EFM8 Busy Bee Family
EFM8BB3 Reference Manual

The EFM8BB3, part of the Busy Bee family of MCUs, is a performance line of 8-bit microcontrollers with a comprehensive analog and digital feature set in small packages.

These devices offer state-of-the-art performance by integrating 12-bit ADC, internal temperature sensor, and up to four 12-bit DACs into small packages, making them ideal for general purpose applications. With an efficient, pipelined 8051 core with maximum operating frequency at 50 MHz, various communication interfaces, and four channels of configurable logic, the EFM8BB3 family is optimal for many embedded applications.

EFM8BB3 applications include the following:

- Consumer electronics
- Precision instrumentation
- Power management and control
- Industrial control and automation
- Smart sensors
- Automotive control

KEY FEATURES

- Pipelined 8-bit 8051 MCU Core with 50 MHz operating frequency
- Up to 29 multifunction I/O pins
- One 12-bit/10-bit ADC
- Four 12-bit DACs with synchronization and PWM capabilities
- Two low-current analog comparators with built-in reference DACs
- Internal temperature sensor
- Internal 49 MHz and 24.5 MHz oscillators accurate to ±2%
- Four channels of Configurable Logic
- 6-channel PWM / PCA
- Six 16-bit general-purpose timers

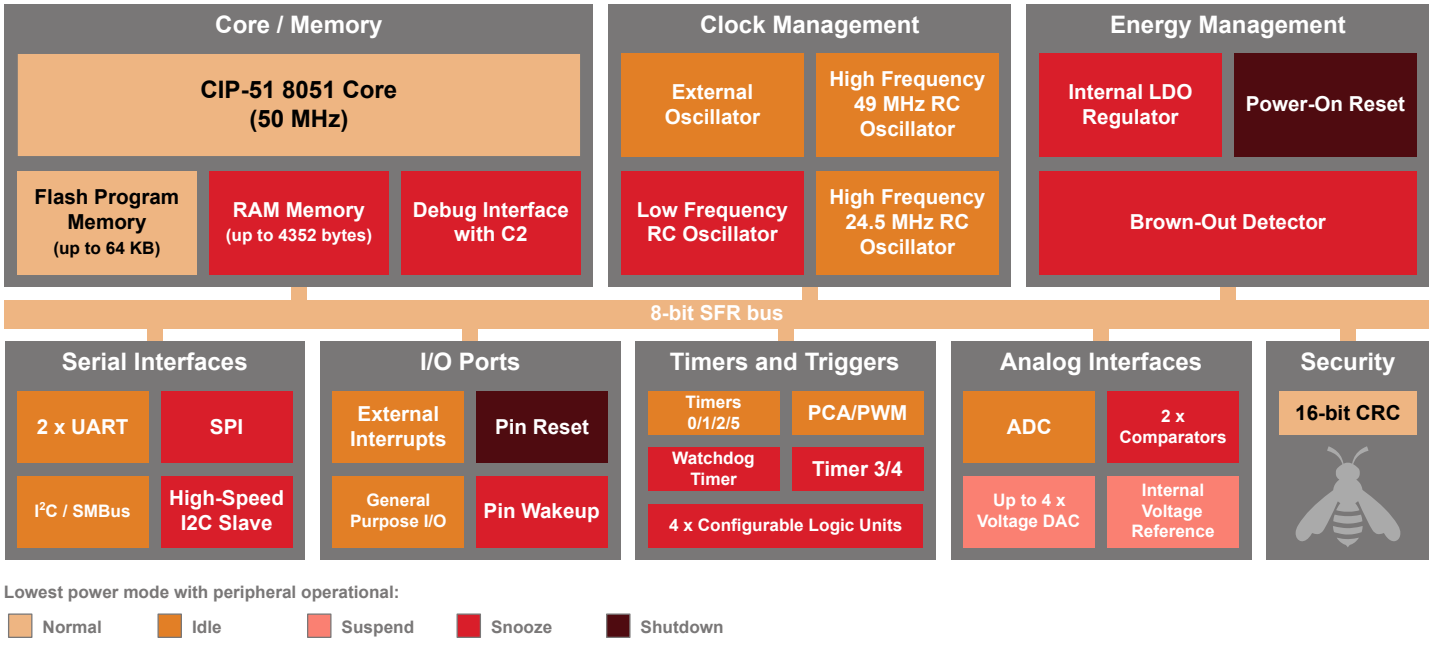


Table of Contents

1. System Overview

1.1 Introduction

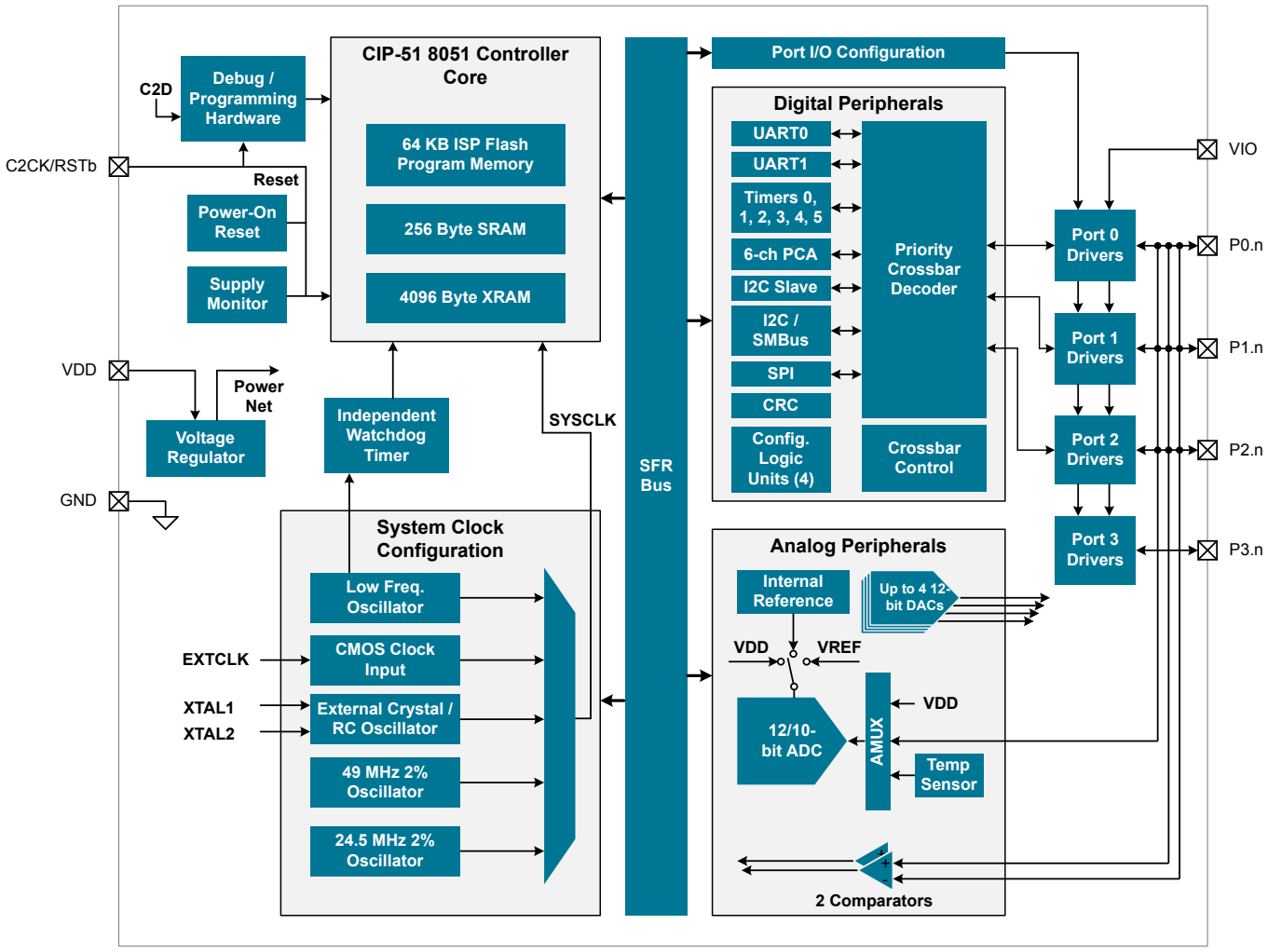


Figure 1.1. Detailed EFM8BB3 Block Diagram

This section describes the EFM8BB3 family at a high level.

