt mask	Go
1090h	RIS	Raw interrupt status	Go
1098h	MIS	Masked interrupt status	Go
10A0h	ISET	Interrupt set	Go
10A8h	ICLR	Interrupt clear	Go
10E0h	EVT_MODE	Event Mode	Go
10E4h	INTCTL	Interrupt control register	Go
10FCh	DESC	Module Description	Go
1200h	GFCTL	I2C Glitch Filter Control	Go
1204h	TIMEOUT_CTL	I2C Timeout Count Control Register	Go
1208h	TIMEOUT_CNT	I2C Timeout Count Register	Go
1210h	MSA	I2C Controller Target Address Register	Go
1214h	MCTR	I2C Controller Control Register	Go
1218h	MSR	I2C Controller Status Register	Go
121Ch	MRXDATA	I2C Controller RXData	Go
1220h	MTXDATA	I2C Controller TXData	Go
1224h	MTPR	I2C Controller Timer Period	Go
1228h	MCR	I2C Controller Configuration	Go
1234h	MBMON	I2C Controller Bus Monitor	Go
1238h	MFIFOCTL	I2C Controller FIFO Control	Go
123Ch	MFIFOSR	I2C Controller FIFO Status Register	Go

Table 13-21. I2C Registers (continued)

Offset	Acronym	Register Name	Section
1250h	SOAR	I2C Target Own Address	Go
1254h	SOAR2	I2C Target Own Address 2	Go
1258h	SCTR	I2C Target Control Register	Go
125Ch	SSR	I2C Target Status Register	Go
1260h	SRXDATA	I2C Target RXData	Go
1264h	STXDATA	I2C Target TXData	Go
126Ch	SFIFOCTL	I2C Target FIFO Control	Go
1270h	SFIFOSR	I2C Target FIFO Status Register	Go

Complex bit access types are encoded to fit into small table cells. Table 13-22 shows the codes that are used for access types in this section.

Table 13-22. I2C Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

13.3.1 PWREN Register (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 13-18 and described in Table 13-23.

Return to the Summary Table.

Register to control the power state

Figure 13-18. PWREN Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/WK-0h

Table 13-23. PWREN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R	0h	
0	ENABLE	R/WK	0h	Enable the power KEY must be set to 26h to write to this bit. 0h = Disable Power 1h = Enable Power

13.3.2 RSTCTL Register (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 13-19 and described in Table 13-24.

Return to the Summary Table.

Register to control reset assertion and de-assertion

Figure 13-19. RSTCTL Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
R-0h						WK-0h	WK-0h

Table 13-24. RSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	R	0h	
1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Clear reset sticky bit
0	RESETASSERT	WK	0h	Assert reset to the peripheral KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Assert reset

13.3.3 CLKCFG Register (Offset = 808h) [Reset = 00000000h]

CLKCFG is shown in Figure 13-20 and described in Table 13-25.

Return to the Summary Table.

Peripheral Clock Configuration Register

Figure 13-20. CLKCFG Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							BLOCKASYNC
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 13-25. CLKCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to Allow State Change -- 0xA9 A9h = key value to allow change field of GPRCM
23-9	RESERVED	R	0h	
8	BLOCKASYNC	R/W	0h	Async Clock Request is blocked from starting SYSOSC or forcing bus clock to 32MHz 0h = Not block async clock request 1h = Block async clock request
7-0	RESERVED	R	0h	

13.3.4 STAT Register (Offset = 814h) [Reset = 00000000h]

STAT is shown in Figure 13-21 and described in Table 13-26.

Return to the Summary Table.

peripheral enable and reset status

Figure 13-21. STAT Register

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

Table 13-26. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	0x0	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

13.3.5 CLKDIV Register (Offset = 1000h) [Reset = 00000000h]

CLKDIV is shown in Figure 13-22 and described in Table 13-27.

Return to the Summary Table.

This register is used to specify module-specific divide ratio of the functional clock

Figure 13-22. CLKDIV Register

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

Table 13-27. CLKDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	
2-0	RATIO	R/W	0h	Selects divide ratio of module clock 0h = Do not divide clock source 1h = Divide clock source by 2 2h = Divide clock source by 3 3h = Divide clock source by 4 4h = Divide clock source by 5 5h = Divide clock source by 6 6h = Divide clock source by 7 7h = Divide clock source by 8

13.3.6 CLKSEL Register (Offset = 1004h) [Reset = 00000000h]

CLKSEL is shown in Figure 13-23 and described in Table 13-28.

Return to the Summary Table.

Clock source selection.

Figure 13-23. CLKSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUSCLK_SEL	MFCLK_SEL	RESERVED	
R-0h				R/W-0h	R/W-0h	R-0h	

Table 13-28. CLKSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	BUSCLK_SEL	R/W	0h	Selects BUSCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
2	MFCLK_SEL	R/W	0h	Selects MFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
1-0	RESERVED	R	0h	

13.3.7 PDBGCTL Register (Offset = 1018h) [Reset = 00000000h]

PDBGCTL is shown in Figure 13-24 and described in Table 13-29.

Return to the Summary Table.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 13-24. PDBGCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R-0h						R/W-0h	R/W-0h

Table 13-29. PDBGCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	SOFT	R/W	0h	Soft halt boundary control. This function is only available, if FREE is set to 'STOP' 0h = The peripheral will halt immediately, even if the resultant state will result in corruption if the system is restarted 1h = The peripheral blocks the debug freeze until it has reached a boundary where it can resume without corruption
0	FREE	R/W	0h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

13.3.8 IIDX Register (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 13-25 and described in Table 13-30.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index.

Figure 13-25. IIDX Register

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 13-30. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	I2C Module Interrupt Vector Value. This register provides the highest priority interrupt index. A read clears the corresponding interrupt flag in RIS and MISC. 15h-1Fh = Reserved 00h = No interrupt pending 01h = Controller data received 02h = Controller data transmitted 03h = Controller receive FIFO Trigger Level 04h = Controller transmit FIFO Trigger level 5h = RX FIFO FULL Event/interrupt pending 6h = Transmit FIFO/Buffer Empty Event/interrupt pending 08h = Address/Data NACK 09h = Start Event 0Ah = Stop Event 0Bh = Arbitration Lost Ch = DMA DONE on Channel TX Dh = DMA DONE on Channel RX Eh = Controller PEC Receive Error Event Fh = Timeout A Event 10h = Timeout B Event 11h = Target Data Event 12h = Target Data Event 13h = Target receive FIFO Trigger Level 14h = Target transmit FIFO Trigger level 15h = RX FIFO FULL Event/interrupt pending 16h = Transmit FIFO/Buffer Empty Event/interrupt pending 17h = Start Event 18h = Stop Event 19h = General Call Event 1Ah = DMA DONE on Channel TX 1Bh = DMA DONE on Channel RX 1Ch = Target PEC receive error event 1Dh = Target TX FIFO underflow 1Eh = Target RX FIFO overflow event 1Fh = Target arbitration lost event 20h = Interrupt overflow event

13.3.9 IMASK Register (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 13-26 and described in Table 13-31.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 13-26. IMASK Register

31	30	29	28	27	26	25	24
INTR_OVFL	SARBLOST	SRX_OVFL	STX_UNFL	SPEC_RX_ERR	SDMA_DONE_RX	SDMA_DONE_TX	SGENCALL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SSTOP	SSTART	STXEMPTY	SRXFIFOFULL	STXFIFOTRG	SRXFIFOTRG	STXDONE	SRXDONE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TIMEOUTB	TIMEOUTA	MPEC_RX_ERR	MDMA_DONE_RX	MDMA_DONE_TX	MARBLOST	MSTOP	MSTART
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MNACK	RESERVED	MTXEMPTY	MRXFIFOFULL	MTXFIFOTRG	MRXFIFOTRG	MTXDONE	MRXDONE
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-31. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTR_OVFL	R/W	0h	Interrupt Overflow Interrupt Mask 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
30	SARBLOST	R/W	0h	Target Arbitration Lost 0h = Clear Set Interrupt Mask 1h = Set Interrupt Mask
29	SRX_OVFL	R/W	0h	Target RX FIFO overflow 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
28	STX_UNFL	R/W	0h	Target TX FIFO underflow 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
27	SPEC_RX_ERR	R/W	0h	Target RX Pec Error Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
26	SDMA_DONE_RX	R/W	0h	Target DMA Done on Event Channel RX 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
25	SDMA_DONE_TX	R/W	0h	Target DMA Done on Event Channel TX 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
24	SGENCALL	R/W	0h	General Call Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
23	SSTOP	R/W	0h	Stop Condition Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 13-31. IMASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	SSTART	R/W	0h	Start Condition Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
21	STXEMPTY	R/W	0h	Target Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
20	SRXFIFOFULL	R/W	0h	RXFIFO full event. This interrupt is set if an Target RX FIFO is full. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
19	STXFIFOTRG	R/W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
18	SRXFIFOTRG	R/W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
17	STXDONE	R/W	0h	Target Transmit Transaction completed Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
16	SRXDONE	R/W	0h	Target Receive Data Interrupt Signals that a byte has been received 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
15	TIMEOUTB	R/W	0h	Timeout B Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
14	TIMEOUTA	R/W	0h	Timeout A Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
13	MPEC_RX_ERR	R/W	0h	Controller RX Pec Error Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
12	MDMA_DONE_RX	R/W	0h	DMA Done on Event Channel RX 0h = Interrupt disabled 1h = Set Interrupt Mask
11	MDMA_DONE_TX	R/W	0h	DMA Done on Event Channel TX 0h = Interrupt disabled 1h = Set Interrupt Mask
10	MARBLOST	R/W	0h	Arbitration Lost Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
9	MSTOP	R/W	0h	STOP Detection Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
8	MSTART	R/W	0h	START Detection Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
7	MNACK	R/W	0h	Address/Data NACK Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
6	RESERVED	R	0h	
5	MTXEMPTY	R/W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 13-31. IMASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	MRXFIFOFULL	R/W	0h	RXFIFO full event. This interrupt is set if an RX FIFO is full. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3	MTXFIFOTRG	R/W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	MRXFIFOTRG	R/W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXDONE	R/W	0h	Controller Transmit Transaction completed Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXDONE	R/W	0h	Controller Receive Transaction completed Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.10 RIS Register (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 13-27 and described in Table 13-32.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 13-27. RIS Register

31	30	29	28	27	26	25	24
INTR_OVFL	SARBLOST	SRX_OVFL	STX_UNFL	SPEC_RX_ERR	SDMA_DONE_RX	SDMA_DONE_TX	SGENCALL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
SSTOP	SSTART	STXEMPTY	SRXFIFOFULL	STXFIFOTRG	SRXFIFOTRG	STXDONE	SRXDONE
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
TIMEOUTB	TIMEOUTA	MPEC_RX_ERR	MDMA_DONE_RX	MDMA_DONE_TX	MARBLOST	MSTOP	MSTART
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
MNACK	RESERVED	MTXEMPTY	MRXFIFOFULL	MTXFIFOTRG	MRXFIFOTRG	MTXDONE	MRXDONE
R-0h	R-0h	R-0h	R-0h	R/W-0h	R/W-0h	R-0h	R-0h

Table 13-32. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTR_OVFL	R	0h	Interrupt overflow interrupt It is set when SSTART or SSTOP interrupts overflow (occur twice without being serviced) 0h = Interrupt did not occur 1h = Interrupt occurred
30	SARBLOST	R	0h	Target Arbitration Lost 0h = Interrupt did not occur 1h = Interrupt occurred
29	SRX_OVFL	R	0x0	Target RX FIFO overflow 0h = Interrupt did not occur 1h = Interrupt Occurred
28	STX_UNFL	R	0x0	Target TX FIFO underflow 0h = Interrupt did not occur 1h = Interrupt occurred
27	SPEC_RX_ERR	R	0x0	Target RX Pec Error Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
26	SDMA_DONE_RX	R	0x0	DMA Done on Event Channel RX 0h = Clear interrupt 1h = Set interrupt
25	SDMA_DONE_TX	R	0x0	DMA Done on Event Channel TX 0h = Clear interrupt 1h = Set interrupt
24	SGENCALL	R	0x0	General Call Interrupt 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 13-32. RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	SSTOP	R	0x0	Stop Condition Interrupt 0h = Clear Interrupt 1h = Set interrupt
22	SSTART	R	0x0	Start Condition Interrupt 0h = Clear interrupt 1h = Set Interrupt
21	STXEMPTY	R	0x0	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Interrupt did not occur 1h = Set Interrupt Mask
20	SRXFIFOFULL	R	0x0	RXFIFO full event. This interrupt is set if an RX FIFO is full. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
19	STXFIFOTRG	R	0x0	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
18	SRXFIFOTRG	R	0x0	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
17	STXDONE	R	0x0	Target Transmit Transaction completed Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
16	SRXDONE	R	0x0	Target Receive Data Interrupt Signals that a byte has been received 0h = Interrupt did not occur 1h = Set Interrupt Mask
15	TIMEOUTB	R	0x0	Timeout B Interrupt 0h = Interrupt did not occur 1h = Interrupt occurred
14	TIMEOUTA	R	0x0	Timeout A Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
13	MPEC_RX_ERR	R	0x0	Controller RX Pec Error Interrupt 0h = Interrupt did not occur 1h = Interrupt Occurred
12	MDMA_DONE_RX	R	0x0	DMA Done on Event Channel RX 0h = Interrupt disabled 1h = Set Interrupt Mask
11	MDMA_DONE_TX	R	0x0	DMA Done on Event Channel TX 0h = Interrupt disabled 1h = Set Interrupt Mask
10	MARBLOST	R	0x0	Arbitration Lost Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
9	MSTOP	R	0x0	STOP Detection Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
8	MSTART	R	0x0	START Detection Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
7	MNACK	R	0x0	Address/Data NACK Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
6	RESERVED	R	0h	

Table 13-32. RIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	MTXEMPTY	R	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Interrupt did not occur 1h = Set Interrupt Mask
4	MRXFIFOFULL	R	0x0	RXFIFO full event. This interrupt is set if an RX FIFO is full. 0h = Interrupt did not occur 1h = Set Interrupt Mask
3	MTXFIFOTRG	R/W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	MRXFIFOTRG	R/W	0x0	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXDONE	R	0x0	Controller Transmit Transaction completed Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
0	MRXDONE	R	0x0	Controller Receive Transaction completed Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask

13.3.11 MIS Register (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 13-28 and described in Table 13-33.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 13-28. MIS Register

31	30	29	28	27	26	25	24
INTR_OVFL	SARBLOST	SRX_OVFL	STX_UNFL	SPEC_RX_ERR	SDMA_DONE_RX	SDMA_DONE_TX	SGENCALL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SSTOP	SSTART	STXEMPTY	SRXFIFOFULL	STXFIFOTRG	SRXFIFOTRG	STXDONE	SRXDONE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TIMEOUTB	TIMEOUTA	MPEC_RX_ERR	MDMA_DONE_RX	MDMA_DONE_TX	MARBLOST	MSTOP	MSTART
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MNACK	RESERVED	MTXEMPTY	MRXFIFOFULL	MTXFIFOTRG	MRXFIFOTRG	MTXDONE	MRXDONE
R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-33. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTR_OVFL	R/W	0h	Interrupt overflow 0h = Interrupt did not occur 1h = Interrupt occurred
30	SARBLOST	R/W	0h	Target Arbitration Lost 0h = Clear interrupt mask 1h = Set interrupt mask
29	SRX_OVFL	R/W	0h	Target RX FIFO overflow 0h = Clear interrupt mask 1h = Set interrupt mask
28	STX_UNFL	R/W	0h	Target TX FIFO underflow 0h = Clear interrupt mask 1h = Set interrupt mask
27	SPEC_RX_ERR	R/W	0h	Target RX Pec Error Interrupt 0h = Clear interrupt mask 1h = Set interrupt mask
26	SDMA_DONE_RX	R/W	0h	DMA Done on Event Channel RX 0h = Clear MIS 1h = Set MIS
25	SDMA_DONE_TX	R/W	0h	DMA Done on Event Channel TX 0h = Clear MIS 1h = Set MIS
24	SGENCALL	R/W	0h	General Call Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
23	SSTOP	R/W	0h	Target STOP Detection Interrupt 0h = Clear MIS 1h = Set MIS
22	SSTART	R/W	0h	Target START Detection Interrupt 0h = Clear MIS 1h = Set MIS

Table 13-33. MIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	STXEMPTY	R/W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Interrupt did not occur 1h = Set Interrupt Mask
20	SRXFIFOFULL	R/W	0h	RXFIFO full event. This interrupt is set if an RX FIFO is full. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
19	STXFIFOTRG	R/W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
18	SRXFIFOTRG	R/W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
17	STXDONE	R/W	0h	Target Transmit Transaction completed Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
16	SRXDONE	R/W	0h	Target Receive Data Interrupt Signals that a byte has been received 0h = Interrupt did not occur 1h = Set Interrupt Mask
15	TIMEOUTB	R/W	0h	Timeout B Interrupt 0h = Clear interrupt mask 1h = Set interrupt mask
14	TIMEOUTA	R/W	0h	Timeout A Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
13	MPEC_RX_ERR	R/W	0h	Controller RX Pec Error Interrupt 0h = Clear interrupt mask 1h = Set interrupt mask
12	MDMA_DONE_RX	R/W	0h	DMA Done on Event Channel RX 0h = Interrupt disabled 1h = Set Interrupt Mask
11	MDMA_DONE_TX	R/W	0h	DMA Done on Event Channel TX 0h = Interrupt disabled 1h = Set Interrupt Mask
10	MARBLOST	R/W	0h	Arbitration Lost Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
9	MSTOP	R/W	0h	STOP Detection Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
8	MSTART	R/W	0h	START Detection Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
7	MNACK	R/W	0h	Address/Data NACK Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
6	RESERVED	R	0h	
5	MTXEMPTY	R/W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Interrupt did not occur 1h = Set Interrupt Mask
4	MRXFIFOFULL	R/W	0h	RXFIFO full event. This interrupt is set if the RX FIFO is full. 0h = Interrupt did not occur 1h = Set Interrupt Mask

Table 13-33. MIS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	MTXFIFOTRG	R/W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	MRXFIFOTRG	R/W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXDONE	R/W	0h	Controller Transmit Transaction completed Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask
0	MRXDONE	R/W	0h	Controller Receive Data Interrupt 0h = Interrupt did not occur 1h = Set Interrupt Mask

13.3.12 ISET Register (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 13-29 and described in Table 13-34.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 13-29. ISET Register

31	30	29	28	27	26	25	24
INTR_OVFL	SARBLOST	SRX_OVFL	STX_UNFL	SPEC_RX_ERR	SDMA_DONE_RX	SDMA_DONE_TX	SGENCALL
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
SSTOP	SSTART	STXEMPTY	SRXFIFOFULL	STXFIFOTRG	SRXFIFOTRG	STXDONE	SRXDONE
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
TIMEOUTB	TIMEOUTA	MPEC_RX_ERR	MDMA_DONE_RX	MDMA_DONE_TX	MARBLOST	MSTOP	MSTART
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
MNACK	RESERVED	MTXEMPTY	MRXFIFOFULL	MTXFIFOTRG	MRXFIFOTRG	MTXDONE	MRXDONE
W-0h	R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 13-34. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTR_OVFL	W	0h	Interrupt overflow 0h = No effect 1h = Set interrupt
30	SARBLOST	W	0h	Target Arbitration Lost 0h = Writing 0 has no effect 1h = Set interrupt
29	SRX_OVFL	W	0h	Target RX FIFO overflow 0h = Writing 0 has no effect 1h = Set interrupt
28	STX_UNFL	W	0h	Target TX FIFO underflow 0h = Writing 0 has no effect 1h = Set interrupt
27	SPEC_RX_ERR	W	0h	Target RX Pec Error Interrupt 0h = Writing 0 has no effect 1h = Set interrupt
26	SDMA_DONE_RX	W	0h	DMA Done on Event Channel RX 0h = Writing 0 has no effect 1h = Set interrupt
25	SDMA_DONE_TX	W	0h	DMA Done on Event Channel TX 0h = Writing 0 has no effect 1h = Set interrupt
24	SGENCALL	W	0h	General Call Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
23	SSTOP	W	0h	Stop Condition Interrupt 0h = Writing 0 has no effect 1h = Set interrupt

Table 13-34. ISET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	SSTART	W	0h	Start Condition Interrupt 0h = Writing 0 has no effect 1h = Set interrupt
21	STXEMPTY	W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Writing 0 has no effect 1h = Set Interrupt Mask
20	SRXFIFOFULL	W	0h	RXFIFO full event. This interrupt is set if an RX FIFO is full. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
19	STXFIFOTRG	W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
18	SRXFIFOTRG	W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
17	STXDONE	W	0h	Target Transmit Transaction completed Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
16	SRXDONE	W	0h	Target Receive Data Interrupt Signals that a byte has been received 0h = Writing 0 has no effect 1h = Set Interrupt Mask
15	TIMEOUTB	W	0h	Timeout B Interrupt 0h = Writing 0 has no effect 1h = Set interrupt
14	TIMEOUTA	W	0h	Timeout A interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
13	MPEC_RX_ERR	W	0h	Controller RX Pec Error Interrupt 0h = Writing 0 has no effect 1h = Set interrupt
12	MDMA_DONE_RX	W	0h	DMA Done on Event Channel RX 0h = Interrupt disabled 1h = Set Interrupt Mask
11	MDMA_DONE_TX	W	0h	DMA Done on Event Channel TX 0h = Interrupt disabled 1h = Set Interrupt Mask
10	MARBLOST	W	0h	Arbitration Lost Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
9	MSTOP	W	0h	STOP Detection Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
8	MSTART	W	0h	START Detection Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
7	MNACK	W	0h	Address/Data NACK Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
6	RESERVED	R	0h	
5	MTXEMPTY	W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Writing 0 has no effect 1h = Set Interrupt Mask

Table 13-34. ISET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	MRXFIFOFULL	W	0h	RXFIFO full event. 0h = Writing 0 has no effect 1h = Set Interrupt Mask
3	MTXFIFOTRG	W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	MRXFIFOTRG	W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXDONE	W	0h	Controller Transmit Transaction completed Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
0	MRXDONE	W	0h	Controller Receive Data Interrupt Signals that a byte has been received 0h = Writing 0 has no effect 1h = Set Interrupt Mask

13.3.13 ICLR Register (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 13-30 and described in Table 13-35.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 13-30. ICLR Register

31	30	29	28	27	26	25	24
INTR_OVFL	SARBLOST	SRX_OVFL	STX_UNFL	SPEC_RX_ERR	SDMA_DONE_RX	SDMA_DONE_TX	SGENCALL
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
SSTOP	SSTART	STXEMPTY	SRXFIFOFULL	STXFIFOTRG	SRXFIFOTRG	STXDONE	SRXDONE
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
15	14	13	12	11	10	9	8
TIMEOUTB	TIMEOUTA	MPEC_RX_ERR	MDMA_DONE_RX	MDMA_DONE_TX	MARBLOST	MSTOP	MSTART
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
MNACK	RESERVED	MTXEMPTY	MRXFIFOFULL	MTXFIFOTRG	MRXFIFOTRG	MTXDONE	MRXDONE
W-0h	R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 13-35. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	INTR_OVFL	W	0h	Interrupt overflow 0h = No effect 1h = Clear interrupt
30	SARBLOST	W	0h	Target Arbitration Lost 0h = Writing 0 has no effect 1h = Clear Interrupt
29	SRX_OVFL	W	0h	Target RX FIFO overflow 0h = Writing 0 has no effect 1h = Clear Interrupt
28	STX_UNFL	W	0h	Target TX FIFO underflow 0h = Writing 0 has no effect 1h = Clear Interrupt
27	SPEC_RX_ERR	W	0h	Target RX Pec Error Interrupt 0h = Writing 0 has no effect 1h = Clear Interrupt
26	SDMA_DONE_RX	W	0h	DMA Done on Event Channel RX 0h = Writing 0 has no effect 1h = Clear interrupt
25	SDMA_DONE_TX	W	0h	DMA Done on Event Channel TX 0h = Writing 0 has no effect 1h = Clear interrupt
24	SGENCALL	W	0h	General Call Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
23	SSTOP	W	0h	Target STOP Detection Interrupt 0h = Writing 0 has no effect 1h = Clear interrupt
22	SSTART	W	0h	Target START Detection Interrupt 0h = Writing 0 has no effect 1h = Clear interrupt

Table 13-35. ICLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	STXEMPTY	W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Writing 0 has no effect 1h = Set Interrupt Mask
20	SRXFIFOFULL	W	0h	RXFIFO full event. This interrupt is set if an RX FIFO is full. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
19	STXFIFOTRG	W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
18	SRXFIFOTRG	W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
17	STXDONE	W	0h	Target Transmit Transaction completed Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
16	SRXDONE	W	0h	Target Receive Data Interrupt Signals that a byte has been received 0h = Writing 0 has no effect 1h = Set Interrupt Mask
15	TIMEOUTB	W	0h	Timeout B Interrupt 0h = Writing 0 has no effect 1h = Clear Interrupt
14	TIMEOUTA	W	0h	Timeout A interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
13	MPEC_RX_ERR	W	0h	Controller RX Pec Error Interrupt 0h = Writing 0 has no effect 1h = Clear Interrupt
12	MDMA_DONE_RX	W	0h	DMA Done on Event Channel RX 0h = Interrupt disabled 1h = Set Interrupt Mask
11	MDMA_DONE_TX	W	0h	DMA Done on Event Channel TX 0h = Interrupt disabled 1h = Set Interrupt Mask
10	MARBLOST	W	0h	Arbitration Lost Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
9	MSTOP	W	0h	STOP Detection Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
8	MSTART	W	0h	START Detection Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
7	MNACK	W	0h	Address/Data NACK Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
6	RESERVED	R	0h	
5	MTXEMPTY	W	0h	Transmit FIFO Empty interrupt mask. This interrupt is set if all data in the Transmit FIFO have been shifted out and the transmit goes into idle mode. 0h = Writing 0 has no effect 1h = Set Interrupt Mask
4	MRXFIFOFULL	W	0h	RXFIFO full event. 0h = Writing 0 has no effect 1h = Set Interrupt Mask

Table 13-35. ICLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	MTXFIFOTRG	W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	MRXFIFOTRG	W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXDONE	W	0h	Controller Transmit Transaction completed Interrupt 0h = Writing 0 has no effect 1h = Set Interrupt Mask
0	MRXDONE	W	0h	Controller Receive Data Interrupt Signals that a byte has been received 0h = Writing 0 has no effect 1h = Set Interrupt Mask

13.3.14 IIDX Register (Offset = 1050h) [Reset = 00000000h]

IIDX is shown in Figure 13-31 and described in Table 13-36.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index.

Figure 13-31. IIDX Register

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 13-36. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	I2C Module Interrupt Vector Value. This register provides the highest priority interrupt index. A read clears the corresponding interrupt flag in RIS and MISC. 15h-1Fh = Reserved 00h = No interrupt pending 01h = Controller receive FIFO Trigger Level 02h = Controller transmit FIFO Trigger level 03h = Target receive FIFO Trigger Level 04h = Target transmit FIFO Trigger level

13.3.15 IMASK Register (Offset = 1058h) [Reset = 00000000h]

IMASK is shown in Figure 13-32 and described in Table 13-37.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 13-32. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-37. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R/W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R/W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R/W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R/W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.16 RIS Register (Offset = 1060h) [Reset = 00000000h]

RIS is shown in Figure 13-33 and described in Table 13-38.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 13-33. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-38. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R/W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R/W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R/W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R/W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.17 MIS Register (Offset = 1068h) [Reset = 00000000h]

MIS is shown in Figure 13-34 and described in Table 13-39.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 13-34. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 13-39. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.18 ISET Register (Offset = 1070h) [Reset = 00000000h]

ISET is shown in Figure 13-35 and described in Table 13-40.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 13-35. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				W-0h	W-0h	W-0h	W-0h

Table 13-40. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.19 ICLR Register (Offset = 1078h) [Reset = 00000000h]

ICLR is shown in Figure 13-36 and described in Table 13-41.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 13-36. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				W-0h	W-0h	W-0h	W-0h

Table 13-41. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	W	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	W	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	W	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	W	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.20 IIDX Register (Offset = 1080h) [Reset = 00000000h]

IIDX is shown in Figure 13-37 and described in Table 13-42.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index.

Figure 13-37. IIDX Register

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 13-42. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	I2C Module Interrupt Vector Value. This register provides the highest priority interrupt index. A read clears the corresponding interrupt flag in RIS and MISC. 15h-1Fh = Reserved 00h = No interrupt pending 01h = Controller receive FIFO Trigger Level 02h = Controller transmit FIFO Trigger level 03h = Target receive FIFO Trigger Level 04h = Target transmit FIFO Trigger level

13.3.21 IMASK Register (Offset = 1088h) [Reset = 00000000h]

IMASK is shown in Figure 13-38 and described in Table 13-43.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 13-38. IMASK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 13-43. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.22 RIS Register (Offset = 1090h) [Reset = 00000000h]

RIS is shown in Figure 13-39 and described in Table 13-44.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 13-39. RIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 13-44. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.23 MIS Register (Offset = 1098h) [Reset = 00000000h]

MIS is shown in Figure 13-40 and described in Table 13-45.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 13-40. MIS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 13-45. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.24 ISET Register (Offset = 10A0h) [Reset = 00000000h]

ISET is shown in Figure 13-41 and described in Table 13-46.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 13-41. ISET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 13-46. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.25 ICLR Register (Offset = 10A8h) [Reset = 00000000h]

ICLR is shown in Figure 13-42 and described in Table 13-47.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 13-42. ICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				STXFIFOTRG	SRXFIFOTRG	MTXFIFOTRG	MRXFIFOTRG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 13-47. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	STXFIFOTRG	R	0h	Target Transmit FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
2	SRXFIFOTRG	R	0h	Target Receive FIFO Trigger 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
1	MTXFIFOTRG	R	0h	Controller Transmit FIFO Trigger Trigger when Transmit FIFO contains <= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	MRXFIFOTRG	R	0h	Controller Receive FIFO Trigger Trigger when RX FIFO contains >= defined bytes 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

13.3.26 EVT_MODE Register (Offset = 10E0h) [Reset = 00000000h]

EVT_MODE is shown in Figure 13-43 and described in Table 13-48.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 13-43. EVT_MODE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		EVT2_CFG		INT1_CFG		INT0_CFG	
R-0h		R-0h		R-0h		R-0h	

Table 13-48. EVT_MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5-4	EVT2_CFG	R	0h	Event line mode select for event corresponding to none.DMA_TRIG0 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
3-2	INT1_CFG	R	0h	Event line mode select for event corresponding to none.DMA_TRIG1 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
1-0	INT0_CFG	R	0h	Event line mode select for event corresponding to CPU_INT 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

13.3.27 INTCTL Register (Offset = 10E4h) [Reset = 00000000h]

INTCTL is shown in Figure 13-44 and described in Table 13-49.

Return to the Summary Table.

Interrupt control register

Figure 13-44. INTCTL Register

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

Table 13-49. INTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	INTEVAL	W	0h	Writing a 1 to this field re-evaluates the interrupt sources. 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS.

13.3.28 DESC Register (Offset = 10FCh) [Reset = 00000000h]

DESC is shown in Figure 13-45 and described in Table 13-50.

Return to the Summary Table.

This register identifies the peripheral and its exact version.

Figure 13-45. DESC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				INSTNUM				MAJREV				MINREV			
R-0h				R/W-0h				R-0h				R-0h			

Table 13-50. DESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R/W	0h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness. 0h = Smallest value FFFFh = Highest possible value
15-12	FEATUREVER	R	0x0	Feature Set for the module *instance* 0h = Smallest value Fh = Highest possible value
11-8	INSTNUM	R/W	0x0	Instance Number within the device. This will be a parameter to the RTL for modules that can have multiple instances 0h = Smallest value Fh = Highest possible value
7-4	MAJREV	R	0h	Major rev of the IP 0h = Smallest value Fh = Highest possible value
3-0	MINREV	R	0x0	Minor rev of the IP 0h = Smallest value Fh = Highest possible value

13.3.29 GFCTL Register (Offset = 1200h) [Reset = 00000000h]

GFCTL is shown in Figure 13-46 and described in Table 13-51.

Return to the Summary Table.

This register controls the glitch filter on the SCL and SDA lines

Figure 13-46. GFCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CHAIN	AGFSEL		AGFEN
R-0h				R/W-0h	R/W-0h		R/W-0h
7	6	5	4	3	2	1	0
RESERVED					DGFSEL		
R-0h					R/W-0h		

Table 13-51. GFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	
11	CHAIN	R/W	0h	Analog and digital noise filters chaining enable. 0h = When 0, chaining is disabled and only digital filter output is available to IP logic for oversampling 1h = When 1, analog and digital glitch filters are chained and the output of the combination is made available to IP logic for oversampling
10-9	AGFSEL	R/W	0h	Analog Glitch Suppression Pulse Width This field controls the pulse width select for the analog glitch suppression on SCL and SDA lines. See device data sheet for exact values. (ULP I2C only) 0h = Pulses shorter then 5ns length are filtered. 1h = Pulses shorter then 10ns length are filtered. 2h = Pulses shorter then 25ns length are filtered. 3h = Pulses shorter then 50ns length are filtered.
8	AGFEN	R/W	0h	Analog Glitch Suppression Enable 0h = Analog Glitch Filter disable 1h = Analog Glitch Filter enable
7-3	RESERVED	R	0h	
2-0	DGFSEL	R/W	0x0	Glitch Suppression Pulse Width This field controls the pulse width select for glitch suppression on the SCL and SDA lines. The following values are the glitch suppression values in terms of functional clocks. (Core Domain only) 0h = Bypass 1h = 1 clock 2h = 2 clocks 3h = 3 clocks 4h = 4 clocks 5h = 8 clocks 6h = 16 clocks 7h = 31 clocks

13.3.30 TIMEOUT_CTL Register (Offset = 1204h) [Reset = 00020002h]

TIMEOUT_CTL is shown in Figure 13-47 and described in Table 13-52.

Return to the Summary Table.

This register contains controls for Timeout Counters A and B

Figure 13-47. TIMEOUT_CTL Register

31	30	29	28	27	26	25	24
TCNTBEN	RESERVED						
R/W-0h	R-0h						
23	22	21	20	19	18	17	16
TCNTLB							
R/W-2h							
15	14	13	12	11	10	9	8
TCNTAEN	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
TCNTLA							
R/W-2h							

Table 13-52. TIMEOUT_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TCNTBEN	R/W	0h	Timeout Counter B Enable 0h = Disable Timeout Counter B 1h = Enable Timeout Counter B
30-24	RESERVED	R	0h	
23-16	TCNTLB	R/W	2h	Timeout Count B Load: Counter B is used for SCL High Detection. This field contains the upper 8 bits of a 12-bit preload value for the Timeout B count. NOTE: The value of CNTLB must be greater than 1h. Each count is equal to 1* clock period. For example, with 10MHz functional clock one timeout period will be equal to1*100ns. 0h = Smallest possible value FFh = Highest possible value
15	TCNTAEN	R/W	0h	Timeout Counter A Enable 0h = Disable Timeout Counter B 1h = Enable Timeout Counter B
14-8	RESERVED	R	0h	
7-0	TCNTLA	R/W	2h	Timeout counter A load value Counter A is used for SCL low detection. This field contains the upper 8 bits of a 12-bit preload value for the Timeout A count. NOTE: The value of CNTLA must be greater than 1h. Each count is equal to 520 times the timeout period of functional clock. For example, with 8MHz functional clock and a 100KHz operating I2C clock, one timeout period will be equal to (1 / 8MHz) * 520 or 65 us. 0h = Smallest Value FFh = Highest possible value

13.3.31 TIMEOUT_CNT Register (Offset = 1208h) [Reset = 00020002h]

TIMEOUT_CNT is shown in Figure 13-48 and described in Table 13-53.

Return to the Summary Table.

This register contains the upper 8 bits of a 12-bit current counter values for counter A and B. The lower four bits of the counter are not user visible and are always 0h.

Figure 13-48. TIMEOUT_CNT Register

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
RESERVED								TCNTB								RESERVED								TCNTA							
R-0h								R-2h								R-0h								R-2h							

Table 13-53. TIMEOUT_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	
23-16	TCNTB	R	2h	Timeout Count B Current Count: This field contains the upper 8 bits of a 12-bit current counter for timeout counter B 0h = Smallest Value FFh = Highest possible value
15-8	RESERVED	R	0h	
7-0	TCNTA	R	2h	Timeout Count A Current Count: This field contains the upper 8 bits of a 12-bit current counter for timeout counter A 0h = Smallest Value FFh = Highest possible value

13.3.32 MSA Register (Offset = 1210h) [Reset = 00000000h]

MSA is shown in Figure 13-49 and described in Table 13-54.

Return to the Summary Table.

I2C Controller Target Address Register

Figure 13-49. MSA Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MMODE	RESERVED				SADDR		
R/W-0h	R-0h				R/W-0h		
7	6	5	4	3	2	1	0
SADDR							DIR
R/W-0h							R-0h

Table 13-54. MSA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	MMODE	R/W	0h	This bit selects the addressing mode to be used in Controller mode When 0, 7-bit addressing is used. When 1, 10-bit addressing is used. 0h = 7-bit addressing mode 1h = 10-bit addressing mode
14-11	RESERVED	R	0h	
10-1	SADDR	R/W	0h	I2C Target Address This field specifies bits A9 through A0 of the Target address. In 7-bit addressing mode as selected by MSA.MODE bit, the top 3 bits are don't care 0h = Smallest value 3FFh = Highest possible value
0	DIR	R	0h	Receive/Send The DIR bit specifies if the next Controller operation is a Receive (High) or Transmit (Low). 0h = Transmit 1h = Receive 0h = The Controller is in transmit mode. 1h = The Controller is in receive mode.

13.3.33 MCTR Register (Offset = 1214h) [Reset = 00000000h]

MCTR is shown in Figure 13-50 and described in Table 13-55.

Return to the Summary Table.

This control register configures the I2C controller operation. The START bit generates the START or REPEATED START condition. The STOP bit determines if the cycle stops at the end of the data cycle or continues to the next transfer cycle, which could be a repeated START. To generate a single transmit cycle, the I2C Controller Target Address (MSA) register is written with the desired address, the RS bit is cleared, and this register is written with ACK = X (0 or 1), STOP = 1, START = 1, and RUN = 1 to perform the operation and stop. When the operation is completed (or aborted due an error), an byte transaction completed interrupt becomes active and the data may be read from the MRXDATA register. When the I2C module operates in Controller receiver mode, a set ACK bit causes the I2C bus controller to transmit an acknowledge automatically after each byte. This bit must be cleared when the I2C bus controller requires no further data to be transmitted from the Target transmitter.

Figure 13-50. MCTR Register

31	30	29	28	27	26	25	24
RESERVED				MBLEN			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
MBLEN							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RD_ON_TXEMPTY	MACKOEN	ACK	STOP	START	BURSTRUN
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-55. MCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	
27-16	MBLEN	R/W	0h	I2C transaction length This field contains the programmed length of bytes of the Transaction. 0h = Smallest value FFFh = Highest possible value
15-6	RESERVED	R	0h	
5	RD_ON_TXEMPTY	R/W	0h	Read on TX Empty 0h = No special behavior 1h = When 1 the Controller will transmit all bytes from the TX FIFO before continuing with the programmed Burst Run Read. If the DIR is not set to Read in the MSA then this bit is ignored. The Start must be set in the MCTR for proper I2C protocol. The Controller will first send the Start Condition, I2C Address with R/W bit set to write, before sending the bytes in the TX FIFO. When the TX FIFO is empty, the I2C transaction will continue as programmed in MTCR and MSA without sending a Stop Condition. This is intended to be used to perform simple I2C command based reads transition that will complete after initiating them without having to get an interrupt to turn the bus around.

Table 13-55. MCTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	MACKOEN	R/W	0h	Controller ACK override Enable 0h = No special behavior 1h = When 1 and the Controller is receiving data and the number of bytes indicated in MBLEN have been received, the state machine will generate an RXDONE interrupt and wait at the start of the ACK for FW to indicate if an ACK or NACK should be sent. The ACK or NACK is selected by writing the MCTR register and setting ACK accordingly. The other fields in this register can also be written at this time to continue on with the transaction. If a NACK is sent the state machine will automatically send a Stop.
3	ACK	R/W	0h	Data Acknowledge Enable. Software needs to configure this bit to send the ACK or NACK. 0h = The last received data byte of a transaction is not acknowledged automatically by the Controller. 1h = The last received data byte of a transaction is acknowledged automatically by the Controller.
2	STOP	R/W	0h	Generate STOP 0h = The controller does not generate the STOP condition. 1h = The controller generates the STOP condition.
1	START	R/W	0h	Generate START 0h = The controller does not generate the START condition. 1h = The controller generates the START or repeated START condition.
0	BURSTRUN	R/W	0h	I2C Controller Enable and start transaction 0h = In standard mode, this encoding means the Controller is unable to transmit or receive data. 1h = The Controller is able to transmit or receive data.

13.3.34 MSR Register (Offset = 1218h) [Reset = 00000000h]

MSR is shown in Figure 13-51 and described in Table 13-56.

Return to the Summary Table.

The status register indicates the state of the I2C bus controller.

Figure 13-51. MSR Register

31	30	29	28	27	26	25	24
RESERVED				MBCNT			
R-0h				R-0h			
23	22	21	20	19	18	17	16
MBCNT							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	BUSBSY	IDLE	ARBLST	DATACK	ADRACK	ERR	BUSY
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-56. MSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	
27-16	MBCNT	R	0x0	I2C Controller Transaction Count This field contains the current count-down value of the transaction. 0h = Smallest value FFFh = Highest possible value
15-7	RESERVED	R	0h	
6	BUSBSY	R	0x0	I2C Bus is Busy Controller State Machine will wait until this bit is cleared before starting a transaction. When first enabling the controller in multiple-controller environments, FW should wait for one I2C clock period after setting ACTIVE high before writing to the MTCR register to start the transaction so that if SCL goes low it will trigger the BUSBSY. 0h = The I2C bus is idle. 1h = 'This Status bit is set on a START or when SCL goes low. It is cleared on a STOP, or when a SCL high bus busy timeout occurs and SCL and SDA are both high. This status is cleared when the ACTIVE bit is low. Note that the Controller State Machine will wait until this bit is cleared before starting an I2C transaction. When first enabling the Controller in multiple-controller environments, FW should wait for one I2C clock period after setting ACTIVE high before writing to the MTCR register to start the transaction so that if SCL goes low it will trigger the BUSBSY.
5	IDLE	R	0h	I2C Idle 0h = The I2C controller is not idle. 1h = The I2C controller is idle.
4	ARBLST	R	0x0	Arbitration Lost 0h = The I2C controller won arbitration. 1h = The I2C controller lost arbitration.
3	DATACK	R	0x0	Acknowledge Data 0h = The transmitted data was acknowledged 1h = The transmitted data was not acknowledged.

Table 13-56. MSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	ADRACK	R	0x0	Acknowledge Address 0h = The transmitted address was acknowledged 1h = The transmitted address was not acknowledged.
1	ERR	R	0x0	Error The error can be from the Target address not being acknowledged or the transmit data not being acknowledged. 0h = No error was detected on the last operation. 1h = An error occurred on the last operation.
0	BUSY	R	0x0	I2C Controller FSM Busy The BUSY bit is set during an ongoing transaction, so is set during the transmit/receive of the amount of data set in MBLEN including START, RESTART, Address and STOP signal generation when required for the current transaction. 0h = The controller is idle. 1h = The controller is busy.

13.3.35 MRXDATA Register (Offset = 121Ch) [Reset = 00000000h]

MRXDATA is shown in Figure 13-52 and described in Table 13-57.

Return to the Summary Table.

I2C Controller RX FIFO Read Data Byte
This field contains the current byte being read in the RX FIFO stack.
If the FIFO is disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register.

Figure 13-52. MRXDATA Register

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
RESERVED																								VALUE							
R-0h																								R-0h							

Table 13-57. MRXDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	VALUE	R	0h	Received Data. This field contains the last received data. 0h = Smallest value FFh = Highest possible value

13.3.36 MTXDATA Register (Offset = 1220h) [Reset = 00000000h]

MTXDATA is shown in Figure 13-53 and described in Table 13-58.

Return to the Summary Table.

I2C Controller Transmit Data Register.
This register is the transmit data register (the interface to the FIFOs). For transmitted data, if the FIFO is enabled, data written to this location is pushed onto the transmit FIFO. If the FIFO is disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO).

Figure 13-53. MTXDATA Register

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
RESERVED																								VALUE							
R-0h																								R/W-0h							

Table 13-58. MTXDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	VALUE	R/W	0h	Transmit Data This byte contains the data to be transferred during the next transaction. 0h = Smallest value FFh = Highest possible value

13.3.37 MTPR Register (Offset = 1224h) [Reset = 00000001h]

MTPR is shown in Figure 13-54 and described in Table 13-59.

Return to the Summary Table.

This register is programmed to set the timer period for the SCL clock and assign the SCL clock to standard mode.

Figure 13-54. MTPR Register

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
RESERVED																									TPR						
R-0h																									R/W-1h						

Table 13-59. MTPR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	
6-0	TPR	R/W	1h	Timer Period This field is used in the equation to configure SCL_PERIOD : SCL_PERIOD = (1 + TPR) × (SCL_LP + SCL_HP) × INT_CLK_PRD where: SCL_PRD is the SCL line period (I2C clock). TPR is the Timer Period register value (range of 1 to 127). SCL_LP is the SCL Low period (fixed at 6). SCL_HP is the SCL High period (fixed at 4). CLK_PRD is the functional clock period in ns. 0h = Smallest value 7Fh = Highest possible value

13.3.38 MCR Register (Offset = 1228h) [Reset = 00000000h]

MCR is shown in Figure 13-55 and described in Table 13-60.

Return to the Summary Table.

Controller configuration register

Figure 13-55. MCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							LPBK
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED					CLKSTRETCH	MMST	ACTIVE
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 13-60. MCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	LPBK	R/W	0h	I2C Loopback 0h = Normal operation. 1h = The controller in a test mode loopback configuration.
7-3	RESERVED	R	0h	
2	CLKSTRETCH	R/W	0h	Clock Stretching. This bit controls the support for clock stretching of the I2C bus. 0h = Disables the clock stretching detection. This can be disabled if no Target on the bus does support clock stretching, so that the maximum speed on the bus can be reached. 1h = Enables the clock stretching detection. Enabling the clock stretching ensures compliance to the I2C standard but could limit the speed due the clock stretching.
1	MMST	R/W	0h	MultiController mode. In MultiController mode the SCL high time counts once the SCL line has been detected high. If this is not enabled the high time counts as soon as the SCL line has been set high by the I2C controller. 0h = Disable MultiController mode. 1h = Enable MultiController mode.
0	ACTIVE	R/W	0h	Device Active After this bit has been set, it should not be set again unless it has been cleared by writing a 0 or by a reset, otherwise transfer failures may occur. 0h = Disables the I2C Controller operation. 1h = Enables the I2C Controller operation.

13.3.39 MBMON Register (Offset = 1234h) [Reset = 00000003h]

MBMON is shown in Figure 13-56 and described in Table 13-61.

Return to the Summary Table.

This register is used to determine the SCL and SDA signal status.

Figure 13-56. MBMON Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														SDA	SCL
R-0h														R-1h	R-1h

Table 13-61. MBMON Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	SDA	R	1h	I2C SDA Status 0h = The I2CSDA signal is low. 1h = The I2CSDA signal is high. Note: During and right after reset, the SDA pin is in GPIO input mode without the internal pull enabled. For proper I2C operation, the user should have the external pull-up resistor in place before starting any I2C operations.
0	SCL	R	1h	I2C SCL Status 0h = The I2CSCL signal is low. 1h = The I2CSCL signal is high. Note: During and right after reset, the SCL pin is in GPIO input mode without the internal pull enabled. For proper I2C operation, the user should have the external pull-up resistor in place before starting any I2C operations.

13.3.40 MFIFOCTL Register (Offset = 1238h) [Reset = 00000000h]

MFIFOCTL is shown in Figure 13-57 and described in Table 13-62.

Return to the Summary Table.

I2C Controller FIFO Control

Figure 13-57. MFIFOCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RXFLUSH	RESERVED					RXTRIG	
R/W-0h	R-0h					R/W-0h	
7	6	5	4	3	2	1	0
TXFLUSH	RESERVED					TXTRIG	
R/W-0h	R-0h					R/W-0h	

Table 13-62. MFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	RXFLUSH	R/W	0h	RX FIFO Flush Setting this bit will Flush the RX FIFO. Before clearing this bit to stop Flush the RXFIFOCNT should be checked to be 0 and indicating that the Flush has completed. 0h = Do not Flush FIFO 1h = Flush FIFO
14-11	RESERVED	R	0h	
10-8	RXTRIG	R/W	0h	RX FIFO Trigger Indicates at what fill level in the RX FIFO a trigger will be generated. Note: Programming RXTRIG to 0x0 has no effect since no data is present to transfer out of RX FIFO. 0h = Trigger when RX FIFO contains >= 1 byte 1h = Trigger when RX FIFO contains >= 2 byte 2h = Trigger when RX FIFO contains >= 3 byte 3h = Trigger when RX FIFO contains >= 4 byte 4h = Trigger when RX FIFO contains >= 5 byte 5h = Trigger when RX FIFO contains >= 6 byte 6h = Trigger when RX FIFO contains >= 7 byte 7h = Trigger when RX FIFO contains >= 8 byte
7	TXFLUSH	R/W	0h	TX FIFO Flush Setting this bit will Flush the TX FIFO. Before clearing this bit to stop Flush the TXFIFOCNT should be checked to be 8 and indicating that the Flush has completed. 0h = Do not Flush FIFO 1h = Flush FIFO
6-3	RESERVED	R	0h	

Table 13-62. MFIFOCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	TXTRIG	R/W	0h	TX FIFO Trigger Indicates at what fill level in the TX FIFO a trigger will be generated. 0h = Trigger when the TX FIFO is empty. 1h = Trigger when TX FIFO contains ≤ 1 byte 2h = Trigger when TX FIFO contains ≤ 2 byte 3h = Trigger when TX FIFO contains ≤ 3 byte 4h = Trigger when TX FIFO contains ≤ 4 byte 5h = Trigger when TX FIFO contains ≤ 5 byte 6h = Trigger when TX FIFO contains ≤ 6 byte 7h = Trigger when TX FIFO contains ≤ 7 byte

13.3.41 MFIFOSR Register (Offset = 123Ch) [Reset = 00000800h]

MFIFOSR is shown in Figure 13-58 and described in Table 13-63.

Return to the Summary Table.

I2C Controller FIFO Status Register
Note: this Register should only be read when BUSY is 0

Figure 13-58. MFIFOSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TXFLUSH	RESERVED			TXFIFOCNT			
R/W-0h	R-0h			R-8h			
7	6	5	4	3	2	1	0
RXFLUSH	RESERVED			RXFIFOCNT			
R/W-0h	R-0h			R-0h			

Table 13-63. MFIFOSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	TXFLUSH	R/W	0h	TX FIFO Flush When this bit is set a Flush operation for the TX FIFO is active. Clear the TXFLUSH bit in the control register to stop. 0h = FIFO Flush not active 1h = FIFO Flush active
14-12	RESERVED	R	0h	
11-8	TXFIFOCNT	R	8h	Number of Bytes which could be put into the TX FIFO 0h = Smallest value 8h = Highest possible value
7	RXFLUSH	R/W	0h	RX FIFO Flush When this bit is set a Flush operation for the RX FIFO is active. Clear the RXFLUSH bit in the control register to stop. 0h = FIFO Flush not active 1h = FIFO Flush active
6-4	RESERVED	R	0h	
3-0	RXFIFOCNT	R	0h	Number of Bytes which could be read from the RX FIFO 0h = Smallest value 8h = Highest possible value

13.3.42 SOAR Register (Offset = 1250h) [Reset = 00004000h]

SOAR is shown in Figure 13-59 and described in Table 13-64.

Return to the Summary Table.

This register consists of seven address bits that identify the I2C device on the I2C bus.

Figure 13-59. SOAR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
SMODE	OAREN	RESERVED				OAR	
R/W-0h	R/W-1h	R-0h				R/W-0h	
7	6	5	4	3	2	1	0
OAR							
R/W-0h							

Table 13-64. SOAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	SMODE	R/W	0h	This bit selects the addressing mode to be used in Target mode. When 0, 7-bit addressing is used. When 1, 10-bit addressing is used. 0h = Enable 7-bit addressing 1h = Enable 10-bit addressing
14	OAREN	R/W	1h	I2C Target Own Address Enable 0h = Disable OAR address 1h = Enable OAR address
13-10	RESERVED	R	0h	
9-0	OAR	R/W	0h	I2C Target Own Address: This field specifies bits A9 through A0 of the Target address. In 7-bit addressing mode as selected by I2CSOAR.MODE bit, the top 3 bits are don't care 0h = Smallest value 3FFh = Highest possible value

13.3.43 SOAR2 Register (Offset = 1254h) [Reset = 00000000h]

SOAR2 is shown in Figure 13-60 and described in Table 13-65.

Return to the Summary Table.

This register consists of seven address bits that identify the alternate address for the I2C device on the I2C bus.

Figure 13-60. SOAR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	OAR2_MASK						
R-0h	R/W-0h						
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OAR2EN	OAR2						
R/W-0h	R/W-0h						

Table 13-65. SOAR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	
22-16	OAR2_MASK	R/W	0h	I2C Target Own Address 2 Mask: This field specifies bits A6 through A0 of the Target address. The bits with value '1' in SOAR2.OAR2_MASK field will make the corresponding incoming address bits to match by default regardless of the value inside SOAR2.OAR2; that is, the corresponding SOAR2.OAR2 bit is a don't care. 0h = Minimum Value 7Fh = Maximum Value
15-8	RESERVED	R	0h	
7	OAR2EN	R/W	0h	I2C Target Own Address 2 Enable 0h = The alternate address is disabled. 1h = Enables the use of the alternate address in the OAR2 field.
6-0	OAR2	R/W	0h	I2C Target Own Address 2 This field specifies the alternate OAR2 address. 0h = Smallest value 7Fh = Highest possible value

13.3.44 SCTR Register (Offset = 1258h) [Reset = 00000404h]

SCTR is shown in Figure 13-61 and described in Table 13-66.

Return to the Summary Table.

I2C Target Control Register

Figure 13-61. SCTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					SWUEN	EN_DEFDEVADR	EN_ALRESPADR
R-0h					R/W-1h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EN_DEFHOSTADR	RXFULL_ON_REQ	TXWAIT_STALE_TXFIFO	TXTRIG_TXMODE	TXEMPTY_ON_TREQ	SCLKSTRETCH	GENCALL	ACTIVE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-1h	R/W-0h	R/W-0h

Table 13-66. SCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	
10	SWUEN	R/W	1h	Target Wakeup Enable 0h = When 0, the Target is not allowed to clock stretch on START detection 1h = When 1, the Target is allowed to clock stretch on START detection and wait for faster clock to be available. This allows clean wake up support for I2C in low power mode use cases
9	EN_DEFDEVADR	R/W	0h	Enable Default device address 0h = When this bit is 0, the default device address is not matched. NOTE: it may still be matched if programmed inside SOAR/SOAR2. 1h = When this bit is 1, default device address of 7'h110_0001 is always matched by the Target address match logic.
8	EN_ALRESPADR	R/W	0h	Enable Alert Response Address 0h = When this bit is 0, the alert response address is not matched. NOTE: it may still be matched if programmed inside SOAR/SOAR2 1h = When this bit is 1, alert response address of 7'h000_1100 is always matched by the Target address match logic.
7	EN_DEFHOSTADR	R/W	0h	Enable Default Host Address 0h = When this bit is 0, the default host address is not matched NOTE: it may still be matched if programmed inside SOAR/SOAR2 1h = When this bit is 1, default host address of 7'h000_1000 is always matched by the Target address match logic.

Table 13-66. SCTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RXFULL_ON_RREQ	R/W	0h	Rx full interrupt generated on RREQ condition as indicated in SSR 0h = When 0, RIS:SRXFULL will be set when only the Target RX FIFO is full. This allows the SRXFULL interrupt to be used to indicate that the I2C bus is being clock stretched and that the FW must either read the RX FIFO or ACK/NACK the current Rx byte. 1h = When 1, RIS:SRXFULL will be set when the Target State Machine is in the RX_WAIT or RX_ACK_WAIT states which occurs when the I2C transaction is clock stretched because the RX FIFO is full or the ACKOEN has been set and the state machine is waiting for FW to ACK/NACK the current byte.
5	TXWAIT_STALE_TXFIFO	R/W	0h	Tx transfer waits when stale data in Tx FIFO. This prevents stale bytes left in the TX FIFO from automatically being sent on the next I2C packet. Note: this should be used with TXEMPTY_ON_TREQ set to prevent the Target State Machine from waiting for TX FIFO data without an interrupt notification when the FIFO data is stale. 0h = When 0, the TX FIFO empty signal to the Target State Machine indicates that the TX FIFO is empty. 1h = When 1, the TX FIFO empty signal to the Target State Machine will indicate that the TX FIFO is empty or that the TX FIFO data is stale. The TX FIFO data is determined to be stale when there is data in the TX FIFO when the Target State Machine leaves the TXMODE as defined in the SSR register. This can occur is a Stop or timeout occur when there are bytes left in the TX FIFO.
4	TXTRIG_TXMODE	R/W	0h	Tx Trigger when Target FSM is in Tx Mode 0h = No special behavior 1h = When 1, RIS:TXFIFOTRG will be set when the Target TX FIFO has reached the trigger level AND the Target State Machine is in the TXMODE as defined in the SSR register. When cleared RIS:TXFIFOTRG will be set when the Target TX FIFO is at or above the trigger level. This setting can be used to hold off the TX DMA until a transaction starts. This allows the DMA to be configured when the I2C is idle but have it wait till the transaction starts to load the Target TX FIFO, so it can load from a memory buffer that might be changing over time.
3	TXEMPTY_ON_TREQ	R	0h	Tx Empty Interrupt on TREQ 0h = When 0, RIS:STXEMPTY will be set when only the Target TX FIFO is empty. This allows the STXEMPTY interrupt to be used to indicate that the I2C bus is being clock stretched and that Target TX data is required. 1h = When 1, RIS:STXEMPTY will be set when the Target State Machine is in the TX_WAIT state which occurs when the TX FIFO is empty AND the I2C transaction is clock stretched waiting for the FIFO to receive data.
2	SCLKSTRETCH	R/W	1h	Target Clock Stretch Enable 0h = Target clock stretching is disabled 1h = Target clock stretching is enabled
1	GENCALL	R/W	0h	General call response enable Modify only when UCSWRST = 1. 0b = Do not respond to a general call 1b = Respond to a general call 0h = Do not respond to a general call 1h = Respond to a general call
0	ACTIVE	R/W	0h	Device Active. Setting this bit enables the Target functionality. 0h = Disables the I2C Target operation. 1h = Enables the I2C Target operation.

13.3.45 SSR Register (Offset = 125Ch) [Reset = 00000000h]

SSR is shown in Figure 13-62 and described in Table 13-67.

Return to the Summary Table.

This register functions as a control register when written, and a status register when read.

Figure 13-62. SSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					ADDRMATCH		
R-0h					R-0h		
15	14	13	12	11	10	9	8
ADDRMATCH							STALE_TXFIFO
R-0h							R-0h
7	6	5	4	3	2	1	0
TXMODE	BUSBSY	QCMDRW	QCMDST	OAR2SEL	RXMODE	TREQ	RREQ
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-67. SSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	
18-9	ADDRMATCH	R	0h	Indicates the address for which Target address match happened 0h = Minimum Value 3FFh = Maximum Value
8	STALE_TXFIFO	R	0h	Stale Tx FIFO 0h = Tx FIFO is not stale 1h = The TX FIFO is stale. This occurs when the TX FIFO was not emptied during the previous I2C transaction.
7	TXMODE	R	0h	Target FSM is in TX MODE 0h = The Target State Machine is not in TX_DATA, TX_WAIT, TX_ACK or ADDR_ACK state with the bus direction set to read. 1h = The Target State Machine is in TX_DATA, TX_WAIT, TX_ACK or ADDR_ACK state with the bus direction set to read.
6	BUSBSY	R	0h	I2C bus is busy 0h = The I2C Bus is not busy 1h = The I2C Bus is busy. This is cleared on a timeout.
5	QCMDRW	R	0h	Quick Command Read / Write This bit only has meaning when the QCMDST bit is set. Value Description: 0: Quick command was a write 1: Quick command was a read 0h = Quick command was a write 1h = Quick command was a read
4	QCMDST	R	0h	Quick Command Status Value Description: 0: The last transaction was a normal transaction or a transaction has not occurred. 1: The last transaction was a Quick Command transaction 0h = The last transaction was a normal transaction or a transaction has not occurred. 1h = The last transaction was a Quick Command transaction.

Table 13-67. SSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	OAR2SEL	R	0h	OAR2 Address Matched This bit gets reevaluated after every address comparison. 0h = Either the OAR2 address is not matched or the match is in legacy mode. 1h = OAR2 address matched and ACKed by the Target.
2	RXMODE	R	0h	Target FSM is in Rx MODE 0h = The Target State Machine is not in the RX_DATA, RX_ACK, RX_WAIT, RX_ACK_WAIT or ADDR_ACK state with the bus direction set to write. 1h = The Target State Machine is in the RX_DATA, RX_ACK, RX_WAIT, RX_ACK_WAIT or ADDR_ACK state with the bus direction set to write.
1	TREQ	R	0h	Transmit Request 0h = No outstanding transmit request. 1h = The I2C controller has been addressed as a Target transmitter and is using clock stretching to delay the Controller until data has been written to the STXDATA FIFO (Target TX FIFO is empty).
0	RREQ	R	0h	Receive Request 0h = No outstanding receive data. 1h = The I2C controller has outstanding receive data from the I2C Controller and is using clock stretching to delay the Controller until the data has been read from the SRXDATA FIFO (Target RX FIFO is full).

13.3.46 SRXDATA Register (Offset = 1260h) [Reset = 00000000h]

SRXDATA is shown in Figure 13-63 and described in Table 13-68.

Return to the Summary Table.

I2C Target RX FIFO Read Data Byte
This field contains the current byte being read in the RX FIFO stack.
If the FIFO is disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register.

Figure 13-63. SRXDATA Register

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
RESERVED																								VALUE							
R-0h																								R-0h							

Table 13-68. SRXDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	VALUE	R	0h	Received Data. This field contains the last received data. 0h = Smallest value FFh = Highest possible value

13.3.47 STXDATA Register (Offset = 1264h) [Reset = 00000000h]

STXDATA is shown in Figure 13-64 and described in Table 13-69.

Return to the Summary Table.

I2C Target Transmit Data Register.
This register is the transmit data register (the interface to the FIFOs). For transmitted data, if the FIFO is enabled, data written to this location is pushed onto the transmit FIFO. If the FIFO is disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO).

Figure 13-64. STXDATA Register

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
RESERVED																								VALUE							
R-0h																								R/W-0h							

Table 13-69. STXDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	VALUE	R/W	0h	Transmit Data This byte contains the data to be transferred during the next transaction. 0h = Smallest value FFh = Highest possible value

13.3.48 SFIFOCTL Register (Offset = 126Ch) [Reset = 00000000h]

SFIFOCTL is shown in Figure 13-65 and described in Table 13-70.

Return to the Summary Table.

I2C Target FIFO Control

Figure 13-65. SFIFOCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RXFLUSH	RESERVED					RXTRIG	
R/W-0h	R-0h					R/W-0h	
7	6	5	4	3	2	1	0
TXFLUSH	RESERVED					TXTRIG	
R/W-0h	R-0h					R/W-0h	

Table 13-70. SFIFOCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	RXFLUSH	R/W	0h	RX FIFO Flush Setting this bit will Flush the RX FIFO. Before clearing this bit to stop Flush the RXFIFOCNT should be checked to be 0 and indicating that the Flush has completed. 0h = Do not Flush FIFO 1h = Flush FIFO
14-11	RESERVED	R	0h	
10-8	RXTRIG	R/W	0h	RX FIFO Trigger Indicates at what fill level in the RX FIFO a trigger will be generated. Note: Programming RXTRIG to 0x0 has no effect since no data is present to transfer out of RX FIFO. 4h = Trigger when RX FIFO contains >= 5 byte 5h = Trigger when RX FIFO contains >= 6 byte 6h = Trigger when RX FIFO contains >= 7 byte 7h = Trigger when RX FIFO contains >= 8 byte
7	TXFLUSH	R/W	0h	TX FIFO Flush Setting this bit will Flush the TX FIFO. Before clearing this bit to stop Flush the TXFIFOCNT should be checked to be 8 and indicating that the Flush has completed. 0h = Do not Flush FIFO 1h = Flush FIFO
6-3	RESERVED	R	0h	
2-0	TXTRIG	R/W	0h	TX FIFO Trigger Indicates at what fill level in the TX FIFO a trigger will be generated. 4h = Trigger when TX FIFO contains ≤ 4 byte 5h = Trigger when TX FIFO contains ≤ 5 byte 6h = Trigger when TX FIFO contains ≤ 6 byte 7h = Trigger when TX FIFO contains ≤ 7 byte

13.3.49 SFIFOSR Register (Offset = 1270h) [Reset = 00000800h]

SFIFOSR is shown in Figure 13-66 and described in Table 13-71.

Return to the Summary Table.

I2C Target FIFO Status Register
Note: this Register should only be read when BUSY is 0

Figure 13-66. SFIFOSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TXFLUSH	RESERVED			TXFIFOCNT			
R/W-0h	R-0h			R-8h			
7	6	5	4	3	2	1	0
RXFLUSH	RESERVED			RXFIFOCNT			
R/W-0h	R-0h			R-0h			

Table 13-71. SFIFOSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15	TXFLUSH	R/W	0h	TX FIFO Flush When this bit is set a Flush operation for the TX FIFO is active. Clear the TXFLUSH bit in the control register to stop. 0h = FIFO Flush not active 1h = FIFO Flush active
14-12	RESERVED	R	0h	
11-8	TXFIFOCNT	R	8h	Number of Bytes which could be put into the TX FIFO 0h = Smallest value 8h = Highest possible value
7	RXFLUSH	R/W	0h	RX FIFO Flush When this bit is set a Flush operation for the RX FIFO is active. Clear the RXFLUSH bit in the control register to stop. 0h = FIFO Flush not active 1h = FIFO Flush active
6-4	RESERVED	R	0h	
3-0	RXFIFOCNT	R	0h	Number of Bytes which could be read from the RX FIFO 0h = Smallest value 8h = Highest possible value

Chapter 14

CRC

The cyclic redundancy check (CRC) accelerator generates signatures for a given data sequence based on the CRC16-CCITT polynomial.

14.1 CRC Overview

The CRC accelerator produces CRC signatures for given sequences of data. The CRC16-CCITT functions are supported. Identical input data sequences result in identical CRC signatures when the CRC is initialized with a fixed seed value. Different sequences of input data, in general, result in different signatures for a given CRC function.

Key features of the CRC accelerator include:

- Support for CRC16-CCITT
- Fast single-cycle computation of new CRC output for each data input (no wait states)
- Support for input/output bit reversal
- Support for little or big endian operation
- Byte, half-word, or word input to CRCIN
- 512-word CRCIN_IDX input field in which all addresses are mapped to CRCIN, supporting use of a standard C-style memcpy() routine to load data into the CRC module for data lengths up to 2KB

14.1.1 CRC16-CCITT

For CRC16-CCITT, the CRC signature is generated based on the polynomial given in the 16-bit CCITT standard, as shown in the equation below.

$$f(x)=x^{16}+x^{12}+x^{5}+1 \quad (12)$$

The CRC16-CCITT digest size is 16 bits (half-word).

14.2 CRC Operation

The CRC generator is initialized by configuring the desired mode of operation in the CRCCTRL register, followed by writing the seed value to the CRCSEED register. After the seed is loaded to the CRCSEED register, the CRCOUT register will reflect the SEED value loaded to CRCSEED.

Note

If the endianness mode is configured to big endian before the seed value is written to CRCSEED, then the byte order of the seed value written to CRCSEED is swapped when the seed value is loaded into the CRC module.

Once initialized, data can be input into the CRC generator by writing to the CRCIN register using byte (8-bit), half-word (16-bit), or word (32-bit) writes. The CPU or the DMA can be used to move input data into the CRC accelerator.

Note

Byte writes need not be word aligned; a byte write to any byte location in CRCIN will be interpreted the same way (adding the 8 written bits to the computed CRC). Half-word writes also need not be word aligned, but they must be half-word aligned. For example, a half-word can be written to BIT0-BIT15 or BIT16-BIT32 of CRCIN, but not to BIT8-BIT23.

When using the CRC generator to verify a data set, all data to be included in the CRC calculation must be written to the CRCIN register in the same order as was used to calculate the original CRC signature. When using the CRC generator to create a new signature to be used in the future for verification, be sure to load the data the same way and with the same settings when performing verification.

The current CRC output can be read at any time by reading the CRCOUT register.

14.2.1 CRC Generator Implementation

The CRC generator is implemented with a set of XOR trees. After a set of 8, 16, or 32 bits is provided to the CRC accelerator by writing to the CRCIN register, a calculation for the whole set of input bits is performed. When

new data is written to CRCIN, the CRC generator updates the CRC output in a single cycle. Bus wait states are not required to load data back-to-back into the CRC generator.

14.2.2 Configuration

The CRC accelerator supports polynomial selection, bit reversal selection, and byte order (endianness) selection. This section describes these configuration aspects.

The CRC accelerator must be enabled before being used through the PWREN register (see peripheral power enable).

The CRC accelerator is in power domain 1 (PD1), and as such can only be active in RUN or SLEEP mode. If the CRC accelerator is configured to be enabled by application software, and the device enters STOP or STANDBY mode, SYSCTL will force the CRC into a disabled state until the device exists STOP or STANDBY mode. All CRC register contents are retained when the CRC is forced to a disabled state in STOP or STANDBY mode.

The CRC module only runs from the PD1 bus clock (MCLK).

14.2.2.1 Bit Order

The various CRC standards were defined in the era of main frame computers. At that time, BIT0 was treated as the MSB. In modern computing, BIT0 is typically the LSB.

The Arm Cortex-M0+ CPU treats BIT0 as the LSB, as is typical in modern CPUs and MCUs. This sometimes causes confusion, because BIT0 has been treated as the LSB in some cases and as the MSB in other cases. Therefore, the CRC accelerator provides a bit order reversal capability to support both conventions.

Bit order reversal can be enabled by setting the BITREVERSE bit in the CRCCTRL register, giving the following behavior for input and output data:

- **Input data:** The bit order of each input byte written to the CRCIN register is reversed before it is passed to the CRC generator to be used for CRC calculation
- **Output data:** The bit order of the 16-bit CRC result is reversed when read from the CRCOUT register

Note

If input data must be provided bit-reversed, but the output is to be read not reversed, the BITREVERSE bit can be set before loading data to CRCIN and then cleared after all data is written to CRCIN but before the resulting signature is read from CRCOUT. Likewise, it is possible to load input data to CRCIN with BITREVERSE cleared (not reversed), and flip the output before reading it (by setting BITREVERSE before reading CRCOUT).

14.2.2.2 Byte Swap

The bit OUTPUT_BYTESWAP in the register CRCCTRL can be used to enable or disable CRC output byte swap. This bit controls whether the output is byte-swapped upon a read of the CRCOUT register. If CRCOUT is accessed as a half-word, and the OUTPUT_BYTESWAP is set to 1, then the two bytes in the 16-bit access are swapped and returned. Using B0, B1, B2 and B3 to identify Byte 0, Byte 1, Byte 2, Byte 3. B1 is returned as B0 and B0 is returned as B1. If CRCOUT is accessed as a word, and the OUTPUT_BYTESWAP is set to 1, then the four bytes in the 32-bit read are swapped. B3 is returned as B0, B2 is returned as B1, B1 is returned as B2 and B0 is returned as B3.

Note that if the CRC POLYSIZE is 16-bit and a 32-bit read of CRCOUT is performed with OUTPUT_BYTESWAP enabled, then the output is: MSB LSB 0x0 0x0 B0 B1. If the CRC POLYSIZE is 16-bit and a 32-bit read of CRCOUT is performed with OUTPUT_BYTESWAP disabled, then the output is: MSB LSB 0x0 0x0 B1 B0.

14.2.2.3 Byte Order

When working with half-word or word input data, the input byte order can be configured as either little endian or big endian. The default configuration is little endian. To reverse the byte order when using half-word or word inputs, set the INPUT_ENDIANNESS bit in the CRCCTRL register.

Reversing the endianness will cause the following translation for half-word and word writes:

Table 14-1. CRCIN Byte Order Translation

Endianness	Data Written to CRCIN	Data Applied to CRC Logic
0 (little)	0x1234	0x1234
1 (big)	0x1234	0x3412
0 (little)	0x12345678	0x12345678
1 (big)	0x12345678	0x78563412

Note

If the ENDIANNESS bit is set before the seed value is written to CRCSEED, then the byte order of the seed value written to CRCSEED is also reversed when it is loaded into the CRC, and the seed value will read back reversed when reading the CRCOUT register after writing to the CRCSEED register.

14.2.2.4 CRC C Library Compatibility

To simplify loading of data by software into the CRC, the CRC accelerator provides a 512-word (2KB) CRCIN_IDX region. Within the CRCIN_IDX region, a write to any word is re-mapped as, and functionally equivalent to, a write to the CRCIN register. This mechanism enables the use of the standard C library *memcpy()* routine to copy data from SRAM or flash into the CRC, provided that the source data is less than 2KB (2,048 bytes).

14.3 CRC Registers

Table 14-2 lists the memory-mapped registers for the CRC registers. All register offset addresses not listed in Table 14-2 should be considered as reserved locations and the register contents should not be modified.

Table 14-2. CRC Registers

Offset	Acronym	Register Name	Group	Section
800h	PWREN	Power enable		Go
804h	RSTCTL	Reset Control		Go
814h	STAT	Status Register		Go
1004h	CLKSEL	Clock Select		Go
10FCh	DESC	Module Description		Go
1100h	CRCCTRL	CRC Control Register		Go
1104h	CRCSEED	CRC Seed Register		Go
1108h	CRCIN	CRC Input Data Register		Go
110Ch	CRCOUT	CRC Output Result Register		Go
1800h + formula	CRCIN_IDX[y]	CRC Input Data Array Register		Go

Complex bit access types are encoded to fit into small table cells. Table 14-3 shows the codes that are used for access types in this section.

Table 14-3. CRC Access Type Codes

Access Type	Code	Description
Read Type		
H	H	Set or cleared by hardware
R	R	Read
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.3.1 PWREN (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 14-1 and described in Table 14-4.

Return to the Summary Table.

Register to control the power state

Figure 14-1. PWREN

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R/W-0h							R/WK-0h

Table 14-4. PWREN Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R/W	0h	
0	ENABLE	R/WK	0h	Enable the power KEY must be set to 26h to write to this bit. 0h = Disable Power 1h = Enable Power

14.3.2 RSTCTL (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 14-2 and described in Table 14-5.

Return to the Summary Table.

Register to control reset assertion and de-assertion

Figure 14-2. RSTCTL

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
R-0h						WK-0h	WK-0h

Table 14-5. RSTCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	R	0h	Reserved
1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Clear reset sticky bit
0	RESETASSERT	WK	0h	Assert reset to the peripheral KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Assert reset

14.3.3 STAT (Offset = 814h) [Reset = 00000000h]

STAT is shown in Figure 14-3 and described in Table 14-6.

Return to the Summary Table.

peripheral enable and reset status

Figure 14-3. STAT

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

Table 14-6. STAT Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	RESETSTKY	R	0h	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	Reserved

14.3.4 CLKSEL (Offset = 1004h) [Reset = 00000001h]

CLKSEL is shown in Figure 14-4 and described in Table 14-7.

Return to the Summary Table.

Clock source selection

Figure 14-4. CLKSEL

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

Table 14-7. CLKSEL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	MCLK_SEL	R	1h	Selects main clock (MCLK) if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source

14.3.5 DESC (Offset = 10FCh) [Reset = 20117010h]

DESC is shown in Figure 14-5 and described in Table 14-8.

Return to the Summary Table.

This register identifies the peripheral and its exact version.

Figure 14-5. DESC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-2011h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				INSTNUM				MAJREV				MINREV			
R-7h				R-0h				R-1h				R-0h			

Table 14-8. DESC Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	2011h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness. 0h = Smallest value FFFFh = Highest possible value
15-12	FEATUREVER	R	7h	Feature Set for the module *instance* 0h = Smallest value Fh = Highest possible value
11-8	INSTNUM	R	0h	Instance Number within the device. This will be a parameter to the RTL for modules that can have multiple instances 0h = Smallest value Fh = Highest possible value
7-4	MAJREV	R	1h	Major rev of the IP 0h = Smallest value Fh = Highest possible value
3-0	MINREV	R	0h	Minor rev of the IP 0h = Smallest value Fh = Highest possible value

14.3.6 CRCCTRL (Offset = 1100h) [Reset = X]

CRCCTRL is shown in Figure 14-6 and described in Table 14-9.

Return to the Summary Table.

CRC Control Register. Configuration control of the CRC.

Figure 14-6. CRCCTRL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			OUTPUT_BYTESWAP	RESERVED	INPUT_ENDIANNESS	BITREVERSE	POLYSIZE
R-0h			R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-9. CRCCTRL Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	OUTPUT_BYTESWAP	R/W	0h	CRC Output Byteswap Enable. This bit controls whether the output is byte-swapped upon a read of the CRCOUT register. If CRCOUT is accessed as a half-word, and the OUTPUT_BYTESWAP is set to 1, then the two bytes in the 16-bit access are swapped and returned. B1 is returned as B0 B0 is returned as B1 If CRCOUT is accessed as a word, and the OUTPUT_BYTESWAP is set to 1, then the four bytes in the 32-bit read are swapped. B3 is returned as B0 B2 is returned as B1 B1 is returned as B2 B0 is returned as B3 Note that if the CRC POLYSIZE is 16-bit and a 32-bit read of CRCOUT is performed with OUTPUT_BYTESWAP enabled, then the output is: MSB LSB 0x0 0x0 B0 B1 If the CRC POLYSIZE is 16-bit and a 32-bit read of CRCOUT is performed with OUTPUT_BYTESWAP disabled, then the output is: MSB LSB 0x0 0x0 B1 B0 0h = Output byteswapping is disabled 1h = Output byteswapping is enabled.
3	RESERVED	R	0h	Reserved
2	INPUT_ENDIANNESS	R/W	0h	CRC Endian. This bit indicates the byte order within a word or half word of input data. 0h = LSB is lowest memory address and first to be processed. 1h = LSB is highest memory address and last to be processed.

Table 14-9. CRCCTRL Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	BITREVERSE	R/W	0h	CRC Bit Input and output Reverse. This bit indicates that the bit order of each input byte used for the CRC calculation is reversed before it is passed to the generator, and that the bit order of the calculated CRC is be reversed when read from CRC_RESULT. 0h = Bit order is not reversed. 1h = Bit order is reversed.
0	POLYSIZE	R/W	0h	This bit indicates which CRC calculation is performed by the generator. 0h = CRC-32 ISO-3309 calculation is performed 1h = CRC-16 CCITT is performed

14.3.7 CRCSEED (Offset = 1104h) [Reset = 00000000h]

CRCSEED is shown in Figure 14-7 and described in Table 14-10.

Return to the Summary Table.

CRC Seed Register. The Data written to this register is used to initialize the CRC result with this SEED value. Note that in 16-bit mode only the lower 16-bits of this value are used. After writing this register the CRC Output Result Register will reflect this value.

Figure 14-7. CRCSEED

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
SEED																															
W-0h																															

Table 14-10. CRCSEED Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEED	W	0h	Seed Data 00000000h = Minimum value FFFFFFFFh = Maximum value

14.3.8 CRCIN (Offset = 1108h) [Reset = 00000000h]

CRCIN is shown in Figure 14-8 and described in Table 14-11.

Return to the Summary Table.

CRC Input Data Register. The Data written to this register is used along with the current CRC result to calculate the next CRC result. This is done in a single clock cycle and requires no wait states. This register can by written as a byte, half word or word transfer and the correct number of bits will be used for the next CRC result.
This register is also mapped to a range of registers starting at 0xTDB_X000 and ending at 0xTDB_XFFF to allow memcpy to be used instead of DMA for CRC calculations that do not exceed the bounds of the memory range of this register.

Figure 14-8. CRCIN

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
DATA																															
W-0h																															

Table 14-11. CRCIN Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	W	0h	Input Data 00000000h = Minimum value FFFFFFFFh = Maximum value

14.3.9 CRCOUT (Offset = 110Ch) [Reset = 00000000h]

CRCOUT is shown in Figure 14-9 and described in Table 14-12.

Return to the Summary Table.

CRC Output Result Register. This register stores the result of the current CRC calculation. Note when configured for 16-bit mode the upper bits will read back 0. Note that if output inversion is set in the CRC Control register it will be applied.

Figure 14-9. CRCOUT

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
RESULT																															
R-0h																															

Table 14-12. CRCOUT Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESULT	R	0h	Result 00000000h = Minimum value FFFFFFFFh = Maximum value

14.3.10 CRCIN_IDX[y] (Offset = 1800h + formula) [Reset = 00000000h]

CRCIN_IDX[y] is shown in Figure 14-10 and described in Table 14-13.

Return to the Summary Table.

This register is dual mapped to CRCIN and is intended to allow operation with C memcpy routine.

Offset = 1800h + (y * 4h); where y = 0h to 1FFh

Figure 14-10. CRCIN_IDX[y]

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
DATA																															
HW-0h																															

Table 14-13. CRCIN_IDX[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	HW	0h	Input Data 00000000h = Minimum value FFFFFFFFh = Maximum value

Chapter 15

Timers (TIMx)

The timer module (TIMx) is a timer counting module with multiple compare/capture blocks. Based on the device, two types of timers are available: general-purpose timers (TIMG) and advanced control timers (TIMA). Both timers include many common features that can be used for a variety of functions such as measuring the input signal edge and period (capture mode) or generating output waveforms (compare mode output) like PWMs. See the device-specific data sheet to determine which timers and timer instances are available.

15.1 TIMx Overview

The timer module (TIMx) is a timer counting module with multiple compare/capture blocks. Based on the device, two types of timers are available: general-purpose timers (TIMG) and advanced control timers (TIMA). Both timers use a common timer architecture, which allows for easy migration between timer instances with common functions. This minimizes the need to write extra software for timer-based applications and allows for easy porting and maintenance between TIMx instances.

Note

See Section 15.1.3 to determine the common features available between TIMG and TIMA instances.

In this section:

- “TIMx” indicates a common feature available on TIMG and TIMA.
- "TIMA" indicates a feature available only on TIMA.
- "TIMG" indicates a feature available only on TIMG.

15.1.1 TIMG Overview

The TIMG module consists of 16-bit auto reload counters driven by a programmable prescaler with two capture/compare (CC) blocks for multiple capture/compares, PWM outputs, and interval timing. TIMG also has extensive event generation capabilities, including counter overflow, reload, and capture/compare actions for a variety of use cases.

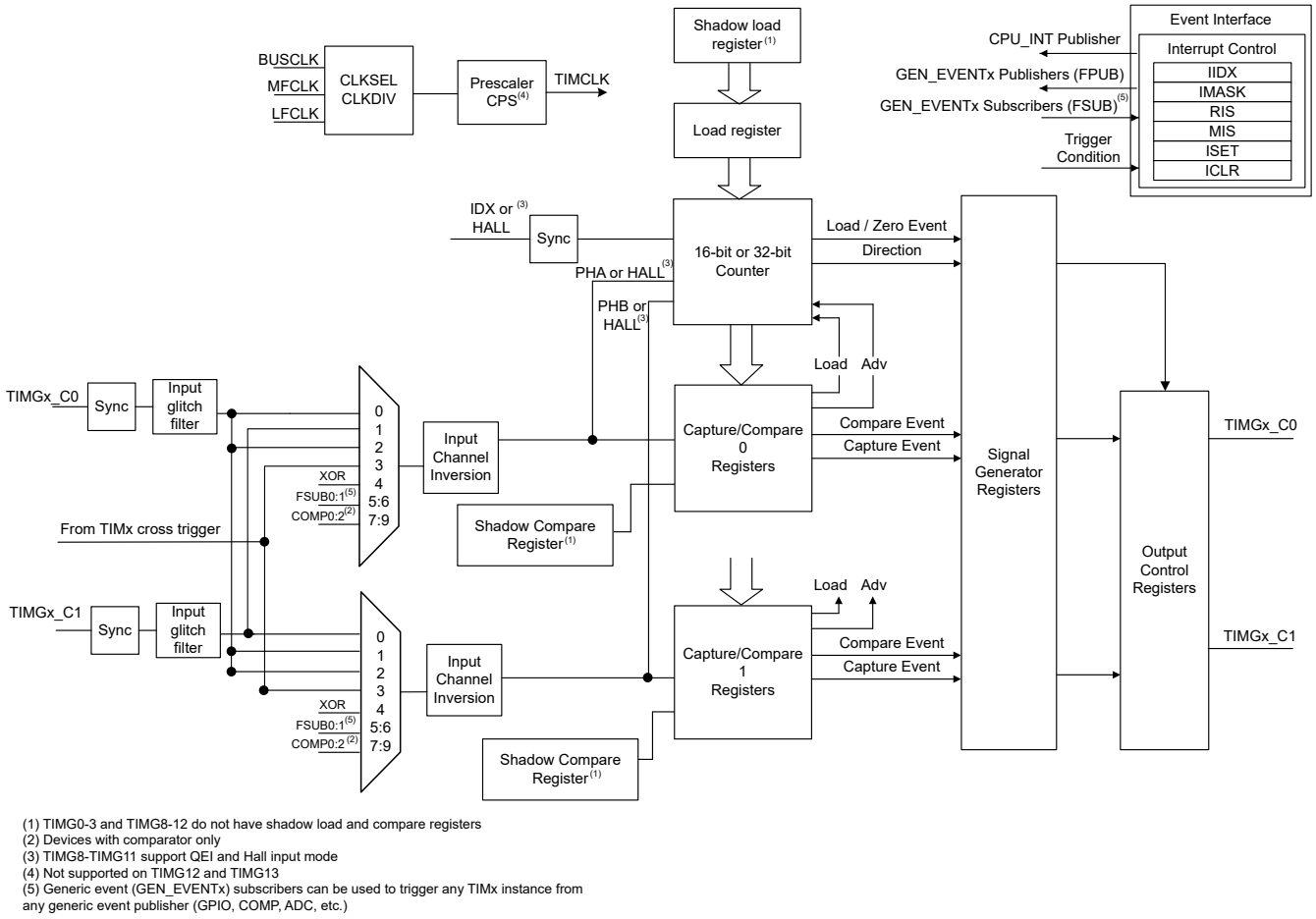
15.1.1.1 TIMG Features

Specific features for TIMG include:

- 16-bit up, down, or up-down counter, with repeat-reload mode
- 8-bit programmable prescaler to divide the counter clock frequency
- Up to two independent channels for
  - Output compare
  - Input capture
  - PWM output (Edge-Aligned and Center-Aligned)
  - One-shot mode
- Shadow register mode for load and compare values (see Section 15.2.4)
- Support for quadrature encoder interface (QEI) (see Section 15.2.3.1.3)
- 3-input Hall sensor mode for position sensing and speed computation (see Section 15.2.3.1.4)
- Support synchronization and cross trigger among different TIMx instances in the same power domain (see Section 15.2.7)
- Support CPU interrupt generation and cross peripherals (such as ADC, DAC, etc.) using the Event (see Section 15.2.9)

15.1.1.2 Functional Block Diagram

Figure 15-1 shows the TIMG block diagram.



(1) TIMG0-3 and TIMG8-12 do not have shadow load and compare registers
(2) Devices with comparator only
(3) TIMG8-TIMG11 support QEI and Hall input mode
(4) Not supported on TIMG12 and TIMG13
(5) Generic event (GEN_EVENTx) subscribers can be used to trigger any TIMx instance from any generic event publisher (GPIO, COMP, ADC, etc.)

Figure 15-1. TIMG Functional Block Diagram

15.1.2 TIMA Overview

The TIMA module consists of a 16-bit auto reload counter driven by a programmable prescaler with up to four capture/compare (CC) blocks for multiple capture/compares, PWM outputs with deadband insertion, and interval timing. TIMA has extensive event generation capabilities from different counter events such as overflow, reload, and from each of the capture/compare registers. It also has the hardware design to handle the fault signal generated by internal or external circuitry to indicate a fault in the system.

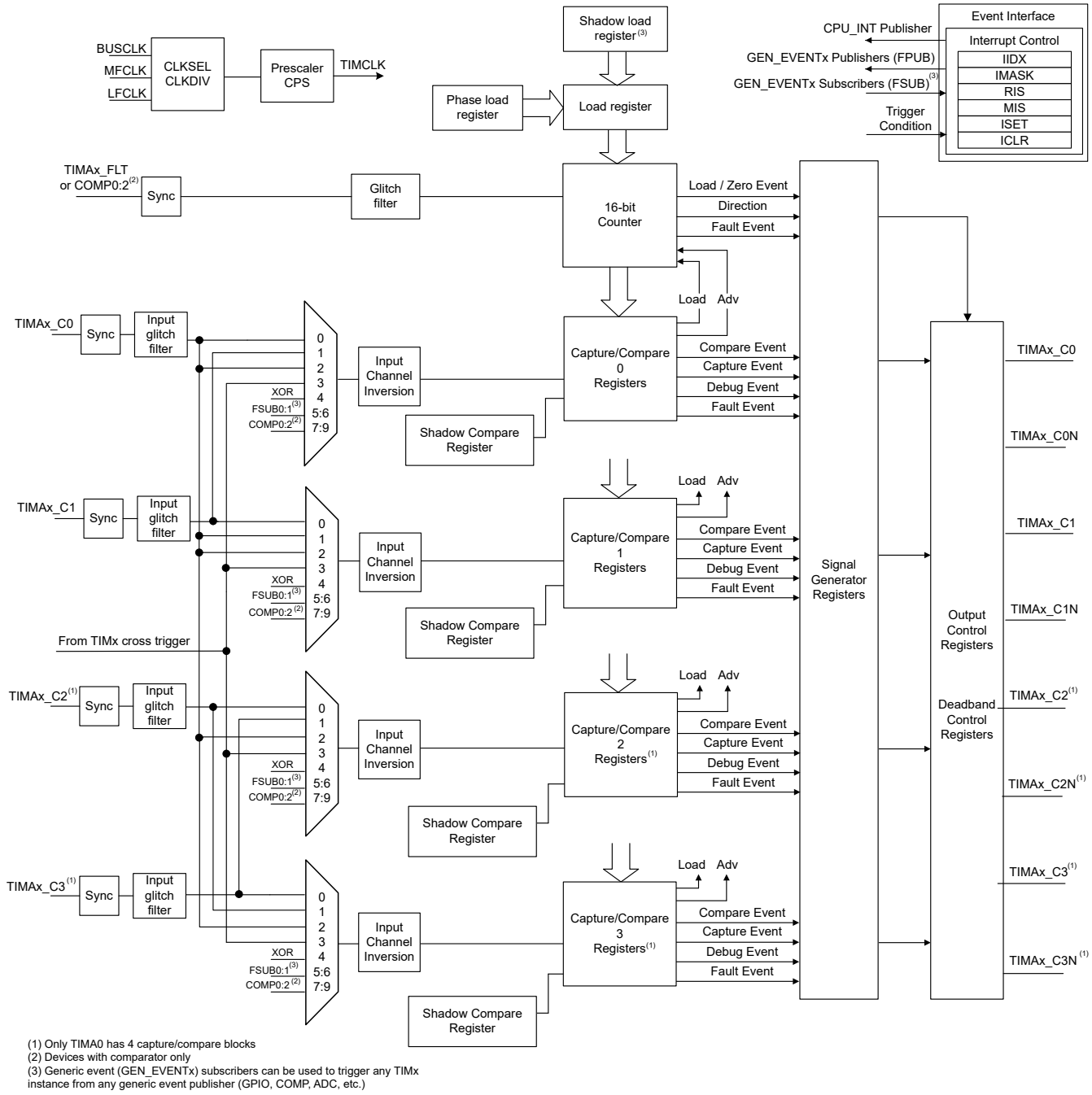
15.1.2.1 TIMA Features

- 16-bit up, down, or up-down counter, with repeat-reload mode
- Selectable and configurable clock source
- 8-bit programmable prescaler to divide the counter clock frequency
- Repeat counter to generate an interrupt or event only after a given number of cycles of the counter (see Section 15.2.1.2)
- Up to four independent channels for:
  - Output compare
  - Input capture
  - PWM output (Edge-Aligned and Center-Aligned)
  - One-shot mode
- Two additional capture/compare channels for internal events (CC4/CC5)
- Shadow register for load and compare values (see Section 15.2.4)
- Complementary PWM output with programmable deadband insertion (see Section 15.2.5.2.4)
- Asymmetric PWM output (see Section 15.2.2.5)

- Fault handling mechanisms to ensure the output signals are in a safe user-defined state when a fault condition is encountered (see Section 15.2.6)
- Support synchronization and cross trigger among different TIMx instances in the same power domain (see Section 15.2.7)
- Support CPU interrupt generation and cross peripherals (such as ADC, DAC, etc.) using the Event (see Section 15.2.9)

15.1.2.2 Functional Block Diagram

Figure 15-2 shows the TIMA block diagram.



(1) Only TIMA0 has 4 capture/compare blocks
(2) Devices with comparator only
(3) Generic event (GEN_EVENTx) subscribers can be used to trigger any TIMx instance from any generic event publisher (GPIO, COMP, ADC, etc.)

Figure 15-2. TIMA Block Diagram

15.1.3 TIMx Instance Configuration

Table 15-1 shows the TIMx instance configuration for TIMA and TIMG instances.

Table 15-1. TIMx Instance Configuration

Instance	Counter Resolution	Prescaler	Repeat Counter	CCP Channels (External/ Internal)	External PWM Channels	Phase Load	Shadow Load	Shadow CCs	Deadband	Fault Handler	QEI / Hall Input Mode
TIMG0	16-bit	8-bit	-	2	2	-	-	-	-	-	-
TIMG1	16-bit	8-bit	-	2	2	-	-	-	-	-	-
TIMG2	16-bit	8-bit	-	2	2	-	-	-	-	-	-
TIMG3	16-bit	8-bit	-	2	2	-	-	-	-	-	-
TIMG4	16-bit	8-bit	-	2	2	-	Yes	Yes	-	-	-
TIMG5	16-bit	8-bit	-	2	2	-	Yes	Yes	-	-	-
TIMG6	16-bit	8-bit	-	2	2	-	Yes	Yes	-	-	-
TIMG7	16-bit	8-bit	-	2	2	-	Yes	Yes	-	-	-
TIMG8	16-bit	8-bit	-	2	2	-	-	-	-	-	Yes
TIMG9	16-bit	8-bit	-	2	2	-	-	-	-	-	Yes
TIMG10	16-bit	8-bit	-	2	2	-	-	-	-	-	Yes
TIMG11	16-bit	8-bit	-	2	2	-	-	-	-	-	Yes
TIMG12	32-bit	-	-	2	2	-	-	Yes	-	-	-
TIMG13	32-bit	-	-	2	2	-	-	Yes	-	-	-
TIMG14	16-bit	8-bit	-	4	4	-	-	-	-	-	-
TIMA0	16-bit	8-bit	Yes	4/2	8	Yes	Yes	Yes	Yes	Yes	-
TIMA1	16-bit	8-bit	Yes	2/2	4	Yes	Yes	Yes	Yes	Yes	-

Note

On TIMA instances, external PWM channels are pairs of complementary PWM output signals with deadband generation with respect to the CC block instance, such as TIMA0_C0 and TIMA0_C0N. For independent PWM output generation, separate noninverting channels must be used, such as TIMA0_C0 and TIMA0_C1.

Note

Internal CC channels 4 and 5 (CC_45) can be used for internal compare events and are available in TIMA only.

Note

Look at the device-specific data sheet to check which TIMx instances are available on the device and their respective power domains

15.2 TIMx Operation

The TIMx module is configured with user software. The setup and operation of TIMx is discussed in the following sections.

Note

There are register arrays in the timer module to group registers with same bit fields for different capture/compare ports. For example, TIMx.CC_01[0/1] is a register array that contains the capture/compare registers for both CCP0 and CCP1. Access TIMx.CC_01[0] to read the CC0 capture/compare value, and access TIMx.CC_01[1] to read the CC1 capture/compare value.

Note

TIMx supports event subscribers and event publishers through event register arrays. For more information, please see the Events chapter.

15.2.1 Timer Counter

All TIMx instances have 16-bit counter blocks except for TIMG12 and TIMG13, which have 32-bit counter blocks. The timer counter register (TIMx.CTR) can count down, up-and-down, or up depending on the operation mode. It can also be read or written with software. Each count occurs with each rising edge of the TIMCLK signal or with both edges of external signals.

Enabling the TIMx Counter

The counter is clocked by the prescaler output TIMCLK. The counter enable bit TIMx.CTRCTL.EN can be enabled in two ways:

- Set in software manually
- After a load condition (LCOND) or zero condition (ZCOND) is met, and the counter value after enable (CVAE) is changed to the load value or zero value, respectively.

Note

The ability to write the counter register while TIMx is running is possible but should be avoided because the user write can collide with a load, zero, or advance event. Depending on the prescaler ratio, the application cannot predict the timing of the write, which can affect the correct timer period.

15.2.1.1 Clock Source Select and Prescaler

The TIMx clock (TIMCLK) can be sourced from an internal clock or an external signal trigger to advance the clock.

15.2.1.1.1 Internal Clock and Prescaler

The TIMx clock (TIMCLK) can be sourced from BUSCLK, MFCLK and LFCLK by setting the TIMx.CLKSEL register. It can also be divided by a ratio by setting the TIMx.CLKDIV register and a prescaler (if present). The selected source clock is always available and the frequency depends on the power mode. For more information, see the Clock Module (CKM) section.

The TIMCLK can come from the following sources:

- BUSCLK: the current bus clock is selected as the source for TIMx. The current bus clock depends on power domain.
  - If the TIMx instance is in Power Domain 1 (PD1), refer to MCLK.
  - If the TIMx instance is in Power Domain 0 (PD0), refer to ULPCLK.
- MFCLK: MFCLK is selected as the source for TIMx, refer to MFCLK.
- LFCLK: LFCLK is selected as the source for TIMx, refer to LFCLK.

The selected clock source can be passed directly to TIMx or divided by the 8-bit prescaler. The prescaler setting can be configured with register TIMx.CPS.PCNT bit. The selected TIMCLK source is divided by a value of (PCNT+1). A PCNT value of 0 divides TIMCLK by 1, effectively bypassing the divider. A PCNT value of greater than 0 divides the TIMCLK source to generate a slower clock.

TIMx also has a software mechanism for disabling the timer clock. Set TIMx.CCLKCTL.CLKEN to 0 to put the timer in an IDLE state.

TIMCLK Configuration

To configure the clock source, divider, and prescaler:

1. Select the TIMCLK clock source (BUSCLK, MFCLK, or LFCLK) using the CLKSEL register.
2. Optionally divide the TIMCLK using CLKDIV.RATIO.

3. In TIMx instances with prescalers, optionally set a prescaler using CPS.PCNT.
4. Enable the TIMCLK by setting CCLKCTL.CLKEN = 1.

The frequency of TIMCLK is determined using Equation 13.

$$f_{TIMCLK} = \frac{f_{CLK\_SOURCE}}{((CLKDIV.RATIO + 1)*(CPS.PCNT + 1))} \quad (13)$$

15.2.1.1.2 External Signal Trigger

The counter can also advance (increment or decrement) by using an external signal on the timer input pin or by triggering from an event from other peripherals in the system. This can be configured by using the advance condition setting (TIMx.CCCTL_xy[0/1].ACOND) to specify what creates the advance event. To specify what event advances the counter, use the TIMx.CTRCTL.CAC setting.

The counter can advance using the internal clock TIMCLK, a different edge of an timer external input, or an internal trigger event from other peripherals.

15.2.1.2 Repeat Counter (TIMA only)

In TIMA only, the repeat counter (RC) is an 8-bit counter that provides the mechanism to suppress unnecessary events and generate real events for interrupt generation. Specifically, the repeat counter can suppress Load, Compare, and Zero events in the case where the timer is generating events that repeat for a known number of cycles, such a periodic PWM output waveform. This prevents generating excessive and unnecessary interrupts every timer period.

When the timer counter (TIMA.CTR) is advancing, the repeat counter (TIMA.RC) advances when the counter reloads (TIMA.CTR = 0). Software can set the how many timer counter reloads occur until generating the interrupts and events by setting the TIMA.RCLD register. When TIMA.RC = TIMA.RCLD, the repeat counter is reset back to zero and a Repeat Counter Zero event occurs (REPC) in the Interrupt and Event Status registers.

Note

If software configures the counter to stop counting in a debug or fault condition, also stop the repeat counter. See Section 15.2.6 and Section 15.2.10 for more details.

Additionally, the repeat counter provides the ability to suppress generation of Zero, Load, and Compare events when TIMA.RC does not equal zero.

- Zero and Load events are suppressed by setting TIMA.CTRCTL.SLZERCNEZ register bit
- Compare events (see Table 15-15) are suppressed by setting the TIMA.CCCTL_xy[0/1].SCERCNEZ bit

Table 15-2 shows the repeat counter behavior with respect to the timer counter and repeat counter load value.

Table 15-2. Repeat Counter Behavior

TIMA.CTR is Advancing (+1)	Counter value	TIMA.RC = TIMA.RCLD	Repeat Counter Behavior	Suppress Load and Zero Events (SLZERCNEZ = 1)	Suppress Compare Events (SCERCNEZ = 1)
No	-	-	Does not advance	Yes	Yes
Yes	TIMA.CTR ≠ 0	-	Does not advance	Yes	Yes
Yes	TIMA.CTR = 0	No	Advance (+1)	Yes	Yes
Yes	TIMA.CTR = 0	Yes	TIMA.RC = 0	No	No

Repeat counter example

As shown in Figure 15-3, the TIMA.CTR is configured for down-counting mode and zero events are generated once TIMA.CTR = 0. To suppress interrupt generation until 4 timer reloads occur, set TIMA.RCLD = 4 and TIMA.CTRCTL.SLZERCNEZ = 1 to suppress zero and load events until RC = 0 (which occurs once TIMA.RC = TIMA.RCLD).

Note

The use of the repeat counter does not affect the output signal generation. All events are generated regardless of the repeat counter value sent to the signal generator unit.

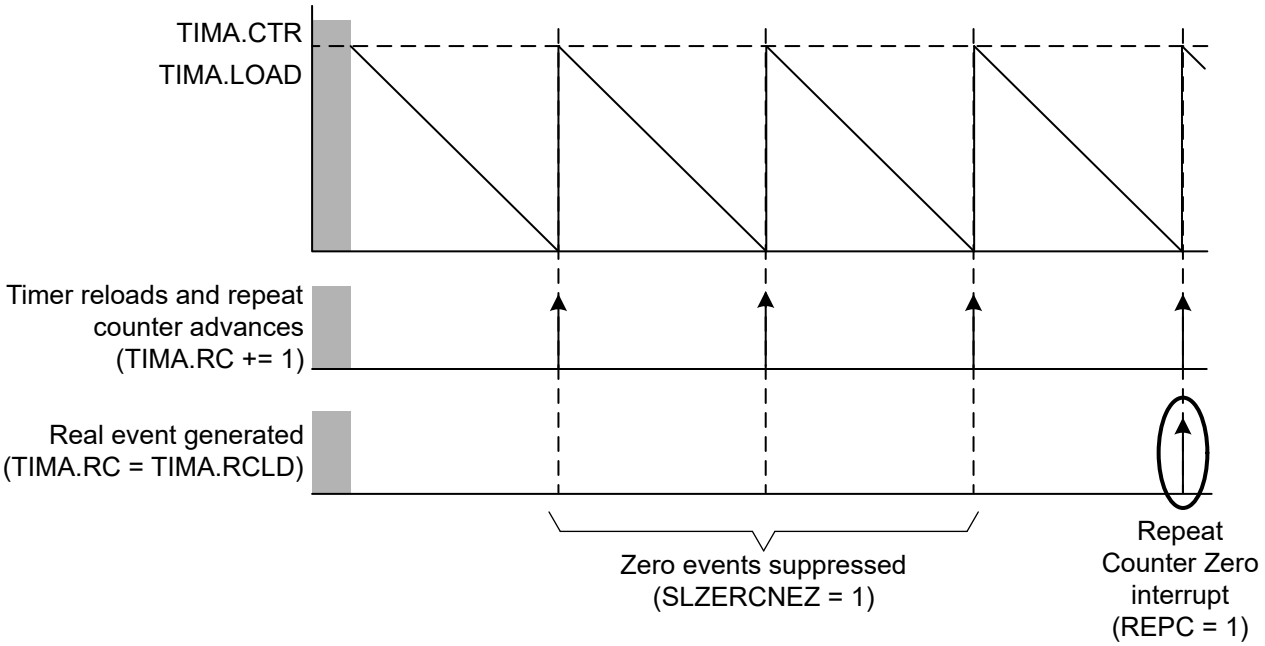


Figure 15-3. Event Suppressed by Repeat Counter with TIMx.RCLD = 4 in Down Counting Mode

15.2.2 Counting Mode Control

When the device is out of reset, TIMx is disabled. Writing 1 to the TIMx.CTRCTL.EN bit enables the counter. This bit is automatically cleared if TIMx.CTRCTL.REPEAT=0 (do not automatically reload), and the counter value equals zero.