For more information on the device packages and pinout, electrical specifications, and typical connection diagrams, see the EFM8BB3 Data Sheet. For more information on each module including register definitions, see the EFM8BB3 Reference Manual. For more information on any errata, see the EFM8BB3 Errata.

1.2 Power

All internal circuitry draws power from the VDD supply pin. External I/O pins are powered from the VIO supply voltage (or VDD on devices without a separate VIO connection), while most of the internal circuitry is supplied by an on-chip LDO regulator. Control over the device power can be achieved by enabling/disabling individual peripherals as needed. Each analog peripheral can be disabled when not in use and placed in low power mode. Digital peripherals, such as timers and serial buses, have their clocks gated off and draw little power when they are not in use.

Table 1.1. Power Modes

Power Mode	Details	Mode Entry	Wake-Up Sources
Normal	Core and all peripherals clocked and fully operational		
Idle	• Core halted • All peripherals clocked and fully operational • Code resumes execution on wake event	Set IDLE bit in PCON0	Any interrupt
Suspend	• Core and peripheral clocks halted • HFOSC0 and HFOSC1 oscillators stopped • Regulator in normal bias mode for fast wake • Timer 3 and 4 may clock from LFOSC0 • Code resumes execution on wake event	1. Switch SYSCLK to HFOSC0 2. Set SUSPEND bit in PCON1	• Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Low Level • CLUn Interrupt-Enabled Event
Stop	• All internal power nets shut down • Pins retain state • Exit on any reset source	1. Clear STOPCF bit in REG0CN 2. Set STOP bit in PCON0	Any reset source