TIMx has three counting modes when enabled: down, up/down, and up. The operating mode is selected by TIMx.CTRCTL.CM bit (shown in Table 15-3). After the counter is enabled, the timer will start counting from the TIMx.CTRCTL.CVAE setting.

Table 15-3. TIMx Counting Modes (CM)

TIMx.CTRCTL.CM	Counting Mode
0	Down
1	Up/Down
2	Up

Table 15-4. TIMx Counter Value after Enable (CVAE)

Count Value After Enable (CVAE)	Description	Supported Counting Modes
0	LOAD value	Up, up/down, down
1	Unchanged from current value	Up/down
2	Zero value	Up, up/down, down

15.2.2.1 One-shot and Periodic Modes

Figure 15-4 shows TIMx working in both one-shot mode and periodic mode.

One-shot Mode

When TIMx.CTRCTL.REPEAT bit is set to 0, TIMx does not advance when:

- TIMx.CTR value reaches 0 in either down- or up/down-counting mode
- TIMx.CTR value reaches TIMx.LOAD in up-counting mode

Periodic Mode (Counter Reload)

When TIMx.CTRCTL.REPEAT is set to 1h, TIMx automatically repeats once a zero event occurs. This happens when:

- TIMx.CTR reaches 0 in either down- or up/down-counting mode
  - In down-counting mode, a zero event is followed by a load event at the next advance condition
  - In up/down-counting mode, a zero event is followed by an advance event (+1)
- TIMx.CTR reaches TIMx.LOAD in up-counting mode
  - A load event is followed by a zero at the next advance condition

TIMx.CTRCTL.REPEAT can also be set to 3h for TIMx to repeat only when not in a debug condition. If there is a debug condition, TIMx will count to the zero event and repeat only once the debug condition is removed.

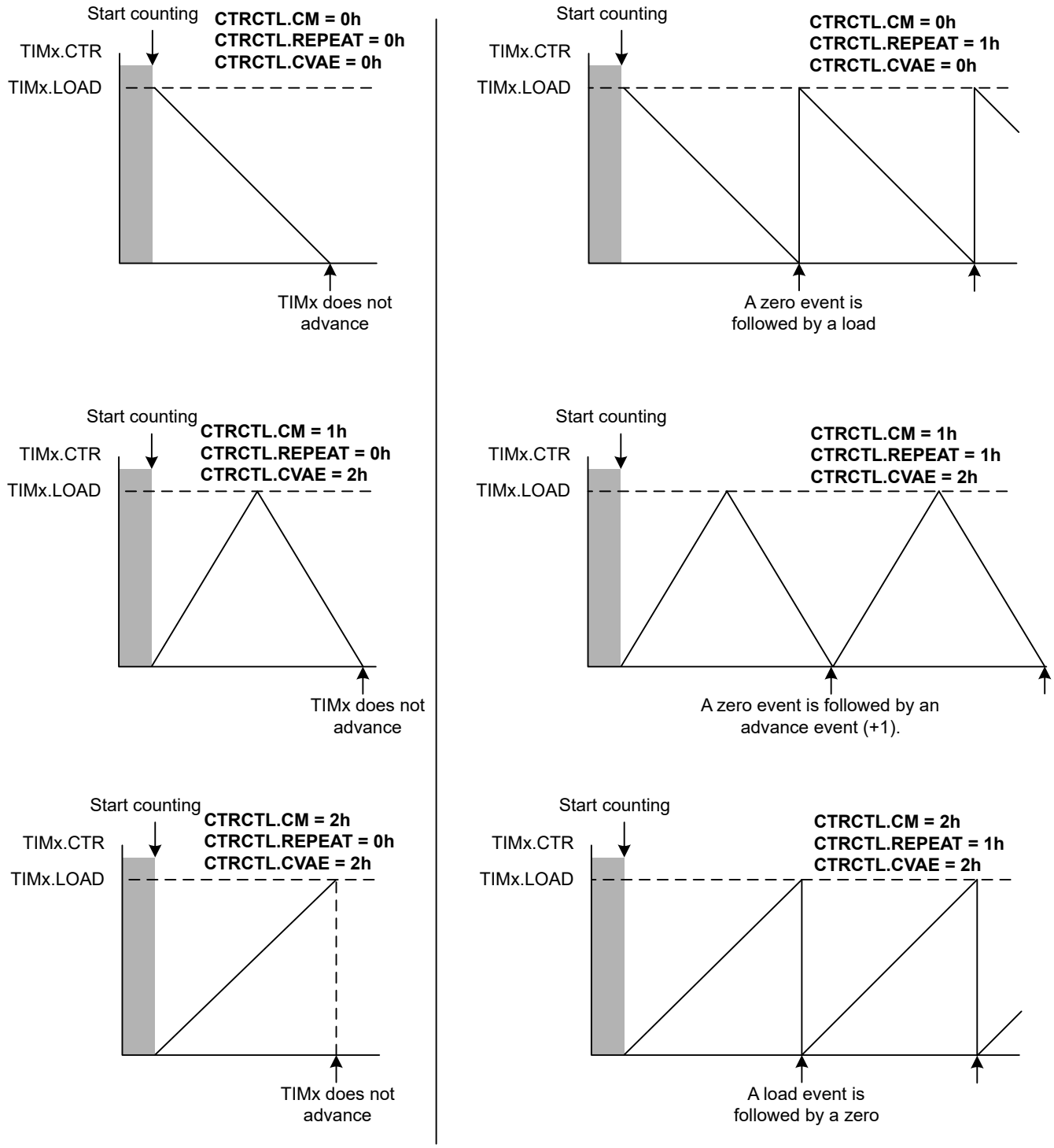


Figure 15-4. One-shot and Periodic Mode Behavior

15.2.2.2 Down Counting Mode

In down counting mode (CM = 0), TIMx counts from the value defined in TIMx.LOAD (CVAE = 0) down to zero. When the TIMx.CTR value equals zero and TIMx.CTRCTL.REPEAT is set to 1, the TIMx.LOAD value is loaded into TIMx.CTR and the timer repeats the down counting pattern as shown in Figure 15-5.

A Zero event is generated when TIMx counts to zero. A Load event is generated when TIMx counts from zero to the TIMx.LOAD value. Figure 15-6 shows the event generating cycle.

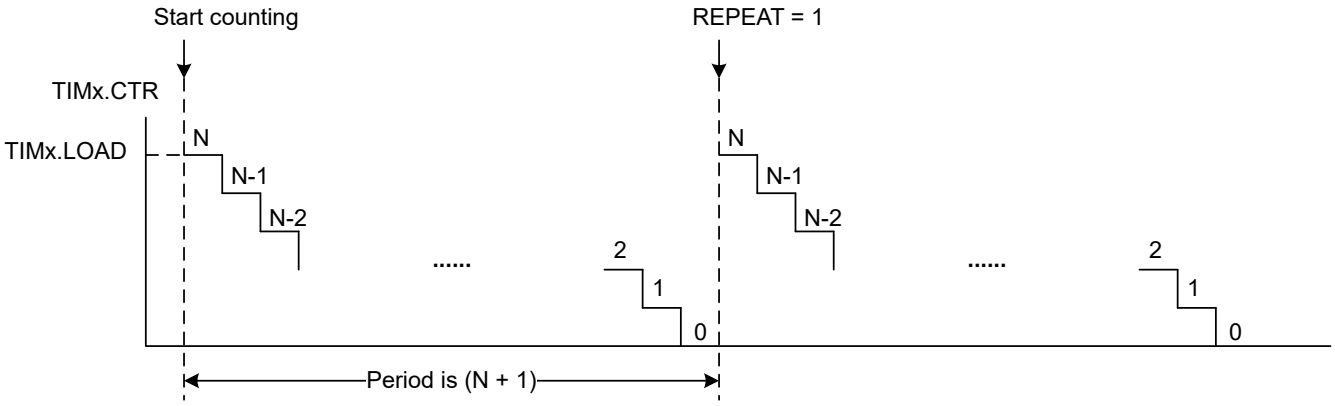


Figure 15-5. Down Counting Mode, CVAE = 0

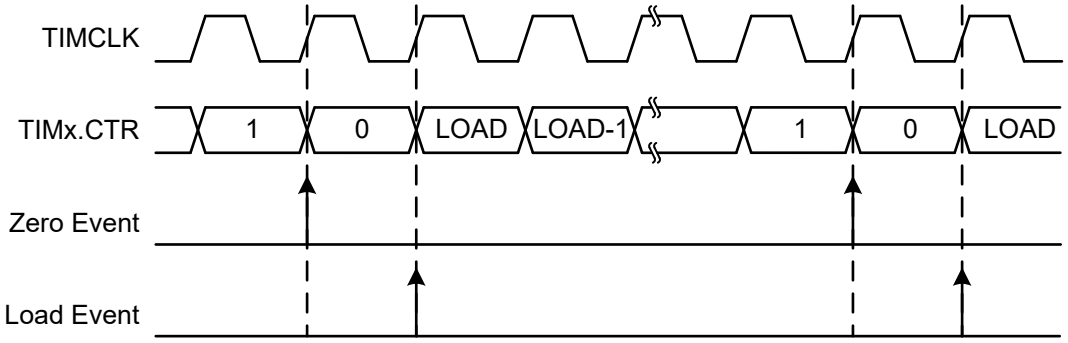


Figure 15-6. Down Counting Mode Event Generation

15.2.2.3 Up/Down Counting Mode

The Up/Down counting mode can count in an down-up direction or an up-down direction depending on TIMx.CTRCTL.CVAE value. The TIMx.CTRCTL.CVAE bits specify the initialization condition of the counter.

Table 15-5. Counter Value after Enable Initialization Conditions

TIMx.CTRCTL.CVAE Value	Counter Value After Enable
0x0	Load Value
0x1	No Change
0x2	Zero

Counting in down-up direction

When TIMx.CTRCTL.CVAE = 0, TIMx.CTR is set to TIMx.LOAD register value and TIMx counts in the down direction. When it reaches zero, a Zero event is generated and TIMx counts back up to TIMx.LOAD value. A Load event is generated when it reaches TIMx.LOAD value.

Figure 15-7 shows TIMx counting in the down-up direction when TIMx.CTRCTL.CVAE = 0.

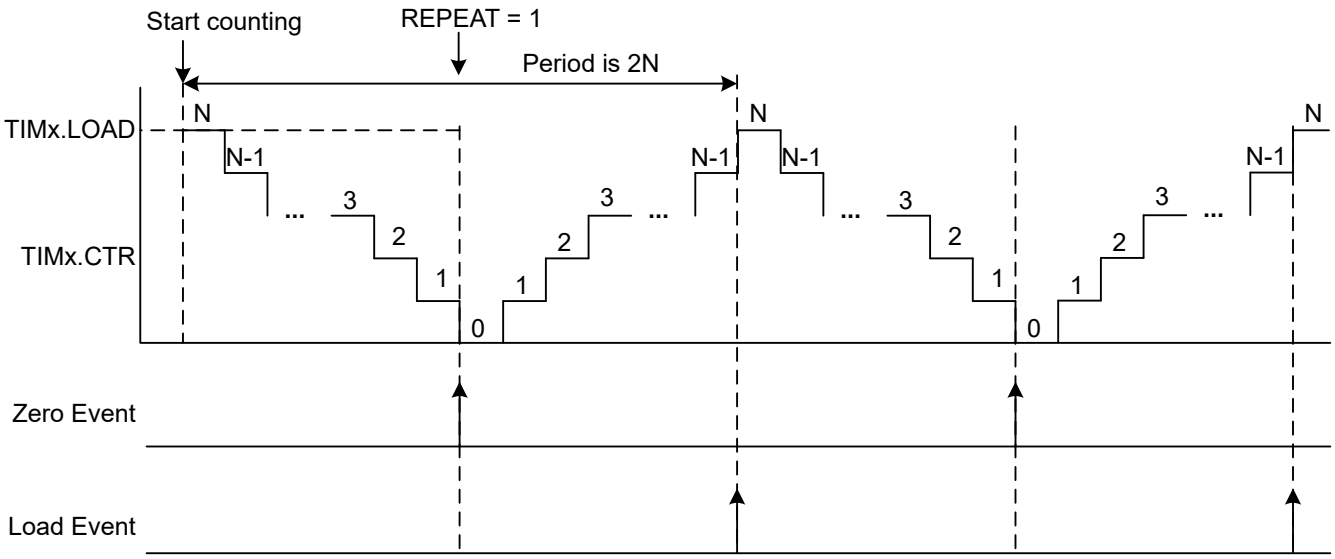


Figure 15-7. Down-up Counting Mode and Event Generation, CVAE = 0

Counting in up-down direction

When TIMx.CTRCTL.CVAE = 2, TIMx.CTR is set to zero and TIMx counts in the up direction. When it reaches TIMx.LOAD, a Load event is generated and TIMx counts back down to zero. A Zero event is generated when it reaches zero.

Figure 15-8 shows TIMx counting in up-down direction when TIMxCTRCTL.CVAE = 2.

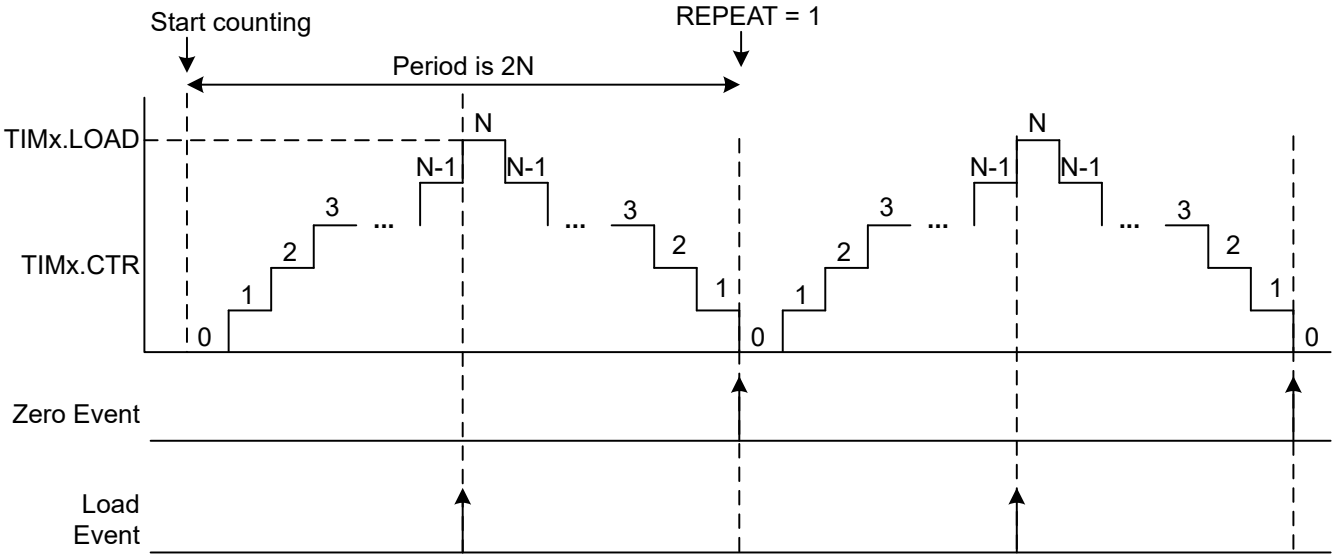


Figure 15-8. Up-down Counting Mode and Event Generation, CVAE = 2

15.2.2.4 Up Counting Mode

In up counting mode, TIMx counts from zero up to the value defined in TIMx.LOAD. When the TIMx.CTR value equals TIMx.LOAD and TIMx.CTRCTL.REPEAT is not equal to 0, the zero is loaded into TIMx.CTR and the timer repeats the up counting pattern as shown in Figure 15-5.

A Load event is generated when TIMx counts to TIMx.LOAD. A Zero event is generated when TIMx counts from TIMx.LOAD to the zero value. Figure 15-6 shows the event generating cycle.

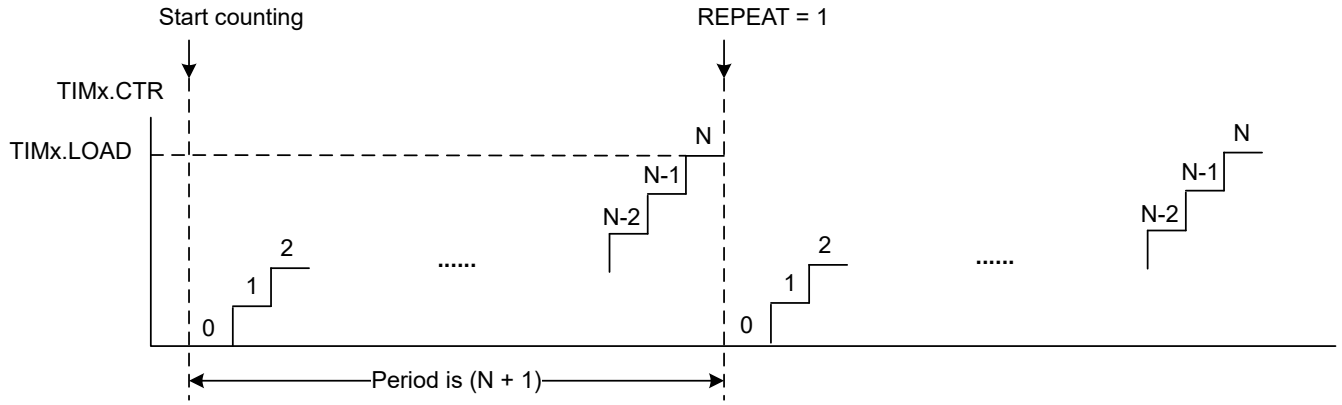


Figure 15-9. Up Counting Mode, CVAE = 2

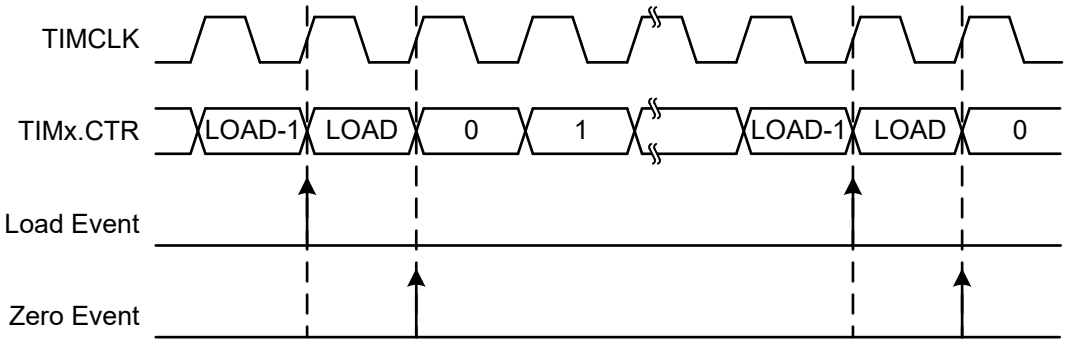


Figure 15-10. Up Counting Mode Event Generation

15.2.2.5 Phase Load (TIMA only)

In TIMA only, the phase load register TIMA.PL provides the capability for TIMA.CTR to count from a value other than zero or TIMA.LOAD in Up/Down counting mode. Phase load is used to generate asymmetric center-aligned PWM output signals with a controlled phase shift between different timer instances.

When TIMA.PL is nonzero, phase load is enabled by setting TIMA.CTRCTL.PLEN = 1 and triggered when TIMA.CTTRIG.TRIG = 1. When phase load is triggered while TIMA.CTRCTL.CVAE = 0, the timer counts from the TIMA.PL value in the down direction. When phase load is triggered while TIMA.CTRCTL.CVAE = 0, the TIMx counts from the TIMA.PL value in the up direction.

TIMA.PL is latched when the timer starts, and TIMA.PL is synchronized every time when the counter reaches the previously latched TIMA.PL value. Figure 15-11 shows how the phase load register works when the timer is counting in the up-down direction and the phase load value changes to a new value.

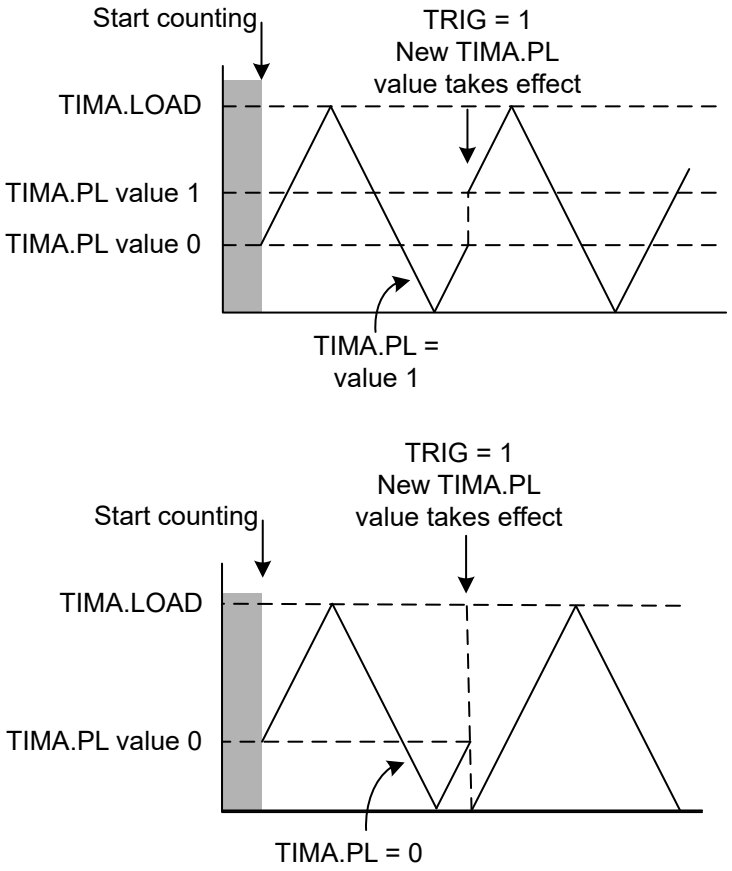


Figure 15-11. Phase Load Register Synchronization in Up-Down Mode

15.2.3 Capture/Compare Module

The capture/compare (CC) block is used for capture events or compare events. TIMG has up to 2 identical capture/compare blocks and TIMA has up to 4 identical capture/compare blocks present to support external or internal signals. Additionally, TIMA provides 2 additional CC blocks (CC4 and CC5) for compare events only from internal signals. Any of the TIMx capture/compare blocks may be used to capture timings of an input signal or to generate time intervals.

15.2.3.1 Capture Mode

Capture mode is selected when the TIMx.CCCTL_xy[0/1].COC bit is set to 1. Capture mode is used to generate capture events and record time intervals, which is useful for speed computation or time measurements.

Key registers for configuring capture mode:

- **TIMx.LOAD**: the contents of this register are copied to counter (TIMx.CTR) on any operation designated to do a "load". This value is also used to compare with the counter value for generating a "Load Event" which can be used for interrupt, trigger, or signal generation actions.
- **TIMx.CC_xy[0/1]**: this is a register that can be used as either a capture register to acquire or record the next counter value on an event, or a compare register to the current counter to create an event.
- **TIMx.CCCTL_xy[0/1]**: this register controls the operations of the respective CC (capture/compare) blocks. In capture mode, it can configure whether a rising edge or falling edge generates a load, zero, advance, or capture condition. In compare mode, it controls which sources generate different types of compare events.
- **TIMx.CTRCTL**: this register provides control over the counter operation in different conditions.
- **TIMx.IFCTL_xy[0/1]**: this register controls the input filtering (FE, FP, CPV), selection (ISEL), and inversion (INV) for the associated CC block.

15.2.3.1.1 Input Selection, Counter Conditions, and Inversion

The TIMx.IFCTL register is used for selecting input source, filtering, and final inversion options for the capture/compare block.

Figure 15-12 shows the block diagram for the TIMx capture block with two CC channels.

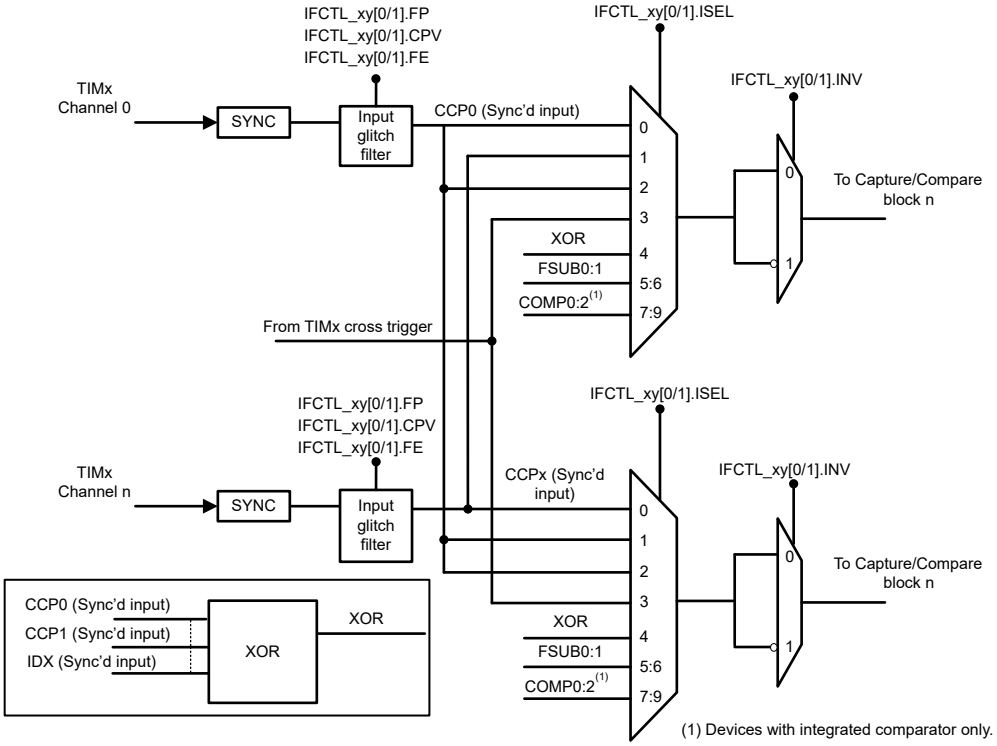


Figure 15-12. TIMx Capture Block Diagram

15.2.3.1.1.1 CCP Input Edge Synchronization

When using a capture/compare pin (CCP) as an input, configure the pin control management register (PINCMx) for the TIMx CCP input. Refer to the device data sheet for TIMx CCP pinmux input options, such as TIMG0_C0.

The CCP input signal is always passed through a synchronizer, and the input state (high or low) must be greater than one TIMCLK clock period for the synchronizer to detect the edge. CCP input edge detection requires at least one TIMCLK cycle to synchronize the edge input. Timing in the first TIMCLK cycle is uncertain because the edge detection cannot be predicted in the first TIMCLK period.

When the capture condition occurs, an additional TIMCLK cycle is required to generate the capture event.

15.2.3.1.1.2 CCP Input Pulse Conditions

The TIMx.CCCTL_xy[0/1] register can control whether each timer instance generates a zero, load, capture, or advance pulse based on the edge or polarity of the CCP input signal or trigger edge. The conditions that can be generated are:

- Advance condition (ACOND)
- Load condition (LCOND)
- Zero condition (ZCOND)
- Capture condition (CCOND)

Advance conditions

By default, the timer advances based on each TIMCLK. (ACOND = 0h). However, the timer can also advance based off the specified TIMx.CCCTL_xy[0/1].ACOND settings below.

Table 15-6. Advance pulse condition settings (ACOND)

ACOND	Condition
0h	Each TIMCLK
1h	Rising edge of CCP or trigger assertion edge
2h	Falling edge of CCP or trigger de-assertion edge
3h	Either edge of CCP or trigger
5h	CCP high or trigger assertion

Load, zero, and capture conditions

Load, zero, and capture pulses can be generated the LCOND, ZCOND, and CCOND condition settings below in the TIMx.CCCTL_xy[0/1] register.

Table 15-7. Load, zero, and capture condition settings (LCOND, ZCOND, CCOND)

LCOND	ZCOND	CCOND	Condition
N/A	N/A	0h	None
1h	1h	1h	Rising edge of CCP or trigger assertion edge
2h	2h	2h	Falling edge of CCP or trigger de-assertion edge
3h	3h	3h	Either edge of CCP or trigger

15.2.3.1.1.3 Counter Control Operation

To specify what CC instance controls the load, zero, or advance event, the CZC, CAC, and CLC fields are used for configuration in the TIMx.CTRCTL register.

See Table 15-8 for counter zero control settings. For example, if a timer triggers a ZCOND event in Channel 1, then CZC should be set to 1h to register that a ZCOND event in channel 1 triggers the zero event.

Table 15-8. Counter Zero Control (CZC) settings

TIMx.CTRCTL.CZC	Setting
0h	Channel 0 ZCOND event zeroes the TIMx instance
1h	Channel 1 ZCOND event zeroes the TIMx instance
2h	Channel 2 ZCOND event zeroes the TIMx instance (4 CC timer only)
3h	Channel 3 ZCOND event zeroes the TIMx instance (4 CC timer only)
4h	2-input QEI mode. See Section 15.2.3.1.3
5h	3-input QEI mode. SeeSection 15.2.3.1.3

See Table 15-9 for counter load control settings. For example, if a timer triggers a LCOND event in Channel 2, then CLC should be set to 2h to register that a LCOND event in channel 2 triggers the load event.

Table 15-9. Counter Load Control (CLC) settings

TIMx.CTRCTL.CLC	Setting
0h	Channel 0 LCOND event loads the TIMx instance
1h	Channel 1 LCOND event loads the TIMx instance
2h	Channel 2 LCOND event loads the TIMx instance (4 CC timer only)
3h	Channel 3 LCOND event loads the TIMx instance (4 CC timer only)
4h	2-input QEI mode. See Section 15.2.3.1.3
5h	3-input QEI mode. SeeSection 15.2.3.1.3

See Table 15-10 for counter advance control settings. For example, if a timer triggers a ACOND event in Channel 3, then CAC should be set to 3h to register that a ACOND event in channel 3 triggers the advance event.

Table 15-10. Counter Advance Control (CAC) settings

TIMx.CTRCTL.CAC	Setting
0h	Channel 0 ACOND event advances the TIMx instance
1h	Channel 1 ACOND event advances the TIMx instance
2h	Channel 2 ACOND event advances the TIMx instance (4 CC timer only)
3h	Channel 3 ACOND event advances the TIMx instance (4 CC timer only)
4h	2-input QEI mode. See Section 15.2.3.1.3
5h	3-input QEI mode. SeeSection 15.2.3.1.3

15.2.3.1.1.4 CCP Input Filtering

The input glitch filter can be enabled by setting the TIMx. IFCTL_01[0/1].FE bit. The filter period is configured by setting the TIMx. IFCTL_01[0/1].FP bit.

A consecutive period or majority voting format selected by the TIMx.IFCTL_xy[0/1].CPV bit is used to select the criteria for a CCP input signal.

- **Consecutive period** - The CCP input signal must be at the specified level for the defined number of FP timer clocks for the CCP input to be processed.
- **Majority voting** - The filter ignores one clock of opposite logic over the filter period. For example, over the number of FP samples of the input, up to 1 sample can be of an opposite logic value (glitch) without affecting the output.

The example shown in Figure 15-13 shows the difference between consecutive period and majority voting formats with a digital filter implemented to capture a CCP input signal of 3 TIMCLK periods.

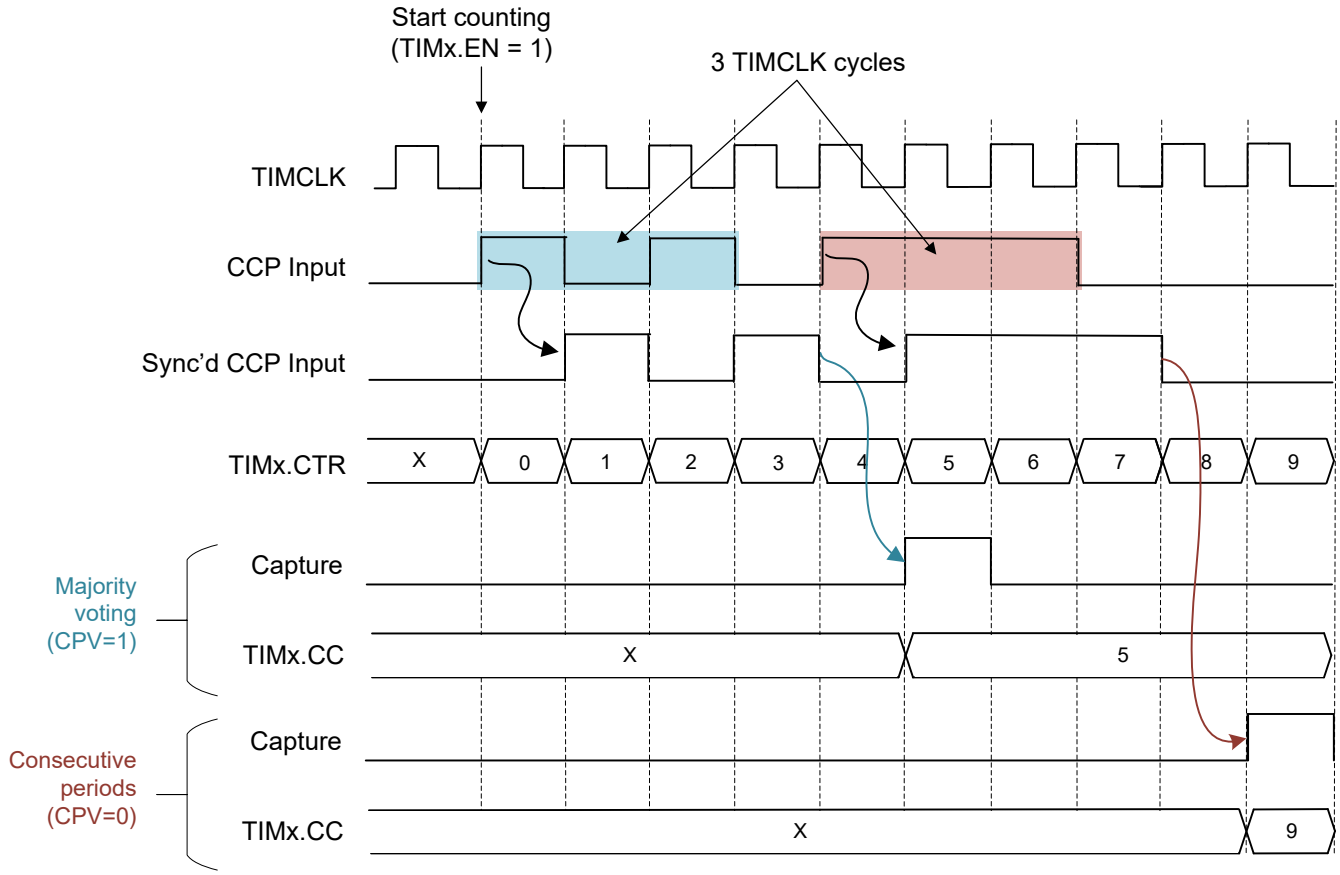


Figure 15-13. Consecutive Period and Majority Voting for Input Glitch Filtering using FP = 0 (3 TIMCLK cycles)

15.2.3.1.1.5 Input Selection

The input select bits TIMx.IFCTL_xy[0/1].ISEL select the input source to the filter input as the corresponding CCP input, CCP0 for cross-triggering across CC blocks, an external trigger, XOR (used in Hall input mode), event subscribers, or comparator inputs.

Table 15-11. Input Selection Options for TIMx CC Instances

Input selection (TIMx.IFCTL_xy[0/1].ISEL)	Source
0h	TIMx CCP of the corresponding capture compare unit
1h	Input pair CCPx of the capture compare unit. [CCP0/CCP1, CCP2/CCP3]
2h	TIMx CCP0
3h	Cross trigger signal
4h	XOR of CCP inputs. Used in Hall input mode. See Section 15.2.3.1.4
5h	Subscriber event 0 (FSUB0). See Section 15.2.9.2
6h	Subscriber event 1 (FSUB1). See Section 15.2.9.2
7h	COMP0_OUT (devices with comparator only)
8h	COMP1_OUT (devices with comparator only)
9h	COMP2_OUT (devices with comparator only)

15.2.3.1.2 Use Cases

Several different use cases can be achieved in capture mode and are discussed in the following sections.

15.2.3.1.2.1 Edge Time Capture

Edge time capture measures the time (in TIMCLK cycles) from the start of the capture operation to the signal edge. The counter is loaded when TIMx is enabled and counts with each TIMCLK period until the CCP edge is detected, which triggers the capture of the timer value and generates a capture event. The capture edge time is equivalent to the difference between the starting value of the counter and the capture value in TIMx.CC_xy[0/1] register.

Edge Time Capture Configuration

1. Set the TIMx.LOAD value.
2. In the CTRCTL register, set the desired counter control settings for:
   a. Counting mode (CM) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Zero (CZC), advance (CAC), and load control (CLC) to specify what condition controls zeroing, advancing, or loading the counter
   c. Repeat or one-shot mode (REPEAT)
3. Set TIMx.CCCTL_xy[0/1].COC = 1 for capture mode.
4. Configure CCP as an input for the CC block by setting respective bit in the CCPD registers. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
5. For the corresponding CC block control register (CCCTL_01[0/1]), set CCOND to the corresponding setting to capture events based off the input signal condition (rising and/or falling edge). Additionally, set ZCOND or LCOND depending on the counting mode used.
6. Configure input capture settings in the TIMx.IFCTL_xy[0/1] register as described in Section 15.2.3.1.1.
7. Enable the counter by setting EN = 1 or waiting for a capture event to occur from the input edge.

Example using up-counting mode for rising edge capture

In up-counting mode starting from zero (CM = 2, CVAE = 2), TIMx can be configured to generate a zero pulse and start the counter from the configured capture event (CCOND) by setting ZCOND to 1.

The expected internal timing for a rising or positive edge time capture in up-counting mode is shown in Figure 15-14.

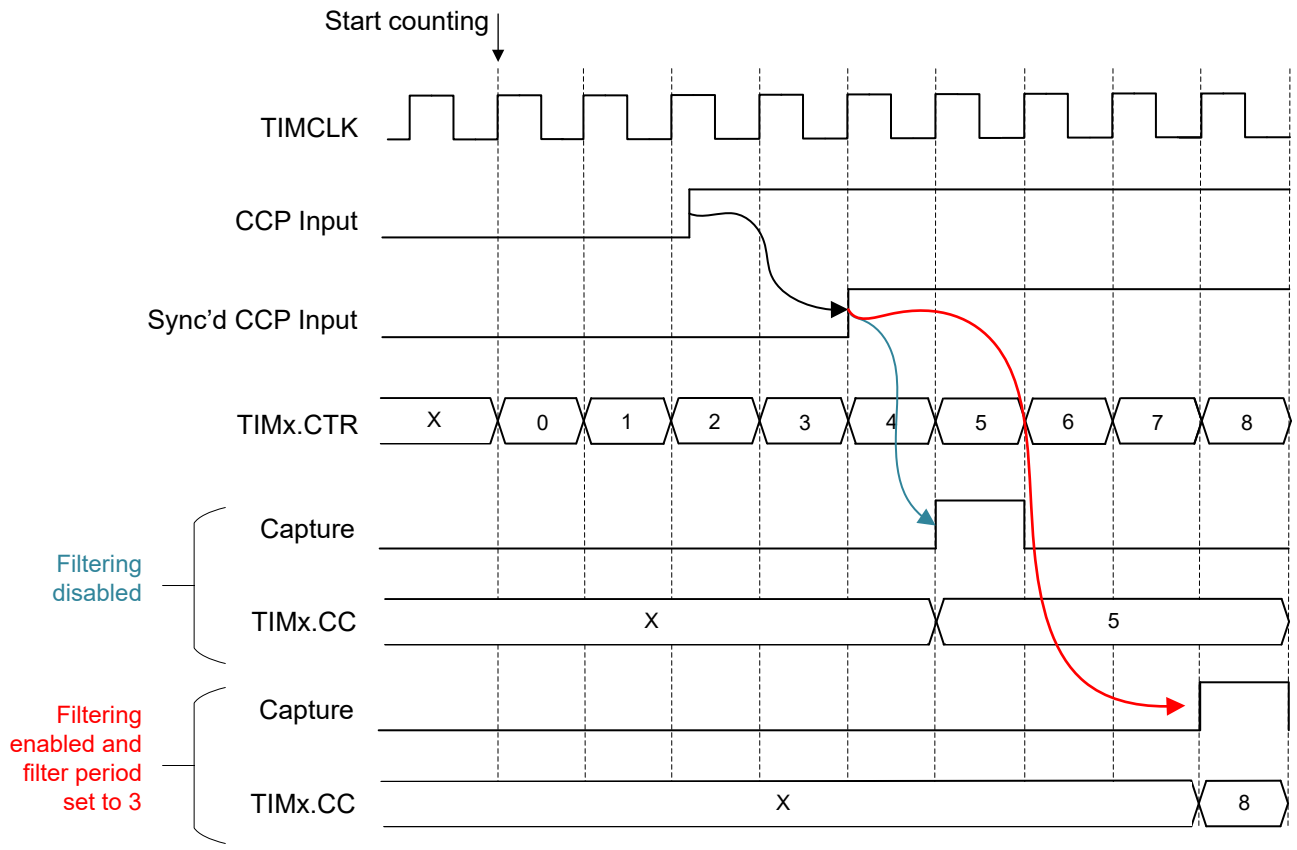


Figure 15-14. Edge Time Capture Mode in Up-Counting Mode, CVAE = 2

15.2.3.1.2.2 Period Capture

Period capture measures the period of a signal on an input CCP in TIMCLK cycles. On each positive (or negative) edge of the CCP input, the TIMx.CTR value is both captured into the TIMx.CC register to generate a capture event. The period capture time is equivalent to the difference between the starting value of the counter generated to the capture event, or time between reoccurring capture events for a periodic input signal.

Period Capture Configuration

1. Set the TIMx.LOAD value.
2. In the CTRCTL register, set the desired counter control settings for:
   a. Counting mode (CM) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Advance (CAC) to specify what condition controls advancing the counter
   c. Repeat or one-shot mode (REPEAT)
3. Set TIMx.CCCTL_xy[0/1].COC = 1 for capture mode.
4. Configure CCP as an input for the CC block by setting respective bit in the CCPD registers. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
5. For the corresponding CC block control register (CCCTL_01[0/1]),
   a. Set CCOND to the corresponding setting to capture events based off the input signal condition (rising and/or falling edge)
   b. Set ZCOND or LCOND depending on the counting mode used
6. Configure input capture settings in the TIMx.IFCTL_xy[0/1] register as described in Section 15.2.3.1.1.
7. Enable the counter by setting EN = 1 or waiting for a capture event to occur from the input edge.

Example using up-counting mode for rising-edge period capture

In up-counting mode starting from zero (CM = 2, CVAE = 2), TIMx channel 0 can be configured to generate a capture event from a rising edge input by setting CCOND = 1h. After enabling the counter, when a rising

edge input is detected, the counter will capture the counter value in TIMx.CC. After the capture event, set the TIMx.LOAD value back to 0 to reset the timer counter for the periodic CCP input signal.

The expected internal timing for a period capture in up-counting mode using two rising edges is shown in Figure 15-14.

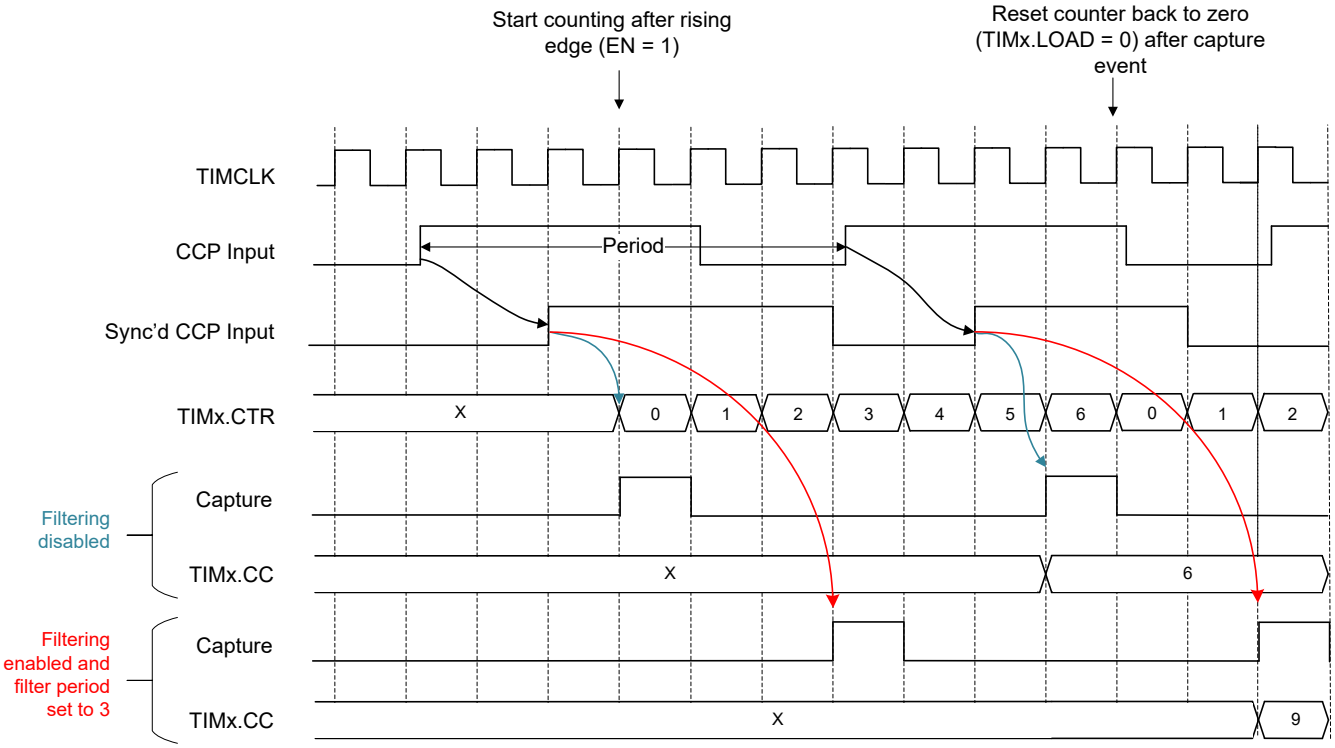


Figure 15-15. Period Capture Mode in Up-Counting Mode, CVAE = 2

15.2.3.1.2.3 Pulse Width Capture

Pulse width capture measures the high-time of a signal on CCP. The high time is the number of TIMCLK periods from rising edge to falling edge of the CCP input, and is useful for applications such as measuring the duty cycle of an PWM input signal. The counter is loaded at the positive edge and captured at the negative edge (capture event is generated).

Pulse-Width Capture Configuration

1. Set the TIMx.LOAD value.
2. In the CTRCTL register, set the desired counter control settings for:
   a. Counting mode (CM) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Zero (CZC), advance (CAC), and load control (CLC) to specify what condition controls zeroing, advancing, or loading the counter
   c. Repeat or one-shot mode (REPEAT)
3. Set TIMx.CCCTL_xy[0/1].COC = 1 for capture mode.
4. Configure CCP as an input for the CC block by setting respective bit in the CCPD registers. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
5. For the corresponding CC block control register (CCCTL_01[0/1]), set CCOND to the corresponding setting to capture events based off the input signal condition (rising and/or falling edge). Additionally, set ZCOND or LCOND depending on the counting mode used.
6. Configure input capture settings in the TIMx.IFCTL_xy[0/1] register as described in Section 15.2.3.1.1.
7. Enable the counter by setting EN = 1 or waiting for a capture event to occur from the input edge.

Example using up-counting mode for pulse width capture

In up-counting mode starting from zero (CM = 2, CVAE = 2), TIMx channel 0 can be configured to generate a zero pulse and start the counter from the configured capture event (CCOND) by setting ZCOND to 1. To start the counter, a load condition can be triggered from the CCP rising edge input by setting LCOND = 1.

The expected internal timing for a pulse width capture in up-counting mode using a rising and falling edge is shown in Figure 15-14.

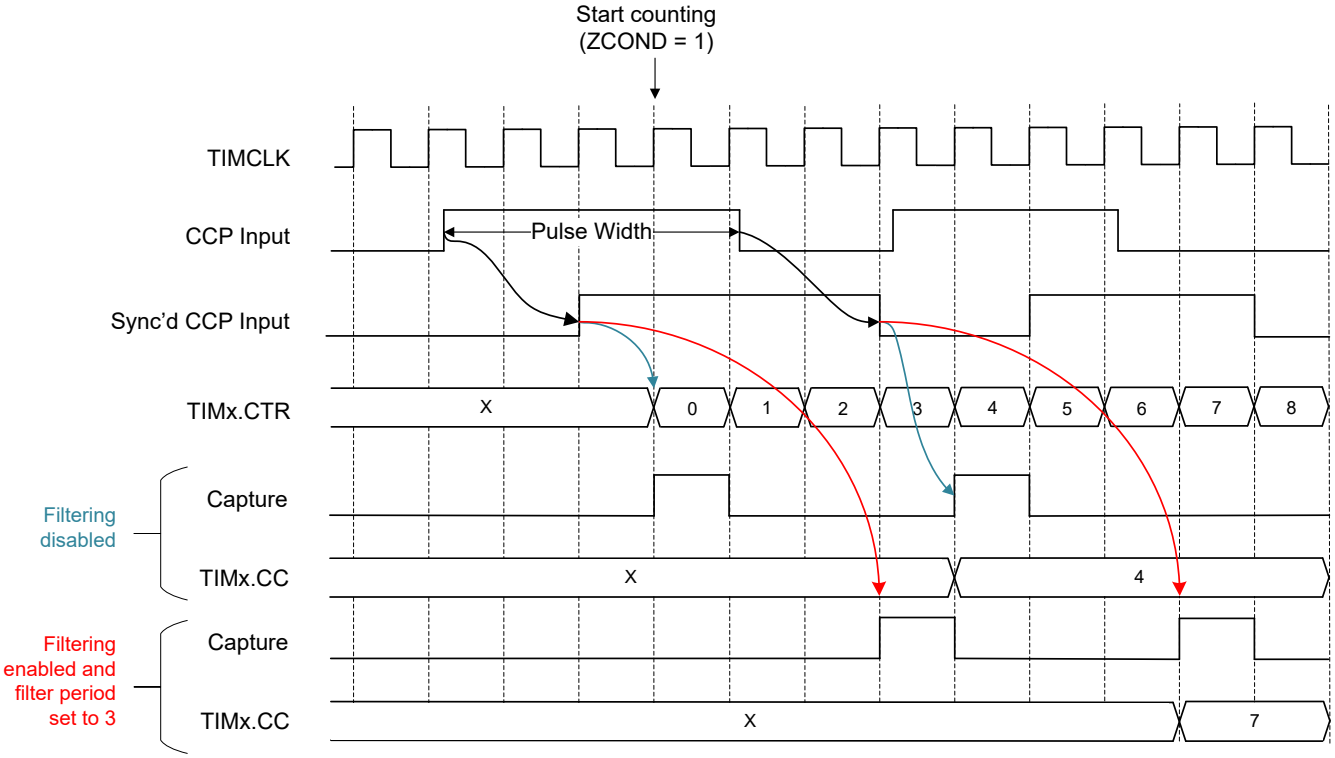


Figure 15-16. Pulse-Width Capture Mode

15.2.3.1.2.4 Combined Pulse Width and Period Time

Using two capture registers can combine pulse-width and period capture of a single input waveform. The input signal can be externally connected to CCP channel 0, and the IFCTL_01[1] register can be configured to have the input connected to CCP channel 1 internally so capture register 0 (TIMx.CC0) captures pulse width and capture register 1 (TIMx.CC1) captures period. The expected internal timing for combined pulse-width and period capture is shown in Figure 15-17.

Pulse-Width and Period Capture Configuration

1. Set the TIMx.LOAD value.
2. In the CTRCTL register, set the desired counter control settings for:
   a. Counting mode (CM) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Zero (CZC), advance (CAC), and load control (CLC) to specify what condition controls zeroing, advancing, or loading the counter
   c. Repeat or one-shot mode (REPEAT)
3. Set TIMx.CCCTL_xy[0/1].COC = 1 for capture mode for each CC channel.
4. Configure CCP as an input for each CC block by setting respective bits in the CCPD register. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
5. For the corresponding CC block control register (CCCTL_01[0/1]),
   a. Set CCOND to the corresponding setting to capture events based off the input signal condition (rising and/or falling edge)
   b. Set ZCOND or LCOND depending on the counting mode used.

6. Configure input capture settings in the TIMx.IFCTL_xy[0/1] register as described in Section 15.2.3.1.1.
7. Enable the counter by setting EN = 1 or waiting for a capture event to occur from the input edge.

Example using pulse-width and period time capture

In up counting mode, TIMx can be configured to generate a zero pulse and start the counter from the configured capture event (CCOND) by setting ZCOND to 1.

The expected internal timing for a pulse-width and period capture in up-counting mode using two CC blocks is shown in Figure 15-14.

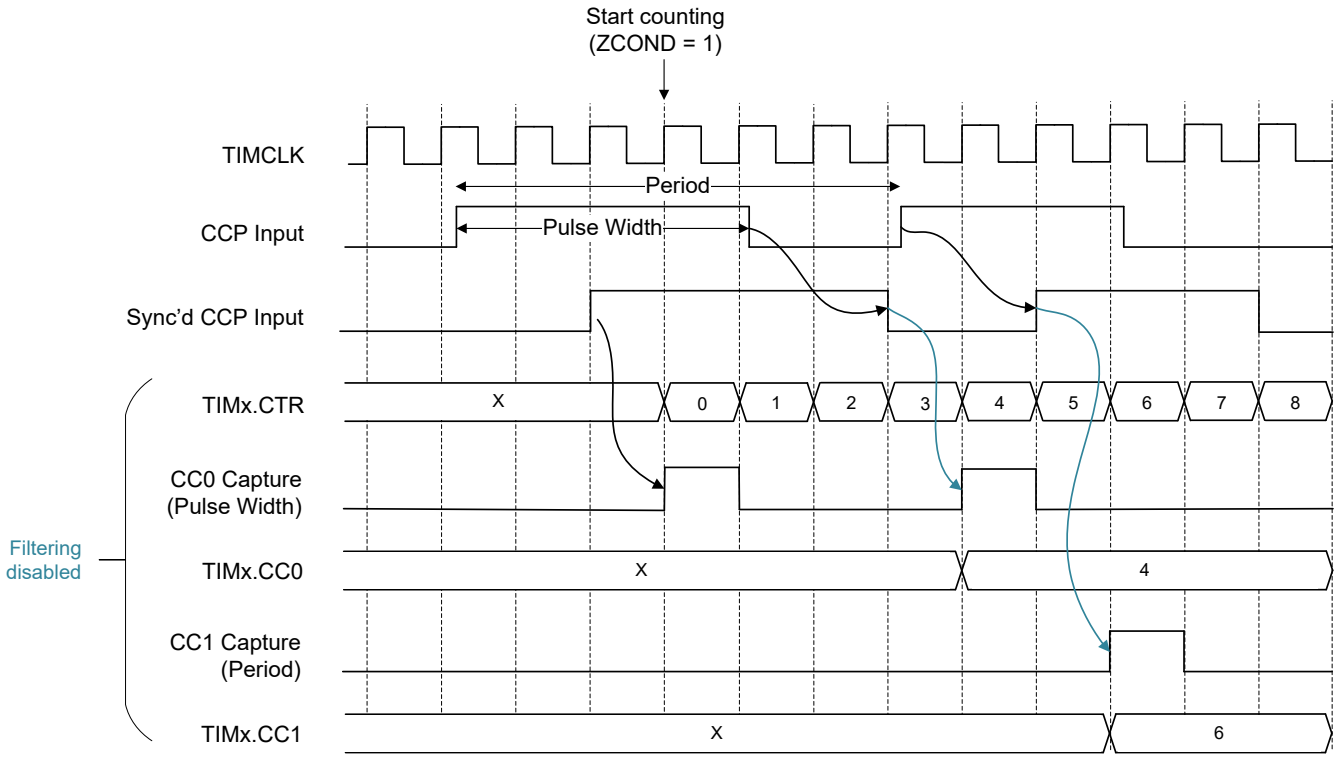


Figure 15-17. Combined Pulse-Width and Period Capture

15.2.3.1.3 QEI Mode (TIMG with QEI support only)

In TIMGx instances with QEI support, Quadrature Encoder Interface (QEI) mode provides an interface to the output of a quadrature encoder. It decodes the quadrature encoded data to provide the information on the relative positioning and movement of a linear or rotary motion.

The QEI consists of two Gray coded quadrature input signals PHA and PHB, and an index input signal IDX. All input signals go to CCP inputs of a single counter, such that PHA and PHB are mapped to CCP0 and CCP1, and IDX is brought in as a separate input. An error detection mechanism can report erroneous transitions to avoid improper signal decodings.

Note

See Section 15.1.3 and the device-specific data sheet for TIMG instances that support QEI / Hall Input Mode.

15.2.3.1.3.1 QEI With 2-Signal

QEI is used to decode signals from optical position encoders. Optical position encoders typically output 2 signals (PHA/PHB) which are often used to measure rotary or linear movement of physical items with rotating shafts,

such as 3-phase motors. The measurement provided by the interface is incremental, which means as movement occurs, the interface provides the ability to capture the relative change from the previous position.

When operating in QEI mode, a counter accumulates the incremental updates, deriving current position from initial position and the accumulated change. The initial position can be determined by the signal of the IDX input (3-signal QEI mode) or by other software means (software directly setting the initial position). The capture and compare register can be used to store the position value by defining a capture condition.

When a direction change (DC) occurs, a DC interrupt is generated in the RIS register.

Figure 15-18 shows the state machine in the QEI interface to detect the directional rotation from the two CCP input signals, PHA and PHB.

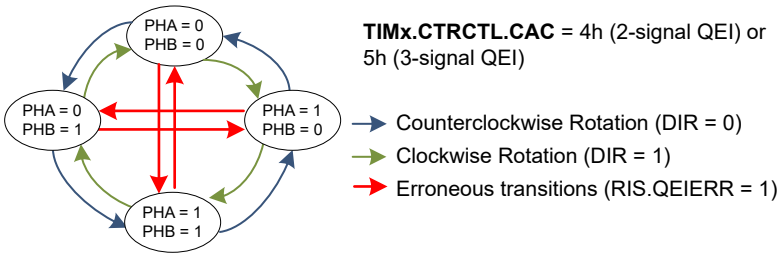


Figure 15-18. State Machine for 2-signal and 3-signal QEI mode

QEI 2-Signal Mode Configuration

1. Configure PINCMx for TIMGx_C0 (PHA) and TIMGx_C1 (PHB).
2. Set TIMG.CCCTL_01[0].COC = 1 and TIMG.CCCTL_01[1].COC = 1 for capture mode for both CCP channels 0 and 1 (PHA and PHB).
3. Configure CCP as an input for each CC block by setting respective bits in the CCPD register. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
4. Set the TIMx.LOAD to the encoder resolution, such as 4000.
5. In the CTRCTL register, set the CAC, CZC, and CLC bits to 4h.
6. Configure input capture settings as described in Section 15.2.3.1.1, if desired.
7. Enable the counter to start counting by setting EN = 1.

Example using QEI mode with 2 input signals

The behavior of PHA/PHB follows Table 15-12 and Figure 15-19.

Table 15-12. PHA/PHB State Table and Counter Actions

Current State (PHA, PHB)	Next State (PHA, PHB)	Direction (DIR)	Counter Action
00	10	1 (Up)	+1 (If TIMx.CTR = LOAD, then CTR = 0)
10	11		
11	01		
01	00		
00	01	0 (Down)	-1 (If TIMx.CTR = 0, then CTR = TIMx.LOAD)
01	11		
11	10		
10	00		

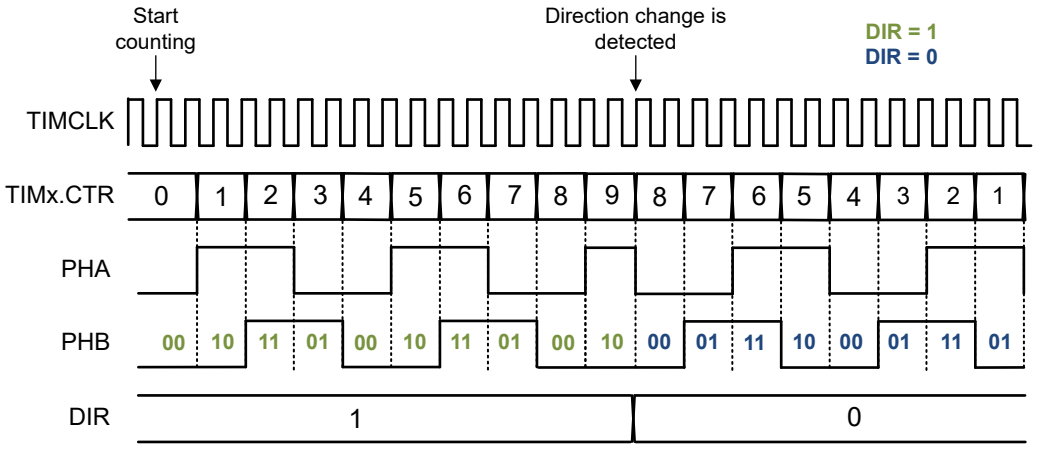


Figure 15-19. 2-Signal QEI Operation

15.2.3.1.3.2 QEI With Index Input

3-signal QEI mode is similar to the 2-signal mode with additional index input signal IDX. Generally, IDX input pulses once per rotation which can be used to reset the counter. It is used in one of two applications:

- One IDX pulse is generated each movement cycle. In this case, the accumulated position represents the fraction of the movement cycle.
- An IDX pulse is generated at a specific position in a noncyclic movement.

QEI 3-Signal Mode Configuration

1. Configure PINCMx for TIMGx_C0 (PHA), TIMGx_C1 (PHB), and TIMGx_IDX (IDX).
2. Set TIMG.CCCTL_01[0].COC = 1 and TIMG.CCCTL_01[1].COC = 1 for capture mode for both CCP channels 0 and 1 (PHA and PHB).
3. Configure CCP as an input for each CC block by setting respective bits in the CCPD register. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
4. Set the TIMx.LOAD value to the encoder resolution, such as 4000.
5. In the CTRCTL register,set the CAC, CZC, and CLC bits to 5h.
6. Configure input capture settings as described in Section 15.2.3.1.1, if desired.
7. Enable the counter to start counting by setting EN = 1.

The IDX input is sampled when a rising edge is detected. The IDX signal affects the counter value depending on the direction as shown inTable 15-13 and

Table 15-13. Relation of IDX Input and Counter Value

Direction (DIR)	IDX	Counter Action
1	Rising	Zero (TIMx.CTR is set to zero)
0	Rising	Load (TIMx.CTR is set to TIMx.LOAD)

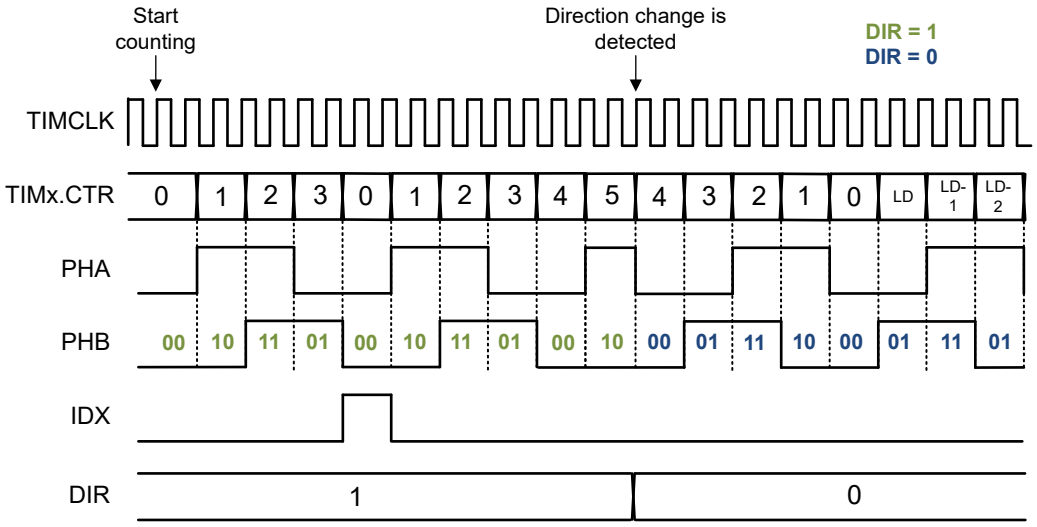


Figure 15-20. 2-Signal QEI with Index Input Operation

15.2.3.1.3.3 QEI Error Detection

The QEI module can detect erroneous transactions or state errors as shown in Figure 15-18. A QEIERR interrupt is generated and the counter or direction signal does not change in the error state.

Note

QEIERR can occur if the TIMCLK period is slower than the period of the PHA or PHB signals.

15.2.3.1.4 Hall Input Mode (TIMG with QEI support only)

In TIMGx instances with QEI support, three digital Hall signals can be input into CCP channel 0 (CCP0), CCP channel input 1 (CCP1), and IDX for position control of 3-phase Hall-sensored motor applications. Hall signals are used to detect real-time motor position in motor control applications and can be used for speed computation measurements, position control, or motor stall status.

Note

See Section 15.1.3 and the device-specific data sheet for TIMG instances that support QEI / Hall Input Mode.

Table 15-14 shows the signal mapping for Hall signals A (U), B (V), and C (W) to TIMG capture/compare input signals.

Table 15-14. Hall Input and TIMx Input Signal Mapping

Hall input signal	TIMx input
HALL A / HALL U	CCP0
HALL B / HALL V	CCP1
HALL C / HALL W	IDX

Note

Hall input signals should be digital inputs from Hall sensor ICs with a pullup to VCC.

As shown in Figure 15-21, the input capture module provides a 3-input XOR of synced CCP0, CCP1, and IDX signals to create a frequency generator (FG) signal. The XOR output signal is selected when IFCTL_xy[0/1].ISEL is set to 4h. See Figure 15-21 for XOR option in the input capture block diagram.

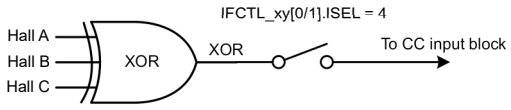


Figure 15-21. Hall 3-input XOR for Frequency Generator (FG) Signal to CC input block

The XOR'ed output signal is propagated to the CC block and a period or pulse-width capture can be used to compute the linear motor speed in relation to the calculated period or pulse width in the TIMx.CC register. See Section 15.2.3.1.2.2 and Section 15.2.3.1.2.3 on how to calculate period and pulse-width captures based on the XOR'ed input signal.

Figure 15-22 shows the input signal to the CC block which can be used for speed calculations.

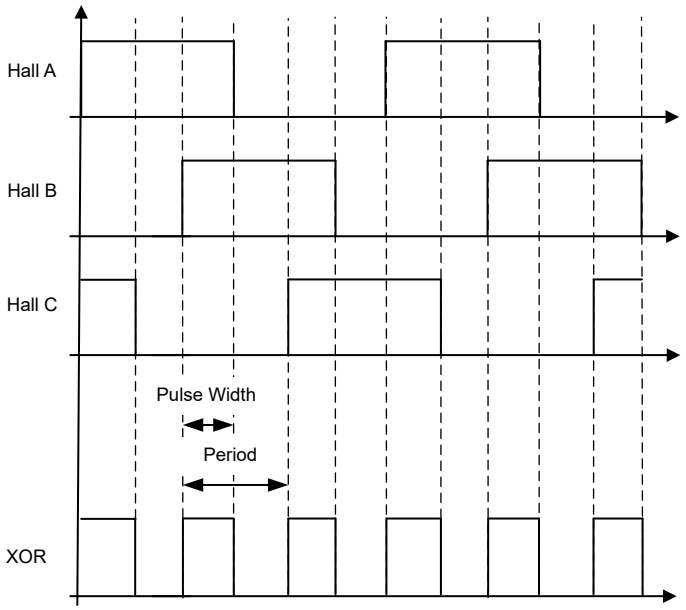


Figure 15-22. Hall 3-Input XOR Signal used with Pulse Width or Period Capture for Speed Computation

Hall Input Mode Configuration

1. Configure PINCMx for TIMGx_C0 (HALLA), TIMGx_C1 (HALLB), and TIMGx_IDX (HALLC).
2. Set TIMG.CCCTL_01[0].COC = 1 and TIMG.CCCTL_01[1].COC = 1 for capture mode for both CCP channels 0 and 1 (HALLA and HALLB).
3. Configure CCP as an input for each CC block by setting respective bits in the CCPD register. For instance, if TIMx Channel 0 is an input, set CCPD.C0CCP0 = 0.
4. Set the TIMx.LOAD value.
5. Set TIMG.IFCTL_xy[0/1].ISEL = 4h to select the XOR option for Hall signals.
6. Enable the counter to start counting by setting EN = 1.

15.2.3.2 Compare Mode

Compare mode is selected when TIMx.CCCTL_xy[0/1].COC = 0. Compare mode is used to generate a compare event to output PWM output signals at specific time intervals. Compare events can be used to generate a timing base internally or generate a PWM output with specific profiles using active, inactive, or toggle action behaviors, and optional deadband insertion.

Many types of compare mode events can be generated based on the configuration of the CC action register CCACT_xy[0/1]:

- Compare events: occurs for a CC channel when TIMx.CTR counts up (CCU) or down (CCD) to the value in TIMx.CC_xy[0/1]

- Secondary compare events: occurs when a secondary CCx block compare up (CC2U) or down (CC2U) is configured for another CCx block's CCU event or CCD event, respectively. This enables more flexible output generation from other external events such as comparator outputs or fault signals. It can be useful for real-time control applications, like digital power or motor control.

In TIMA only, an additional internal 5th and 6th CC block (TIMA.CC_45[0/1]) can be used for internal secondary compare events while continuing to use CC channels with dedicated output pins for external PWM signal generation.

Table 15-15 shows the types of compare mode events that can be generated and conditions to generate the events.

Table 15-15. Compare Mode Events

Event	Name	Event Condition
CCDn (n = CC channel)	Capture/compare down event	When timer is counting down, TIMx.CTR = TIMx.CC_xy[0/1]
CCUn (n = CC channel)	Capture/compare up event	When timer is counting up, TIMx.CTR = TIMx.CC_xy[0/1]
CC2Dn (n = CC channel)	Secondary capture/compare down event	Occurs when CC2SELD is configured for the source of the CCDn event
CC2Un (n = CC channel)	Secondary capture/compare up event	Occurs when CC2SELU is configured for the source of the CCUn event

Note

Look at the device specific data sheet to check how many CC channels are available in each TIMx instance on the device.

15.2.3.2.1 Edge Count

In addition to event or PWM output generation, compare mode can also be used for input signal edge counting to determine when a number of edges has been detected. In edge count operation, a CCP input edge can advance the counter based on the ACOND condition. The counter register is initialized with the starting value, and the number of detected CCP input edges at any time can increment or decrement depending on the counting mode configuration. The user can count rising edges, falling edges, or both edges by configuring the CCOND value.

Edge Count Configuration

1. Set TIMx.CCCTL_xy[0/1].COC = 0 for compare mode.
2. Optionally set the corresponding TIMx.CC_xy[0/1] to a compare value to generate a compare interrupt when the counter reaches this value.
3. In the CTRCTL register, set the desired counter control settings for:
   a. Counting mode (CM) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Zero (CZC), advance (CAC), and load control (CLC) to specify what condition controls zeroing, advancing, or loading the counter
   c. Repeat or one-shot mode (REPEAT)
4. Set ACOND to a setting to advance the counter based on the input edge polarity.
5. Configure input capture settings as described in Section 15.2.3.1.1, if desired.
6. Enable the counter by setting EN = 1.

Example using edge count operation using up-counting mode

In up-counting mode starting from zero (CM = 2, CVAE = 2), the expected internal timing for rising edge count operation to increment the counter is shown in Figure 15-23.

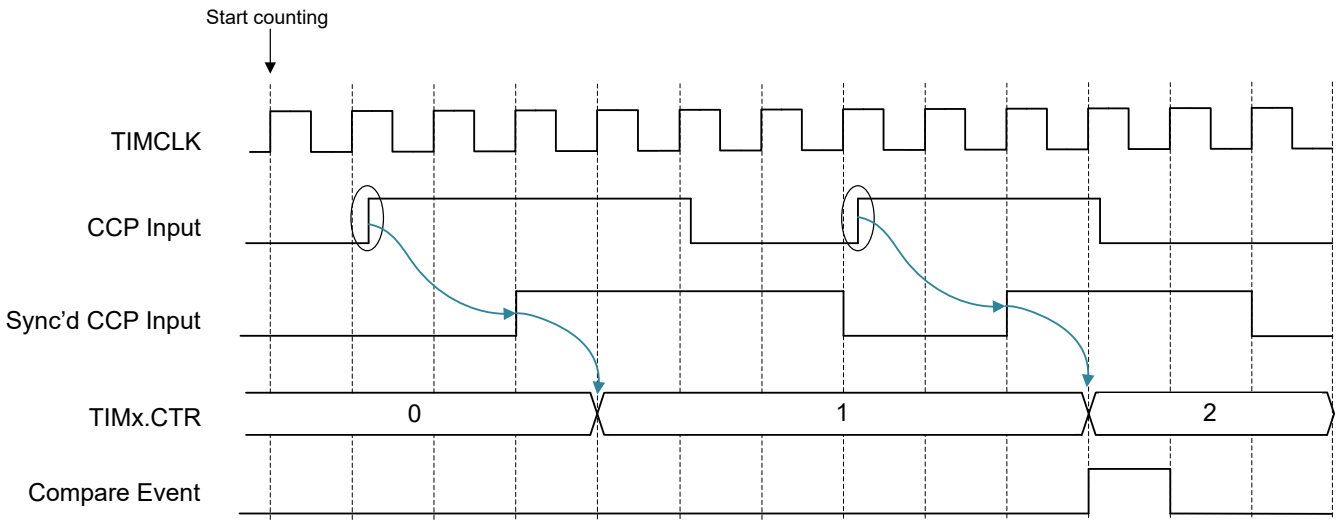


Figure 15-23. Edge Count Operation to Generate Compare Event (TIMx.CC = 2)

15.2.4 Shadow Load and Shadow Compare

Some timer modules have a shadow load and shadow compare register feature which gives the user the flexibility of holding the update of load and CC values until a certain event occurs. This is useful in timing-critical applications where PWM control signals need to be updated with correct timings, such as duty cycle updates. Refer to the timer features for specific configurations of the TIMx modules.

Note

See Section 15.1.3 and the device-specific data sheet for TIMx instances that support Shadow Load and Shadow Compare.

15.2.4.1 Shadow Load (TIMG4-7, TIMA only)

On shadow-load capable timers, the shadow load feature allows holding the update of load values until a zero event occurs. To enable shadow loading, set the TIMx.GCTL.SHDWLDEN bit while the timer is enabled (EN = 1).

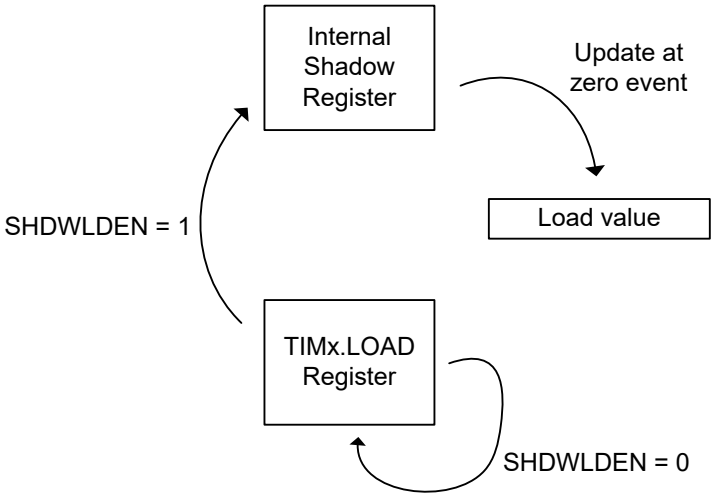
If the TIMx module has a shadow load feature, there is an internal shadow register for the load value (TIMx.LOAD). The shadow register will update the load value at a zero event as shown in Figure 15-24.

Note

On shadow-load capable timers, SHDWLDEN must be set or else the load value will not update.

If the timer instance does not have shadow load capability (standard timer), the load value will update immediately when TIMx.LOAD is written to.

Shadow-load capable timers update the load value from the internal shadow register when:
- TIMx.CTRCTL.EN = 1
- GCTL.SHDWLDEN = 1
- Available for TIMG4-7, TIMAx



Standard timers update the load value from TIMx.LOAD immediately when:
- TIMx.CTRCTL.EN = 1
- Available for TIMG0-3, TIMG8-14

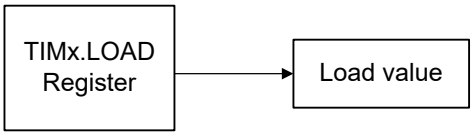


Figure 15-24. Load value updates for TIMx instances when EN=1

Note

To update the load value for shadow load capable timers, the timer must be disabled before changing the TIMx.LOAD value.

When TIMx.GCTL.SHDWLDEN = 1, load values update at zero events for all counting modes. Consult the counting mode operations below to determine if a shadow load is needed:

- In down-counting modes, since TIMx.LOAD value is updated when a zero event occurs, a shadow load is not needed in these modes.
- In up/down counting mode, TIMx.LOAD is compared with the counter value to determine if the peak is reached and when to start to counting down. A shadow load is necessary to ensure that TIMx counts up to the load value before the zero event, or else the load value can update immediately and cause incorrect timings.
- In up-counting mode, the timer counts to TIMx.LOAD. A shadow load is necessary to ensure that TIMx counts up to the load value before the zero event, or else the load value can update immediately and cause incorrect timings.

Figure 15-25 shows an example of how shadow load and shadow compare takes effect at the zero event for both the TIMx.LOAD and TIMx.CC value in up/down counting mode.

15.2.4.2 Shadow Compare (TIMG4-7, TIMG12-13, TIMA only)

When shadow compare is enabled for updating the capture/compare register (TIMx.CC), the value written to the respective compare register is first stored into a shadow compare register and then transferred to the compare register at different events configured by setting the TIMx.CCCTL_xy[0/1].CCUPD bits.

Additionally, the capture/compare action register (TIMx.CCACT) has the ability to update the action at different events configured by setting the TIMx.CCCTL_xy[0/1].CCACTUPD bits.

Table 15-16 shows the settings for configuring when shadow compare and actions occur at different events.

Table 15-16. Shadow Compare and Action Update Behavior

Bit Field	Value	Description/Comment
CCUPD / CCACTUPD	0	The value written to TIMx.CC register take effect immediately.
	1	The value written to the TIMx.CC register is stored in a shadow compare register and gets transferred to the TIMx.CC register in the TIMCLK cycle following a zero event (TIMx.CTR value equals 0).
	2	The value written to the TIMx.CC register is stored in a shadow compare register and gets transferred to the TIMx.CC register in the TIMCLK cycle following a compare (down) event (TIMx.CTR value equals TIMx.CC)
	3	The value written to the TIMx.CC register is stored in a shadow compare register and gets transferred to the TIMx.CC register in the TIMCLK cycle following a compare (up) event (TIMx.CTR value equals TIMx.CC)
	4	The value written to the TIMx.CC register is stored in a shadow compare register and gets transferred to the TIMx.CC register in the TIMCLK cycle following a zero or load event (TIMx.CTR value equals 0 or TIMx.CTR equals TIMx.LOAD). **Note: this update mechanism is defined for use only in up/down counting mode.**
	5	The value written to the TIMx.CC register is stored in a shadow compare register and gets transferred to the TIMx.CC register in the TIMCLK cycle following a zero event and the repeat count equaling zero (TIMx.CTR value equals 0 and TIMx.RC equals 0)
	6	The value written to the TIMx.CC register is stored in a shadow compare register, and gets transferred to the TIMx.CC register in the TIMCLK cycle following a trigger pulse. See Section 15.2.7.

Figure 15-25 shows an example of how shadow load and shadow compare takes effect at the zero event for both the TIMx.LOAD and TIMx.CC value in up/down counting mode.

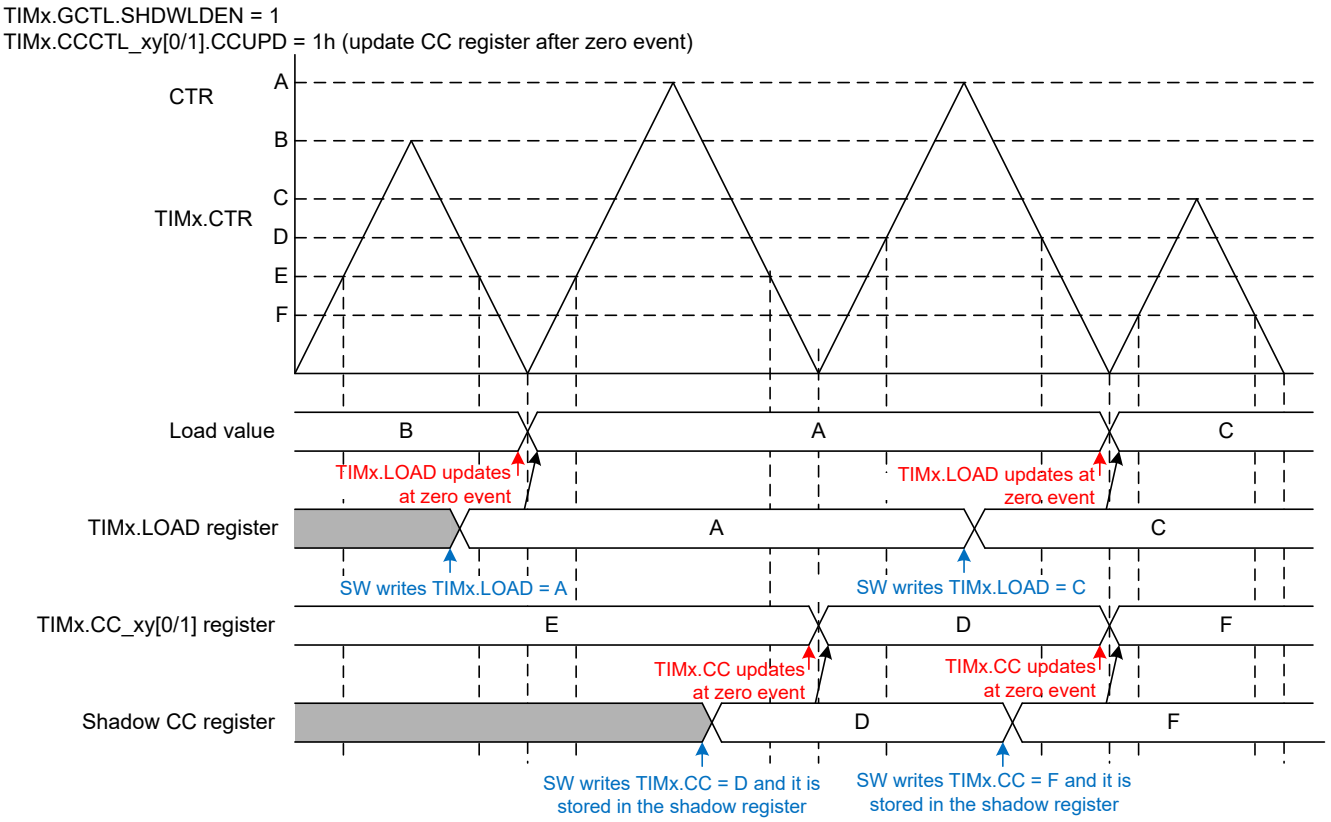


Figure 15-25. Shadow Load and Shadow Compare Taking Effect at Zero Event in Up/Down Mode

15.2.5 Output Generator

The output signal generation unit can be used with the counter and capture/compare modules to generate desired pulse-width modulation (PWM) output waveforms, event signals, synchronized capture inputs, or the

counter direction. Many output waveforms are generated from counter events (load, zero, counter direction) and the capture/compare block (compare match).

TIMA and TIMG have many common features in the output generation signal unit. Additionally, TIMA has advanced output generation features such as complimentary output signals, deadband insertion, and fault generation.

Figure 15-27 shows the TIMG output block diagram.

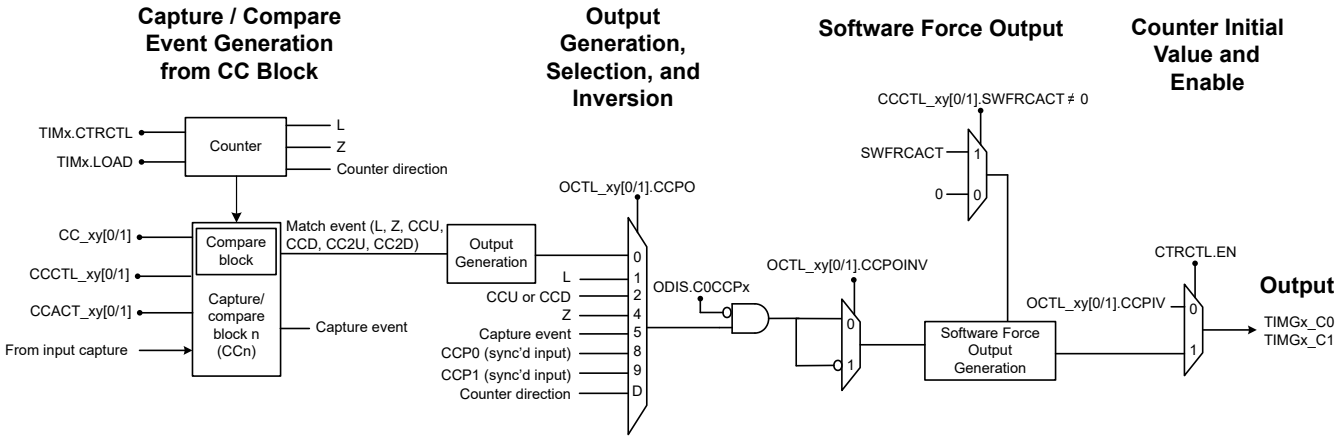


Figure 15-26. Output Connection for TIMG

Figure 15-27 shows the TIMA output block diagram.

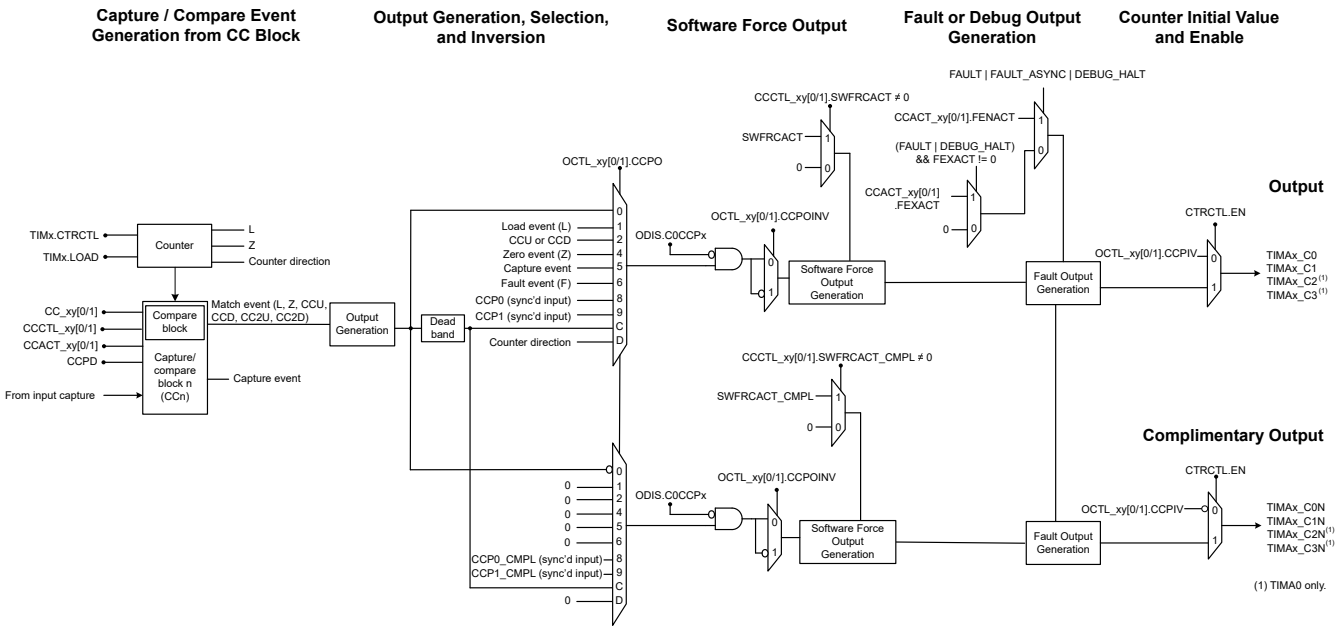


Figure 15-27. Output Connection for TIMA

Signal Generator Actions

Table 15-17 shows the types of signal generator actions capable by the output generator. Signal generator actions are configured in the CCACT_xy[0/1] register for zero, load, and compare events. For types of compare events, see Table 15-15.

Table 15-17. Signal Generator Actions from Compare Event

Value	Action
0h	Event is disabled and a lower priority event is selected if asserting
1h	CCP output value is set high
2h	CCP output value is set low
3h	CCP output value is toggled

The key registers for generation of output signals are:

- **LOAD**: the contents of this register are copied to the counter (TIMx.CTR) on any operation designated to do a "load". This value is also used to compare with the counter value for generating a "Load Event" that can be used for interrupt, trigger, or signal generator actions.
- **CCPD:** this register configures the direction of the CCP pins as inputs or outputs.
- **CC_xy[0/1]**: this is a register used as a compare value to the current counter to create an match event.
- **CTRCTL**: this register provides control over the counter operation in different conditions.
- **CCCTL_xy[0/1]**: this register controls the operations of the respective CC registers and the counter
- **OCTL_xy[0/1]**: this register controls the output of the capture-compare portion of the counter. This includes the ability to select the source of what is driven out along with initial condition values and final inversion options.
- **CCACT_xy[0/1]**: this register controls the actions of the signal generator of the capture-compare portion based on the events created in the counter block, the capture and compare block, and debug events.
- **ODIS:** this register disables the output signal selected by OCTL.CCPO (before conditional inversion) to allow software the ability to hold the CCP output low during configuration or shutdown.

These are key registers for configuring the compare mode to generate PWM signals:

15.2.5.1 Configuration

There are five stages to configuring output signal generation in TIMx devices:

- Counter and CC Block Event Generation
- Output Generation, Selection and Inversion
- Software Force Output
- Fault Output Generation (TIMA only)
- Counter Initial Value and Enable

Counter and CC Block Event Generation

The counter block contains the counter and produces a load event (L), zero event (Z), and direction of counting based on the counting mode used.

The CC blocks contain the CC register and can generate two types of output signals: compare match events and capture events. Please see Table 15-15 for the compare events that can be generated.

Output Generation, Selection and Inversion

The TIMx.CCACT register specifies the waveform generation of a CCP output depending on the counting mode and counter compare actions.

TIMx.OCTL_xy[0/1].CCPO controls the CCP output selection from the output generation unit, output generation unit with deadband (TIMA only), counter events, compare events, capture events, fault events, or signal inputs. The output disable register (ODIS) can optionally disable the CCP output to optionally hold the CCP output low during configuration or shutdown. TIMx.OCTL_xy[0/1].INV controls final inversion options.

Note

Mux selections for synchronized inputs are tied to 0 for TIMAx CC2 and CC3 instances. Do not use TIMAx.OCTL_23[0/1].CCPO = 8 or 9.

On TIMA devices only, CCP complimentary output channels can be generated from the output generation unit (denoted by "N" in the signal name). For instance, TIMA0 channel 2 (TIMA0_C2) can also produce a complimentary output (TIMA0_C2N). The CCPO and INV bits also controls the selection and inversion options for the complementary output.

Complimentary outputs with deadband insertion are a common use case for inverter-based applications with half-bridge topologies. For more information, please see Section 15.2.5.2.4.

Software Force Output

The output of the signal generator can be overwritten in software by setting CCCTL_xy[0/1].SWFRCACT to a nonzero setting. For TIMA devices only, the complementary output of the signal generator can be overwritten by setting CCCTL_xy[0/1].SWFRCACT_CMPL to a nonzero setting.

For more information, see Section 15.2.5.3.

Fault / Debug Output Generation (TIMA only)

On TIMA devices, the CCP output can be overwritten after the software force output block if there is a system fault (FAULT), fault condition upon exit (FEXACT), fault condition upon entry (FENACT), an asynchronous fault (FAULT_ASYNC), or the debugger is halted (DEBUG_HALT).

For more information, see Section 15.2.6 and Section 15.2.10.

Counter Compare Initial Value and Enable

To specify an initial value for the CCP output while the counter is disabled, set OCTL_xy[0/1].CCPIV to 0 for a low value or 1 for a high value. This is useful for applications where CCP outputs need to be in a default state before enabling the counter.

To enable the counter, set TIMx.CTRCTL.EN to 1.

15.2.5.2 Use Cases

Several different use cases can be achieved with the output generator and are discussed in the following sections.

15.2.5.2.1 Edge-Aligned PWM

To generate edge-aligned PWMs, TIMx can be configured for up- or down-counting mode and the PWM period in TIMCLK cycles is TIMx.LOAD + 1. The waveform uses load, zero, and compare events to drive the CCPx output high or low depending on the configuration settings of the compare/capture block and counter.

Edge-Aligned PWM Configuration

To generate edge-aligned PWMs using compare match events from the counter:

1. In the TIMx.CTRCTL register, set the desired counter control settings for:
   a. Up-counting (CM = 2) or down-counting mode (CM = 0) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Zero (CZC), advance (CAC), and load control (CLC) to specify what condition controls zeroing, advancing, or loading the counter
   c. Repeat or one-shot mode (REPEAT)
2. Set the TIMx.LOAD value to configure the PWM period.
3. Set the TIMx.CC_xy[0/1] value to configure the duty cycle.
4. Set TIMx.CCCTL_xy[0/1].COC = 1 for compare mode.
5. Configure CCP as an output for the CC block by setting respective bit in the CCPD registers. For instance, if TIMx Channel 0 is an output, set CCPD.C0CCP0 = 1.
6. In TIMx.CCACT_xy[0/1], set the CCP output action settings for compare events, zero events, load events, software force action, or fault events (TIMA only).
7. In TIMx.OCTL_xy[0/1], set CCPO = 0 to select the signal generator output.
8. Enable the corresponding CCP output by setting ODIS.C0CCPn to 1 for the corresponding counter n.

9. Configure polarity of the signal using the CCPOINV bit, and configure CCPIV to specify the CCP output state while disabled.
10. Enable the counter by setting TIMx.CTRCTL.EN = 1.

Example using edge-aligned PWM in down-counting mode

A typical 2-channel edge-aligned PWM generation for down-counting mode is shown in Figure 15-28 with the following edge-aligned PWM output waveforms:

- CCP0 output generates:
  - High pulse-width from TIMx.LOAD to TIMx.CC0 value (LACT = 1h)
  - Low pulse-width from TIMx.CC0 value to zero (CDACT = 2h)
- CCP1 output generates:
  - High pulse-width from TIMx.LOAD to TIMx.CC1 value (LACT = 1h)
  - Low pulse-width from TIMx.CC1 value to zero (CDACT = 2h)

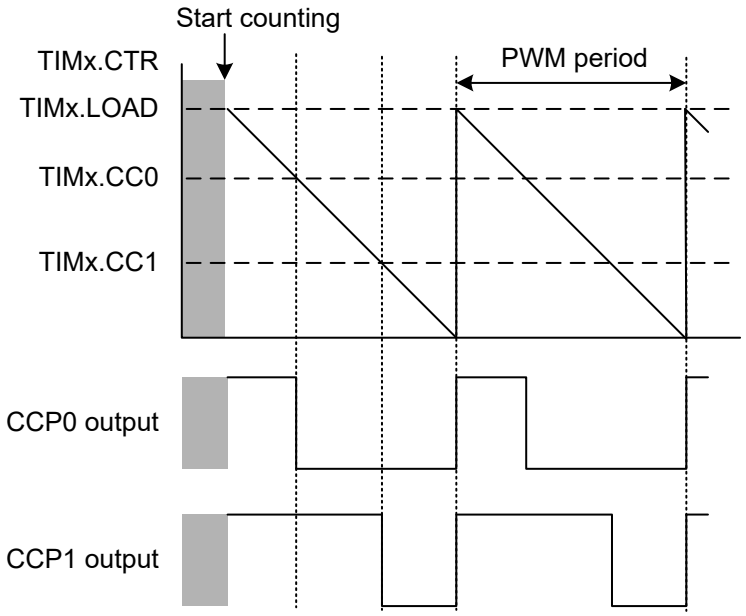


Figure 15-28. Edge-Aligned PWM Signals in Down-Counting Mode

Example using edge-aligned PWM in up-counting mode

A typical 2-channel edge-aligned PWM generation for up-counting mode is shown in Figure 15-28 with the following edge-aligned PWM output waveforms:

- CCP0 output generates:
  - High pulse-width from zero to TIMx.CC0 value (ZACT = 1h)
  - Low pulse-width from TIMx.CC0 value to TIMx.LOAD (CUACT = 2h)
- CCP1 output generates:
  - High pulse-width from zero to TIMx.CC1 value (ZACT = 1h)
  - Low pulse-width from TIMx.CC1 value to TIMx.LOAD (CUACT = 2h)

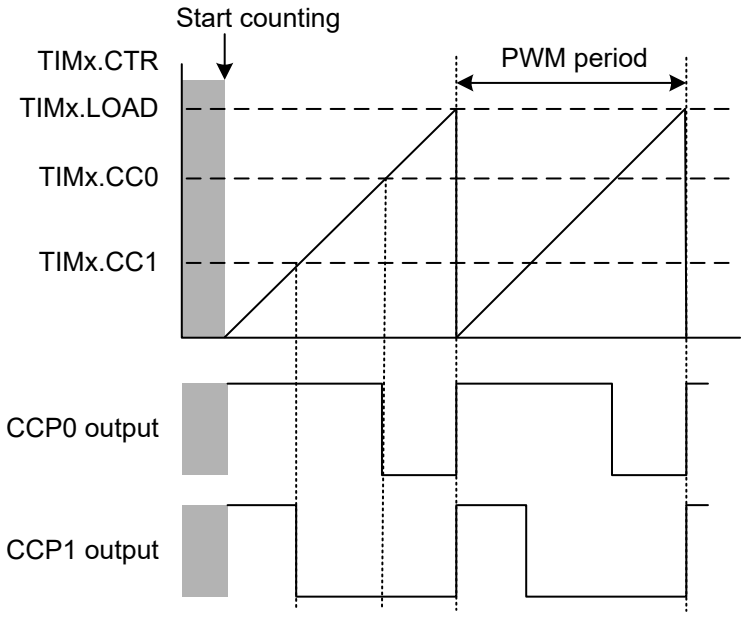


Figure 15-29. Edge-Aligned PWM Signals in Up-Counting Mode

15.2.5.2.2 Center-Aligned PWM

To generate center-aligned PWMs, TIMx is configured for up/down counting mode and the TIMx.LOAD value contains the half-period. The waveform uses up compare events and down compare events to drive the CCPx output high or low depending on the configuration settings of the compare/capture block and counter.

In TIMCLK cycles, the PWM period is (2 * TIMx.LOAD) and the duty cycle is 1 - (TIMx.CC_xy[0/1] / TIMx.LOAD).

Center-Aligned PWM Configuration

To generate center-aligned PWMs using compare match events from the counter:

1. In the TIMx.CTRCTL register, set the desired counter control settings for:
   a. Up/down counting mode (CM = 1) and counter value after enable (CVAE) (see as described in Section 15.2.2)
   b. Zero (CZC), advance (CAC), and load control (CLC) to specify what condition controls zeroing, advancing, or loading the counter
   c. Repeat or one-shot mode (REPEAT)
2. Set the TIMx.LOAD value to configure the PWM period.
3. Set the TIMx.CC_xy[0/1] value to configure the duty cycle.
4. Set TIMx.CCCTL_xy[0/1].COC = 1 for compare mode.
5. Configure CCP as an output for the CC block by setting respective bit in the CCPD registers. For instance, if TIMx Channel 0 is an output, set CCPD.C0CCP0 = 1.
6. In TIMx.CCACT_xy[0/1], set the CCP output action settings for compare events, zero events, load events, software force action, or fault events (TIMA only).
7. In TIMx.OCTL_xy[0/1], set CCPO = 0 to select the signal generator output.
8. Enable the corresponding CCP output by setting ODIS.C0CCPn to 1 for the corresponding counter n.
9. Configure polarity of the signal using the CCPOINV bit, and configure CCPIV to specify the CCP output state while disabled.
10. Enable the counter by setting TIMx.CTRCTL.EN = 1.

Example using center-aligned PWM in up/down counting mode

A typical 2-channel center-aligned PWM generation using up/down counting mode is shown in Figure 15-30 with the following center-aligned PWM output waveforms:

- CCP0 output generates:

  - High pulse-width from TIMx.CC0 compare up event to TIMx.CC0 compare down event (CUACT = 1h)
  - Low pulse-width from TIMx.CC0 compare down event to TIMx.CC0 compare up event (CDACT = 2h)
- CCP1 output generates:
  - High pulse-width from TIMx.CC0 compare up event to TIMx.CC0 compare down event (CUACT = 1h)
  - Low pulse-width from TIMx.CC0 compare down event to TIMx.CC0 compare up event (CDACT = 2h)

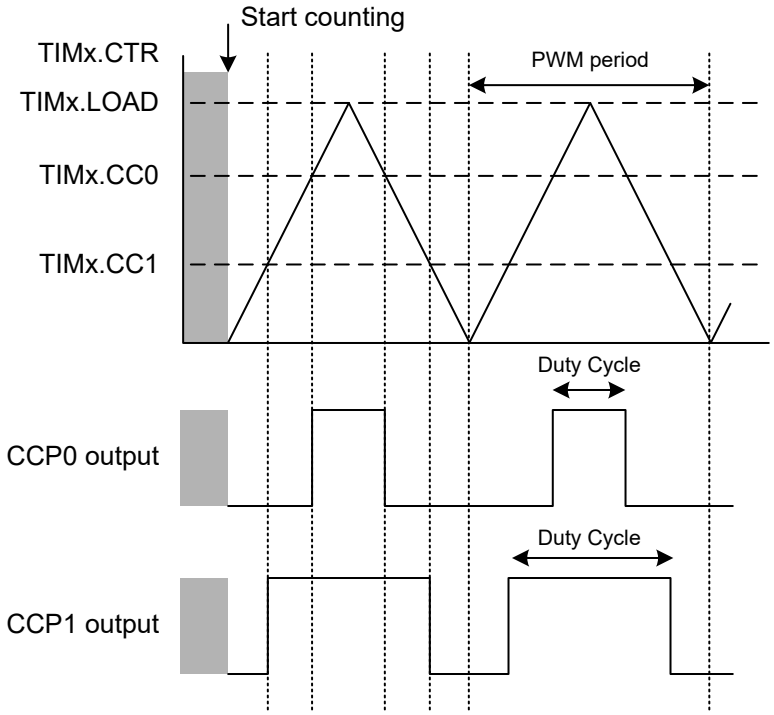


Figure 15-30. Center-Aligned PWM

15.2.5.2.3 Asymmetric PWM (TIMA only)

In TIMA only, asymmetric PWMs can be generated by generating two synchronized center-aligned PWM signals with a controlled phase shift. To generate the asymmetric PWM signals, the phase load feature is used as described in Section 15.2.2.5.

Asymmetric PWM Configuration

To generate asymmetric PWMs using compare match events from the counter:

1. Synchronize TIMA0 and TIMA1 using a cross trigger as described in Section 15.2.7.
2. Configure two center-aligned PWMs as described in Section 15.2.5.2.2 using TIMA0 and TIMA1. TIMA0 and TIMA1 should have the same load value (TIMA.LOAD) and compare value (TIMA.CC_xy[0/1]) to generate the same PWM frequency and duty cycle.
3. Add a phase shift value for TIMA0 or TIMA1 by configuring the phase load value TIMA.PL as described in Section 15.2.2.5.
4. Enable the counter by setting TIMA.CTRCTL.EN = 1.

Figure 15-31 shows an example of asymmetric PWM configuration using CCP channel 0 of TIMA0 and TIMA1.

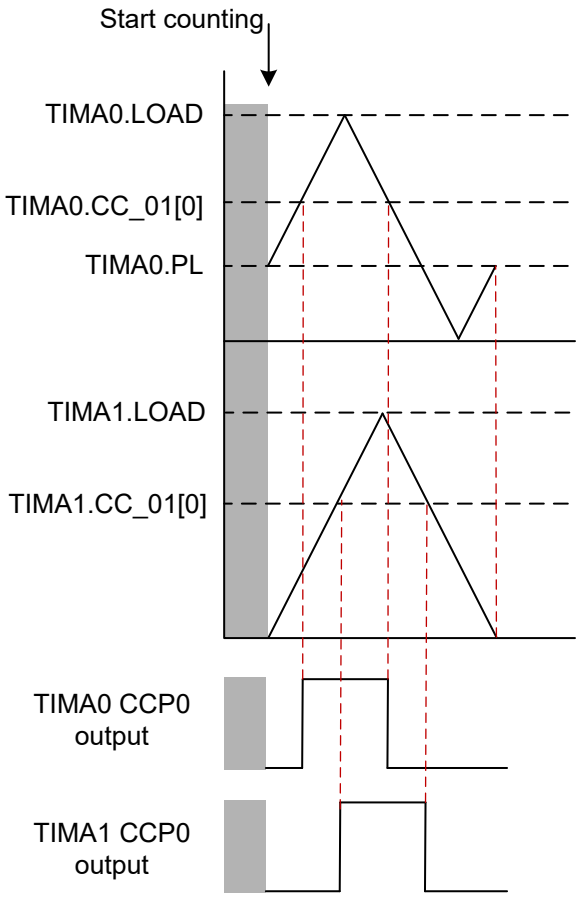


Figure 15-31. Asymmetric PWM Configuration with Phase Load for CCP channel 0 of TIMA0 and TIMA1

15.2.5.2.4 Complementary PWM With Deadband Insertion (TIMA only)

TIMA provides the option of generating complimentary PWM outputs with deadband insertion (nonoverlapping transitions in complimentary PWM signals) from a signal PWM reference signal. Deadband is useful for applications with half-bridge control to avoid shoot-through conditions, such as motor driver or inverter-based applications.

The reference signal is generated on the TIMAx_Cy signal, and the complimentary signal is generated on TIMAx_CyN, where x is the timer instance and y is the CCP output channel. For instances, a reference PWM signal generated on TIMA0 CCP output channel 2 will produce complimentary output signals on TIMA0_C2 and TIMA0_C2N.

The deadband control register (TIMAx.DBCTL) is programmed with the deadband mode and timing information. The deadband mode is selected using the M1_ENABLE bit, and the timing information to control the deadband width in TIMCLK cycles is selected by the RISEDELAY and FALLDELAY bit fields. RISEDELAY and FALLDELAY are a function of the rising or falling edge delay to or from the reference PWM.

See Table 15-18 for the configuration and relationship between deadband mode and deadband width settings.

Table 15-18. Deadband Modes and Delay Timing Configuration in DBCTL register

Deadband Mode	Bitfield	Description	Counting Mode
Mode 0	M1_ENABLE = 0	RISEDELAY is applied from **rising edge of reference PWM** to rising edge of TIMAx_Cy signal. FALLDELAY is applied from **falling edge of reference PWM** to rising edge of CxN signal.	Any

Table 15-18. Deadband Modes and Delay Timing Configuration in DBCTL register (continued)

Deadband Mode	Bitfield	Description	Counting Mode
Mode 1	M1_ENABLE = 1	RISEDELAY is applied from falling edge of TIMAx_CyN signal to **rising edge of reference PWM**. FALLDELAY is applied to **falling edge of reference PWM** to rising edge of TIMAx_CyN signal.	Up/down counting mode only

Deadband timing equation and example

The equations for configuring RISEDELAY and FALLDELAY from TIMCLK frequency and deadband timing is shown in Equation 14 and Equation 15.

$$RISEDELAY = f_{TIMCLK} \times t_{dead\_rise} \quad (14)$$

$$FALLDELAY = f_{TIMCLK} \times t_{dead\_fall} \quad (15)$$

For example, if 400ns of deadband is required when using a TIMCLK frequency of 80MHz, and Mode 1 is used with center-aligned PWMs to generate equal deadband every PWM period, then RISEDELAY = FALLDELAY = (80MHz) * (400ns) = 32.

Complimentary PWM with Deadband Configuration

1. Configure a PWM output for an edge-aligned PWM (Section 15.2.5.2.1) or center-aligned PWM (Section 15.2.5.2.2) for any CCP output channel in TIMA.
2. Configure TIMA.DBCTL with the specified deadband mode (M1_ENABLE) and deadband width RISEDELAY and FALLDELAY, depending on the deadband mode.
3. In TIMx.OCTL_xy[0/1], set CCPO = 0xC to select the signal generator with deadband output.
4. Enable the counter by setting TIMx.CTRCTL.EN = 1.

Example 1 - Complimentary PWM outputs with deadband using edge-aligned PWM in down-counting mode

For edge-aligned PWM, Mode 0 can only be used for deadband insertion mode. See Figure 15-32 for inserting configurable deadband using down counting mode, TIMA output channel 0, and edge-aligned PWM.

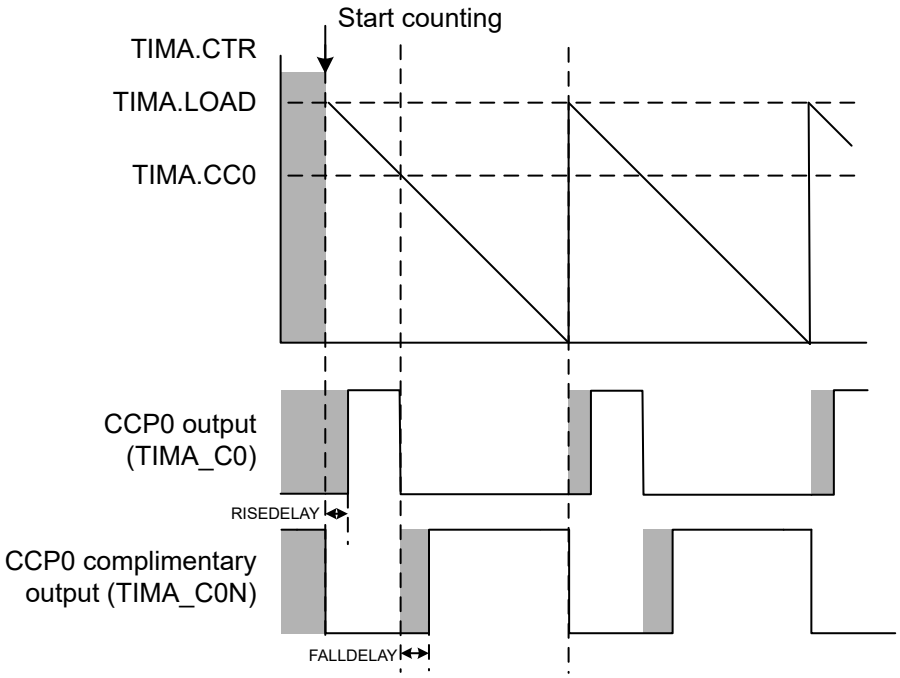


Figure 15-32. Deadband Insertion (Mode 0) for PWM in Down-Counting Mode

Example 2- Complimentary PWM outputs with deadband using center-aligned PWM

For center-aligned PWM, Mode 0 or Mode 1 can be used for deadband insertion mode. See Figure 15-33 for inserting configurable deadband using up/down counting mode, TIMA output channel 0, and center-aligned PWMs with deadband.

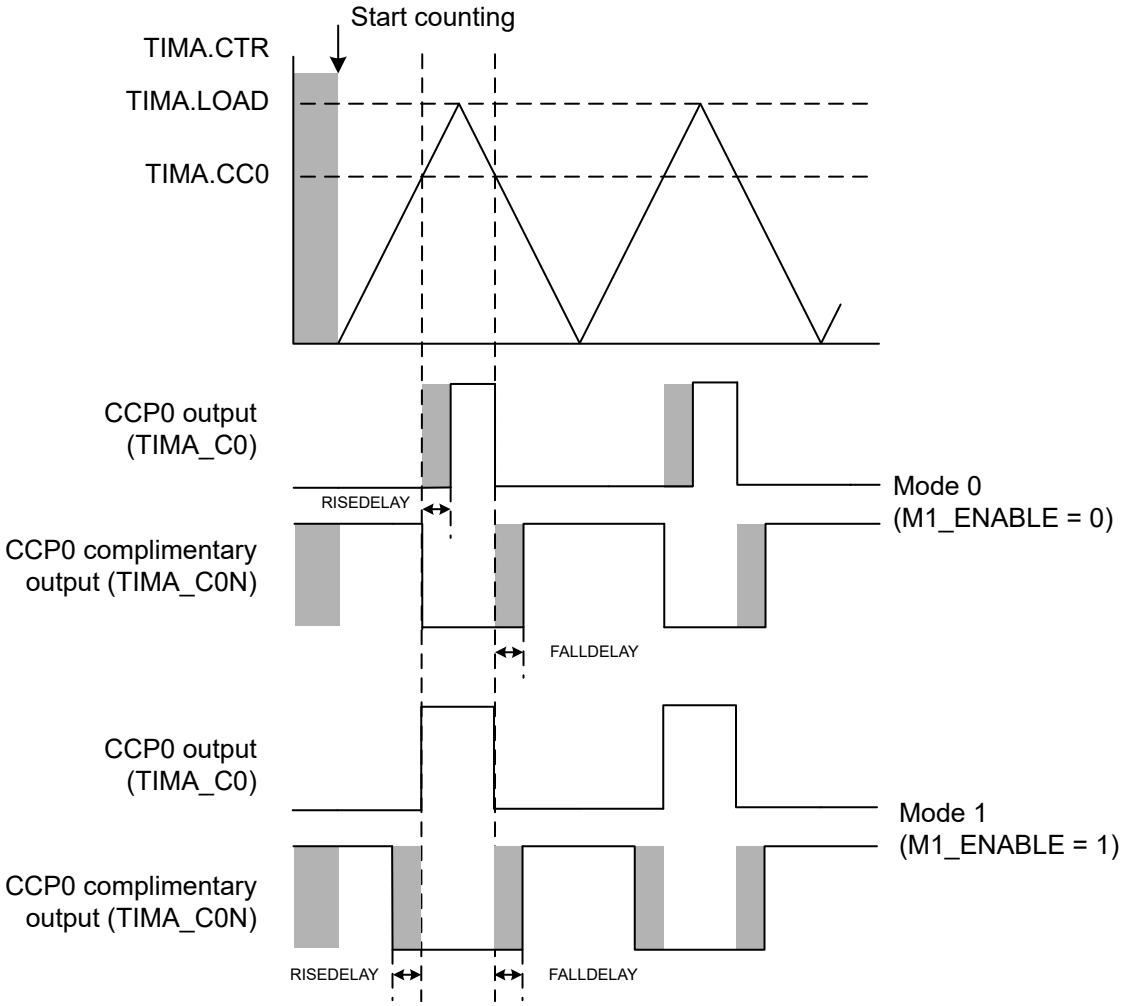


Figure 15-33. Dead Band Insertion for Center-Aligned PWM (Mode 0 and Mode 1)

15.2.5.3 Forced Output

Each output channel signal can be forced to a high or low level directly by software, independently of any comparison between the compare register and the counter. A shadow register exists to ensure the forced output action occurs at the end of the timer period.

The output of the CCP channel can be forced to high or low by setting the SWFRCACT bit in the TIMx.CCACT_xy[0/1] register.

In TIMA only, the complimentary output channel can also be forced to high or low by setting the SWFRCACT_CMPL bit in the TIMx.CCACT_xy[0/1] register.

Table 15-19 shows the software force output action configuration options.

Table 15-19. Force Output Action Configuration

Bit Field	Value	Description/Comment
SWFRCACT / SWFRCACT_CMPL	0	No forced output. Output is directly from the signal generation block.
	1	Force output high
	2	Force output low

15.2.6 Fault Handler (TIMA only)

In TIMA only, there are internal and external fault inputs which can be used to control the generation of PWM signals. The intended use of these inputs is as a mechanism for internal or external circuitry to indicate a fault in the system. This allows the hardware to react quickly to the external fault while optionally signaling an interrupt for software correction and leaving the output signals in a safe state.

It is important to consider the following basic properties of faults in a system, such as:

- Fault input selection (fault signal from external IC, internal signal, etc.)
- How long a fault condition lasts, or the fault condition duration
- How the counter reacts to the entry and exit of a fault condition
- How the output signal reacts to the entry and exit of a fault condition

Fault conditions are synchronously detected using TIMCLK or asynchronously detected. Synchronous faults have a configurable glitch filter and can generate a latched fault event. Asynchronous faults cannot be latched and do not generate a fault event, with a latency of 1-2 TIMCLK cycles to detect the fault and perform a configured action. The CCP output can be configured for either type of fault upon entry and exit conditions.

The fault handler logic diagrams are split into three parts: asynchronous faults, synchronous faults, and fault output generation.

Figure 15-34 shows the asynchronous fault handler logic connections.

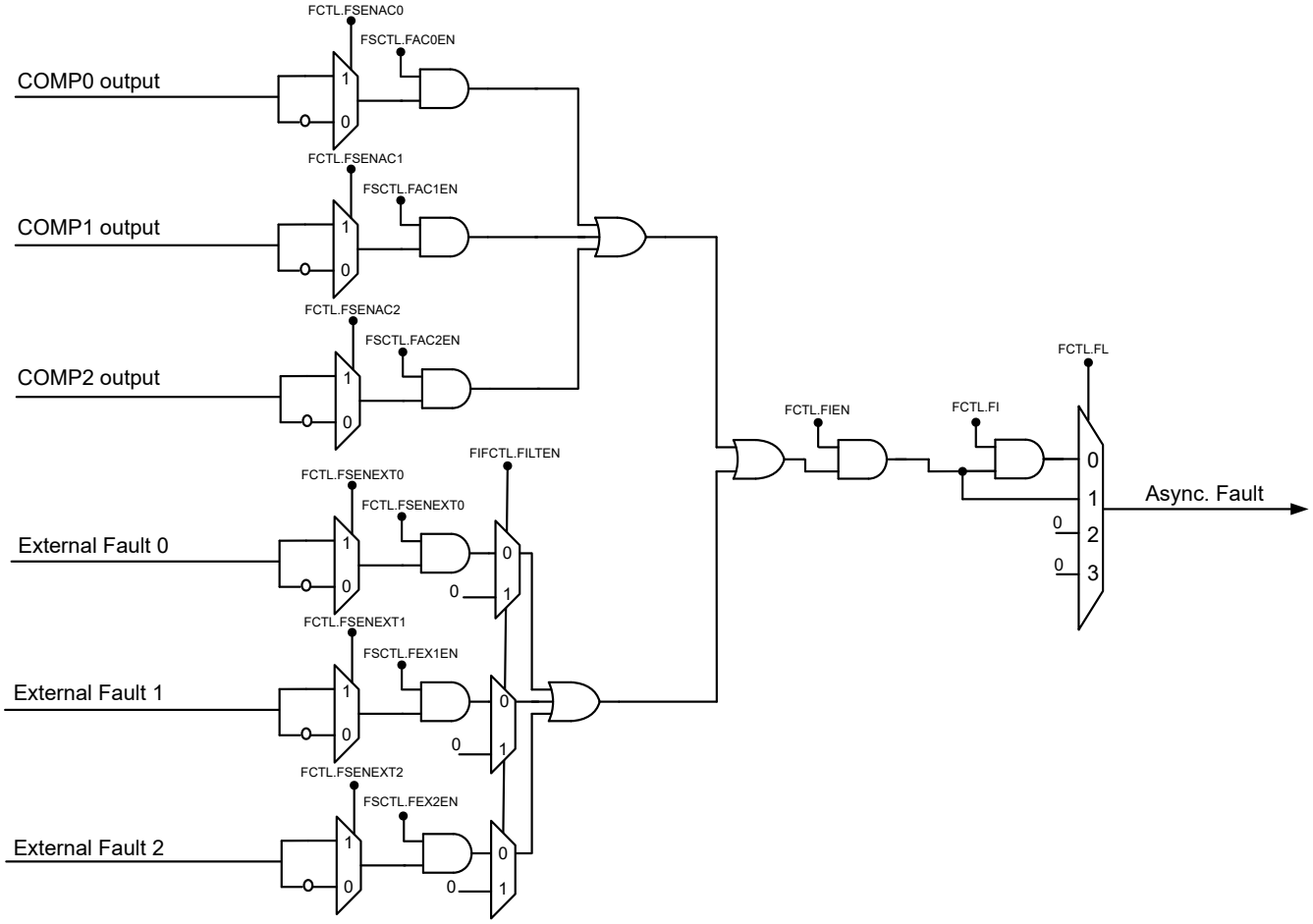


Figure 15-34. Asynchronous Fault Handler Connections

Figure 15-35 shows the synchronous fault handler logic connections.

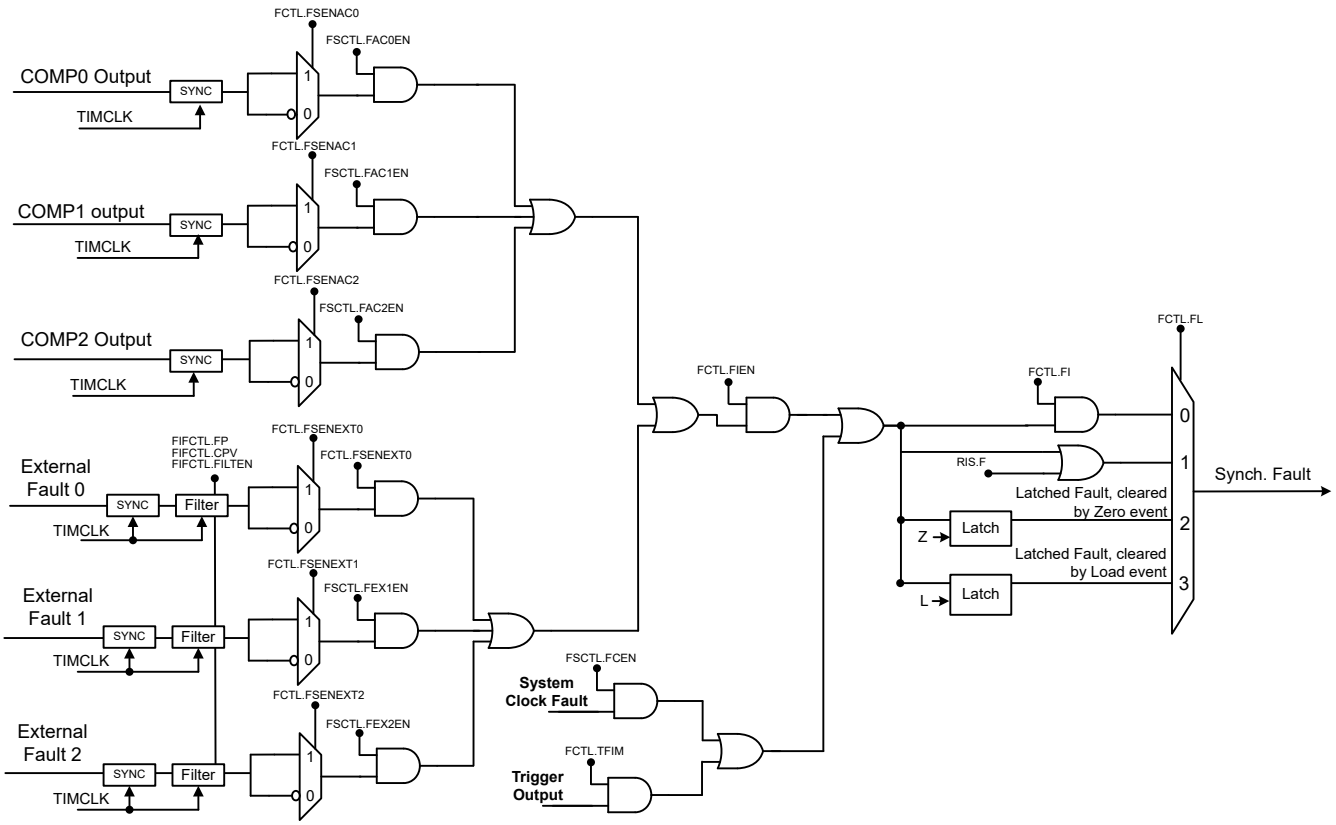


Figure 15-35. Synchronous Fault Handler Connections

Figure 15-36 shows the fault output generation logic connections.

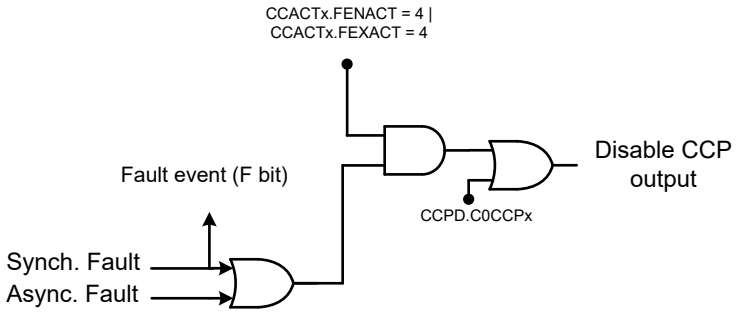


Figure 15-36. Fault Output Generation Connections

Key registers for configuring the fault handler are:

- **TIMA.FCTL**: this register controls the fault inputs, fault detection, and error handling behavior.
- **TIMA.FSCTL**: this register controls the fault source selection and enable.
- **TIMA.FIFCTL**: this register controls the input filtering (FILTEN, FP, CPV) for the fault input.
- **TIMA.CCACT_xy[0/1]**: this register controls the actions of the signal generator of the capture-compare portion based on fault events.

15.2.6.1 Fault Input Conditioning

The comparator and external fault pin fault source input passes through a two TIMCLK synchronization stage and the fault input can be filtered through the glitch filter using the fault input filter (TIMA.FIFCTL) register.

The fault input glitch filter can be enabled by setting the TIMA.FIFCTL.FILTEN bit. The filter period is configured by setting the TIMA.FIFCTL.FP bit.

A consecutive period or majority voting format selected by the TIMA.FIFCTL.CPV bit is used to select the criteria for a CCP input signal.

- **Consecutive period** - The fault input signal must be at the specified level for the defined number of FP timer clocks for the fault input to be processed.
- **Majority voting** - The filter ignores one clock of opposite logic over the filter period. For example, over the number of FP samples of the fault input, up to 1 sample may be of an opposite logic value (glitch) without affecting the output.

The example shown in Figure 15-13 shows the difference between consecutive period and majority voting formats with a digital filter implemented to capture a fault input of 3 TIMCLK periods.

15.2.6.2 Fault Input Sources

The fault control (FCTL) and fault source control (FSCTL) registers are used to select the polarity and enable various fault input sources as shown in Table 15-20.

To enable the final input for fault detection, set TIMA.FTCL.FIEN = 1.

There are four types of fault input sources that are available for synchronous or asynchronous fault detection:

Comparator (COMP) output

The comparator output is useful for fault detection when COMPs are used for detecting overcurrent or overvoltage events. To enable the comparator output for fault detection, set the TIMA.FSCTL.FACxEN bit, and configure the polarity to detect the fault using TIMA.FCTL.FSENACx bit (x = 0, 1, or 2 for COMP instance).

External Fault Pin

Many IC devices include a fault detection pin (i.e. nFAULT) that an MCU can detect when there is a fault condition in the system. There are 3 fault external signal pins (TIMA_FLTx) connected to every TIMA module, where x = 0, 1, or 2. Each signal pin can be enabled by setting the TIMA.FSCTL.FEXxEN bit, and the polarity of this signal to trigger a fault condition can be configured by using TIMA.FCTL.FSENEXTx bit (where x = 0, 1, or 2 for each TIMA_FLTx pin).

System Clock Fault

Any system clock fault can be used to trigger the PWM output(s) to a Hi-Z state. This can be enabled by setting the TIMA.FSCTL.FCEN bit.

Note

When a SYSCLK fault occurs, a device reset is generated. Various TIMA fault entry and exit options are invalid while the device is in reset.

Trigger

A trigger can be configured to generate a fault condition is detected. This is useful for performing diagnostics or creating fault dependencies from other peripherals in the event fabric. For trigger configuration, please see Section 15.2.7. The fault input mask can be enabled by setting the TIMA.FSCTL.TFIM bit.

Table 15-20. Fault Input Sources and Configuration

Signal name	Input source	Fault type	Polarity Bit	Enable Bit
COMP0_OUT	COMP0 output	Synchronous or Asynchronous	FSENAC0	FAC0EN
COMP1_OUT	COMP1 output		FSENAC1	FAC1EN
COMP2_OUT	COMP2 output		FSENAC2	FAC2EN
TIMA_FLT0	External Fault 0		FSENEXT0	FSENEXT0
TIMA_FLT1	External Fault 1		FSENEXT1	FSENEXT1
TIMA_FLT2	External Fault 2		FSENEXT2	FSENEXT2
SYSCLK	System Clock	Synchronous	-	FCEN
TRIG	Trigger Output		-	TFIM

15.2.6.3 Counter Behavior With Fault Conditions

There are two settings for specifying the counter behavior in fault conditions: TIMA.CTRCTL.FB (during fault behavior) and TIMA.CTRCTL.FRB (fault resume behavior). The counter should continue to be enabled (TIMA.CTRCTL.EN = 1) during the fault handler behavior.

The counter behavior of the fault condition is described in Table 15-21 and Figure 15-37.

Table 15-21. Counter Behavior in Fault Condition With TIMA.CTRCTL Register

Bit Fields				Counter Behavior
FB	FRB	CVAE	REPEAT	
0	X	X	0	Ignores fault mode. Counter continues to count during fault and stops when reaches zero.
			1/3	Ignores fault mode. Counter continues to count during fault and repeat.
1	0	X	0/1/3	Reacts immediately to fault mode. The counter stops counting immediately and throughout the fault mode. Upon exit of fault mode, the counter continues counting from where it left off.
	1	0	X	Reacts immediately to fault mode. The counter stops counting immediately and throughout the debug mode. Upon exit of fault mode, the counter restarts from LOAD value (restarts a down count).
		1		Reacts immediately to fault mode. The counter stops counting immediately and throughout the debug mode. Upon exit of debug mode, the counter restarts from where it paused at fault entry.
		2		Reacts immediately to fault mode. The counter stops counting immediately and throughout the fault mode. Upon exit of fault mode, it restarts from 0 value (restarts an up/down count).

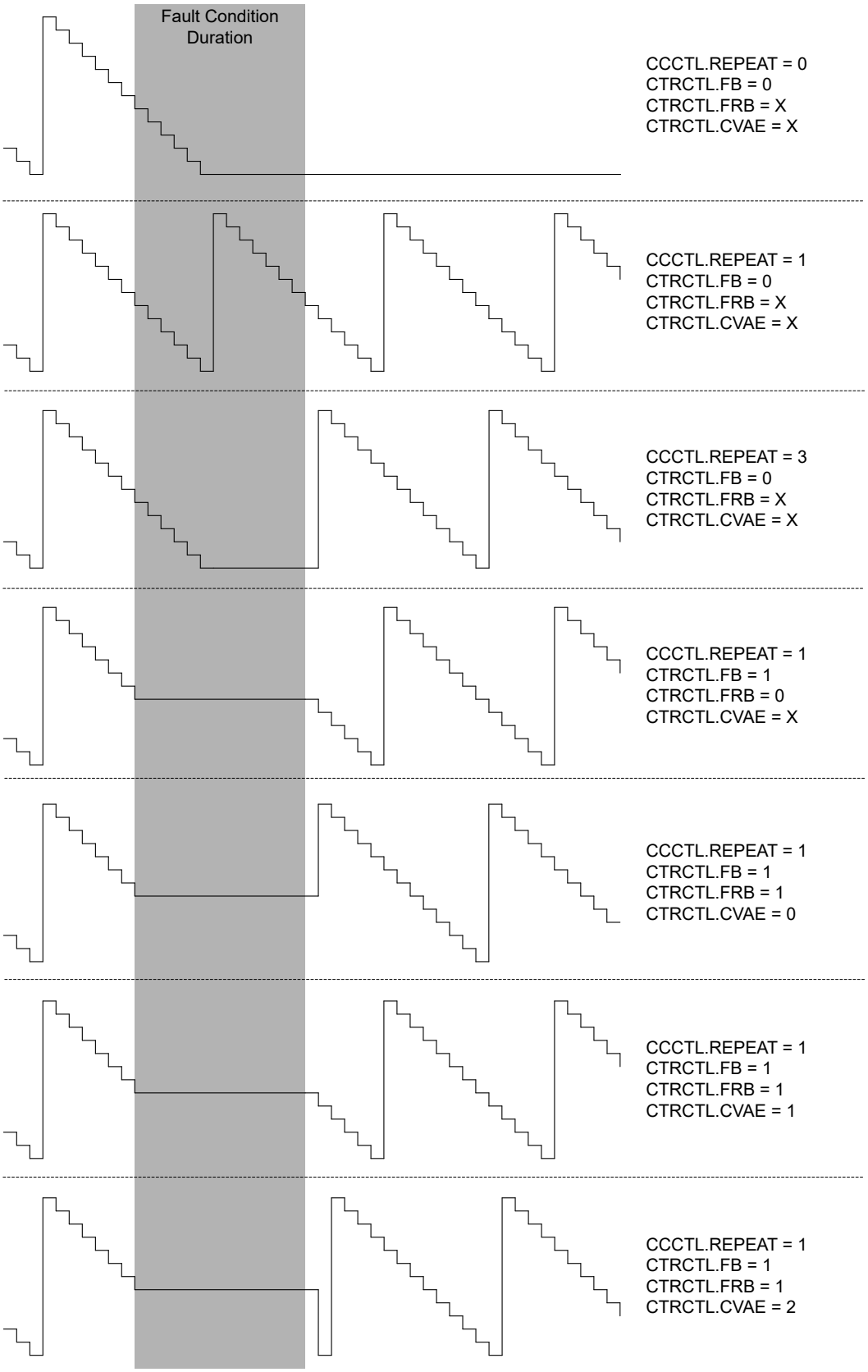


Figure 15-37. Counter Behavior in Fault Condition with TIMx.CTRCTL Register

15.2.6.4 Output Behavior With Fault Conditions

There are two settings for the CCP output channel behavior in a fault condition, TIMA.CCACT_01[0/1].FEXACT (fault exit behavior) and TIMA.CCACT_01[0/1].FENACT (fault entry behavior). The output behavior of fault condition is described in Table 15-22.

Table 15-22. CCP Output Behavior in Fault Condition With TIMA.CCACT Register

Bit Fields		Output Behavior
FEXACT	FENACT	
0		The CCP output value is unaffected by the fault event
1		CCP output value is set high
2		CCP output value is set low
3		CCP output value toggles
4		CCP output is tristated (Hi-Z)

Note

Fault entry and exit behavior is not dependent on the counter being enabled. They will always update the output when in fault mode. Enabling the counter through a software write when in fault mode will generate a load event that will be captured in RIS but the counter will not proceed. Load events will not affect the output when in fault mode. Fault events takes the absolute priority and no events can update the output until you are out of fault mode.

Note

If using the TIMA_FALx pin with the CCP capture channel inputs (such as adding deadband to complimentary PWMs) for low latency actions, an external connection should be in hardware between the TIMA_FALx pin and TIMAx_Cy pin (x = timer instance, y = CC channel).

15.2.7 Synchronization With Cross Trigger

When using a main-secondary timer configuration by connecting multiple timers together, the cross-trigger feature can instruct multiple timer modules in the same power domain or across different power domains using the event fabric to start counting simultaneously.

Cross-triggers can be enabled using software, compare events from other timer instances, zero or load events, or generic subscriber events. Some applications may require more than one counter block that can be simultaneously started across the same power domain (such as TIMA0 and TIMA1) or different power domains (such as TIMA0 and TIMG0).

This configuration uses cross triggers from a main timer module as the input trigger condition for the secondary timers. The timer cross trigger is essentially the combined logic of the hardware and software conditions that control the EN bit in the TIMx.CTRCTL register.

The cross triggers that are outputted from the main timer are connected to the external trigger input of other secondary timer modules. As shown in Figure 15-38, TIMGx is the main timer and TIMAx is the secondary timer that will be cross triggered in the configuration example.

Note

For power domains and cross trigger selection sources enabled for timer instances, refer to the device-specific data sheet.

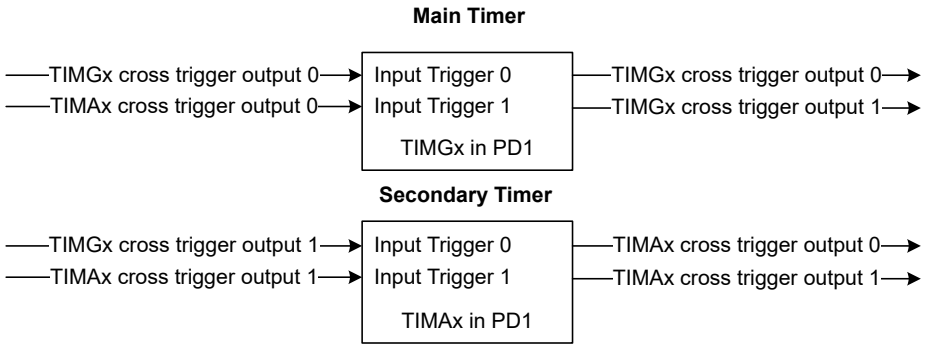


Figure 15-38. Cross Trigger Connections for Main Timer (TIMGx) and Secondary Timer (TIMAx) in Power Domain 1

15.2.7.1 Main Timer Cross Trigger Configuration

The following steps are used to configure the main timer cross trigger (which is TIMGx in this example):

1. Configure the main timer (which triggers other secondary timers) for the desired function, such as PWM output generation or using compare mode, to trigger other peripherals. See Section 15.2.5 for how to configure for PWM generation.
2. Select which cross trigger output needs to be generated. For example, in Figure 15-38, TIMGx cross trigger 1 can be used to trigger TIMAx and TIMGx cross trigger 0 can be used to trigger itself.
3. Enable the cross trigger output function by setting TIMx.CTTRIGCTL.CTEN bit to 1.
4. Choose how to trigger the start of these connected timers, which can be a software trigger or hardware trigger from a subscriber port, zero, load, or compare event.
   a. For a software event trigger, set the TIMx.CTTRIG.TRIG bit.
   b. For a hardware trigger event, select the source for the trigger using TIMx.CTTRIGCTL.EVTCTTRIGSEL and enable the hardware trigger by setting TIMx.CTTRIGCTL.EVTCTEN.

Figure 15-39 shows the connection logic and registers.

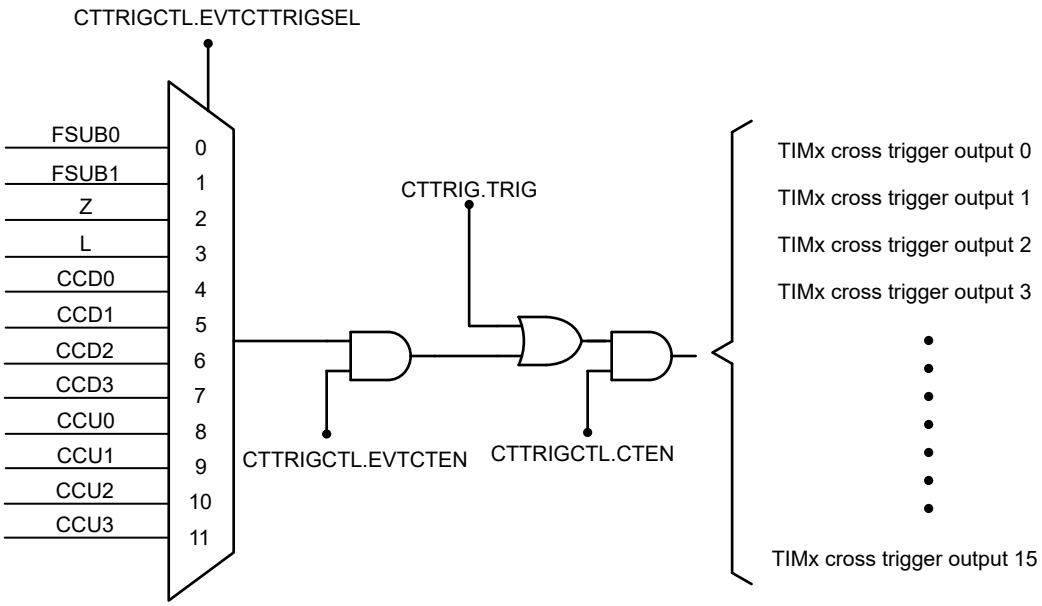


Figure 15-39. Main Timer Cross Trigger Output Configuration

15.2.7.2 Secondary Timer Cross Trigger Configuration

The following steps are used to configure the secondary timer cross trigger (which is TIMAx in this example):

1. Configure the secondary timer (triggered by the main timer) for the desired function for this timer, such as PWM output generation or using compare mode, to trigger other peripherals. See Section 15.2.5 for how to configure for PWM generation.
2. Select which input trigger to use according to the device-specific data sheet. Using the example connection in Figure 15-38, TIMAx must be triggered by TIMGx and the cross trigger output 1 of TIMGx is connected to input trigger 0 of TIMAx. Therefore, select input trigger 0 of TIMAx by setting TIMA.TSEL.ETSEL bit to 0.
3. Enable the input trigger function by setting the TIMA.TSEL.TE bit to 1.
4. Set TIMAx.IFCTL_01[0].ISEL = 3 and TIMAx.IFCTL_01[1].ISEL = 3 to select the trigger as the input source.
   a. For a center-aligned PWM, set the TIMA.CCCTL_01[0].ZCOND and TIMA.CCCTL_01[1].ZCOND bits to 1 to use a trigger assertion edge for a zero event.
   b. For an edge-aligned PWM, set the TIMA.CCCTL_01[0].LCOND and TIMA.CCCTL_01[1].LCOND bits to 1 to use the trigger assertion edge for a load event.
5. The TIMx.CTRCTL.EN bit is set as the result of an LCOND or ZCOND condition being met, and the counter value changes to the load value or zero value, respectively.

As the main timer TIMGx must also trigger itself, complete the previous configuration steps for TIMAx to trigger TIMGx itself.

Figure 15-40 shows the logic connection.

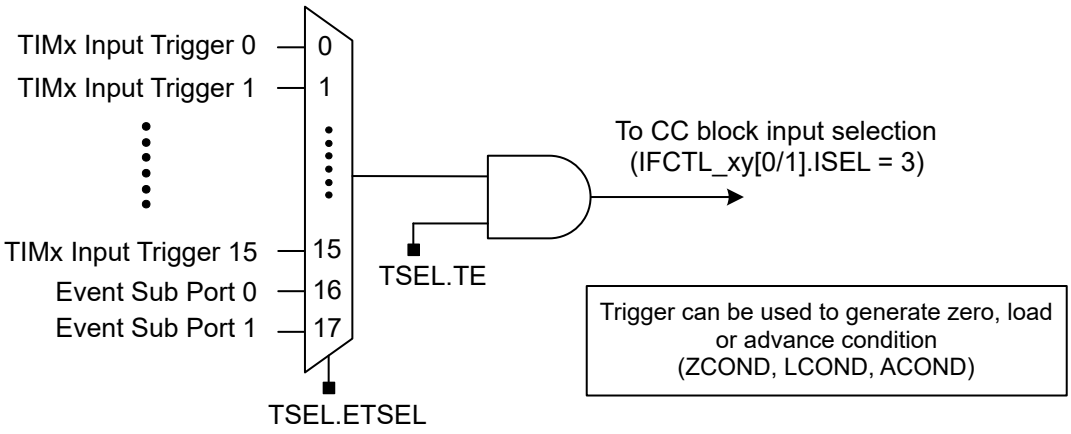


Figure 15-40. Secondary Timer Cross Trigger Input Configuration

Note

Refer to "TIMx Cross Trigger Map" in the device-specific data sheet for enabled cross trigger mapping using the ETSEL bit. For instance, if the timer instances of a device all use trigger input 0 (TRIG0) to cross trigger other timers, then only TRIG0 can be used to cross trigger other instances.

15.2.8 Low Power Operation

For detailed information on the low power modes in terms of available clock source and behaviors, refer to Section 2.1.1.

Timer modules in power domain PD0 can be active and configured to continue counting in all power modes except SHUTDOWN mode. See Section 2.1.1 for the available clock sources in each low-power mode. The user needs to configure the proper clock to source the timer in low-power mode.

Timer modules in power domain PD1 can only be active in RUN and SLEEP modes. When the system goes to STOP or STANDBY mode, the timer modules will be forced to a disabled state and resume when the systems moves back to RUN or SLEEP modes.

15.2.9 Interrupt and Event Support

TIMx interrupts and events can be configured to any peripheral of the device using the Event Manager. The timer can generate interrupts or events as an **event publisher**, and be triggered from other peripheral events (such as

GPIO, comparator, ADC, etc.) as an **event subscriber**. See Section 6.2.5 for guidance on configuring the event registers for CPU interrupts or generic events.

The TIMx module contains three event publishers and two event subscribers.

- One event publisher (CPU_INT) manages TIMx interrupt requests (IRQs) to the CPU subsystem through a static event route.
- The second and third event (GEN_EVENT0 and GEN_EVENT1) are used to setup the generic event publishers and subscribers through Generic route.

TIMx events are summarized in Table 15-23.

Table 15-23. TIMx Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU interrupt	Publisher	TIMx	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from TIMx to CPU
Generic publisher event	Publisher	TIMx	Other peripherals	Generic route	GEN_EVENT0 and FPUB_0 registers	Configurable interrupt route from TIMx to other peripherals
Generic publisher event	Publisher	TIMx	Other peripherals	Generic route	GEN_EVENT 1 and FPUB_1 registers	Configurable interrupt route from TIMx to other peripherals
Generic subscriber event	Subscriber	Other peripherals	TIMx	Generic route	FSUB_0	Configurable interrupt route from other peripherals to TIMx
Generic subscriber event	Subscriber	Other peripherals	TIMx	Generic route	FSUB_1	Configurable interrupt route from other peripherals to TIMx

15.2.9.1 CPU Interrupt Event Publisher (CPU_INT)

The TIMx module provides 18 interrupt sources (depending on the specific TIMx module features) which can be configured to source a CPU interrupt event. The CPU interrupt event configuration is managed with the CPU_INT event management registers. Table 15-24 lists the CPU interrupt events from the TIMx in order of decreasing interrupt priority.

Table 15-24. TIMx CPU Interrupt Event Conditions (CPU_INT)

IIDX STAT	Name	Description	Timer Module
0x01	Z	Zero event interrupt. This interrupt is set when there is a zero event.	TIMx
0x02	L	Load event interrupt. This interrupt is set when there is a load event.	TIMx
0x05	CCD0	Capture or compare 0 down event. This interrupt is set when there is a down compare match event at CC0.	TIMx
0x06	CCD1	Capture or compare 1 down event. This interrupt is set when there is a down compare match event at CC1.	TIMx
0x07	CCD2	Capture or compare 2 down event. This interrupt is set when there is a down compare match event at CC2. This interrupt is only available for TIMA0.	TIMx
0x08	CCD3	Capture or compare 3 down event. This interrupt is set when there is a down compare match event at CC3. This interrupt is only available for TIMA0.	TIMx
0x09	CCU0	Capture or compare 0 up event. This interrupt is set when there is a up compare match event at CC0.	TIMx
0x0A	CCU1	Capture or compare 1 up event. This interrupt is set when there is a up compare match event at CC1.	TIMx
0x0B	CCU2	Capture or compare 2 up event. This interrupt is set when there is a up compare match event at CC2.	TIMx
0x0C	CCU3	Capture or compare 3 up event. This interrupt is set when there is a up compare match event at CC3.	TIMx
0x0D	CCD4	Capture or compare 4 down event. This interrupt is set when there is a down compare match event at CC4. This interrupt is only available for TIMA modules.	TIMA

Table 15-24. TIMx CPU Interrupt Event Conditions (CPU_INT) (continued)

IIDX STAT	Name	Description	Timer Module
0x0E	CCD5	Capture or compare 5 down event. This interrupt is set when there is a down compare match event at CC5. This interrupt is only available for TIMA modules.	TIMA
0x0F	CCU4	Capture or compare 4 up event. This interrupt is set when there is a up compare match event at CC4. This interrupt is only available for TIMA modules.	TIMA
0x10	CCU5	Capture or compare 5 up event. This interrupt is set when there is a up compare match event at CC5. This interrupt is only available for TIMA modules.	TIMA
0x19	F	Fault event interrupt. This interrupt is set when there is a fault condition event. See Section 15.2.6. This interrupt is only available for TIMA modules with fault handler features.	TIMA
0x1A	TOV	Trigger overflow interrupt. This interrupt is set if a trigger event is generated while the associated trigger channel is active.	TIMx
0x1B	REPC	Repeat counter zero interrupt. This bit controls the generation of an interrupt if the repeat counter value transitions from a nonzero value to zero. This interrupt is only available for TIMA modules with a repeat counter feature.	TIMA
0x1C	DC	Direction change interrupt, used in QEI mode. This interrupt is only available for TIMG modules with QEI features.	TIMG
0X1D	QEIERR	Direction change interrupt, used in QEI mode. This interrupt is only available for TIMG modules with QEI features.	TIMG

See Section 6.2.5 for guidance on configuring the event registers for CPU interrupts.

15.2.9.2 Generic Event Publisher and Subscriber (GEN_EVENT0 and GEN_EVENT1)

A generic route is a route in which the comparator peripheral publishing the event is configured to use one of several available generic route channels to publish its event to another entity (or entities, in the case of a splitter route), where an entity can be another peripheral, a generic DMA trigger event, or a generic CPU event.

The GEN_EVENT0 and GEN_EVENT1 registers are used to select a peripheral condition (Table 15-25) to use for publishing or subscribing an event. FPUB_0 and FPUB_1 are the publisher port registers and are used to configure which generic route channel to use to broadcast the event. FSUB_0 and FSUB_1 are the subscriber port registers and are used to configure which generic route channel to use to subscribe the event. Other peripherals, the DMA, or the CPU can subscribe to this event by configuring its subscriber port to listen on the same generic route channel which the publishing peripheral is connected to.

For example, through the use of a generic event channel, it is possible to directly start an ADC conversion from a TIMx event by connecting a TIMx FPUB_x and ADC FSUB_0 to the same generic event channel. Refer to Section 6.1.3.3 and Section 6.2.3 for how generic event route works.

Table 15-25. TIMx Generic Event Conditions (GEN_EVENT0 and GEN_EVENT1)

IIDX STAT	Name	Description	Timer Module
0x01	Z	Zero event interrupt. This interrupt is set when there is a zero event.	TIMx
0x02	L	Load event interrupt. This interrupt is set when there is a load event.	TIMx
0x05	CCD0	Capture or compare 0 down event. This interrupt is set when there is a down compare match event at CC0.	TIMx
0x06	CCD1	Capture or compare 1 down event. This interrupt is set when there is a down compare match event at CC1.	TIMx
0x07	CCD2	Capture or compare 2 down event. This interrupt is set when there is a down compare match event at CC2. This interrupt is only available for TIMA0.	TIMx
0x08	CCD3	Capture or compare 3 down event. This interrupt is set when there is a down compare match event at CC3. This interrupt is only available for TIMA0.	TIMx
0x09	CCU0	Capture or compare 0 up event. This interrupt is set when there is a up compare match event at CC0.	TIMx
0x0A	CCU1	Capture or compare 1 up event. This interrupt is set when there is a up compare match event at CC1.	TIMx
0x0B	CCU2	Capture or compare 2 up event. This interrupt is set when there is a up compare match event at CC2.	TIMx

Table 15-25. TIMx Generic Event Conditions (GEN_EVENT0 and GEN_EVENT1) (continued)

IIDX STAT	Name	Description	Timer Module
0x0C	CCU3	Capture or compare 3 up event. This interrupt is set when there is a up compare match event at CC3.	TIMx
0x0D	CCD4	Capture or compare 4 down event. This interrupt is set when there is a down compare match event at CC4. This interrupt is only available for TIMA modules.	TIMA
0x0E	CCD5	Capture or compare 5 down event. This interrupt is set when there is a down compare match event at CC5. This interrupt is only available for TIMA modules.	TIMA
0x0F	CCU4	Capture or compare 4 up event. This interrupt is set when there is a up compare match event at CC4. This interrupt is only available for TIMA modules.	TIMA
0x10	CCU5	Capture or compare 5 up event. This interrupt is set when there is a up compare match event at CC5. This interrupt is only available for TIMA modules.	TIMA
0x19	F	Fault event interrupt. This interrupt is set when there is a fault condition event. See Section 15.2.6. This interrupt is only available for TIMA modules with fault handler features.	TIMA
0x1A	TOV	Trigger overflow interrupt. This interrupt is set if a trigger event is generated while the associated trigger channel is active.	TIMx
0x1B	REPC	Repeat counter zero interrupt. This bit controls the generation of an interrupt if the repeat counter value transitions from a nonzero value to zero. This interrupt is only available for TIMA modules with a repeat counter feature.	TIMA
0x1C	DC	Direction change interrupt, used in QEI mode. This interrupt is only available for TIMG modules with QEI features.	TIMG
0x1D	QEIERR	Direction change interrupt, used in QEI mode. This interrupt is only available for TIMG modules with QEI features.	TIMG

See Section 6.2.5 for guidance on configuring the event registers.

15.2.9.3 Generic Subscriber Event Example (COMP to TIMx)

A common use case is to directly trigger a timer action (reset, PWM output, etc.) from the comparator output. The Event Manager can be used to set up the comparator as a generic event publisher (FPUB_1) and the timer as a generic event subscriber (FSUB_0 or FSUB_1) listening for an event from the publisher.

Comparator configuration (Publisher):

1. Set COMPIFG bit in GEN_EVENT0 IMASK register to mask the comparator output interrupt. For more info, see the COMP chapter.
2. Configure FPUB_1 register in comparator module to connect to event channel y.
3. Configure and enable the comparator.

TIMx configuration (Subscriber):

1. Configure FSUB_0 or FSUB_1 register in the TIMx module to connect to event channel y. Channel y must not be in use by another peripheral.
2. Set TIMx.IFCTL_xy[0/1].ISEL = 5 or 6 to select the CC input source as FSUB0 or FSUB1 for the TIMx module. See Figure 15-12

   and Section 15.2.3.1.1.5.
3. Configure and enable the timer.

Note

The comparator output (COMP0:2) has two more options for cross-peripherals:

- Input to the TIMx CC block directly using TIMx.IFCTL_xy[0/1].ISEL = 7h, 8h, or 9h.
  - This is a lower latency path and is useful for applications such as cycle-by-cycle overcurrent limiting. See Figure 15-12.
- Using the timer's cross trigger path to configure the event subscriber port as a trigger source by configuring the TIMx.TSEL.ETSEL = 0x10 for FSUB0 or to 0x11 for FSUB1.
  - Enable the input trigger function by setting the TIMx.TSEL.TE bit to 1.
  - Set TIMx.IFCTL_xy[0/1].ISEL bit to 3 to select the cross trigger as input source for the TIMx module.
  - This is useful for using an event subscriber to trigger multiple timer instances in the same power domain, such as COMP to multiple TIMG instances.

15.2.10 Debug Handler (TIMA Only)

In TIMA only, the counter behavior in CPU halt debug mode and debug resume can also be configured by software using the PDBGCTL and CTRCTL registers.

TIMA can be configured to stop counting or continue counting when the CPU is halted for debug by the debug subsystem. By default, TIMA stops counting when the CPU is halted for debug and the device is in a debug state. To allow TIMA to continue to free run when the CPU is stopped for debug, set the FREE bit in the PDBGCTL register. See Section 17.2.2 for more information.

The CTRCTL register lets the device to resume counting or perform the action specified by the CVAE field following the exit of debug mode using the DRB bit.

15.3 TIMx Registers

Table 15-26 lists the memory-mapped registers for the TIMx registers. All register offset addresses not listed in Table 15-26 should be considered as reserved locations and the register contents should not be modified.

Table 15-26. TIMX Registers

Offset	Acronym	Register Name	Group	Section
400h	FSUB_0	Subsciber Port 0		Go
404h	FSUB_1	Subscriber Port 1		Go
444h	FPUB_0	Publisher Port 0		Go
448h	FPUB_1	Publisher Port 1		Go
800h	PWREN	Power enable		Go
804h	RSTCTL	Reset Control		Go
814h	STAT	Status Register		Go
1000h	CLKDIV	Clock Divider		Go
1008h	CLKSEL	Clock Select for Ultra Low Power peripherals		Go
1018h	PDBGCTL	Peripheral Debug Control		Go
1020h	IIDX	Interrupt index	CPU_INT	Go
1028h	IMASK	Interrupt mask	CPU_INT	Go
1030h	RIS	Raw interrupt status	CPU_INT	Go
1038h	MIS	Masked interrupt status	CPU_INT	Go
1040h	ISET	Interrupt set	CPU_INT	Go
1048h	ICLR	Interrupt clear	CPU_INT	Go
1050h	IIDX	Interrupt index	GEN_EVENT0	Go
1058h	IMASK	Interrupt mask	GEN_EVENT0	Go
1060h	RIS	Raw interrupt status	GEN_EVENT0	Go
1068h	MIS	Masked interrupt status	GEN_EVENT0	Go
1070h	ISET	Interrupt set	GEN_EVENT0	Go
1078h	ICLR	Interrupt clear	GEN_EVENT0	Go
1080h	IIDX	Interrupt index	GEN_EVENT1	Go
1088h	IMASK	Interrupt mask	GEN_EVENT1	Go
1090h	RIS	Raw interrupt status	GEN_EVENT1	Go
1098h	MIS	Masked interrupt status	GEN_EVENT1	Go
10A0h	ISET	Interrupt set	GEN_EVENT1	Go
10A8h	ICLR	Interrupt clear	GEN_EVENT1	Go
10E0h	EVT_MODE	Event Mode		Go
10FCh	DESC	Module Description		Go
1100h	CCPD	CCP Direction		Go
1104h	ODIS	Output Disable		Go
1108h	CCLKCTL	Counter Clock Control Register		Go
110Ch	CPS	Clock Prescale Register		Go
1110h	CPSV	Clock prescale count status register		Go
1114h	CTTRIGCTL	Timer Cross Trigger Control Register		Go
111Ch	CTTRIG	Timer Cross Trigger Register		Go
1120h	FSCTL	Fault Source Control		Go
1124h	GCTL	Global control register		Go
1800h	CTR	Counter Register		Go
1804h	CTRCTL	Counter Control Register		Go

Table 15-26. TIMX Registers (continued)

Offset	Acronym	Register Name	Group	Section
1808h	LOAD	Load Register		Go
1810h + formula	CC_01[y]	Capture or Compare Register 0/1		Go
1818h + formula	CC_23[y]	Capture or Compare Register 2/3		Go
1820h + formula	CC_45[y]	The CC_45 register are a registers which can be used as compare to the current CTR to create an events CC4U, CC4D, CC5U and CC5D.		Go
1830h + formula	CCCTL_01[y]	Capture or Compare Control Registers 0/1		Go
1838h + formula	CCCTL_23[y]	Capture or Compare Control Registers 2/3		Go
1840h + formula	CCCTL_45[y]	Capture or Compare Control Registers 4/5		Go
1850h + formula	OCTL_01[y]	CCP Output Control Registers 0/1		Go
1858h + formula	OCTL_23[y]	CCP Output Control Registers 2/3		Go
1870h + formula	CCACT_01[y]	Capture or Compare Action Registers 0/1		Go
1878h + formula	CCACT_23[y]	Capture or Compare Action Registers 2/3		Go
1880h + formula	IFCTL_01[y]	Input Filter Control Register 0/1		Go
1888h + formula	IFCTL_23[y]	Input Filter Control Register 2/3		Go
18A0h	PL	Phase Load Register		Go
18A4h	DBCTL	Dead Band insertion control register		Go
18B0h	TSEL	Trigger Select Register		Go
18B4h	RC	Repeat counter Register		Go
18B8h	RCLD	Repeat counter load Register		Go
18BCh	QDIR	QEI Count Direction Register		Go
18D0h	FCTL	Fault Control Register		Go
18D4h	FIFCTL	Fault input Filter control register		Go

Complex bit access types are encoded to fit into small table cells. Table 15-27 shows the codes that are used for access types in this section.

Table 15-27. TIMx Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 15-27. TIMx Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.3.1 FSUB_0 (Offset = 400h) [Reset = 00000000h]

FSUB_0 is shown in Figure 15-41 and described in Table 15-28.

Return to the Summary Table.

Subscriber port

Figure 15-41. FSUB_0

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

Table 15-28. FSUB_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

15.3.2 FSUB_1 (Offset = 404h) [Reset = 00000000h]

FSUB_1 is shown in Figure 15-42 and described in Table 15-29.

Return to the Summary Table.

Subscriber port

Figure 15-42. FSUB_1

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

Table 15-29. FSUB_1 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

15.3.3 FPUB_0 (Offset = 444h) [Reset = 00000000h]

FPUB_0 is shown in Figure 15-43 and described in Table 15-30.

Return to the Summary Table.

Publisher port

Figure 15-43. FPUB_0

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

Table 15-30. FPUB_0 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

15.3.4 FPUB_1 (Offset = 448h) [Reset = 00000000h]

FPUB_1 is shown in Figure 15-44 and described in Table 15-31.

Return to the Summary Table.

Publisher port

Figure 15-44. FPUB_1

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

Table 15-31. FPUB_1 Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3-0	CHANID	R/W	0h	0 = disconnected. 1-15 = connected to channelID = CHANID. 0h = A value of 0 specifies that the event is not connected Fh = Consult your device data sheet as the actual allowed maximum may be less than 15.

15.3.5 PWREN (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 15-45 and described in Table 15-32.

Return to the Summary Table.

Register to control the power state

Figure 15-45. PWREN

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/WK-0h

Table 15-32. PWREN Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R	0h	
0	ENABLE	R/WK	0h	Enable the power KEY must be set to 26h to write to this bit. 0h = Disable Power 1h = Enable Power

15.3.6 RSTCTL (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 15-46 and described in Table 15-33.

Return to the Summary Table.

Register to control reset assertion and de-assertion

Figure 15-46. RSTCTL

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
R-0h						WK-0h	WK-0h

Table 15-33. RSTCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	R	0h	
1	RESETSTKYCLR	WK	0h	Clear the RESETSTKY bit in the STAT register KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Clear reset sticky bit
0	RESETASSERT	WK	0h	Assert reset to the peripheral KEY must be set to B1h to write to this bit. 0h = Writing 0 has no effect 1h = Assert reset

15.3.7 STAT (Offset = 814h) [Reset = 00000000h]

STAT is shown in Figure 15-47 and described in Table 15-34.

Return to the Summary Table.

peripheral enable and reset status

Figure 15-47. STAT

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

Table 15-34. STAT Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	0h	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

15.3.8 CLKDIV (Offset = 1000h) [Reset = 00000000h]

CLKDIV is shown in Figure 15-48 and described in Table 15-35.

Return to the Summary Table.

This register is used to specify module-specific divide ratio of the functional clock

Figure 15-48. CLKDIV

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

Table 15-35. CLKDIV Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	
2-0	RATIO	R/W	0h	Selects divide ratio of module clock 0h = Do not divide clock source 1h = Divide clock source by 2 2h = Divide clock source by 3 3h = Divide clock source by 4 4h = Divide clock source by 5 5h = Divide clock source by 6 6h = Divide clock source by 7 7h = Divide clock source by 8

15.3.9 CLKSEL (Offset = 1008h) [Reset = 00000000h]

CLKSEL is shown in Figure 15-49 and described in Table 15-36.

Return to the Summary Table.

Clock Source Select Register

Figure 15-49. CLKSEL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	BUSCLK_SEL	MFCLK_SEL	LFCLK_SEL	RESERVED
R-0h			R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 15-36. CLKSEL Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4	RESERVED	R	0h	Reserved
3	BUSCLK_SEL	R/W	0h	Selects BUSCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
2	MFCLK_SEL	R/W	0h	Selects MFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
1	LFCLK_SEL	R/W	0h	Selects LFCLK as clock source if enabled 0h = Does not select this clock as a source 1h = Select this clock as a source
0	RESERVED	R	0h	

15.3.10 PDBGCTL (Offset = 1018h) [Reset = 00000000h]

PDBGCTL is shown in Figure 15-50 and described in Table 15-37.

Return to the Summary Table.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 15-50. PDBGCTL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						SOFT	FREE
R-0h						R/W-0h	R/W-0h

Table 15-37. PDBGCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	SOFT	R/W	0h	Soft halt boundary control. This function is only available, if FREE is set to 'STOP' 0h = The peripheral will halt immediately, even if the resultant state will result in corruption if the system is restarted 1h = The peripheral blocks the debug freeze until it has reached a boundary where it can resume without corruption
0	FREE	R/W	0h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

15.3.11 IIDX (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 15-51 and described in Table 15-38.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 15-51. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 15-38. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No interrupt pending 01h = Interrupt Source: Zero event (Z) 02h = Interrupt Source: Load event (L) 05h = Interrupt Source: Capture or compare down event (CCD0) 06h = Interrupt Source: Capture or compare down event (CCD1) 07h = Interrupt Source: Capture or compare down event (CCD2) 08h = Interrupt Source: Capture or compare down event (CCD3) 09h = Interrupt Source: Capture or compare up event (CCU0) 0Ah = Interrupt Source: Capture or compare up event (CCU1) 0Bh = Interrupt Source: Capture or compare up event (CCU2) 0Ch = Interrupt Source: Capture or compare up event (CCU3) 0Dh = Interrupt Source: Compare down event (CCD4) 0Eh = Interrupt Source: Compare down event (CCD5) 0Fh = Interrupt Source: Compare down event (CCU4) 10h = Interrupt Source: Compare down event (CCU5) 19h = Interrupt Source: Fault Event generated an interrupt. (F) 1Ah = Interrupt Source: Trigger overflow (TOV) 1Bh = Interrupt Source: Repeat Counter Zero (REPC) 1Ch = Interrupt Source: Direction Change (DC) 1Dh = Interrupt Source:QEI Incorrect state transition error (QEIERR)

15.3.12 IMASK (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 15-52 and described in Table 15-39.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS."

Figure 15-52. IMASK

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 15-39. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R/W	0h	QEIERR Event mask 0h = Disable Event 1h = Enable Event
27	DC	R/W	0h	Direction Change Event mask 0h = Disable Event 1h = Enable Event
26	REPC	R/W	0h	Repeat Counter Zero Event mask 0h = Disable Event 1h = Enable Event
25	TOV	R/W	0h	Trigger Overflow Event mask 0h = Disable Event 1h = Enable Event
24	F	R/W	0h	Fault Event mask 0h = Disable Event 1h = Enable Event
23-16	RESERVED	R	0h	
15	CCU5	R/W	0h	Compare UP event mask CCP5 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
14	CCU4	R/W	0h	Compare UP event mask CCP4 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
13	CCD5	R/W	0h	Compare DN event mask CCP5 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
12	CCD4	R/W	0h	Compare DN event mask CCP4 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 15-39. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	R/W	0h	Capture or Compare UP event mask CCP3 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
10	CCU2	R/W	0h	Capture or Compare UP event mask CCP2 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
9	CCU1	R/W	0h	Capture or Compare UP event mask CCP1 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
8	CCU0	R/W	0h	Capture or Compare UP event mask CCP0 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
7	CCD3	R/W	0h	Capture or Compare DN event mask CCP3 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
6	CCD2	R/W	0h	Capture or Compare DN event mask CCP2 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
5	CCD1	R/W	0h	Capture or Compare DN event mask CCP1 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
4	CCD0	R/W	0h	Capture or Compare DN event mask CCP0 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3-2	RESERVED	R	0h	
1	L	R/W	0h	Load Event mask 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	Z	R/W	0h	Zero Event mask 0h = Disable Event 1h = Enable Event

15.3.13 RIS (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 15-53 and described in Table 15-40.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 15-53. RIS

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 15-40. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R	0h	QEIERR, set on an incorrect state transition on the encoder interface. 0h = Event Cleared 1h = Event Set
27	DC	R	0h	Direction Change 0h = Event Cleared 1h = Event Set
26	REPC	R	0h	Repeat Counter Zero 0h = Event Cleared 1h = Event Set
25	TOV	R	0h	Trigger overflow 0h = Event Cleared 1h = Event Set
24	F	R	0h	Fault 0h = Event Cleared 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	R	0h	Compare up event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
14	CCU4	R	0h	Compare up event generated an interrupt CCU4 0h = Event Cleared 1h = Event Set
13	CCD5	R	0h	Compare down event generated an interrupt CCD5 0h = Event Cleared 1h = Event Set

Table 15-40. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	CCD4	R	0h	Compare down event generated an interrupt CCD4 0h = Event Cleared 1h = Event Set
11	CCU3	R	0h	Capture or compare up event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
10	CCU2	R	0h	Capture or compare up event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
9	CCU1	R	0h	Capture or compare up event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
8	CCU0	R	0h	Capture or compare up event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
7	CCD3	R	0h	Capture or compare down event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
6	CCD2	R	0h	Capture or compare down event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
5	CCD1	R	0h	Capture or compare down event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
4	CCD0	R	0h	Capture or compare down event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
3-2	RESERVED	R	0h	
1	L	R	0h	Load event generated an interrupt. 0h = Event Cleared 1h = Event Set
0	Z	R	0h	Zero event generated an interrupt. 0h = Event Cleared 1h = Event Set

15.3.14 MIS (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 15-54 and described in Table 15-41.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 15-54. MIS

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 15-41. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R	0h	QEIERR 0h = Event Cleared 1h = Event Set
27	DC	R	0h	Direction Change 0h = Event Cleared 1h = Event Set
26	REPC	R	0h	Repeat Counter Zero 0h = Event Cleared 1h = Event Set
25	TOV	R	0h	Trigger overflow 0h = Event Cleared 1h = Event Set
24	F	R	0h	Fault 0h = Event Cleared 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	R	0h	Compare up event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
14	CCU4	R	0h	Compare up event generated an interrupt CCP4 0h = Event Cleared 1h = Event Set
13	CCD5	R	0h	Compare down event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
12	CCD4	R	0h	Compare down event generated an interrupt CCP4 0h = Event Cleared 1h = Event Set

Table 15-41. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	R	0h	Capture or compare up event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
10	CCU2	R	0h	Capture or compare up event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
9	CCU1	R	0h	Capture or compare up event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
8	CCU0	R	0h	Capture or compare up event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
7	CCD3	R	0h	Capture or compare down event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
6	CCD2	R	0h	Capture or compare down event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
5	CCD1	R	0h	Capture or compare down event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
4	CCD0	R	0h	Capture or compare down event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
3-2	RESERVED	R	0h	
1	L	R	0h	Load event generated an interrupt. 0h = Event Cleared 1h = Event Set
0	Z	R	0h	Zero event generated an interrupt. 0h = Event Cleared 1h = Event Set

15.3.15 ISET (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 15-55 and described in Table 15-42.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 15-55. ISET

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
W-0h	W-0h	W-0h	W-0h	R-0h		W-0h	W-0h

Table 15-42. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	W	0h	QEIERR event SET 0h = Writing 0 has no effect. 1h = Event Set
27	DC	W	0h	Direction Change event SET 0h = Writing 0 has no effect. 1h = Event Set
26	REPC	W	0h	Repeat Counter Zero event SET 0h = Writing 0 has no effect. 1h = Event Set
25	TOV	W	0h	Trigger Overflow event SET 0h = Writing 0 has no effect. 1h = Event Set
24	F	W	0h	Fault event SET 0h = Writing 0 has no effect. 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	W	0h	Compare up event 5 SET 0h = Writing 0 has no effect. 1h = Event Set
14	CCU4	W	0h	Compare up event 4 SET 0h = Writing 0 has no effect. 1h = Event Set
13	CCD5	W	0h	Compare down event 5 SET 0h = Writing 0 has no effect. 1h = Event Set
12	CCD4	W	0h	Compare down event 4 SET 0h = Writing 0 has no effect. 1h = Event Set

Table 15-42. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
10	CCU2	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
9	CCU1	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
8	CCU0	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
7	CCD3	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
6	CCD2	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
5	CCD1	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
4	CCD0	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
3-2	RESERVED	R	0h	
1	L	W	0h	Load event SET 0h = Writing 0 has no effect. 1h = Event Set
0	Z	W	0h	Zero event SET 0h = Writing 0 has no effect. 1h = Event Set

15.3.16 ICLR (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 15-56 and described in Table 15-43.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 15-56. ICLR

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
W-0h	W-0h	W-0h	W-0h	R-0h		W-0h	W-0h

Table 15-43. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	W	0h	QEIERR event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
27	DC	W	0h	Direction Change event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
26	REPC	W	0h	Repeat Counter Zero event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
25	TOV	W	0h	Trigger Overflow event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
24	F	W	0h	Fault event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
23-16	RESERVED	R	0h	
15	CCU5	W	0h	Compare up event 5 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
14	CCU4	W	0h	Compare up event 4 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
13	CCD5	W	0h	Compare down event 5 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
12	CCD4	W	0h	Compare down event 4 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear

Table 15-43. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
10	CCU2	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
9	CCU1	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
8	CCU0	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
7	CCD3	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
6	CCD2	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
5	CCD1	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
4	CCD0	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
3-2	RESERVED	R	0h	
1	L	W	0h	Load event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
0	Z	W	0h	Zero event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear

15.3.17 IIDX (Offset = 1050h) [Reset = 00000000h]

IIDX is shown in Figure 15-57 and described in Table 15-44.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 15-57. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 15-44. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No interrupt pending 01h = Interrupt Source: Zero event (Z) 02h = Interrupt Source: Load event (L) 05h = Interrupt Source: Capture or compare down event (CCD0) 06h = Interrupt Source: Capture or compare down event (CCD1) 07h = Interrupt Source: Capture or compare down event (CCD2) 08h = Interrupt Source: Capture or compare down event (CCD3) 09h = Interrupt Source: Capture or compare up event (CCU0) 0Ah = Interrupt Source: Capture or compare up event (CCU1) 0Bh = Interrupt Source: Capture or compare up event (CCU2) 0Ch = Interrupt Source: Capture or compare up event (CCU3) 0Dh = Interrupt Source: Compare down event (CCD4) 0Eh = Interrupt Source: Compare down event (CCD5) 0Fh = Interrupt Source: Compare down event (CCU4) 10h = Interrupt Source: Compare down event (CCU5) 19h = Interrupt Source: Fault Event generated an interrupt. (F) 1Ah = Interrupt Source: Trigger overflow (TOV) 1Bh = Interrupt Source: Repeat Counter Zero (REPC) 1Ch = Interrupt Source: Direction Change (DC) 1Dh = Interrupt Source:QEI Incorrect state transition error (QEIERR)

15.3.18 IMASK (Offset = 1058h) [Reset = 00000000h]

IMASK is shown in Figure 15-58 and described in Table 15-45.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS."

Figure 15-58. IMASK

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 15-45. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R/W	0h	QEIERR Event mask 0h = Disable Event 1h = Enable Event
27	DC	R/W	0h	Direction Change Event mask 0h = Disable Event 1h = Enable Event
26	REPC	R/W	0h	Repeat Counter Zero Event mask 0h = Disable Event 1h = Enable Event
25	TOV	R/W	0h	Trigger Overflow Event mask 0h = Disable Event 1h = Enable Event
24	F	R/W	0h	Fault Event mask 0h = Disable Event 1h = Enable Event
23-16	RESERVED	R	0h	
15	CCU5	R/W	0h	Compare UP event mask CCP5 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
14	CCU4	R/W	0h	Compare UP event mask CCP4 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
13	CCD5	R/W	0h	Compare DN event mask CCP5 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
12	CCD4	R/W	0h	Compare DN event mask CCP4 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 15-45. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	R/W	0h	Capture or Compare UP event mask CCP3 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
10	CCU2	R/W	0h	Capture or Compare UP event mask CCP2 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
9	CCU1	R/W	0h	Capture or Compare UP event mask CCP1 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
8	CCU0	R/W	0h	Capture or Compare UP event mask CCP0 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
7	CCD3	R/W	0h	Capture or Compare DN event mask CCP3 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
6	CCD2	R/W	0h	Capture or Compare DN event mask CCP2 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
5	CCD1	R/W	0h	Capture or Compare DN event mask CCP1 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
4	CCD0	R/W	0h	Capture or Compare DN event mask CCP0 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3-2	RESERVED	R	0h	
1	L	R/W	0h	Load Event mask 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	Z	R/W	0h	Zero Event mask 0h = Disable Event 1h = Enable Event

15.3.19 RIS (Offset = 1060h) [Reset = 00000000h]

RIS is shown in Figure 15-59 and described in Table 15-46.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 15-59. RIS

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 15-46. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R	0h	QEIERR, set on an incorrect state transition on the encoder interface. 0h = Event Cleared 1h = Event Set
27	DC	R	0h	Direction Change 0h = Event Cleared 1h = Event Set
26	REPC	R	0h	Repeat Counter Zero 0h = Event Cleared 1h = Event Set
25	TOV	R	0h	Trigger overflow 0h = Event Cleared 1h = Event Set
24	F	R	0h	Fault 0h = Event Cleared 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	R	0h	Compare up event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
14	CCU4	R	0h	Compare up event generated an interrupt CCU4 0h = Event Cleared 1h = Event Set
13	CCD5	R	0h	Compare down event generated an interrupt CCD5 0h = Event Cleared 1h = Event Set

Table 15-46. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	CCD4	R	0h	Compare down event generated an interrupt CCD4 0h = Event Cleared 1h = Event Set
11	CCU3	R	0h	Capture or compare up event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
10	CCU2	R	0h	Capture or compare up event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
9	CCU1	R	0h	Capture or compare up event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
8	CCU0	R	0h	Capture or compare up event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
7	CCD3	R	0h	Capture or compare down event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
6	CCD2	R	0h	Capture or compare down event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
5	CCD1	R	0h	Capture or compare down event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
4	CCD0	R	0h	Capture or compare down event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
3-2	RESERVED	R	0h	
1	L	R	0h	Load event generated an interrupt. 0h = Event Cleared 1h = Event Set
0	Z	R	0h	Zero event generated an interrupt. 0h = Event Cleared 1h = Event Set

15.3.20 MIS (Offset = 1068h) [Reset = 00000000h]

MIS is shown in Figure 15-60 and described in Table 15-47.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 15-60. MIS

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 15-47. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R	0h	QEIERR 0h = Event Cleared 1h = Event Set
27	DC	R	0h	Direction Change 0h = Event Cleared 1h = Event Set
26	REPC	R	0h	Repeat Counter Zero 0h = Event Cleared 1h = Event Set
25	TOV	R	0h	Trigger overflow 0h = Event Cleared 1h = Event Set
24	F	R	0h	Fault 0h = Event Cleared 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	R	0h	Compare up event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
14	CCU4	R	0h	Compare up event generated an interrupt CCP4 0h = Event Cleared 1h = Event Set
13	CCD5	R	0h	Compare down event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
12	CCD4	R	0h	Compare down event generated an interrupt CCP4 0h = Event Cleared 1h = Event Set

Table 15-47. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	R	0h	Capture or compare up event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
10	CCU2	R	0h	Capture or compare up event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
9	CCU1	R	0h	Capture or compare up event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
8	CCU0	R	0h	Capture or compare up event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
7	CCD3	R	0h	Capture or compare down event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
6	CCD2	R	0h	Capture or compare down event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
5	CCD1	R	0h	Capture or compare down event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
4	CCD0	R	0h	Capture or compare down event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
3-2	RESERVED	R	0h	
1	L	R	0h	Load event generated an interrupt. 0h = Event Cleared 1h = Event Set
0	Z	R	0h	Zero event generated an interrupt. 0h = Event Cleared 1h = Event Set

15.3.21 ISET (Offset = 1070h) [Reset = 00000000h]

ISET is shown in Figure 15-61 and described in Table 15-48.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 15-61. ISET

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
W-0h	W-0h	W-0h	W-0h	R-0h		W-0h	W-0h

Table 15-48. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	W	0h	QEIERR event SET 0h = Writing 0 has no effect. 1h = Event Set
27	DC	W	0h	Direction Change event SET 0h = Writing 0 has no effect. 1h = Event Set
26	REPC	W	0h	Repeat Counter Zero event SET 0h = Writing 0 has no effect. 1h = Event Set
25	TOV	W	0h	Trigger Overflow event SET 0h = Writing 0 has no effect. 1h = Event Set
24	F	W	0h	Fault event SET 0h = Writing 0 has no effect. 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	W	0h	Compare up event 5 SET 0h = Writing 0 has no effect. 1h = Event Set
14	CCU4	W	0h	Compare up event 4 SET 0h = Writing 0 has no effect. 1h = Event Set
13	CCD5	W	0h	Compare down event 5 SET 0h = Writing 0 has no effect. 1h = Event Set
12	CCD4	W	0h	Compare down event 4 SET 0h = Writing 0 has no effect. 1h = Event Set

Table 15-48. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
10	CCU2	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
9	CCU1	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
8	CCU0	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
7	CCD3	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
6	CCD2	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
5	CCD1	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
4	CCD0	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
3-2	RESERVED	R	0h	
1	L	W	0h	Load event SET 0h = Writing 0 has no effect. 1h = Event Set
0	Z	W	0h	Zero event SET 0h = Writing 0 has no effect. 1h = Event Set

15.3.22 ICLR (Offset = 1078h) [Reset = 00000000h]

ICLR is shown in Figure 15-62 and described in Table 15-49.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 15-62. ICLR

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
W-0h	W-0h	W-0h	W-0h	R-0h		W-0h	W-0h

Table 15-49. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	W	0h	QEIERR event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
27	DC	W	0h	Direction Change event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
26	REPC	W	0h	Repeat Counter Zero event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
25	TOV	W	0h	Trigger Overflow event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
24	F	W	0h	Fault event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
23-16	RESERVED	R	0h	
15	CCU5	W	0h	Compare up event 5 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
14	CCU4	W	0h	Compare up event 4 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
13	CCD5	W	0h	Compare down event 5 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
12	CCD4	W	0h	Compare down event 4 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear

Table 15-49. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
10	CCU2	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
9	CCU1	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
8	CCU0	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
7	CCD3	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
6	CCD2	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
5	CCD1	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
4	CCD0	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
3-2	RESERVED	R	0h	
1	L	W	0h	Load event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
0	Z	W	0h	Zero event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear

15.3.23 IIDX (Offset = 1080h) [Reset = 00000000h]

IIDX is shown in Figure 15-63 and described in Table 15-50.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. Value 0x00 means no event pending. Interrupt 1 is the highest priority, IIDX next highest, 4, 8, … IIDX^31 is the least priority. That is, the least bit position that is set to 1 denotes the highest priority pending interrupt. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred. On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in [RIS] and [MIS] are cleared as well. After a read from the CPU (not from the debug interface), the register is updated with the next highest priority interrupt, if none are pending, then it should display 0x0.

Figure 15-63. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 15-50. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 00h = No interrupt pending 01h = Interrupt Source: Zero event (Z) 02h = Interrupt Source: Load event (L) 05h = Interrupt Source: Capture or compare down event (CCD0) 06h = Interrupt Source: Capture or compare down event (CCD1) 07h = Interrupt Source: Capture or compare down event (CCD2) 08h = Interrupt Source: Capture or compare down event (CCD3) 09h = Interrupt Source: Capture or compare up event (CCU0) 0Ah = Interrupt Source: Capture or compare up event (CCU1) 0Bh = Interrupt Source: Capture or compare up event (CCU2) 0Ch = Interrupt Source: Capture or compare up event (CCU3) 0Dh = Interrupt Source: Compare down event (CCD4) 0Eh = Interrupt Source: Compare down event (CCD5) 0Fh = Interrupt Source: Compare down event (CCU4) 10h = Interrupt Source: Compare down event (CCU5) 19h = Interrupt Source: Fault Event generated an interrupt. (F) 1Ah = Interrupt Source: Trigger overflow (TOV) 1Bh = Interrupt Source: Repeat Counter Zero (REPC) 1Ch = Interrupt Source: Direction Change (DC) 1Dh = Interrupt Source:QEI Incorrect state transition error (QEIERR)

15.3.24 IMASK (Offset = 1088h) [Reset = 00000000h]

IMASK is shown in Figure 15-64 and described in Table 15-51.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS."

Figure 15-64. IMASK

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 15-51. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R/W	0h	QEIERR Event mask 0h = Disable Event 1h = Enable Event
27	DC	R/W	0h	Direction Change Event mask 0h = Disable Event 1h = Enable Event
26	REPC	R/W	0h	Repeat Counter Zero Event mask 0h = Disable Event 1h = Enable Event
25	TOV	R/W	0h	Trigger Overflow Event mask 0h = Disable Event 1h = Enable Event
24	F	R/W	0h	Fault Event mask 0h = Disable Event 1h = Enable Event
23-16	RESERVED	R	0h	
15	CCU5	R/W	0h	Compare UP event mask CCP5 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
14	CCU4	R/W	0h	Compare UP event mask CCP4 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
13	CCD5	R/W	0h	Compare DN event mask CCP5 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
12	CCD4	R/W	0h	Compare DN event mask CCP4 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

Table 15-51. IMASK Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	R/W	0h	Capture or Compare UP event mask CCP3 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
10	CCU2	R/W	0h	Capture or Compare UP event mask CCP2 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
9	CCU1	R/W	0h	Capture or Compare UP event mask CCP1 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
8	CCU0	R/W	0h	Capture or Compare UP event mask CCP0 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
7	CCD3	R/W	0h	Capture or Compare DN event mask CCP3 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
6	CCD2	R/W	0h	Capture or Compare DN event mask CCP2 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
5	CCD1	R/W	0h	Capture or Compare DN event mask CCP1 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
4	CCD0	R/W	0h	Capture or Compare DN event mask CCP0 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
3-2	RESERVED	R	0h	
1	L	R/W	0h	Load Event mask 0h = Clear Interrupt Mask 1h = Set Interrupt Mask
0	Z	R/W	0h	Zero Event mask 0h = Disable Event 1h = Enable Event

15.3.25 RIS (Offset = 1090h) [Reset = 00000000h]

RIS is shown in Figure 15-65 and described in Table 15-52.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 15-65. RIS

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 15-52. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R	0h	QEIERR, set on an incorrect state transition on the encoder interface. 0h = Event Cleared 1h = Event Set
27	DC	R	0h	Direction Change 0h = Event Cleared 1h = Event Set
26	REPC	R	0h	Repeat Counter Zero 0h = Event Cleared 1h = Event Set
25	TOV	R	0h	Trigger overflow 0h = Event Cleared 1h = Event Set
24	F	R	0h	Fault 0h = Event Cleared 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	R	0h	Compare up event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
14	CCU4	R	0h	Compare up event generated an interrupt CCU4 0h = Event Cleared 1h = Event Set
13	CCD5	R	0h	Compare down event generated an interrupt CCD5 0h = Event Cleared 1h = Event Set

Table 15-52. RIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	CCD4	R	0h	Compare down event generated an interrupt CCD4 0h = Event Cleared 1h = Event Set
11	CCU3	R	0h	Capture or compare up event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
10	CCU2	R	0h	Capture or compare up event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
9	CCU1	R	0h	Capture or compare up event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
8	CCU0	R	0h	Capture or compare up event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
7	CCD3	R	0h	Capture or compare down event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
6	CCD2	R	0h	Capture or compare down event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
5	CCD1	R	0h	Capture or compare down event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
4	CCD0	R	0h	Capture or compare down event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
3-2	RESERVED	R	0h	
1	L	R	0h	Load event generated an interrupt. 0h = Event Cleared 1h = Event Set
0	Z	R	0h	Zero event generated an interrupt. 0h = Event Cleared 1h = Event Set

15.3.26 MIS (Offset = 1098h) [Reset = 00000000h]

MIS is shown in Figure 15-66 and described in Table 15-53.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 15-66. MIS

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
R-0h	R-0h	R-0h	R-0h	R-0h		R-0h	R-0h

Table 15-53. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	R	0h	QEIERR 0h = Event Cleared 1h = Event Set
27	DC	R	0h	Direction Change 0h = Event Cleared 1h = Event Set
26	REPC	R	0h	Repeat Counter Zero 0h = Event Cleared 1h = Event Set
25	TOV	R	0h	Trigger overflow 0h = Event Cleared 1h = Event Set
24	F	R	0h	Fault 0h = Event Cleared 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	R	0h	Compare up event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
14	CCU4	R	0h	Compare up event generated an interrupt CCP4 0h = Event Cleared 1h = Event Set
13	CCD5	R	0h	Compare down event generated an interrupt CCP5 0h = Event Cleared 1h = Event Set
12	CCD4	R	0h	Compare down event generated an interrupt CCP4 0h = Event Cleared 1h = Event Set

Table 15-53. MIS Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	R	0h	Capture or compare up event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
10	CCU2	R	0h	Capture or compare up event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
9	CCU1	R	0h	Capture or compare up event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
8	CCU0	R	0h	Capture or compare up event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
7	CCD3	R	0h	Capture or compare down event generated an interrupt CCP3 0h = Event Cleared 1h = Event Set
6	CCD2	R	0h	Capture or compare down event generated an interrupt CCP2 0h = Event Cleared 1h = Event Set
5	CCD1	R	0h	Capture or compare down event generated an interrupt CCP1 0h = Event Cleared 1h = Event Set
4	CCD0	R	0h	Capture or compare down event generated an interrupt CCP0 0h = Event Cleared 1h = Event Set
3-2	RESERVED	R	0h	
1	L	R	0h	Load event generated an interrupt. 0h = Event Cleared 1h = Event Set
0	Z	R	0h	Zero event generated an interrupt. 0h = Event Cleared 1h = Event Set

15.3.27 ISET (Offset = 10A0h) [Reset = 00000000h]

ISET is shown in Figure 15-67 and described in Table 15-54.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 15-67. ISET

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
W-0h	W-0h	W-0h	W-0h	R-0h		W-0h	W-0h

Table 15-54. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	W	0h	QEIERR event SET 0h = Writing 0 has no effect. 1h = Event Set
27	DC	W	0h	Direction Change event SET 0h = Writing 0 has no effect. 1h = Event Set
26	REPC	W	0h	Repeat Counter Zero event SET 0h = Writing 0 has no effect. 1h = Event Set
25	TOV	W	0h	Trigger Overflow event SET 0h = Writing 0 has no effect. 1h = Event Set
24	F	W	0h	Fault event SET 0h = Writing 0 has no effect. 1h = Event Set
23-16	RESERVED	R	0h	
15	CCU5	W	0h	Compare up event 5 SET 0h = Writing 0 has no effect. 1h = Event Set
14	CCU4	W	0h	Compare up event 4 SET 0h = Writing 0 has no effect. 1h = Event Set
13	CCD5	W	0h	Compare down event 5 SET 0h = Writing 0 has no effect. 1h = Event Set
12	CCD4	W	0h	Compare down event 4 SET 0h = Writing 0 has no effect. 1h = Event Set

Table 15-54. ISET Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
10	CCU2	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
9	CCU1	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
8	CCU0	W	0h	Capture or compare up event SET 0h = Writing 0 has no effect. 1h = Event Set
7	CCD3	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
6	CCD2	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
5	CCD1	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
4	CCD0	W	0h	Capture or compare down event SET 0h = Writing 0 has no effect. 1h = Event Set
3-2	RESERVED	R	0h	
1	L	W	0h	Load event SET 0h = Writing 0 has no effect. 1h = Event Set
0	Z	W	0h	Zero event SET 0h = Writing 0 has no effect. 1h = Event Set

15.3.28 ICLR (Offset = 10A8h) [Reset = 00000000h]

ICLR is shown in Figure 15-68 and described in Table 15-55.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 15-68. ICLR

31	30	29	28	27	26	25	24
RESERVED			QEIERR	DC	REPC	TOV	F
R-0h			W-0h	W-0h	W-0h	W-0h	W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
CCU5	CCU4	CCD5	CCD4	CCU3	CCU2	CCU1	CCU0
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
CCD3	CCD2	CCD1	CCD0	RESERVED		L	Z
W-0h	W-0h	W-0h	W-0h	R-0h		W-0h	W-0h

Table 15-55. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	
28	QEIERR	W	0h	QEIERR event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
27	DC	W	0h	Direction Change event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
26	REPC	W	0h	Repeat Counter Zero event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
25	TOV	W	0h	Trigger Overflow event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
24	F	W	0h	Fault event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
23-16	RESERVED	R	0h	
15	CCU5	W	0h	Compare up event 5 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
14	CCU4	W	0h	Compare up event 4 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
13	CCD5	W	0h	Compare down event 5 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
12	CCD4	W	0h	Compare down event 4 CLEAR 0h = Writing 0 has no effect. 1h = Event Clear

Table 15-55. ICLR Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	CCU3	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
10	CCU2	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
9	CCU1	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
8	CCU0	W	0h	Capture or compare up event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
7	CCD3	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
6	CCD2	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
5	CCD1	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
4	CCD0	W	0h	Capture or compare down event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
3-2	RESERVED	R	0h	
1	L	W	0h	Load event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear
0	Z	W	0h	Zero event CLEAR 0h = Writing 0 has no effect. 1h = Event Clear

15.3.29 EVT_MODE (Offset = 10E0h) [Reset = 00000029h]

EVT_MODE is shown in Figure 15-69 and described in Table 15-56.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 15-69. EVT_MODE

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		EVT2_CFG		EVT1_CFG		EVT0_CFG	
R-0h		R-2h		R-2h		R-1h	

Table 15-56. EVT_MODE Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5-4	EVT2_CFG	R	2h	Event line mode select for event corresponding to GEN_EVENT1 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
3-2	EVT1_CFG	R	2h	Event line mode select for event corresponding to GEN_EVENT0 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.
1-0	EVT0_CFG	R	1h	Event line mode select for event corresponding to CPU_INT 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

15.3.30 DESC (Offset = 10FCh) [Reset = 00000000h]

DESC is shown in Figure 15-70 and described in Table 15-57.

Return to the Summary Table.

This register identifies the peripheral and its exact version.

Figure 15-70. DESC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				INSTNUM				MAJREV				MINREV			
R-0h				R-0h				R-0h				R-0h			

Table 15-57. DESC Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	0h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness. 0h = Smallest value FFFFh = Highest possible value
15-12	FEATUREVER	R	0h	Feature Set for the module *instance* 0h = Smallest value Fh = Highest possible value
11-8	INSTNUM	R	0h	Instance Number within the device. This will be a parameter to the RTL for modules that can have multiple instances 0h = Smallest value Fh = Highest possible value
7-4	MAJREV	R	0h	Major rev of the IP 0h = Smallest value Fh = Highest possible value
3-0	MINREV	R	0h	Minor rev of the IP 0h = Smallest value Fh = Highest possible value

15.3.31 CCPD (Offset = 1100h) [Reset = 00000000h]

CCPD is shown in Figure 15-71 and described in Table 15-58.

Return to the Summary Table.

CCP Direction. Controls whether CCP is used as an input or an output.

Figure 15-71. CCPD

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	C0CCP2	C0CCP1	C0CCP0
R-0h				R-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-58. CCPD Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	RESERVED	R	0h	Reserved
2	C0CCP2	R/W	0h	CCP2 direction 0h = input 1h = Output
1	C0CCP1	R/W	0h	CCP1 direction 0h = Input 1h = Output
0	C0CCP0	R/W	0h	CCP0 direction 0h = Input 1h = Output

15.3.32 ODIS (Offset = 1104h) [Reset = 00000000h]

ODIS is shown in Figure 15-72 and described in Table 15-59.

Return to the Summary Table.

The ODIS register output is inverted and then ANDed with the output signal selected by the OCTL register CCPO field (before conditional inversion) to allow software the ability to hold the CCP output low during configuration or shutdown.

Figure 15-72. ODIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				C0CCP3	C0CCP2	C0CCP1	C0CCP0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-59. ODIS Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	C0CCP3	R/W	0h	Counter CCP3 Disable Mask Defines whether CCP3 of Counter n is forced low or not 0h = Output function as selected by the OCTL register CCPO field are provided to occpout[2]. 1h = CCP output occpout[3] is forced low.
2	C0CCP2	R/W	0h	Counter CCP2 Disable Mask Defines whether CCP2 of Counter n is forced low or not 0h = Output function as selected by the OCTL register CCPO field are provided to output inversion block. 1h = CCP output occpout[2] is forced low.
1	C0CCP1	R/W	0h	Counter CCP1 Disable Mask Defines whether CCP0 of Counter n is forced low or not 0h = Output function as selected by the OCTL register CCPO field are provided to output inversion block. 1h = CCP output occpout[1] is forced low.
0	C0CCP0	R/W	0h	Counter CCP0 Disable Mask Defines whether CCP0 of Counter n is forced low or not 0h = Output function as selected by the OCTL register CCPO field are provided to output inversion block. 1h = CCP output occpout[0] is forced low.

15.3.33 CCLKCTL (Offset = 1108h) [Reset = 00000000h]

CCLKCTL is shown in Figure 15-73 and described in Table 15-60.

Return to the Summary Table.

The CCLKCTL register provides a SW mechanism for gating the TIMER clock if the module is expected not to be used but the power domain is alive. This effectively puts the IP in an IDLE state

Figure 15-73. CCLKCTL

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

Table 15-60. CCLKCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	CLKEN	R/W	0h	Clock Enable Disables the clock gating to the module. SW has to explicitly program the value to 0 to gate the clock. 0h = Clock is disabled. 1h = Clock is enabled

15.3.34 CPS (Offset = 110Ch) [Reset = 00000000h]

CPS is shown in Figure 15-74 and described in Table 15-61.

Return to the Summary Table.

The CPS register provides the value for the clock pre-scaler.

Figure 15-74. CPS

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
RESERVED																								PCNT							
R-0h																								R/W-0h							

Table 15-61. CPS Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	PCNT	R/W	0h	Pre-Scale Count This field specifies the pre-scale count value. The selected TIMCLK source is divided by a value of (PCNT+1). A PCNT value of 0 divides TIMCLK by 1, effectively bypassing the divider. A PCNT value of greater than 0 divides the TIMCLK source generating a slower clock 0h = Minimum value FFh = Maximum Value

15.3.35 CPSV (Offset = 1110h) [Reset = 00000000h]

CPSV is shown in Figure 15-75 and described in Table 15-62.

Return to the Summary Table.

The CPSV register provides the ability to read the current clock prescale count value.

Figure 15-75. CPSV

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
RESERVED																								CPSVAL							
R-0h																								R-0h							

Table 15-62. CPSV Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	CPSVAL	R	0h	Current Prescale Count Value 0h = Minimum value FFh = Maximum Value

15.3.36 CTTRIGCTL (Offset = 1114h) [Reset = 00000000h]

CTTRIGCTL is shown in Figure 15-76 and described in Table 15-63.

Return to the Summary Table.

Cross Timer Trigger Control Register
This register is used to control the cross trigger connections for enables and faults of different timer instances in the same power domain. Please refer to sections Timer Module Cross Trigger (In/Out) and Fault Cross Triggering for details.

Figure 15-76. CTTRIGCTL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				EVTCTTRIGSEL			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						EVTCTEN	CTEN
R-0h						R/W-0h	R/W-0h

Table 15-63. CTTRIGCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	
19-16	EVTCTTRIGSEL	R/W	0h	Used to Select the subscriber port that should be used for input cross trigger. 0h = Use FSUB0 as cross trigger source. 1h = Use FSUB1 as cross trigger source. 2h = Use Zero event as cross trigger source. 3h = Use Load event as cross trigger source. 4h = Use CCD0 event as cross trigger source. 5h = Use CCD1 event as cross trigger source. 6h = Use CCD2 event as cross trigger source. 7h = Use CCD3 event as cross trigger source. 8h = Use CCU0 event as cross trigger source. 9h = Use CCU1 event as cross trigger source. Ah = Use CCU2 event as cross trigger source. Bh = Use CCU3 event as cross trigger source.
15-2	RESERVED	R	0h	
1	EVTCTEN	R/W	0h	Enable the Input Trigger Conditions to the Timer module as a condition for Cross Triggers. 0h = Cross trigger generation disabled. 1h = Cross trigger generation enabled
0	CTEN	R/W	0h	Timer Cross trigger enable. This field is used to enable whether the SW or HW logic can generate a timer cross trigger event in the system. These cross triggers are connected to the respective timer trigger in of the other timer IPs in the SOC power domain. The timer cross trigger is essentially the combined logic of the HW and SW conditions controlling EN bit in the CTRCTL register. 0h = Cross trigger generation disabled. 1h = Cross trigger generation enabled

15.3.37 CTTRIG (Offset = 111Ch) [Reset = 00000000h]

CTTRIG is shown in Figure 15-77 and described in Table 15-64.

Return to the Summary Table.

Cross Timer Trigger Register
This register is used to trigger the timer instances connected and enabled using CTTRIGCTL and CTTRIGMSK registers.

Figure 15-77. CTTRIG

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

Table 15-64. CTTRIG Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	TRIG	W	0h	Generate Cross Trigger This bit when programmed will generate a synchronized trigger condition all the cross trigger enabled Timer instances including current timer instance. 0h = Cross trigger generation disabled 1h = Generate Cross trigger pulse

15.3.38 FSCTL (Offset = 1120h) [Reset = 00000000h]

FSCTL is shown in Figure 15-78 and described in Table 15-65.

Return to the Summary Table.

The FSCTL register controls the fault source selection and enable. There are 5 input fault sources either through synchronous path processing or asynchronous path.

Figure 15-78. FSCTL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	FEX2EN	FEX1EN	FEX0EN	FAC2EN	FAC1EN	FAC0EN	FCEN
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-65. FSCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	
6	FEX2EN	R/W	0h	This field controls whether the fault is caused by external fault pin 2. 0h = Disable 1h = Enable
5	FEX1EN	R/W	0h	This field controls whether the fault is caused by external fault pin 1. 0h = Disable 1h = Enable
4	FEX0EN	R/W	0h	This field controls whether the fault is caused by external fault pin 0. 0h = Disable 1h = Enable
3	FAC2EN	R/W	0h	This field controls whether the fault is caused by COMP2 output. 0h = Disable 1h = Enable
2	FAC1EN	R/W	0h	This field controls whether the fault is caused by COMP1 output. 0h = Disable 1h = Enable
1	FAC0EN	R/W	0h	This field controls whether the fault signal is caused by COMP0 output. 0h = Disable 1h = Enable
0	FCEN	R/W	0h	This field controls whether the fault is caused by the system clock fault. 0h = Disable 1h = Enable

15.3.39 GCTL (Offset = 1124h) [Reset = 00000000h]

GCTL is shown in Figure 15-79 and described in Table 15-66.

Return to the Summary Table.

Global control register

Figure 15-79. GCTL

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

Table 15-66. GCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	SHDWLDEN	R/W	0h	Enables shadow to active load of buffered registers and register fields. 0h = Disable 1h = Enable

15.3.40 CTR (Offset = 1800h) [Reset = 00000000h]

CTR is shown in Figure 15-80 and described in Table 15-67.

Return to the Summary Table.

This is the TIMER counter register.
This can be set by SW. However, the writes will be unpredictable if the software tries to set a value while the counter is running.

Figure 15-80. CTR

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
RESERVED																CCTR															
R-0h																R/W-0h															

Table 15-67. CTR Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	CCTR	R/W	0h	Current Counter value 0h = Minimum value 00FFFFFFh = Maximum Value

15.3.41 CTRCTL (Offset = 1804h) [Reset = 0000FF80h]

CTRCTL is shown in Figure 15-81 and described in Table 15-68.

Return to the Summary Table.

This register provides control over the counter operation.
The configuration can change as well as setting the EN bit in a single write.
There is no requirement to change the configuration first and then do an additional write to set the EN bit.

Figure 15-81. CTRCTL

31	30	29	28	27	26	25	24
RESERVED		CVAE		RESERVED			PLEN
R-0h		R/W-0h		R-0h			R/W-0h
23	22	21	20	19	18	17	16
SLZERCNEZ	RESERVED			FRB	FB	DRB	RESERVED
R/W-0h	R-0h			R/W-0h	R/W-0h	R/W-0h	R-0h
15	14	13	12	11	10	9	8
CZC				CAC		CLC	
R/W-7h				R/W-7h		R/W-7h	
7	6	5	4	3	2	1	0
CLC	RESERVED	CM		REPEAT			EN
R/W-7h	R-0h	R/W-0h		R/W-0h			R/W-0h

Table 15-68. CTRCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	
29-28	CVAE	R/W	0h	Counter Value After Enable. This field specifies the initialization condition of the counter when the EN bit is changed from 0 to 1 by a write to the CTRCTL register. Note that an external event can also cause the EN bit to go active. 0h = The counter is set to the LOAD register value 1h = The counter value is unchanged from its current value which could have been initialized by software 2h = The counter is set to zero
27-25	RESERVED	R	0h	
24	PLEN	R/W	0h	Phase Load Enable. This bit allows the timer to have phase load feature. 0h = Disabled 1h = Enabled
23	SLZERCNEZ	R/W	0h	Suppress Load and Zero Events if Repeat Counter is Not Equal to Zero. This bit suppresses the generation of the Z (zero) and L (load) events from the counter when the repeat counter (RC) value is not 0. 0h = Disabled. Z and L events are always generated from the counter when their conditions are generated. 1h = Enabled. Z and L events are generated from the counter when their conditions are generated and the RC register value is 0.
22-20	RESERVED	R	0h	
19	FRB	R/W	0h	Fault Resume Behavior This bit specifies what the device does following the release/exit of fault condition. 0h = Resume counting 1h = Perform the action as specified by the CVAE field.

Table 15-68. CTRCTL Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	FB	R/W	0h	Fault Behavior This bit specifies whether the counter continues running or suspends during a fault mode. There is a separate control under REPEAT to indicate whether counting is to suspend at next Counter==0 0h = Continues counting 1h = Suspends counting
17	DRB	R/W	0h	Debug Resume Behavior This bit specifies what the device does following the release/exit of debug mode. 0h = Resume counting 1h = Perform the action as specified by the CVAE field.
16	RESERVED	R	0h	
15-13	CZC	R/W	7h	Counter Zero Control This field specifies what controls the counter operation with respect to zeroing the counter value. Encodings 1-3 are present based on the CCPC parameter value. Bits 4-5 are present based on the HQEI parameter value. Any encodings not provided are documented as reserved. 0h = CCCTL_0 ZCOND 1h = CCCTL_1 ZCOND 2h = CCCTL_2 ZCOND This value exists when there are 4 channels. 3h = CCCTL_3 ZCOND This value exists when there are 4 channels. 4h = Controlled by 2-input QEI mode This value exists when TIMER support QEI feature. 5h = Controlled by 3-input QEI mode This value exists when TIMER support QEI feature.
12-10	CAC	R/W	7h	Counter Advance Control. This field specifies what controls the counter operation with respect to advancing (incrementing or decrementing) the counter value. Encodings 1-3 are present based on the CCPC parameter value. Bits 4-5 are present based on the HQEI parameter value. Any encodings not provided are documented as reserved. 0h = CCCTL_0 ACOND 1h = CCCTL_1 ACOND 2h = CCCTL_2 ACOND This value exists when there are 4 channels. 3h = CCCTL_3 ACOND This value exists when there are 4 channels. 4h = Controlled by 2-input QEI mode This value exists when TIMER support QEI feature. 5h = Controlled by 3-input QEI mode This value exists when TIMER support QEI feature.
9-7	CLC	R/W	7h	Counter Load Control. This field specifies what controls the counter operation with respect to setting the counter to the LD register value. Encodings 1-3 are present based on the CCPC parameter value. Bits 4-5 are present based on the HQEI parameter value. Any encodings not provided are documented as reserved. 0h = CCCTL_0 LCOND 1h = CCCTL_1 LCOND 2h = CCCTL_2 LCOND This value exists when there are 4 channels. 3h = CCCTL_3 LCOND This value exists when there are 4 channels. 4h = Controlled by 2 input QEI mode. This value exists when TIMER support QEI feature. 5h = Controlled by 3 input QEI mode. This value exists when TIMER support QEI feature.
6	RESERVED	R	0h	

Table 15-68. CTRCTL Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	CM	R/W	0h	Count Mode 0h = Down 1h = Up/Down 2h = Counter counts up.
3-1	REPEAT	R/W	0h	Repeat. The repeat bit controls whether the counter continues to advance following a zero event, or the exiting of a debug or fault condition. If counting down, a zero event is followed by a load at the next advance condition. If counting up-down, a zero event is followed by an advance event (+1). The intent of encoding 3 is that if the debug condition is in effect, the generation of the load pulse is deferred until the debug condition is over. This allows the counter to reach zero before counting is suspended. 0h = Does not automatically advance following a zero event. 1h = Continues to advance following a zero event. 2h = Reserved 3h = Continues to advance following a zero event if the debug mode is not in effect, or following the release of the debug mode. 4h = Reserved
0	EN	R/W	0h	Counter Enable. This bit allows the timer to advance This bit is automatically cleared if REPEAT=0 (do not automatically reload) and the counter value equals zero. CPU Write: A register write that sets the EN bit, the counter value is set per the CVAE value. Hardware: This bit may also be set as the result of an LCOND or ZCOND condition being met and the counter value changed to the load value or zero value, respectively. 0h = Disabled 1h = Enabled

15.3.42 LOAD (Offset = 1808h) [Reset = 00000000h]

LOAD is shown in Figure 15-82 and described in Table 15-69.

Return to the Summary Table.

The contents of LOAD register are copied to CTR on any operation designated to do a "LOAD". The LOAD is used to compare with the CTR for generating a "Load Event" that can be used for interrupt, trigger, or signal generator actions.

Figure 15-82. LOAD

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
RESERVED																LD															
R-0h																R/W-0h															

Table 15-69. LOAD Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	LD	R/W	0h	Load Value 0h = Minimum value 00FFFFFFh = Maximum Value

15.3.43 CC_01[y] (Offset = 1810h + formula) [Reset = 00000000h]

CC_01[y] is shown in Figure 15-83 and described in Table 15-70.

Return to the Summary Table.

The CC_01 register is a register that can be used as either a capture register, to capture the next CTR value on an event, or a compare to the current CTR to create an event. It cannot operate concurrently as both. There are two Capture-Compare slices of hardware for each counter, hence there are two CC_01 registers per timer. On a capture event, the next value of the CTR is loaded so that CTR and CC_01 (which captured) will be equal on the cycle that an interrupt or trigger is created from the capture action.

Offset = 1810h + (y * 4h); where y = 0h to 1h

Figure 15-83. CC_01[y]

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
RESERVED																CCVAL															
R-0h																R/W-0h															

Table 15-70. CC_01[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	CCVAL	R/W	0h	Capture or compare value 0h = Minimum value FFFFh = Maximum Value

15.3.44 CC_23[y] (Offset = 1818h + formula) [Reset = 00000000h]

CC_23[y] is shown in Figure 15-84 and described in Table 15-71.

Return to the Summary Table.

The CC_23 register is a register that can be used as either a capture register, to capture the next CTR value on an event, or a compare to the current CTR to create an event. It cannot operate concurrently as both. There are two Capture-Compare slices of hardware for each counter, hence there are two CC_01 registers per timer. On a capture event, the next value of the CTR is loaded so that CTR and CC_01 (which captured) will be equal on the cycle that an interrupt or trigger is created from the capture action.

Offset = 1818h + (y * 4h); where y = 0h to 1h

Figure 15-84. CC_23[y]

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
RESERVED																CCVAL															
R-0h																R/W-0h															

Table 15-71. CC_23[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	CCVAL	R/W	0h	Capture or compare value 0h = Minimum value FFFFh = Maximum Value

15.3.45 CC_45[y] (Offset = 1820h + formula) [Reset = 00000000h]

CC_45[y] is shown in Figure 15-85 and described in Table 15-72.

Return to the Summary Table.

The CC_45 register are a registers which can be used as compare to the current CTR to create an events CC4U, CC4D, CC5U and CC5D.

Offset = 1820h + (y * 4h); where y = 0h to 1h

Figure 15-85. CC_45[y]

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
RESERVED																CCVAL															
R-0h																R/W-0h															

Table 15-72. CC_45[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	CCVAL	R/W	0h	Capture or compare value 0h = Minimum value FFFFh = Maximum Value

15.3.46 CCCTL_01[y] (Offset = 1830h + formula) [Reset = 00000000h]

CCCTL_01[y] is shown in Figure 15-86 and described in Table 15-73.

Return to the Summary Table.

The CCCTL_01 registers control the operations of the respective CC registers and the counter.

Offset = 1830h + (y * 4h); where y = 0h to 1h

Figure 15-86. CCCTL_01[y]

31	30	29	28	27	26	25	24
CC2SELD			CCACTUPD			SCERCNEZ	CC2SELU
R/W-0h			R/W-0h			R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
CC2SELU		RESERVED	CCUPD			COC	RESERVED
R/W-0h		R-0h	R/W-0h			R/W-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	ZCOND			RESERVED	LCOND		
R-0h	R/W-0h			R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	ACOND			RESERVED	CCOND		
R-0h	R/W-0h			R-0h	R/W-0h		

Table 15-73. CCCTL_01[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-29	CC2SELD	R/W	0h	Selects the source second CCD event. 0h = Selects CCD from CC0. 1h = Selects CCD from CC1. 2h = Selects CCD from CC2. 3h = Selects CCD from CC3. 4h = Selects CCD from CC4. 5h = Selects CCD from CC5.

Table 15-73. CCCTL_01[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28-26	CCACTUPD	R/W	0h	CCACT shadow register Update Method This field controls how updates to the CCACT shadow register are performed 0h = Value written to the CCACT register has immediate effect. 1h = Following a zero event (CTR=0) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals 0. 2h = Following a CCD event (CTR=CC_xy) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 3h = Following a CCU event (CTR=CC_xy) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 4h = Following a zero event (CTR=0) or load event (CTR = LOAD) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals 0 or CTR. Equals LDn. Note this update mechanism is defined for use only in configurations using up/down counting. This mode is not intended for use in down count configurations. 5h = Following a zero event (CTR=0) with repeat count also zero (RC=0). Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals 0 and if RC equal 0. 6h = On a TRIG pulse, the value stored in CCACT_xy shadow register is loaded into CCACT_xy register.
25	SCERCNEZ	R/W	0h	Suppress Compare Event if Repeat Counter is Not Equal to Zero This bit suppresses the generation of the compare (CCD, CCU and RC) events from the counter when the repeat counter (RC) value is not 0. 0h = CCD, CCU and RC events are always generated from the counter when their conditions are generated. 1h = CCD, CCU and RC events are generated from the counter when their conditions are generated and the RC register value is 0.
24-22	CC2SELU	R/W	0h	Selects the source second CCU event. 0h = Selects CCU from CC0. 1h = Selects CCU from CC1. 2h = Selects CCU from CC2. 3h = Selects CCU from CC3. 4h = Selects CCU from CC4. 5h = Selects CCU from CC5.

Table 15-73. CCCTL_01[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	RESERVED	R	0h	
20-18	CCUPD	R/W	0h	Capture and Compare Update Method This field controls how updates to the shadow capture and compare register are performed (when operating in compare mode, COC=0). 0h = Writes to the CCx_y register is written to the register directly and has immediate effect. 1h = Following a zero event (CTR=0) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0. 2h = Following a CCD event (CTR=CC_xy) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 3h = Following a CCU event (CTR=CC_xy) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 4h = Following a zero event(CTR=0) or load event (CTR=LOAD) Writes to the CCx_y register are stored in shadow register and transferred to ECCx_y in the TIMCLK cycle following CTR equals 0 or CTR. Equals LD. Note this update mechanism is defined for use only in configurations using up/down counting. This mode is not intended for use in down count configurations. 5h = Following a zero event (CTR=0) with repeat count also zero (RC=0). Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0 and if RC equal 0. 6h = Following a TRIG pulse. Writes to the CCx_y register are stored in shadow register and transferred to CCx_y
17	COC	R/W	0h	Capture or Compare. Specifies whether the corresponding CC register is used as a capture register or a compare register (never both). 0h = Compare 1h = Capture
16-15	RESERVED	R	0h	

Table 15-73. CCCTL_01[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-12	ZCOND	R/W	0h	Zero Condition. This field specifies the condition that generates a zero pulse. 0h = CCP edges have no effect 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge)
11	RESERVED	R	0h	
10-8	LCOND	R/W	0h	Load Condition. Specifies the condition that generates a load pulse. 0h = CCP edges have no effect 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge)
7	RESERVED	R	0h	
6-4	ACOND	R/W	0h	Advance Condition. Specifies the condition that generates an advance pulse. 0h = Each TIMCLK 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge) 5h = CCP High or Trigger assertion (level)
3	RESERVED	R	0h	
2-0	CCOND	R/W	0h	Capture Condition. Specifies the condition that generates a capture pulse. 0h = None (never captures) 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge)

15.3.47 CCCTL_23[y] (Offset = 1838h + formula) [Reset = 00000000h]

CCCTL_23[y] is shown in Figure 15-87 and described in Table 15-74.

Return to the Summary Table.

The CCCTL registers control the operations of the respective CC registers and the counter.

Offset = 1838h + (y * 4h); where y = 0h to 1h

Figure 15-87. CCCTL_23[y]

31	30	29	28	27	26	25	24
CC2SELD			CCACTUPD			SCERCNEZ	CC2SELU
R/W-0h			R/W-0h			R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
CC2SELU		RESERVED	CCUPD			COC	RESERVED
R/W-0h		R-0h	R/W-0h			R/W-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	ZCOND			RESERVED	LCOND		
R-0h	R/W-0h			R-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	ACOND			RESERVED	CCOND		
R-0h	R/W-0h			R-0h	R/W-0h		

Table 15-74. CCCTL_23[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-29	CC2SELD	R/W	0h	Selects the source second CCD event. 0h = Selects CCD from CC0. 1h = Selects CCD from CC1. 2h = Selects CCD from CC2. 3h = Selects CCD from CC3. 4h = Selects CCD from CC4. 5h = Selects CCD from CC5.

Table 15-74. CCCTL_23[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28-26	CCACTUPD	R/W	0h	CCACT shadow register Update Method This field controls how updates to the CCCACT shadow register are performed 0h = Value written to the CCACTx_y register has immediate effect. 1h = Following a zero event (CTR=0) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals 0. 2h = Following a CCD event (CTR=CC_xy) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 3h = Following a CCU event (CTR=cc_xy) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 4h = Following a zero event (CTR=0) or load event (CTR=LOAD) Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals 0 or CTR. Equals LDn. Note this update mechanism is defined for use only in configurations using up/down counting. This mode is not intended for use in down count configurations. 5h = Following a zero event (CTR=0) with repeat count also zero (RC=0). Writes to the CCACTx_y register are stored in shadow register and transferred to CCACTx_y in the TIMCLK cycle following CTR equals 0 and if RC equal 0. 6h = On a TRIG pulse, the value stored in CCACTx_y shadow register is loaded into CCACTx_y active register.
25	SCERCNEZ	R/W	0h	Suppress Compare Event if Repeat Counter is Not Equal to Zero This bit suppresses the generation of the compare (CCD, CCU and RC) events from the counter when the repeat counter (RCn) value is not 0. 0h = CCD, CCU and RC events are always generated from the counter when their conditions are generated. 1h = CCD, CCU and RC events are generated from the counter when their conditions are generated and the RC register value is 0.

Table 15-74. CCCTL_23[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24-22	CC2SELU	R/W	0h	Selects the source second CCU event. 0h = Selects CCU from CC0. 1h = Selects CCU from CC1. 2h = Selects CCU from CC2. 3h = Selects CCU from CC3. 4h = Selects CCU from CC4. 5h = Selects CCU from CC5.
21	RESERVED	R	0h	
20-18	CCUPD	R/W	0h	Capture and Compare Update Method This field controls how updates to the shadow capture and compare register are performed (when operating in compare mode, COC=0). 0h = Writes to the CCx_y register is written to the register directly and has immediate effect. 1h = Following a zero event (CTR=0) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0. 2h = Following a CCD event (CTR=CC_xy) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 3h = Following a CCU event (CTR=CC_xy) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 4h = Following a zero or load event Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0 or CTR. Equals LDn. Note this update mechanism is defined for use only in configurations using up/down counting. This mode is not intended for use in down count configurations. 5h = Following a zero event (CTR=0) with repeat count also zero (RC=0). Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0 and if RC equal 0. 6h = Following a TRIG pulse. Writes to the CCx_y register are stored in shadow register and transferred to CCx_y #xD; 0.

Table 15-74. CCCTL_23[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	COC	R/W	0h	Capture or Compare. Specifies whether the corresponding CC register is used as a capture register or a compare register (never both). 0h = Compare 1h = Capture
16-15	RESERVED	R	0h	
14-12	ZCOND	R/W	0h	Zero Condition. This field specifies the condition that generates a zero pulse. 4h-Fh = Reserved 0h = CCP edges have no effect 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge)
11	RESERVED	R	0h	
10-8	LCOND	R/W	0h	Load Condition. Specifies the condition that generates a load pulse. 4h-Fh = Reserved 0h = CCP edges have no effect 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge)
7	RESERVED	R	0h	
6-4	ACOND	R/W	0h	Advance Condition. Specifies the condition that generates an advance pulse. 6h-Fh = Reserved 0h = Each TIMCLK 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge) 5h = CCP High or Trigger assertion (level)
3	RESERVED	R	0h	
2-0	CCOND	R/W	0h	Capture Condition. Specifies the condition that generates a capture pulse. 4h-Fh = Reserved 0h = None (never captures) 1h = Rising edge of CCP or trigger assertion edge 2h = Falling edge of CCP or trigger de-assertion edge 3h = Either edge of CCP or trigger change (assertion/de-assertion edge)

15.3.48 CCCTL_45[y] (Offset = 1840h + formula) [Reset = 00000000h]

CCCTL_45[y] is shown in Figure 15-88 and described in Table 15-75.

Return to the Summary Table.

The CCCTL registers control the operations of the respective CC registers and the counter.

Offset = 1840h + (y * 4h); where y = 0h to 1h

Figure 15-88. CCCTL_45[y]

31	30	29	28	27	26	25	24
RESERVED						SCERCNEZ	RESERVED
R-0h						R/W-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED			CCUPD			RESERVED	
R-0h			R/W-0h			R-0h	
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 15-75. CCCTL_45[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	
25	SCERCNEZ	R/W	0h	Suppress Compare Event if Repeat Counter is Not Equal to Zero This bit suppresses the generation of the compare (CCD, CCU and RC) events from the counter when the repeat counter (RC) value is not 0. 0h = CCD, CCU and RC events are always generated from the counter when their conditions are generated. 1h = CCD, CCU and RC events are generated from the counter when their conditions are generated and the RC register value is 0.
24-21	RESERVED	R	0h	

Table 15-75. CCCTL_45[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20-18	CCUPD	R/W	0h	Capture and Compare Update Method This field controls how updates to the shadow capture and compare register are performed (when operating in compare mode, COC=0). 0h = Writes to the CCx_y register is written to the register directly and has immediate effect. 1h = Following a zero event (CTR=0) Writes to the CCx_y register are stored in shadow register and transferred to ECCx_y in the TIMCLK cycle following CTR equals 0. 2h = Following a CCD event (CTR=CC_xy) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 3h = Following a CCU event (CTR=CC_xy) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals the CCx_y register value. 4h = Following a zero event (CTR=0) or load event (CTR=LOAD) Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0 or CTR. Equals LD. Note this update mechanism is defined for use only in configurations using up/down counting. This mode is not intended for use in down count configurations. 5h = Following a zero event (CTR=0) with repeat count also zero (RC=0). Writes to the CCx_y register are stored in shadow register and transferred to CCx_y in the TIMCLK cycle following CTR equals 0 and if RC equal 0. 6h = Following a TRIG pulse. Writes to the CCx_y register are stored in shadow register and transferred to CCx_y #xD; 0.
17-0	RESERVED	R	0h	

15.3.49 OCTL_01[y] (Offset = 1850h + formula) [Reset = 00000000h]

OCTL_01[y] is shown in Figure 15-89 and described in Table 15-76.

Return to the Summary Table.

The OCTL_01 register controls the CCP output of the Capture-Compare slice of the counter. This includes the ability to select the source of what is driven out along with initial condition values and final inversion options.

Offset = 1850h + (y * 4h); where y = 0h to 1h

Figure 15-89. OCTL_01[y]

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CCPIV	CCPOINV	CCPO			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 15-76. OCTL_01[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5	CCPIV	R/W	0h	CCP Initial Value This bit specifies the logical value put on the signal generator state while the counter is disabled (CTRCTL.EN == 0). 0h = Low 1h = High
4	CCPOINV	R/W	0h	CCP Output Invert The output as selected by CCPO is conditionally inverted. 0h = No inversion 1h = Invert
3-0	CCPO	R/W	0h	CCP Output Source 0h = Signal generator value (for example, PWM, triggered PWM) 1h = Load event 2h = CCU event or CCD event 4h = Zero event 5h = Capture event 6h = Fault condition 8h = Mirror CCP of first capture and compare register to other capture compare blocks 9h = Mirror CCP of second capture and compare register in other capture compare blocks Ch = Signal generator output after deadband insertion Dh = Counter direction

15.3.50 OCTL_23[y] (Offset = 1858h + formula) [Reset = 00000000h]

OCTL_23[y] is shown in Figure 15-90 and described in Table 15-77.

Return to the Summary Table.

The OCTL register controls the CCP output of the Capture-Compare slice of the counter. This includes the ability to select the source of what is driven out along with initial condition values and final inversion options.

Offset = 1858h + (y * 4h); where y = 0h to 1h

Figure 15-90. OCTL_23[y]

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CCPIV	CCPOINV	CCPO			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 15-77. OCTL_23[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5	CCPIV	R/W	0h	CCP Initial Value This bit specifies the logical value put on the signal generator state while the counter is disabled (CTRCTL.EN == 0). 0h = Low 1h = High
4	CCPOINV	R/W	0h	CCP Output Invert The output as selected by CCPO is conditionally inverted. 0h = No inversion 1h = Invert
3-0	CCPO	R/W	0h	CCP Output Source 0h = Signal generator value (for example, PWM, triggered PWM) 1h = Load condition 2h = CCU event or CCD event 4h = Zero event 5h = Capture event 6h = Fault Condition 8h = Mirror CCP of first capture and compare register in other capture compare blocks 9h = Mirror CCP of second capture and compare register in other capture compare blocks. Ch = Deadband Inserted Output Dh = Counter direction

15.3.51 CCACT_01[y] (Offset = 1870h + formula) [Reset = 00000000h]

CCACT_01[y] is shown in Figure 15-91 and described in Table 15-78.

Return to the Summary Table.

The CCACT_01 register controls the actions of the signal generator of the capture-compare slice based on the events created in the counter block, the capture and compare block and debug events.

Offset = 1870h + (y * 4h); where y = 0h to 1h

Figure 15-91. CCACT_01[y]

31	30	29	28	27	26	25	24
SWFRCACT_CMPL		SWFRCACT		FEXACT			FENACT
R/W-0h		R/W-0h		R/W-0h			R/W-0h
23	22	21	20	19	18	17	16
FENACT		RESERVED					CC2UACT
R/W-0h		R-0h					R/W-0h
15	14	13	12	11	10	9	8
CC2UACT	RESERVED	CC2DACT		RESERVED	CUACT		RESERVED
R/W-0h	R-0h	R/W-0h		R-0h	R/W-0h		R-0h
7	6	5	4	3	2	1	0
CDACT		RESERVED	LACT		RESERVED	ZACT	
R/W-0h		R-0h	R/W-0h		R-0h	R/W-0h	

Table 15-78. CCACT_01[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SWFRCACT_CMPL	R/W	0h	CCP Complimentary output Action on Software Force Output This field describes the resulting action of software force. This action has a shadow register, which will be updated under specific condition. So that this register cannot take into effect immediately. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP Complimentary output value is set high 2h = CCP Complimentary output value is set low
29-28	SWFRCACT	R/W	0h	CCP Output Action on Software Force Output This field describes the resulting action of software force. This action has a shadow register, which will be updated under specific condition. So that this register cannot take into effect immediately. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low
27-25	FEXACT	R/W	0h	CCP Output Action on Fault Exit This field describes the resulting action of the signal generator upon exiting the fault condition. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled 4h = CCP output value is tristated

Table 15-78. CCACT_01[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24-22	FENACT	R/W	0h	CCP Output Action on Fault Entry This field describes the resulting action of the signal generator upon detecting a fault. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled 4h = CCP output value is tristated
21-17	RESERVED	R	0h	
16-15	CC2UACT	R/W	0h	CCP Output Action on CC2U event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
14	RESERVED	R	0h	
13-12	CC2DACT	R/W	0h	CCP Output Action on CC2D event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
11	RESERVED	R	0h	
10-9	CUACT	R/W	0h	CCP Output Action on Compare (Up) This field describes the resulting action of the signal generator upon detecting a compare event while counting up. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
8	RESERVED	R	0h	
7-6	CDACT	R/W	0h	CCP Output Action on Compare (Down) This field describes the resulting action of the signal generator upon detecting a compare event while counting down. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
5	RESERVED	R	0h	
4-3	LACT	R/W	0h	CCP Output Action on Load Specifies what changes occur to CCP output as the result of a load event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
2	RESERVED	R	0h	
1-0	ZACT	R/W	0h	CCP Output Action on Zero Specifies what changes occur to CCP output as the result of a zero event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled

15.3.52 CCACT_23[y] (Offset = 1878h + formula) [Reset = 00000000h]

CCACT_23[y] is shown in Figure 15-92 and described in Table 15-79.

Return to the Summary Table.

The CCACT register controls the actions of the signal generator of the capture-compare slice based on the events created in the counter block, the capture and compare block and debug events.

Offset = 1878h + (y * 4h); where y = 0h to 1h

Figure 15-92. CCACT_23[y]

31	30	29	28	27	26	25	24
SWFRCACT_CMPL		SWFRCACT		FEXACT			FENACT
R/W-0h		R/W-0h		R/W-0h			R/W-0h
23	22	21	20	19	18	17	16
FENACT		RESERVED					CC2UACT
R/W-0h		R-0h					R/W-0h
15	14	13	12	11	10	9	8
CC2UACT	RESERVED	CC2DACT		RESERVED	CUACT		RESERVED
R/W-0h	R-0h	R/W-0h		R-0h	R/W-0h		R-0h
7	6	5	4	3	2	1	0
CDACT		RESERVED	LACT		RESERVED	ZACT	
R/W-0h		R-0h	R/W-0h		R-0h	R/W-0h	

Table 15-79. CCACT_23[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SWFRCACT_CMPL	R/W	0h	CCP Complimentary Output Action on Software Force Output This field describes the resulting action of software force. This action has a shadow register, which will be updated under specific condition. So that this register cannot take into effect immediately. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP Complimentary output value is set high 2h = CCP Complimentary output value is set low
29-28	SWFRCACT	R/W	0h	CCP Output Action on Software Force Output This field describes the resulting action of software force. This action has a shadow register, which will be updated under specific condition. So that this register cannot take into effect immediately. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low
27-25	FEXACT	R/W	0h	CCP Output Action on Fault Exit This field describes the resulting action of the signal generator upon exiting the fault condition. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled 4h = CCP output value is tristated

Table 15-79. CCACT_23[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24-22	FENACT	R/W	0h	CCP Output Action on Fault Entry This field describes the resulting action of the signal generator upon detecting a fault. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled 4h = CCP output value is tristated
21-17	RESERVED	R	0h	
16-15	CC2UACT	R/W	0h	CCP Output Action on CC2U event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
14	RESERVED	R	0h	
13-12	CC2DACT	R/W	0h	CCP Output Action on CC2D event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
11	RESERVED	R	0h	
10-9	CUACT	R/W	0h	CCP Output Action on Compare (Up) This field describes the resulting action of the signal generator upon detecting a compare event while counting up. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
8	RESERVED	R	0h	
7-6	CDACT	R/W	0h	CCP Output Action on Compare (Down) This field describes the resulting action of the signal generator upon detecting a compare event while counting down. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
5	RESERVED	R	0h	
4-3	LACT	R/W	0h	CCP Output Action on Load Specifies what changes occur to CCP output as the result of a load event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled
2	RESERVED	R	0h	
1-0	ZACT	R/W	0h	CCP Output Action on Zero Specifies what changes occur to CCP output as the result of a zero event. 0h = This event is disabled and a lower priority event is selected if asserting. The CCP output value is unaffected by the event. 1h = CCP output value is set high 2h = CCP output value is set low 3h = CCP output value is toggled

15.3.53 IFCTL_01[y] (Offset = 1880h + formula) [Reset = 00000000h]

IFCTL_01[y] is shown in Figure 15-93 and described in Table 15-80.

Return to the Summary Table.

The IFCTL_01 register controls the input selection and inversion for the associated Capture-Compare slice.

Offset = 1880h + (y * 4h); where y = 0h to 1h

Figure 15-93. IFCTL_01[y]

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			FE	CPV	RESERVED	FP	
R-0h			R/W-0h	R/W-0h	R-0h	R/W-0h	
7	6	5	4	3	2	1	0
INV	RESERVED			ISEL			
R/W-0h	R-0h			R/W-0h			

Table 15-80. IFCTL_01[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	
12	FE	R/W	0h	Filter Enable This bit controls whether the input is filtered by the input filter or bypasses to the edge detect. 0h = Bypass. 1h = Filtered.
11	CPV	R/W	0h	Consecutive Period/Voting Select This bit controls whether the input filter uses a stricter consecutive period count or majority voting. 0h = Consecutive Periods The input must be at a specific logic level for the period defined by FP before it is passed to the filter output. 1h = Voting The filter ignores one clock of opposite logic over the filter period. That is, over FP samples of the input, up to 1 sample may be of an opposite logic value (glitch) without affecting the output.
10	RESERVED	R	0h	
9-8	FP	R/W	0h	Filter Period. This field specifies the sample period for the input filter. That is, the input is sampled for FP timer clocks during filtering. 0h = The division factor is 3 1h = The division factor is 5 2h = The division factor is 8

Table 15-80. IFCTL_01[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	INV	R/W	0h	Input Inversion This bit controls whether the selected input is inverted. 0h = Noninverted 1h = Inverted
6-4	RESERVED	R	0h	
3-0	ISEL	R/W	0h	Input Select (CCP0) This field selects the input source to the filter input. 4h-7h = Reserved 0h = CCP of the corresponding capture compare unit 1h = Input pair CCPX of the capture compare unit. For CCP0 input pair is CCP1 and for CCP1 input pair is CCP0. 2h = CCP0 of the counter 3h = Trigger 4h = XOR of CCP inputs as input source (Used in Hall input mode). 5h = subscriber 0 event as input source. 6h = subscriber 1 event as input source. 7h = Comparator 0 output. 8h = Comparator 1 output. 9h = Comparator 2 output.

15.3.54 IFCTL_23[y] (Offset = 1888h + formula) [Reset = 00000000h]

IFCTL_23[y] is shown in Figure 15-94 and described in Table 15-81.

Return to the Summary Table.

The IFCTL register controls the input selection and inversion for the associated Capture-Compare slice.

Offset = 1888h + (y * 4h); where y = 0h to 1h

Figure 15-94. IFCTL_23[y]

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			FE	CPV	RESERVED	FP	
R-0h			R/W-0h	R/W-0h	R-0h	R/W-0h	
7	6	5	4	3	2	1	0
INV	RESERVED			ISEL			
R/W-0h	R-0h			R/W-0h			

Table 15-81. IFCTL_23[y] Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	
12	FE	R/W	0h	Filter Enable This bit controls whether the input is filtered by the input filter or bypasses to the edge detect. 0h = Bypass. 1h = Filtered.
11	CPV	R/W	0h	Consecutive Period/Voting Select This bit controls whether the input filter uses a stricter consecutive period count or majority voting. 0h = Consecutive Periods The input must be at a specific logic level for the period defined by FP before it is passed to the filter output. 1h = Voting The filter ignores one clock of opposite logic over the filter period. That is, over FP samples of the input, up to 1 sample may be of an opposite logic value (glitch) without affecting the output.
10	RESERVED	R	0h	
9-8	FP	R/W	0h	Filter Period. This field specifies the sample period for the input filter. That is, the input is sampled for FP timer clocks during filtering. 0h = The division factor is 3 1h = The division factor is 5 2h = The division factor is 8
7	INV	R/W	0h	Input Inversion This bit controls whether the selected input is inverted. 0h = Noninverted 1h = Inverted
6-4	RESERVED	R	0h	

Table 15-81. IFCTL_23[y] Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	ISEL	R/W	0h	Input Select (CCP0) This field selects the input source to the filter input. 4h-7h = Reserved 0h = CCP of the corresponding capture compare unit 1h = Input pair CCPX of the capture compare unit. For CCP0 input pair is CCP1 and for CCP1 input pair is CCP0. 2h = CCP0 of the counter 3h = Trigger 4h = XOR of CCP inputs as input source (Used in Hall input mode). 5h = subscriber 0 event as input source. 6h = subscriber 1 event as input source. 7h = Comparator 0 output. 8h = Comparator 1 output. 9h = Comparator 2 output.

15.3.55 PL (Offset = 18A0h) [Reset = 00000000h]

PL is shown in Figure 15-95 and described in Table 15-82.

Return to the Summary Table.

This is the phase load register.

Figure 15-95. PL

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
RESERVED																PHASE															
R-0h																R/W-0h															

Table 15-82. PL Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	PHASE	R/W	0h	Phase Load value 0h = Minimum value 00FFFFFFh = Maximum Value

15.3.56 DBCTL (Offset = 18A4h) [Reset = 00000000h]

DBCTL is shown in Figure 15-96 and described in Table 15-83.

Return to the Summary Table.

The DBCTL register controls the dead band insertion of the pulse width modulated output.

Figure 15-96. DBCTL

31	30	29	28	27	26	25	24
RESERVED				FALLDELAY			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
FALLDELAY							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED			M1_ENABLE	RISEDELAY			
R-0h			R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
RISEDELAY							
R/W-0h							

Table 15-83. DBCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	
27-16	FALLDELAY	R/W	0h	Fall Delay The number of TIMCLK periods inserted between the fall edge of CCP signal and the rise edge of CCP complimentary signal. 0h = Minimum value FFFh = Maximum Value
15-13	RESERVED	R	0h	
12	M1_ENABLE	R/W	0h	Dead Band Mode 1 Enable. 0h = Disabled 1h = Enabled
11-0	RISEDELAY	R/W	0h	Rise Delay The number of TIMCLK periods inserted between the fall edge of CCP signal and the rise edge of CCP complimentary signal. 0h = Minimum value FFFh = Maximum Value

15.3.57 TSEL (Offset = 18B0h) [Reset = 00000000h]

TSEL is shown in Figure 15-97 and described in Table 15-84.

Return to the Summary Table.

The TSEL register controls the input trigger enable and selection of the trigger source. Trigger sources are generated by other SoC elements through their respective publisher ports (subscribed in by the timer's subscriber port).

Figure 15-97. TSEL

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						TE	RESERVED				ETSEL				
R-0h						R/W-0h	R-0h				R/W-0h				

Table 15-84. TSEL Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9	TE	R/W	0h	Trigger Enable. This selects whether a trigger is enabled or not for this counter 0x0 = Triggers are not used 0x1 = Triggers are used as selected by the ETSEL field 0h = Triggers are not used. 1h = Triggers are used as selected by the IE, ITSEL and ETSEL fields.
8-5	RESERVED	R	0h	
4-0	ETSEL	R/W	0h	External Trigger Select. This selects which System Event is used if the input filter selects trigger. Triggers 0-15 are used to connect triggers generated by other timer modules. Refer to the SoC data sheet for details related to timer trigger sources. Triggers 16 and 17 are connected to event manager subscriber ports. Event lines 18-31 are reserved for future use. 0h = TRIGx = External trigger input from TIM x. 1h = TRIGx = External trigger input from TIM x. 2h = TRIGx = External trigger input from TIM x. 3h = TRIGx = External trigger input from TIM x. 4h = TRIGx = External trigger input from TIM x. 5h = TRIGx = External trigger input from TIM x. 6h = TRIGx = External trigger input from TIM x. 7h = TRIGx = External trigger input from TIM x. 8h = TRIGx = External trigger input from TIM x. 9h = TRIGx = External trigger input from TIM x. Ah = TRIGx = External trigger input from TIM x. Bh = TRIGx = External trigger input from TIM x. Ch = TRIGx = External trigger input from TIM x. Dh = TRIGx = External trigger input from TIM x. Eh = TRIGx = External trigger input from TIM x. Fh = TRIGx = External trigger input from TIM x. 10h = TRIG_SUBx = External trigger input from subscriber port x. 11h = TRIG_SUBx = External trigger input from subscriber port x.

15.3.58 RC (Offset = 18B4h) [Reset = 00000000h]

RC is shown in Figure 15-98 and described in Table 15-85.

Return to the Summary Table.

Repeat counter is to reduce interrupt overhead. The repeat counter provides the mechanism to suppress unnecessary interrupts;
reducing the number of interrupts generated by each event type to 1 for the program number of periods. Specifically, the repeat timer may suppress Load, Compare (up/down, normal/shadow), and Zero events.

Figure 15-98. RC

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
RESERVED																								RC							
R-0h																								R-0h							

Table 15-85. RC Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	RC	R	0h	Repeat Counter Value 0h = Minimum value FFh = Maximum Value

15.3.59 RCLD (Offset = 18B8h) [Reset = 00000000h]

RCLD is shown in Figure 15-99 and described in Table 15-86.

Return to the Summary Table.

The load register value is transferred to the counter when the counter load input is asserted.

Figure 15-99. RCLD

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
RESERVED																								RCLD							
R-0h																								R/W-0h							

Table 15-86. RCLD Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	RCLD	R/W	0h	Repeat Counter Load Value This field provides the value loaded into the repeat counter at a load event following the repeat counter value equaling 0. 0h = Minimum value FFh = Maximum Value

15.3.60 QDIR (Offset = 18BCh) [Reset = 00000000h]

QDIR is shown in Figure 15-100 and described in Table 15-87.

Return to the Summary Table.

The QDIR register provides the direction of count which is intended for use when operating the counter in QEI.

Figure 15-100. QDIR

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

Table 15-87. QDIR Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	DIR	R	0h	Direction of count 0h = Down (Phase B leads Phase A) 1h = Up (Phase A leads Phase B)

15.3.61 FCTL (Offset = 18D0h) [Reset = 00000000h]

FCTL is shown in Figure 15-101 and described in Table 15-88.

Return to the Summary Table.

The FCTL register controls the fault inputs, fault detection and error handling behavior.

Figure 15-101. FCTL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		FSENEXT2	FSENEXT1	FSENEXT0	FSENAC2	FSENAC1	FSENAC0
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TFIM	RESERVED		FL		FI	RESERVED	FIEN
R/W-0h	R-0h		R/W-0h		R/W-0h	R-0h	R/W-0h

Table 15-88. FCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	
13	FSENEXT2	R/W	0h	Specifies whether the external fault pin2 high/low is treated as fault condition. 0h = Fault Input is active low. 1h = Fault Input is active high.
12	FSENEXT1	R/W	0h	Specifies whether the external fault pin1 high/low is treated as fault condition. 0h = Fault Input is active low. 1h = Fault Input is active high.
11	FSENEXT0	R/W	0h	Specifies whether the external fault pin0 high/low is treated as fault condition. 0h = Fault Input is active low. 1h = Fault Input is active high.
10	FSENAC2	R/W	0h	Specifies whether the COMP2 output high/low is treated as fault condition. 0h = Fault Input is active low. 1h = Fault Input is active high.
9	FSENAC1	R/W	0h	Specifies whether the COMP1 output high/low is treated as fault condition. 0h = Fault Input is active low. 1h = Fault Input is active high.
8	FSENAC0	R/W	0h	Specifies whether the COMP0 output high/low is treated as fault condition. 0h = Fault Input is active low. 1h = Fault Input is active high.
7	TFIM	R/W	0h	Trigger Fault Input Mask Specifies whether the selected trigger participates as a fault input. If enabled and the trigger asserts, the trigger is treated as a fault. 0h = Selected trigger does not participate in fault condition generation 1h = Selected trigger participates in fault condition generation
6-5	RESERVED	R	0h	

Table 15-88. FCTL Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	FL	R/W	0h	Fault Latch mode Specifies whether the fault condition is latched and configures the latch clear conditions. 0h = Overall fault condition is not dependent on the F bit in RIS 1h = Overall fault condition is dependent on the F bit in RIS 2h = Fault condition is latched. Fault condition is cleared on a zero event if the fault input is 0. 3h = Fault condition is latched. Fault condition is cleared on a load event if the fault input is 0.
2	FI	R/W	0h	Fault Input Specifies whether the overall fault condition is dependent on the sensed fault pin. 0h = Overall Fault condition is not dependent on sensed input. 1h = Overall Fault condition is dependent on sensed input.
1	RESERVED	R	0h	
0	FIEN	R/W	0h	Fault Input Enable This bit enables the input for fault detection. 0h = Fault Input Disabled 1h = Fault Input Enabled

15.3.62 FIFCTL (Offset = 18D4h) [Reset = 00000000h]

FIFCTL is shown in Figure 15-102 and described in Table 15-89.

Return to the Summary Table.

The FIFCTL register controls the filtering for the fault input.

Figure 15-102. FIFCTL

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			FILTEN	CPV	RESERVED	FP	
R-0h			R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 15-89. FIFCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4	FILTEN	R/W	0h	Filter Enable This bit controls whether the input is filtered by the input filter or bypasses to go directly to the optional pre-scale filter and then to the edge detect. 0h = Bypass 1h = Filtered.
3	CPV	R/W	0h	Consecutive Period/Voting Select This bit controls whether the input filter uses a stricter consecutive period count or majority voting. 0h = Consecutive Periods. The input must be at a specific logic level for the period defined by FP before it is passed to the filter output. 1h = Voting. The filter ignores one clock of opposite logic over the filter period, meaning that during FP samples of the input, up to 1 sample may be of an opposite logic value (glitch) without affecting the output
2	RESERVED	R	0h	
1-0	FP	R/W	0h	Filter Period This field specifies the sample period for the input filter. The input is sampled for FP timer clocks during filtering. 0h = Filter Period 3 1h = Filter Period 5 2h = Filter Period 8

Chapter 16
WWDT

The window watchdog timer (WWDT) supervises code execution. If the application software does not successfully reset the window watchdog within the programmed open time window, the window watchdog generates a reset.

16.1 WWDT Overview

The primary function of the window watchdog timer (WWDT) is to initiate a reset when correct operation of the device has failed due to an unexpected software or system delay. The WWDT can be programmed with a predefined time window within which the application software must restart the timer, indicating that application execution is proceeding normally. If application software fails to restart the timer within the specified window, the WWDT will issue a WWDT violation signal to SYSCTL to generate a reset.

If watchdog functionality is not required in an application, the WWDT can also be configured as a basic system interval timer which is capable of generating periodic maskable interrupts to the CPU.

Key features of the WWDT include:

- A 25-bit counter with closed and open window
- Counter driven from LFCLK (fixed 32kHz clock path) with a programmable clock divider
- Eight selectable watchdog timer periods
- Optional automatic suspension of counter when operating in low power modes
- Support for standard window watchdog mode or interval timer (nonwatchdog) mode

Devices may have 1 or 2 WWDT instances. A WWDT0 violation generates a BOOTRST, which resets the peripheral and CPU state and also causes the boot configuration routine (BCR) to run. A WWDT1 violation generates a SYSRST, which resets the peripheral and CPU state but does not trigger execution of the BCR. As such, WWDT1 is well suited for recovering from execution stalls that result from software execution, while WWDT0 is well suited for catching larger issues such as a corrupted trim value, at the expense of a longer reset time.

16.1.1 Watchdog Mode

In watchdog mode, the WWDT is configured to count up to the specified WWDT period. The WWDT counter must be restarted with the configured open window of the WWDT period, or the WWDT will assert a WWDT violation to SYSCTL and a reset will be generated.

The window watchdog timer supports detecting both a "too late" response as well as a "too early" response through the use of an optional closed window, as shown in Figure 16-1. The WWDT period consists of a closed window period and an open window period. The closed window period begins first, followed by the open window period. The WWDT can only be restarted during the open window period. An attempt to restart the WWDT during the closed window period results in a violation. Following the closed window, if the WWDT is not restarted before the end of the open window, the WWDT period expires and a violation is also generated.

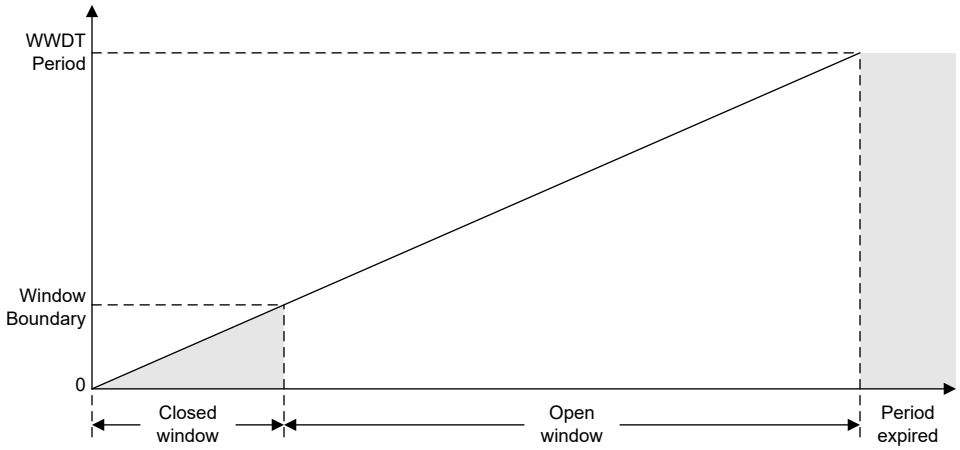


Figure 16-1. WWDT Functionality

If the closed window functionality is not desired, it can be disabled (set to 0%), giving traditional watchdog timer functionality where the WWDT can be reset any time before the WWDT period expires.

16.1.2 Interval Timer Mode

The WWDT can be used in interval timer mode to generate periodic interrupts to the CPU when not using the watchdog functionality. When used in interval timer mode, a WWDT interrupt is generated when the WWDT period expires, or when an incorrect password is applied to the WWDT control registers.

16.2 WWDT Operation

The WWDT must be enabled before being configured for use through the PWREN register (see peripheral power enable).

The WWDT is configured through the WWDTCTL0 and WWDTCTL1 registers. The registers are password protected. Any register access (read or write) must be a 32-bit access. Write access must also include the corresponding password in the most significant byte (0xC9 for WWDTCTL0, and 0xBE for WWDTCTL1). Attempting a register write without the correct password, or attempting a write with an access other than a 32-bit access generates a WWDT violation to SYSCTL. The password byte always reads as 0x00.

The WWDT is disabled and cleared after a SYSRST. The WWDTCTL0 register sets the static configuration of the WWDT, including: the clock divider, the timer period, the two closed window percentages, the timer mode (WWDT or interval), and the stop-in-sleep status. The first write (with a key match) to the WWDTCTL0 register enables the WWDT. Once the WWDT is enabled, the WWDTCTL0 register becomes write protected. Any attempt to write to the WWDTCTL0 register after the WWDT is enabled generates a WWDT violation to SYSCTL. The RUN bit in the WWDTSTAT register indicates that the WWDT is running.

Figure 16-2 shows the WWDT functional block diagram.

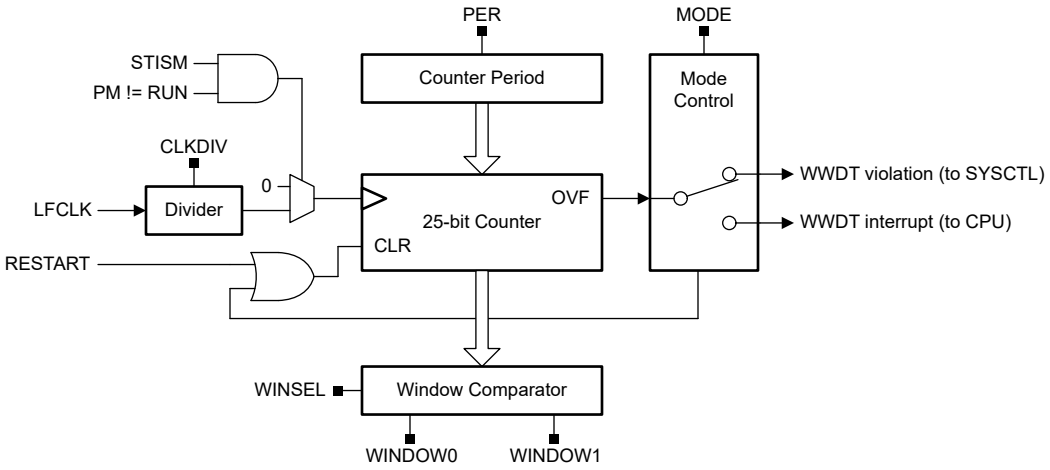


Figure 16-2. WWDT Diagram

16.2.1 Mode Selection

The WWDT operating mode (watchdog mode or interval timer mode) is selected by the MODE bit in the WWDTCTL0 register. Watchdog mode is the default mode (MODE cleared). Setting the MODE bit configures the WWDT for interval mode.

When the WWDT is in watchdog mode, the WWDT counter must be restarted within the open window period by writing the RESTART value (0x000000A7) to the WWDTCNTRST register. After a reset or restart, the WWDT counter will restart from zero. A failure to restart the WWDT within the open window or an attempt to restart the WWDT counter during the closed window will generate a WWDT violation to SYSCTL. Writing any value other than the RESTART value to the WWDTCNTRST register also generates a WWDT violation.

When the WWDT is in interval mode, the timer acts as an interval timer, generating WWDT interrupts to the CPU as specified by the WWDT period. As soon as the WWDT is enabled in interval mode, the WWDT interval timer interrupt will be asserted after the expiration of the timer. It is not necessary to restart the WWDT in interval timer mode.

16.2.2 Clock Configuration

The WWDT runs from the 32kHz low-frequency clock (LFCLK). A clock divider supports dividing the input clock from /1 (no divide) to /8 (divide-by-8) using the CLKDIV field in the WWDTCTL0 register. The default CLKDIV setting is 0x03 (32kHz divided by 4, or 8kHz).

By running from the LFCLK, the WWDT time base is independent of the main clock (MCLK) and CPU clock (CPUCLK) time base, provided that these clocks are not also configured to be running from the LFCLK. While the time base may be considered as independent and derived from a separate oscillator source, LFCLK edges are synchronized to the MCLK before sourcing the WWDT to enable simple access to the memory-mapped registers from application software. A simplified view of the clock scheme is given in Figure 16-3. In Figure 16-3, the internal LFOSC is configured to set the LFCLK rate, and the internal SYSOSC sets the MCLK/CPUCLK rate. Clock selection muxes and dividers are excluded from the figure to simplify the view; the complete clock tree is provided in Section 2.3.3.

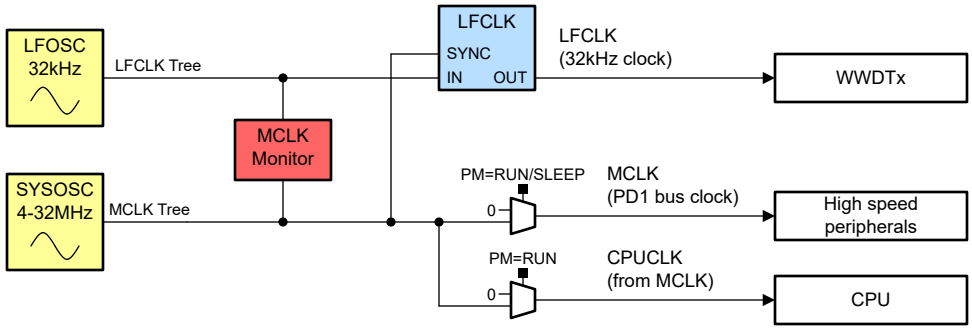


Figure 16-3. WWDT Simplified Clock Source Tree

In the event that the MCLK fails and synchronization of the LFCLK to the MCLK is lost, this failure may be detected by enabling the continuous MCLK monitor. When the MCLK monitor is enabled, a loss of MCLK is always a catastrophic failure which generates a BOOTRST within 12 LFCLK cycles. As a result, a loss of the MCLK tree, and corresponding loss of synchronization, does not prevent a BOOTRST from being generated.

Period Selection

The WWDT has a 25-bit counter which is initially stopped after a SYSRST. The WWDT period (total time interval) is set by the PER field in the WWDTCTL0 register. The total WWDT period is calculated as follows:

$$T_{WWDT} = (CLKDIV + 1) * PER_{COUNT} / 32768Hz \quad (16)$$

The total timer count PER_{COUNT} is selected to be one of 8 possible period count values, with the encoding given in Table 16-1.

Table 16-1. WWDT Period Total Timer Count

PER	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
PER_{COUNT}	2^{25}	2^{21}	2^{18}	2^{15}	2^{12}	2^{10}	2^{8}	2^{6}

The combination of the period selection (PER) and clock divider (CLKDIV) enable a wide range of WWDT periods, from as short as 1.95ms to as long as 136.53 minutes. Table 16-2 gives all possible WWDT periods for a given combination of PER and CLKDIV.

Table 16-2. WWDT Period Timing Options

CLKDIV	PER							
	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
	min	*min*	*s*	*s*	*s*	*ms*	*ms*	*ms*
0x0 (/1)	17.07	1.07	8.00	1.00	0.13	31.25	7.81	1.95

Table 16-2. WWDT Period Timing Options (continued)

CLKDIV	PER							
	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
	min	*min*	*s*	*s*	*s*	*ms*	*ms*	*ms*
0x1 (/2)	34.13	2.13	16.00	2.00	0.25	62.50	15.63	3.91
0x2 (/3)	51.20	3.20	24.00	3.00	0.38	93.75	23.44	5.86
0x3 (/4)	68.27	4.27	32.00	4.00	0.50	125.00	32.25	7.81
0x4 (/5)	85.33	5.33	40.00	5.00	0.63	156.25	39.06	9.77
0x5 (/6)	102.40	6.40	48.00	6.00	0.75	187.50	46.88	11.72
0x6 (/7)	119.47	7.47	56.00	7.00	0.88	218.75	54.69	13.67
0x7 (/8)	136.53	8.53	64.00	8.00	1.00	250.00	62.50	15.63

Synchronization Delay

When starting or re-starting the WWDT counter, a maximum synchronization delay of one 32kHz clock cycle (30.5µs) can occur before the WWDT counter begins counting from zero. The periods given in Table 16-2 do not include this synchronization delay.

Closed Window Selection

Configuration of two closed window periods is supported by setting the WINDOW0 and WINDOW1 fields in the WWDTCTL0 register. The WINSEL bit in the WWDTCTL1 register determines the active window (either WINDOW0 or WINDOW1). Either window can be set to one of 8 possible window settings.

Table 16-3. WWDT Window Options

WINDOW	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
Closed window	0%	12.5%	18.75%	25%	50%	75%	81.25%	87.5%

Setting a WINDOWx value to 0x0 (0% closed, 100% open) is equivalent to disabling the window function of the WWDT. In this configuration, the WWDT can be restarted at any point during the WWDT period.

The active window selection can be changed after the WWDT has been enabled. When the WWDT is restarted by writing to the WWDTCNTRST register, the closed window selection (WINSEL) must not be changed for at least four 32kHz clock cycles (≈122µs).

16.2.3 Low-Power Mode Behavior

The WWDT counter can be configured to continue counting when the device is in a low-power mode (CPU is disabled) or to continue to run when the device is in a low-power mode.

The STISM bit in the WWDTCTL0 register controls if the WWDT counter stops counting in sleep mode. By default, the STISM bit is cleared, indicating that the WWDT continues to count in low-power modes. To stop the WWDT from counting in low-power modes, set the STISM bit when loading the WWDTCTL0 configuration to start the WWDT. In this case, when the low-power mode is exited and the CPU returns to operation, the WWDT counter resumes counting from the same value it held before entering the low-power mode.

16.2.4 Debug Behavior

The WWDT can be configured to stop counting or continue counting when the CPU is halted for debug by the debug subsystem. By default, the WWDT stops counting when the CPU is halted for debug and the device is in a debug state. To allow the WWDT to continue to free run when the CPU is stopped for debug, set the FREE bit in the PDBGCTL register.

16.2.5 WWDT Events

The WWDT module contains one event publisher and no event subscribers. One event publisher (CPU_INT) manages WWDT interrupt requests (IRQs) to the CPU subsystem through a static event route.

Table 16-4 lists the WWDT events.

Table 16-4. WWDT Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU Interrupt Event	Publisher	WWDT	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from WWDT to CPU

16.2.5.1 CPU Interrupt Event Publisher (CPU_INT)

The WWDT module provides one interrupt source which can be configured to source a CPU interrupt event. The WWDT interrupt conditions are given in Table 16-5.

Table 16-5. WWDT CPU Interrupt Conditions (CPU_INT)

Index (IIDX)	Name	Description
0	INTTIM	Indicates that the WWDT interval timer period has expired

The CPU interrupt event configuration is managed with the CPU_INT event management registers. See Section 6.2.5 for guidance on configuring the Event registers for CPU interrupts.

16.3 WWDT Registers

Table 16-6 lists the memory-mapped registers for the WWDT registers. All register offset addresses not listed in Table 16-6 should be considered as reserved locations and the register contents should not be modified.

Table 16-6. WWDT Registers

Offset	Acronym	Register Name	Section
800h	PWREN	Power enable	Section 16.3.1
804h	RSTCTL	Reset Control	Section 16.3.2
814h	STAT	Status Register	Section 16.3.3
1018h	PDBGCTL	Peripheral Debug Control	Section 16.3.4
1020h	IIDX	Interrupt index	Section 16.3.5
1028h	IMASK	Interrupt mask	Section 16.3.6
1030h	RIS	Raw interrupt status	Section 16.3.7
1038h	MIS	Masked interrupt status	Section 16.3.8
1040h	ISET	Interrupt set	Section 16.3.9
1048h	ICLR	Interrupt clear	Section 16.3.10
10E0h	EVT_MODE	Event Mode	Section 16.3.11
10FCh	DESC	Module Description	Section 16.3.12
1100h	WWDTCTL0	Window Watchdog Timer Control Register 0	Section 16.3.13
1104h	WWDTCTL1	Window Watchdog Timer Control Register 0	Section 16.3.14
1108h	WWDTCNTRST	Window Watchdog Timer Counter Reset Register	Section 16.3.15
110Ch	WWDTSTAT	Window Watchdog Timer Status Register	Section 16.3.16

Complex bit access types are encoded to fit into small table cells. Table 16-7 shows the codes that are used for access types in this section.

Table 16-7. WWDT Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
K	K	Write protected by a key
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

16.3.1 PWREN Register (Offset = 800h) [Reset = 00000000h]

PWREN is shown in Figure 16-4 and described in Table 16-8.

Return to the Table 16-6.

Register to control the power state

Figure 16-4. PWREN Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R/W-							K-0h

Table 16-8. PWREN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow Power State Change 26h = KEY to allow write access to this register
23-1	RESERVED	R/W	0h	

0	ENABLE	K	0h	Enable the power
				Note: For safety devices the power cannot be disabled once enabled.
				#none# must be set to 26h to write to this bit.
				0h = Disable Power
				1h = Enable Power

16.3.2 RSTCTL Register (Offset = 804h) [Reset = 00000000h]

RSTCTL is shown in Figure 16-5 and described in Table 16-9.

Return to the Table 16-6.

Register to control reset assertion and de-assertion

Figure 16-5. RSTCTL Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						RESETSTKYCLR	RESETASSERT
W-0h						WK-0h	WK-0h

Table 16-9. RSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	Unlock key B1h = KEY to allow write access to this register
23-2	RESERVED	W	0h	

1	RESETSTKYCLR	WK	0h	Clear #none#
				#none# must be set to B1h to write to this bit.
				0h = Writing 0 has no effect
				1h = Clear reset sticky bit

0	RESETASSERT	WK	0h	Assert reset to the peripheral
				Note: For safety devices a watchdog reset by software is not possible.
				#none# must be set to B1h to write to this bit.
				0h = Writing 0 has no effect
				1h = Assert reset

16.3.3 STAT Register (Offset = 814h) [Reset = 00000000h]

STAT is shown in Figure 16-6 and described in Table 16-10.

Return to the Table 16-6.

peripheral enable and reset status

Figure 16-6. STAT Register

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

Table 16-10. STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	
16	RESETSTKY	R	0h	This bit indicates, if the peripheral was reset, since this bit was cleared by RESETSTKYCLR in the RSTCTL register 0h = The peripheral has not been reset since this bit was last cleared by RESETSTKYCLR in the RSTCTL register 1h = The peripheral was reset since the last bit clear
15-0	RESERVED	R	0h	

16.3.4 PDBGCTL Register (Offset = 1018h) [Reset = 00000000h]

PDBGCTL is shown in Figure 16-7 and described in Table 16-11.

Return to the Table 16-6.

This register can be used by the software developer to control the behavior of the peripheral relative to the 'Core Halted' input

Figure 16-7. PDBGCTL Register

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

Table 16-11. PDBGCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	FREE	R/W	0h	Free run control 0h = The peripheral freezes functionality while the Core Halted input is asserted and resumes when it is deasserted. 1h = The peripheral ignores the state of the Core Halted input

16.3.5 IIDX Register (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 16-8 and described in Table 16-12.

Return to the Table 16-6.

This register provides the highest priority enabled interrupt index.

Figure 16-8. IIDX Register

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
RESERVED																											STAT					
R-0h																											R-0h					

Table 16-12. IIDX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	
4-0	STAT	R	0h	Module Interrupt Vector Value. This register provides the highest priority interrupt index. A read clears the corresponding interrupt flag in RIS and MISC. 0h = No interrupt pending 1h = Interval Timer Interrupt; Interrupt Flag: INTTIM; Interrupt Priority: Highest

16.3.6 IMASK Register (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 16-9 and described in Table 16-13.

Return to the Table 16-6.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS."

Figure 16-9. IMASK Register

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

Table 16-13. IMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	INTTIM	R/W	0h	Interval Timer Interrupt. 0h = Clear Interrupt Mask 1h = Set Interrupt Mask

16.3.7 RIS Register (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 16-10 and described in Table 16-14.

Return to the Table 16-6.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 16-10. RIS Register

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

Table 16-14. RIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	INTTIM	R	0h	Interval Timer Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred

16.3.8 MIS Register (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 16-11 and described in Table 16-15.

Return to the Table 16-6.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 16-11. MIS Register

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

Table 16-15. MIS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	INTTIM	R	0h	Interval Timer Interrupt. 0h = Interrupt did not occur 1h = Interrupt occurred

16.3.9 ISET Register (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 16-12 and described in Table 16-16.

Return to the Table 16-6.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 16-12. ISET Register

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

Table 16-16. ISET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	
0	INTTIM	W	0h	Interval Timer Interrupt. 0h = Writing 0 has no effect 1h = Set Interrupt

16.3.10 ICLR Register (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 16-13 and described in Table 16-17.

Return to the Table 16-6.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 16-13. ICLR Register

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

Table 16-17. ICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	W	0h	
0	INTTIM	W	0h	Interval Timer Interrupt. 0h = Writing 0 has no effect 1h = Clear Interrupt

16.3.11 EVT_MODE Register (Offset = 10E0h) [Reset = 00000001h]

EVT_MODE is shown in Figure 16-14 and described in Table 16-18.

Return to the Table 16-6.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 16-14. EVT_MODE Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-							
23	22	21	20	19	18	17	16
RESERVED							
R/W-							
15	14	13	12	11	10	9	8
RESERVED							
R/W-							
7	6	5	4	3	2	1	0
RESERVED						INT0_CFG	
R/W-						R-1h	

Table 16-18. EVT_MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1-0	INT0_CFG	R	1h	Event line mode select for event corresponding to none.INT_EVENT[0] 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

16.3.12 DESC Register (Offset = 10FCh) [Reset = 1F117010h]

DESC is shown in Figure 16-15 and described in Table 16-19.

Return to the Table 16-6.

This register identifies the peripheral and its exact version.

Figure 16-15. DESC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-1F11h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				INSTNUM				MAJREV				MINREV			
R-7h				R-0h				R-1h				R-0h			

Table 16-19. DESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	1F11h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness. 0h = Smallest value FFFFh = Highest possible value
15-12	FEATUREVER	R	7h	Feature Set for the module *instance* 0h = Smallest value Fh = Highest possible value
11-8	INSTNUM	R	0h	Instance Number within the device. This will be a parameter to the RTL for modules that can have multiple instances 0h = Smallest value Fh = Highest possible value
7-4	MAJREV	R	1h	Major rev of the IP 0h = Smallest value Fh = Highest possible value
3-0	MINREV	R	0h	Minor rev of the IP 0h = Smallest value Fh = Highest possible value

16.3.13 WWDTCTL0 Register (Offset = 1100h) [Reset = 00000043h]

WWDTCTL0 is shown in Figure 16-16 and described in Table 16-20.

Return to the Table 16-6.

Window Watchdog Timer Control 0 Register
NOTE: Write to this register is enabled after System Reset. The first successful write (key match) enables the Watchdog. When the watchdog is enabled all subsequent writes to this register activate the WWDT error signal to the ESM.

Figure 16-16. WWDTCTL0 Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED						STISM	MODE
R/W-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	WINDOW1			RESERVED	WINDOW0		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	PER			RESERVED	CLKDIV		
R/W-0h	R/W-4h			R/W-0h	R/W-3h		

Table 16-20. WWDTCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow write access to this register. Writing to this register with an incorrect key activates the WWDT error signal to the ESM. Read as 0. C9h (W) = KEY to allow write access to this register
23-18	RESERVED	R/W	0h	
17	STISM	R/W	0h	Stop In Sleep Mode. The functionality of this bit requires that POLICY.HWCEN = 0. If POLICY.HWCEN = 1 the WWDT resets during sleep and needs re-configuration. Note: This bit has no effect for the global Window Watchdog as Sleep Mode is not supported. 0h = The WWDT continues to function in Sleep mode. 1h = The WWDT stops in Sleep mode and resumes where it was stopped after wakeup.
16	MODE	R/W	0h	Window Watchdog Timer Mode 0h = Window Watchdog Timer Mode. The WWDT will generate a error signal to the ESM when following conditions occur: - Timer Expiration (Timeout) - Reset WWDT during the active window closed period - Keyword violation 1h = Interval Timer Mode. The WWDT acts as an interval timer. It generates an interrupt on timeout.
15	RESERVED	R/W	0h	

Table 16-20. WWDTCTL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-12	WINDOW1	R/W	0h	Closed window period in percentage of the timer interval. WWDTCTL1.WINSEL determines the active window setting (WWDTCTL0.WINDOW0 or WWDTCTL0.WINDOW1). 0h = 0% (No closed Window) 1h = 12.50% of the total timer period is closed window 2h = 18.75% of the total timer period is closed window 3h = 25% of the total timer period is closed window 4h = 50% of the total timer period is closed window 5h = 75% of the total timer period is closed window 6h = 81.25% of the total timer period is closed window 7h = 87.50% of the total timer period is closed window
11	RESERVED	R/W	0h	
10-8	WINDOW0	R/W	0h	Closed window period in percentage of the timer interval. WWDTCTL1.WINSEL determines the active window setting (WWDTCTL0.WINDOW0 or WWDTCTL0.WINDOW1). 0h = 0% (No closed Window) 1h = 12.50% of the total timer period is closed window 2h = 18.75% of the total timer period is closed window 3h = 25% of the total timer period is closed window 4h = 50% of the total timer period is closed window 5h = 75% of the total timer period is closed window 6h = 81.25% of the total timer period is closed window 7h = 87.50% of the total timer period is closed window
7	RESERVED	R/W	0h	
6-4	PER	R/W	4h	Timer Period of the WWDT. These bits select the total watchdog timer count. 0h = Total timer count is 2^{25} 1h = Total timer count is 2^{21} 2h = Total timer count is 2^{18} 3h = Total timer count is 2^{15} 4h = Total timer count is 2^{12} (default) 5h = Total timer count is 2^{10} 6h = Total timer count is 2^{8} 7h = Total timer count is 2^{6}
3	RESERVED	R/W	0h	
2-0	CLKDIV	R/W	3h	Module Clock Divider, Divide the clock source by CLKDIV+1. Divider values from /1 to /8 are possible. The clock divider is currently 4 bits. Bit 4 has no effect and should always be written with 0. 0h = Minimum value 7h = Maximum value

16.3.14 WWDTCTL1 Register (Offset = 1104h) [Reset = 00000000h]

WWDTCTL1 is shown in Figure 16-17 and described in Table 16-21.

Return to the Table 16-6.

Window Watchdog Timer Control 1 Register

Figure 16-17. WWDTCTL1 Register

31	30	29	28	27	26	25	24
KEY							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							WINSEL
R/W-0h							R/W-0h

Table 16-21. WWDTCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	W	0h	KEY to allow write access to this register. Writing to this register with an incorrect key activates the WWDT error signal to the ESM. Read as 0. BEh (W) = KEY to allow write access to this register
23-1	RESERVED	R/W	0h	
0	WINSEL	R/W	0h	Close Window Select 0h = In window mode field WINDOW0 of WDDTCTL0 defines the closed window size. 1h = In window mode field WINDOW1 of WDDTCTL0 defines the closed window size.

16.3.15 WWDTCNTRST Register (Offset = 1108h) [Reset = 00000000h]

WWDTCNTRST is shown in Figure 16-18 and described in Table 16-22.

Return to the Table 16-6.

Window Watchdog Timer Counter Restart Register

Figure 16-18. WWDTCNTRST Register

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
RESTART																															
R/W-0h																															

Table 16-22. WWDTCNTRST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RESTART	R/W	0h	Window Watchdog Timer Counter Restart Writing 00A7h to this register restarts the WWDT Counter. Writing any other value causes an error generation to the ESM. Read as 0. 0h = Minimum value FFFFFFFFh = Maximum value

16.3.16 WWDTSTAT Register (Offset = 110Ch) [Reset = 00000000h]

WWDTSTAT is shown in Figure 16-19 and described in Table 16-23.

Return to the Table 16-6.

Window Watchdog Timer Status Register
A write to this register has no effect.

Figure 16-19. WWDTSTAT Register

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

Table 16-23. WWDTSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	RUN	R	0h	Watchdog running status flag. 0h = Watchdog counter stopped. 1h = Watchdog running.

Chapter 17

Debug

The debug subsystem (DEBUGSS) is implemented in all MSPM0 devices. The DEBUGSS enables comprehensive debug of application software running on the processor during development by interfacing an external debug probe to the device systems through a serial wire debug (SWD) interface.

17.1 Overview

The debug subsystem (DEBUGSS) interfaces the serial wire debug (SWD) two-wire physical interface to multiple debug functions within the device. MSPM0 devices support debugging of processor execution, the device state, and the power state (through EnergyTrace technology). The DEBUGSS also provides a mailbox system for communicating with software through SWD.

Key features provided by the debug subsystem include:

- Two-wire (SWDIO, SWCLK) debug interface, compatible with both TI and 3rd party debug probes
  - On-chip pullup and pulldown resistors for SWDIO and SWCLK, respectively, enabled by default
  - Support for disabling SWD functions to use SWD pins as general-purpose input/output pins
  - Can wake the device from SHUTDOWN mode on valid SWD activity
- Debug of the processor
  - Run, halt, and step debug support
  - 4 hardware breakpoints (BPU)
  - 2 hardware watchpoints (DWT)
  - Unlimited software breakpoints
- Software-configurable peripheral behavior during processor debug
  - Ability to free run select peripherals through debug halt
  - Ability to halt select peripherals on a debug halt
  - Ability to request reset and mode changes to the PMCU
- Monitoring of CPU state through EnergyTrace technology
- Mailbox (DSSM) for passing data and control signals between the SWD interface and boot ROM (as well as application software)
- Support for various security features, including SWD lockout and password authenticated debugging

17.1.1 Debug Interconnect

The DEBUGSS architecture is given in Figure 17-1.

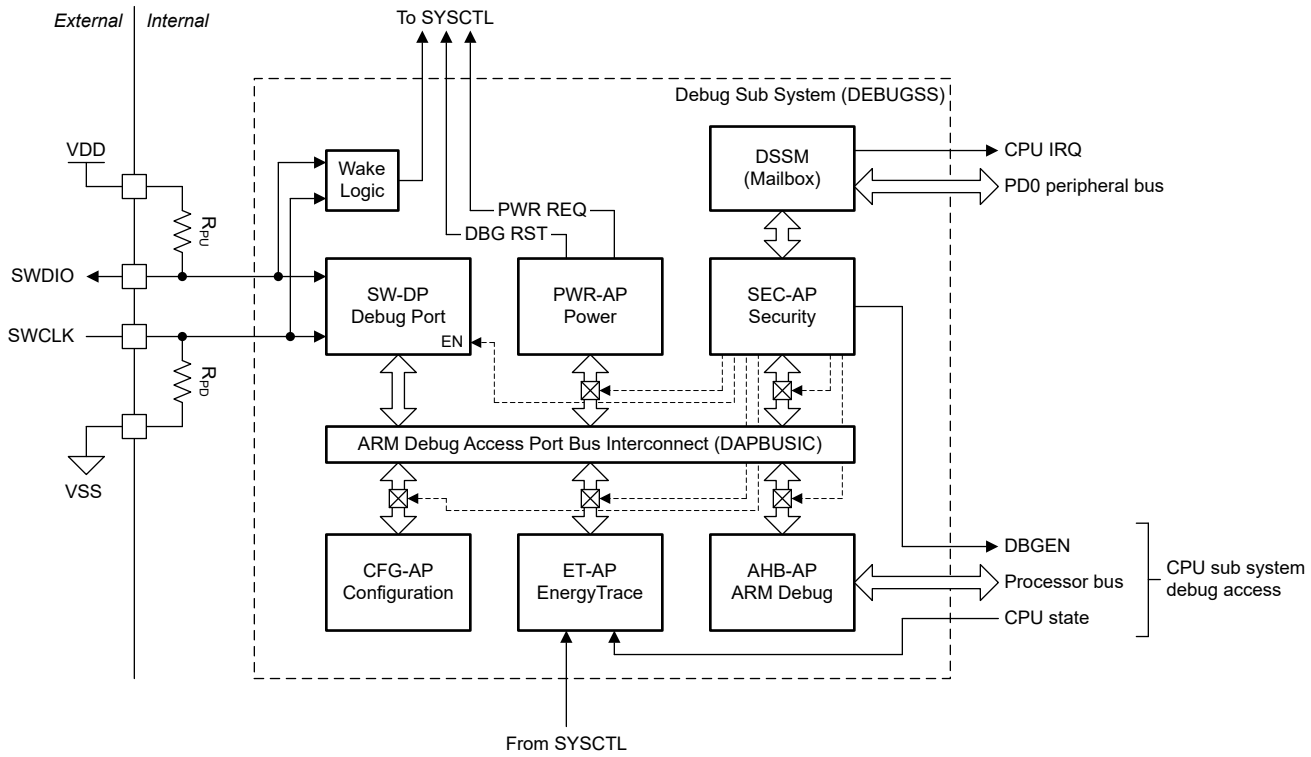


Figure 17-1. Debug Sub System Block Diagram

The SWD physical interface interacts with the Arm serial wire debug port (SW-DP) to gain access to the debug access port bus interconnect (DAPBUSIC) when the SW-DP is enabled. From TI, devices ship with the SW-DP enabled to allow SWD access to the device for development and production programming, but the SW-DP can be configured to be permanently disabled through the boot configuration policy (see Section 17.4).

The DAPBUSIC enables a debug probe to access one or more debug access ports. For a debug probe to be able to communicate with an access port, the SW-DP debug port must not be disabled by the device boot configuration policy, and the target access port must also not be disabled by the boot configuration policy. The available access ports are given in Section 17.1.3.

The SWD and SW-DP also contain signaling to the PMCU module to support debug-generated resets and operating mode changes (see Section 17.3).

17.1.2 Physical Interface

Debug connections to the device are supported through an Arm serial wire debug (SWD) compliant interface. The SWD interface requires two connections:

- A bidirectional data line (SWDIO) used to send data to, and receive data from, the device
- A unidirectional clock line (SWCLK) driven by the debug probe connecting to the device

The SWD interface uses the standard logic levels of the device for SWD communication. See the device-specific data sheet for input and output logic levels for a given supply voltage (VDD). A SWCLK frequency of up to 10MHz is supported by the DEBUGSS.

During SWD operation, the SWDIO line can be driven high or driven low by either the target device or the debug probe. As either device can drive the line, when ownership of the shared SWDIO line is switched between the device and the debug probe, undriven time slots are inserted as a part of the SWD protocol. The primary purpose of the pullup resistor on the SWDIO line, and the pulldown resistor on the SWCLK line, is to place the SWD pins into a known state when no debug probe is attached. A minimum resistance of 100kΩ is recommended by Arm. The internal pullup/pulldown resistors fulfill this requirement and external resistors are not required for correct operation of the SWD interface.

After a power-on reset (POR), MSPM0 devices configure the SWD pins in SWD mode with an internal pullup resistor enabled on the SWDIO line and an internal pulldown resistor enabled on the SWCLK line. If the device configuration has not permanently disabled all SWD access, then the SWD interface is enabled during the boot process and a debug probe can be connected to the DEBUGSS.

In the event that a device was configured by software to enter SHUTDOWN mode, and a debug probe is then connected to the SWD pins with SWCLK active, wakeup logic will trigger an exit from SHUTDOWN mode and cause a BOR. A debug connection can then be established to the DEBUGSS after the BOR completes.

Upon physical connection of a debug probe, a configuration sequence must be sent from the debug probe to the target device to initiate a valid SWD connection with the SW-DP. An invalid sequence will not wake the device from SHUTDOWN mode. Once the sequence is applied and the SWD connection is established, communication with enabled debug access points is possible and the application code is alerted through assertion of the DEBUGSS PWRUPIFG interrupt. When the debug probe is disconnected and the SWD connection is lost, the PWRDWNIFG interrupt is asserted.

It is possible for application software to disable the SWD interface in SYSCTL, freeing the IO to be used for general purpose IO functionality. Review Section 2.4.1.4 in SYSCTL for using the SWD pins for functionality other than SWD. Once software disables SWD functionality, it is not possible to re-enable it other than by triggering a POR. A POR will automatically re-enable the SWD functionality and put the SWD pins into SWD mode with pullup/pulldown resistors enabled. To re-gain debug access to a device which contains software that disables the SWD pins at startup, it is necessary to hold the device in a reset state with the NRST pin during a POR. This will prevent the application software from starting and will allow the debug probe to gain access to the device, at which point a mass erase DSSM command can be sent from the integrated development environment to the device via the debug probe to remove the application software which is disabling the SWD pins.

Note

BOR, BOOTRST, and SYSRST levels do reset the IOMUX logic, which will re-enable the pullup/pulldown resistors on the SWDIO/SWCLK pin. However, the SWD functionality remains disabled until the next POR. Because the device always powers up with the SWDIO pullup and SWCLK pulldown resistors enabled, the hardware design must accommodate this when using the SWD pins for functions other than SWD after startup. After reset, application software may disable the pullup/pulldown resistors in the IOMUX to free the SWD pins for other purposes.

17.1.3 Debug Access Ports

The debug access ports in the DEBUGSS are given in Table 17-1.

Table 17-1. DEBUGSS Access Port Listing

APSEL	AP	Port Description	Purpose
0x0	AHB-AP	MCPUSS debug access port	Debug of the processor and peripherals
0x1	CFG-AP	Configuration access port	Access device type information
0x2	SEC-AP	Security access port	Access the debug mailbox (DSSM)
0x3	ET-AP	EnergyTrace™ technology access port	Read the power state data from EnergyTrace technology for power aware debug
0x4	PWR-AP	Power access port	Configure the device power states (interfaces with PMCU/SYSCTL)

The AHB-AP, PWR-AP, and ET-AP provide the complete device debug functionality (processor debug, peripheral and memory bus access, power state control, and processor state). See Section 17.2 for more information.

The CFG-AP provides device information to the debug probe so that the debug probe can identify device characteristics, including the device part number and the device revision.

The SEC-AP provides access to the mailbox for communicating with software running on the device through SWD. See Section 17.5 for more information.

17.2 Debug Features

The DEBUGSS supports processor debug, processor trace, peripheral debug, and energy state debug.

17.2.1 Processor Debug

The Arm Cortex-M0+ processor supports a wide range of features to simplify debugging of application software during development. Key features supported by MSPM0 MCUs include:

- Ability to halt the processor through a assertion of a halt signal, a configured debug event (such as a hard fault entry or reset), or a BKPT instruction (for software breakpoints)
- Ability to step through instructions (with or without peripheral interrupts enabled)
- Ability to run through instructions (with or without peripheral interrupts enabled)
- Ability to read and write CPU registers when halted
- Ability to read exception information through the Cortex-M0+ system control space (SCS)
- Support for 4 hardware breakpoints
- Support for 2 hardware watchpoints
- Support for accessing the device memory map

17.2.1.1 Breakpoint Unit (BPU)

The breakpoint unit (BPU) provides 4 comparators which can be used to generate a debug event when the address of an instruction fetch matches the address programmed into the respective BPU comparator.

The BPU does not generate a debug event upon an address match for a data read or data write access.

Address matching is possible for half-word (16-bit) instructions and word (32-bit) instructions fetched from the CODE region (0x0000.0000 to 0x1FFF.FFFF).

If a debug scenario requires more than four breakpoints, software breakpoints can be used together with hardware breakpoints using the BKPT instruction. If debugging of code in the SRAM region is desired, hardware breakpoints are not available and software breakpoints must be inserted by the debug probe instead.

```
// Example of a breakpoint function in C (TI Arm CLANG compiler)
__BKPT(0);
```

17.2.1.2 Data Watchpoint and Trace Unit (DWT)

The data watchpoint and trace unit (DWT) provides 2 comparators which both support generating an event upon a data address match (watchpoint event) or an instruction address match (PC watchpoint event).

The DWT comparators support masking of the address, enabling an event to be generated when the processor attempts to access an address within a specified address range.

17.2.2 Peripheral Debug

In addition to processor debug, the DEBUGSS can be used to access the device memory map from the perspective of the processor. Thus, a connected debug probe can be used to read and write memory-mapped peripheral registers, the system SRAM, and the flash memory.

Certain peripherals support advanced debug configuration options. These options are configured by application software (or optionally, the debug probe) by setting/clearing various debug control bits in the memory map of a given peripheral. In general, the debug behavior of a particular peripheral is specified in the PDBGCTL register of each peripheral. Many peripherals offer the option of halting the functional clock to the peripheral when the processor is halted for debug, thus pausing the peripheral together with the processor (default configuration), or letting the peripheral run even when the processor is halted for debug.

For example, the WWDT peripheral supports the FREE bit in the PDBGCTL register. Setting the FREE bit in PDBGCTL for a WWDT causes the WWDT counter to run even if the processor is halted for debug.

17.2.3 EnergyTrace Technology

The DEBUGSS in MSPM0 devices supports EnergyTrace technology. EnergyTrace technology enables power profiling of MCU devices running application code. This is very useful when developing an application which must be optimized for low-power operation.

Development tools from Texas Instruments, including the MSPM0 LaunchPad development tools, support hardware energy measurement of the target MSPM0 over time through EnergyTrace charge counting. This mechanism enables a developer to obtain an energy usage profile for an application, based on real current measurements with a wide dynamic range.

To give context to the energy measurements made by the hardware development tools supporting EnergyTrace technology, MSPM0 MCUs also enable EnergyTrace+. EnergyTrace+ is a component of the DEBUGSS that lets the debug probe log the state of the processor (RUN, SLEEP) and the current program counter value while the device is running. This state information can be then overlaid with energy measurements to determine if the cause of high current is the processor running or some other activity on the device.

TI's Code Composer Studio integrated development environment provides out-of-the-box support for EnergyTrace energy measurement and EnergyTrace+ processor state logging with MSPM0 devices.

17.3 Behavior in Low Power Modes

The DEBUGSS supports maintaining a debug connection through SWD in all operating modes except SHUTDOWN.

Access to device memory and peripherals is possible in RUN mode and SLEEP mode, in which a debug probe can be actively connected to the AHB-AP access port to interface with the processor. In STOP and STANDBY modes, a debug connection can be established and/or maintained with the DEBUGSS, but not with the CPU debug access port.

In SHUTDOWN mode, any active debug connection is terminated as the debug logic is powered down with the device VCORE. While a debug connection to the DEBUGSS is not possible while the device is in SHUTDOWN mode, a debug probe can cause the device to exit SHUTDOWN mode by attempting to communicate with the SWD pins. The device will detect attempted SWD communication even when the device is in SHUTDOWN. If activity is detected, a SHUTDOWN exit is initiated and the device will transition through a BOR state, after which a debug connection can be made to the DEBUGSS through SWD.

The DEBUGSS functionality by operating mode is given in Table 17-2.

Table 17-2. DEBUGSS Functionality by Operating Mode

Capability	RUN	SLEEP	STOP	STANDBY	SHUTDOWN	NRST HOLD
Processor debug	Y	Y	N	N	N	N
Memory map access	Y	Y	N	N	N	N
Debug status through SW-DP	Y	Y	Y	Y	N	Y
Debug state maintained	Y	Y	Y	Y	N	N
Wake from SWD	-	-	-	-	Y	-

17.4 Restricting Debug Access

The debug subsystem supports several methods for restricting access to the device through the SWD interface. The debug access policy is determined by the user configuration specified in the NONMAIN flash region.

There are 3 levels of access control, given in Table 17-3. By default, products shipped from TI arrive in a "debug enabled" state where the device is fully open. This state is not recommended for production. For production, TI recommends changing the debug configuration to password protected or disabled.

Table 17-3. Debug Access Control

DEBUGSS Function	Debug Configuration		
	Debug Enabled (default)	Debug Enabled with Password	Debug Disabled
SW-DP (debug port)	EN	EN	DIS
CFG-AP	EN	EN	DIS
SEC-AP	EN	EN	DIS
ET-AP	EN	EN w/ PW	DIS
AHB-AP (CPU Debug)	EN	EN w/ PW	DIS

When debug is enabled with password, the debug access command together with the user-specified debug access password must be provided to the DEBUGSS mailbox by the debug probe, and a BOOTRST must be issued.

When debug is disabled, the SW-DP will be disabled during the boot process and any commands previously sent to the mailbox are ignored during boot. Following boot, any attempt to connect to the SW-DP is ignored.

It is possible to permanently lock debug access to the device by configuring the NONMAIN flash region to disable debug access while also configuring the NONMAIN flash region as statically write protected (locked). Locking the NONMAIN configuration has the added security of preventing the bootstrap loader (BSL) and application code from changing the debug security policy.

17.5 Mailbox (DSSM)

The debug subsystem mailbox (DSSM) enables a debug probe to pass messages to the target device through the SWD interface, as well as making it possible for the target device to return data to the debug probe.

The DSSM supports the following functions:

- Transmission of commands to the device during boot, including authenticating the debug probe for password-protected debug, mass erase, and factory reset operations

- Communicating with application software running on the target device when no other communication interface is present

Two 32-bit word data buffers are provided for TX data (debug probe to target device) and RX data (target device to debug probe). These data buffers are implemented as 32-bit memory-mapped registers in the DEBUGSS. In addition, TXCTL and RXCTL registers are provided for enabling flow control and indicating status of the mailbox.

Table 17-4. DSSM Register Functions

DSSM Register	Description	Debug Probe	Target Device	Actions
TX_DATA	Data buffer	RW	R	TXCTL.TRANSMIT is set on write by the debug probe, and cleared on a read by the target device; TXIFG is also set on a write by the debug probe
TXCTL	Flow control and status	RW	R	None
RX_DATA	Data buffer	R	RW	RXCTL.RECEIVE is set on write by the target device, and cleared on a read by the debug probe; RXIFG is also set on a write by the target device
RXCTL	Flow control and status	R	RW	None

The TXCTL and RXCTL registers provide TRANSMIT and RECEIVE flags, respectively, in the BIT0 position. The TRANSMIT bit is set in the TXCTL register when a debug probe writes data to the TX_DATA buffer register. The TRANSMIT flag will then remain set until the target device reads TX_DATA or a POR occurs. The RECEIVE flag is set in the RXCTL register when the target device writes data to the RX_DATA buffer register. The RECEIVE flag will then remain set until the debug probe reads the data from RX_DATA.

It is not possible for software running on the target device to write to TX_DATA, and it is also not possible for target software to clear the TRANSMIT flag other than by reading TX_DATA. The upper 31 bits of the TXCTL register contain generic flag bits which can be set or cleared by the debug probe to implement a protocol if desired. Only the debug probe can write to the TRANSMIT_FLAGS field in TXCTL.

In a similar way, only the target device software can write to RX_DATA and RXCTL. The debug probe cannot write to RX_DATA and it can only clear the RECEIVE flag in RXCTL by reading RX_DATA. BIT1 through BIT7 (0xFE) of RXCTL contains the RECEIVE_FLAGS field. Software on the target device can set or clear bits in the RECEIVE_FLAGS field to implement a protocol if desired. These flags can be read by the debug probe but can not be modified by the debug probe.

For a complete listing of DSSM commands which are supported by the boot configuration routine during device startup configuration, see Section 1.4.

17.5.1 DSSM Events

The DSSM contains one event publisher and no event subscribers. One event publisher (CPU_INT) manages DSSM interrupt requests (IRQs) to the CPU subsystem through a static event route.

The DSSM events are summarized in Table 17-5.

Table 17-5. DSSM Events

Event	Type	Source	Destination	Route	Configuration	Functionality
CPU Interrupt Event	Publisher	DEBUGSS	CPU Subsystem	Static route	CPU_INT registers	Fixed interrupt route from DEBUGSS to CPU

17.5.1.1 CPU Interrupt Event (CPU_INT)

The DSSM provides 4 interrupt sources which can be configured to source a CPU interrupt event. In order of decreasing interrupt priority, the CPU interrupt events from the DSSM are given in Table 17-6.

Table 17-6. DSSM CPU Interrupt Event Conditions (CPU_INT)

Index (IIDX)	Name	Description
0	TXIFG	Indicates that the TX_DATA buffer in the DSSM has received data.
1	RXIFG	Indicates that the data in RX_DATA buffer in the DSSM was read.
2	PWRUPIFG	Indicates that the DEBUGSS was started due to a debug probe attaching to the device.
3	PWRDWNIFG	Indicates that the DEBUGSS was stopped due to a debug probe disconnecting from the device.

The CPU interrupt event configuration is managed with the CPU_INT event management registers. See Section 6.2.5 for guidance on configuring the Event registers for CPU interrupts.

17.5.2 DEBUGSS Registers

Table 17-7 lists the memory-mapped registers for the DEBUGSS registers. All register offset addresses not listed in Table 17-7 should be considered as reserved locations and the register contents should not be modified.

Table 17-7. DEBUGSS Registers

Offset	Acronym	Register Name	Group	Section
1020h	IIDX	Interrupt index	CPU_INT	Go
1028h	IMASK	Interrupt mask	CPU_INT	Go
1030h	RIS	Raw interrupt status	CPU_INT	Go
1038h	MIS	Masked interrupt status	CPU_INT	Go
1040h	ISET	Interrupt set	CPU_INT	Go
1048h	ICLR	Interrupt clear	CPU_INT	Go
10E0h	EVT_MODE	Event Mode		Go
10FCh	DESC	Module Description		Go
1100h	TXD	Transmit data register		Go
1104h	TXCTL	Transmit control register		Go
1108h	RXD	Receive data register		Go
110Ch	RXCTL	Receive control register		Go
1200h	SPECIAL_AUTH	Special enable authorization register		Go
1210h	APP_AUTH	Application CPU0 authorization register		Go

Complex bit access types are encoded to fit into small table cells. Table 17-8 shows the codes that are used for access types in this section.

Table 17-8. DEBUGSS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WK	W K	Write Write protected by a key
Reset or Default Value		
-n		Value after reset or the default value

17.5.2.1 IIDX (Offset = 1020h) [Reset = 00000000h]

IIDX is shown in Figure 17-2 and described in Table 17-9.

Return to the Summary Table.

This register provides the highest priority enabled interrupt index. 0xFF means no event pending. Interrupt 0x0 is the highest priority, 0x1 next highest, and 0xFE is the least priority. The priority order is fixed. However, users can implement their own prioritization schemes using other registers that expose the full set of interrupts that have occurred.
On each read, only one interrupt is indicated. On a read, the current interrupt (highest priority) is automatically cleared by the hardware and the corresponding interrupt flag in the RIS and MIS are cleared as well. After a read from the CPU (not from the debug interface), the register must be updated with the next highest priority interrupt, if none are pending, then it displays 0xFF.

Figure 17-2. IIDX

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
RESERVED																								STAT							
R-0h																								R-0h							

Table 17-9. IIDX Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	STAT	R	0h	Interrupt index status 0h = No pending interrupt request 1h = TX interrupt 2h = RX interrupt 3h = Power-up interrupt. A debug session has started. 4h = Power-up interrupt. A debug session has started.

17.5.2.2 IMASK (Offset = 1028h) [Reset = 00000000h]

IMASK is shown in Figure 17-3 and described in Table 17-10.

Return to the Summary Table.

Interrupt Mask. If a bit is set, then corresponding interrupt is unmasked. Unmasking the interrupt causes the raw interrupt to be visible in IIDX, as well as MIS.

Figure 17-3. IMASK

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				PWRDWNIFG	PWRUPIFG	RXIFG	TXIFG
R/W-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-10. IMASK Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3	PWRDWNIFG	R/W	0h	Masks PWRDWNIFG in MIS register 0h = Interrupt is masked out 1h = Interrupt will request an interrupt service routine and corresponding bit in MIS will be set
2	PWRUPIFG	R/W	0h	Masks PWRUPIFG in MIS register 0h = Interrupt is masked out 1h = Interrupt will request an interrupt service routine and corresponding bit in MIS will be set
1	RXIFG	R/W	0h	Masks RXIFG in MIS register 0h = Interrupt is masked out 1h = Interrupt will request an interrupt service routine and corresponding bit in MIS will be set
0	TXIFG	R/W	0h	Masks TXIFG in MIS register 0h = Interrupt is masked out 1h = Interrupt will request an interrupt service routine and corresponding bit in MIS will be set

17.5.2.3 RIS (Offset = 1030h) [Reset = 00000000h]

RIS is shown in Figure 17-4 and described in Table 17-11.

Return to the Summary Table.

Raw interrupt status. Reflects all pending interrupts, regardless of masking. The RIS register allows the user to implement a poll scheme. A flag set in this register can be cleared by writing 1 to the ICLR register bit even if the corresponding IMASK bit is not enabled.

Figure 17-4. RIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PWRDWNIFG	PWRUPIFG	RXIFG	TXIFG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 17-11. RIS Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	PWRDWNIFG	R	0h	Raw interrupt status for PWRDWNIFG 0h = PWRUPIFG did not occur 1h = PWRUPIFG occurred
2	PWRUPIFG	R	0h	Raw interrupt status for PWRUPIFG 0h = PWRUPIFG did not occur 1h = PWRUPIFG occurred
1	RXIFG	R	0h	Raw interrupt status for RXIFG 0h = RXIFG did not occur 1h = RXIFG occurred
0	TXIFG	R	0h	Raw interrupt status for TXIFG 0h = TXIFG did not occur 1h = TXIFG occurred

17.5.2.4 MIS (Offset = 1038h) [Reset = 00000000h]

MIS is shown in Figure 17-5 and described in Table 17-12.

Return to the Summary Table.

Masked interrupt status. This is an AND of the IMASK and RIS registers.

Figure 17-5. MIS

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PWRDWNIFG	PWRUPIFG	RXIFG	TXIFG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 17-12. MIS Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	PWRDWNIFG	R	0h	Masked interrupt status for PWRDWNIFG 0h = PWRUPIFG did not request an interrupt service routine 1h = PWRUPIFG requests an interrupt service routine
2	PWRUPIFG	R	0h	Masked interrupt status for PWRUPIFG 0h = PWRUPIFG did not request an interrupt service routine 1h = PWRUPIFG requests an interrupt service routine
1	RXIFG	R	0h	Masked interrupt status for RXIFG 0h = RXIFG did not request an interrupt service routine 1h = RXIFG requests an interrupt service routine
0	TXIFG	R	0h	Masked interrupt status for TXIFG 0h = TXIFG did not request an interrupt service routine 1h = TXIFG requests an interrupt service routine

17.5.2.5 ISET (Offset = 1040h) [Reset = 00000000h]

ISET is shown in Figure 17-6 and described in Table 17-13.

Return to the Summary Table.

Interrupt set. Allows interrupts to be set by software (useful in diagnostics and safety checks). Writing a 1 to a bit in ISET will set the event and therefore the related RIS bit also gets set. If the interrupt is enabled through the mask, then the corresponding MIS bit is also set.

Figure 17-6. ISET

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED				PWRDWNIFG	PWRUPIFG	RXIFG	TXIFG
W-0h				W-0h	W-0h	W-0h	W-0h

Table 17-13. ISET Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	W	0h	
3	PWRDWNIFG	W	0h	Sets PWRDWNIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to PWRUPIFG is set
2	PWRUPIFG	W	0h	Sets PWRUPIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to PWRUPIFG is set
1	RXIFG	W	0h	Sets RXIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to RXIFG is set
0	TXIFG	W	0h	Sets TXIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to TXIFG is set

17.5.2.6 ICLR (Offset = 1048h) [Reset = 00000000h]

ICLR is shown in Figure 17-7 and described in Table 17-14.

Return to the Summary Table.

Interrupt clear. Write a 1 to clear corresponding Interrupt.

Figure 17-7. ICLR

31	30	29	28	27	26	25	24
RESERVED							
W-0h							
23	22	21	20	19	18	17	16
RESERVED							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
W-0h							
7	6	5	4	3	2	1	0
RESERVED				PWRDWNIFG	PWRUPIFG	RXIFG	TXIFG
W-0h				W-0h	W-0h	W-0h	W-0h

Table 17-14. ICLR Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	W	0h	
3	PWRDWNIFG	W	0h	Clears PWRDWNIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to PWRUPIFG is cleared
2	PWRUPIFG	W	0h	Clears PWRUPIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to PWRUPIFG is cleared
1	RXIFG	W	0h	Clears RXIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to RXIFG is cleared
0	TXIFG	W	0h	Clears TXIFG in RIS register 0h = Writing a 0 has no effect 1h = RIS bit corresponding to TXIFG is cleared

17.5.2.7 EVT_MODE (Offset = 10E0h) [Reset = 00000001h]

EVT_MODE is shown in Figure 17-8 and described in Table 17-15.

Return to the Summary Table.

Event mode register. It is used to select whether each line is disabled, in software mode (software clears the RIS) or in hardware mode (hardware clears the RIS)

Figure 17-8. EVT_MODE

31	30	29	28	27	26	25	24
RESERVED							
R-							
23	22	21	20	19	18	17	16
RESERVED							
R-							
15	14	13	12	11	10	9	8
RESERVED							
R-							
7	6	5	4	3	2	1	0
RESERVED						INT0_CFG	
R-						R-1h	

Table 17-15. EVT_MODE Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1-0	INT0_CFG	R	1h	Event line mode select for peripheral events 0h = The interrupt or event line is disabled. 1h = The interrupt or event line is in software mode. Software must clear the RIS. 2h = The interrupt or event line is in hardware mode. The hardware (another module) clears automatically the associated RIS flag.

17.5.2.8 DESC (Offset = 10FCh) [Reset = 03400000h]

DESC is shown in Figure 17-9 and described in Table 17-16.

Return to the Summary Table.

This register identifies the peripheral and its exact version.

Figure 17-9. DESC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODULEID															
R-340h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEATUREVER				INSTNUM				MAJREV				MINREV			
R-0h				R-0h				R-0h				R-0h			

Table 17-16. DESC Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MODULEID	R	340h	Module identification contains a unique peripheral identification number. The assignments are maintained in a central database for all of the platform modules to ensure uniqueness.
15-12	FEATUREVER	R	0h	Feature Set for the module *instance*
11-8	INSTNUM	R	0h	Instance Number within the device. This will be a parameter to the RTL for modules that can have multiple instances
7-4	MAJREV	R	0h	Major rev of the IP
3-0	MINREV	R	0h	Minor rev of the IP

17.5.2.9 TXD (Offset = 1100h) [Reset = 00000000h]

TXD is shown in Figure 17-10 and described in Table 17-17.

Return to the Summary Table.

This register is used for data transfers from external debug tools to the DSSM module. The register is written by the debug tool and read by the CPU.

Figure 17-10. TXD

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
TX_DATA																															
R-0h																															

Table 17-17. TXD Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX_DATA	R	0h	Contains data written by an external debug tool to the SEC-AP TXDATA register

17.5.2.10 TXCTL (Offset = 1104h) [Reset = 00000000h]

TXCTL is shown in Figure 17-11 and described in Table 17-18.

Return to the Summary Table.

Transmit control register

Figure 17-11. TXCTL

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

Table 17-18. TXCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-1	TRANSMIT_FLAGS	R	0h	Generic TX flags that can be set by external debug tool. Functionality is defined by SW.
0	TRANSMIT	R	0h	Indicates data request in DSSM.TXD, set on write via Debug AP to DSSM.TXD. A read of the DSSM.TXD register by SW will clear the TX field. The tool can check that TXD is empty by reading this field. 0h = TXD is empty 1h = TXD is full

17.5.2.11 RXD (Offset = 1108h) [Reset = 00000000h]

RXD is shown in Figure 17-12 and described in Table 17-19.

Return to the Summary Table.

Receive data register. This register contains the data written by the CPU.
This data is read by external debug tool.

Figure 17-12. RXD

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
RX_DATA																															
R/W-0h																															

Table 17-19. RXD Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RX_DATA	R/W	0h	Contains data written by SM/OW.

17.5.2.12 RXCTL (Offset = 110Ch) [Reset = 00000000h]

RXCTL is shown in Figure 17-13 and described in Table 17-20.

Return to the Summary Table.

Receive control register

Figure 17-13. RXCTL

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RECEIVE_FLAGS							RECEIVE
R/W-0h							R-0h

Table 17-20. RXCTL Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	
7-1	RECEIVE_FLAGS	R/W	0h	Generic RX flags that can be set by SW and read by external debug tool. Functionality is defined by SW.
0	RECEIVE	R	0h	Indicates SW write to the DSSM.RXD register. A read of the DSSM.RXD register by SWD Access Port will clear the RX field. 0h = RXD empty 1h = RXD full

17.5.2.13 SPECIAL_AUTH (Offset = 1200h) [Reset = 00000013h]

SPECIAL_AUTH is shown in Figure 17-14 and described in Table 17-21.

Return to the Summary Table.

This register is used to control ET-AP, DFT-TAP, SWD, CFG-AP and SEC-AP.

Figure 17-14. SPECIAL_AUTH

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	PWRAPEN	AHBAPEN	CFGAPEN	ETAPEN	DFTAPEN	SWDPORTEN	SECAPEN
R-0h	R-0h	R-0h	R-1h	R-0h	R-0h	R-1h	R-1h

Table 17-21. SPECIAL_AUTH Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	
6	PWRAPEN	R	0h	An active high input. When asserted (and SWD access is also permitted), the debug tools can then access the PWR-AP to power and reset state of the CPU. When deasserted, a DAPBUS firewall will isolate the AP and prevent access. 0h = Disable PWR-AP 1h = Enable PWR-AP
5	AHBAPEN	R	0h	Disabling / enabling debug access to the M0+ Core via the AHB-AP DAP bus isolation. 0h = Disable AHB-AP 1h = Enable AHB-AP
4	CFGAPEN	R	1h	An active high input. When asserted (and SWD access is also permitted), the debug tools can use the Config-AP to read device configuration information. When deasserted, a DAPBUS firewall will isolate the AP and prevent access to the Config-AP. 0h = Disable CFG-AP 1h = Enable CFG-AP
3	ETAPEN	R	0h	An active high input. When asserted (and SWD access is also permitted), the debug tools can then access an ET-AP external to the DebugSS lite. When deasserted, a DAPBUS firewall will isolate the AP and prevent access. 0h = Disable ET+ -AP 1h = Enable ET+ -AP
2	DFTAPEN	R	0h	An active high input. When asserted (and SWD access is also permitted), the debug tools can then access the DFT-AP external to the DebugSS lite. When deasserted, a DAPBUS firewall will isolate the AP and prevent access. 0h = Disable DFT-TAP 1h = Enable DFT-TAP
1	SWDPORTEN	R	1h	When asserted, the SW-DP functions normally. When deasserted, the SW-DP effectively disables all external debug access. 0h = Disable SWD port 1h = Enable SWD port

Table 17-21. SPECIAL_AUTH Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SECAPEN	R	1h	An active high input. When asserted (and SWD access is also permitted), the debug tools can use the Security-AP to communicate with security control logic. When deasserted, a DAPBUS firewall will isolate the AP and prevent access to the Security-AP. 0h = Disable SEC-AP 1h = Enable SEC-AP

17.5.2.14 APP_AUTH (Offset = 1210h) [Reset = 00000000h]

APP_AUTH is shown in Figure 17-15 and described in Table 17-22.

Return to the Summary Table.

This register is used to control DBGEN, NIDEN, SPIDEN, and SPNIDEN of Application CPU0. DBGEN, NIDEN are further processed by DSW based on Active and Debug IPF ID.

Figure 17-15. APP_AUTH

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SPNIDEN	SPIDEN	NIDEN	DBGEN
R-0h				R-0h	R-0h	R-0h	R-0h

Table 17-22. APP_AUTH Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	SPNIDEN	R	0h	Secure noninvasive debug enable. 0h = Invasive debug disabled 1h = Invasive debug enabled
2	SPIDEN	R	0h	Secure invasive debug enable. 0h = Invasive debug disabled 1h = Invasive debug enabled
1	NIDEN	R	0h	Controls noninvasive debug enable. 0h = Non-invasive debug disabled 1h = Non-invasive debug enabled
0	DBGEN	R	0h	Controls invasive debug enable. 0h = Invasive debug disabled 1h = Invasive debug enabled

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from May 31, 2024 to July 24, 2024 (from Revision A (May 2024) to Revision B (July 2024)) Page

- Updated NONMAIN register table for the registers BOOTCFG1, FLASHSWP0 and FLASHSWP1.....22

Changes from October 3, 2023 to May 30, 2024 (from Revision * (October 2023) to Revision A (May 2024)) Page

- Changes throughout document for initial release of the MCUs described in this TRM.....9
- Updated Section 2.4.1.7 with note stating RSTCTL register does not reset FPUB and FSUB registers for a given peripheral.....61
- Updated Figure 7-1 with PC connections for Input Logic.....293
- Updated Section 7.2.1 to clarify PC is valid for input and output connections.....296
- Removed ASCRES registers.....418
- Removed enumerations in these registers ASCDONE, ASCVRSEL, ASCSTIME, ASCCHSEL, ASCACT...418
- Updated Table 15-4 showing supported counting modes for CVAE.....756
- Updated Figure 15-24showing shadow load value updates for TIMx instances with and without shadow load capability.....776
- Updated Figure 15-26 bit name from “INV” to “CCPOINV” for output inversion mux.....778
- Updated Figure 15-27 with bubble on CCPIV bit and easier visual of "Complimentary Output" selection.....778
- Updated Section 15.2.5.1 with fixed spelling for SWFRCACT_CMPL.....780
- Updated Table 15-18 to simplify explanation for deadband modes using DBCTL register 785
- Updated Section 15.2.6.4 with note for requiring an external connection to use CCP capture inputs with the fault input pin (TIMA_FALx).....794

This page intentionally left blank.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265