Snooze	• Core and peripheral clocks halted • HFOSC0 and HFOSC1 oscillators stopped • Regulator in low bias current mode for energy savings • Timer 3 and 4 may clock from LFOSC0 • Code resumes execution on wake event	1. Switch SYSCLK to HFOSC0 2. Set SNOOZE bit in PCON1	• Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Low Level • CLUn Interrupt-Enabled Event
Shutdown	• All internal power nets shut down • Pins retain state • Exit on pin or power-on reset	1. Set STOPCF bit in REG0CN 2. Set STOP bit in PCON0	• RSTb pin reset • Power-on reset

1.3 I/O

Digital and analog resources are externally available on the device's multi-purpose I/O pins. Port pins P0.0-P2.3 can be defined as general-purpose I/O (GPIO), assigned to one of the internal digital resources through the crossbar or dedicated channels, or assigned to an analog function. Port pins P2.4 to P3.7 can be used as GPIO. Additionally, the C2 Interface Data signal (C2D) is shared with P3.0 or P3.7, depending on the package option.

The port control block offers the following features:

- Up to 29 multi-function I/O pins, supporting digital and analog functions.
- Flexible priority crossbar decoder for digital peripheral assignment.
- Two drive strength settings for each port.
- State retention feature allows pins to retain configuration through most reset sources.
- Two direct-pin interrupt sources with dedicated interrupt vectors (INT0 and INT1).
- Up to 24 direct-pin interrupt sources with shared interrupt vector (Port Match).

1.4 Clocking

The CPU core and peripheral subsystem may be clocked by both internal and external oscillator resources. By default, the system clock comes up running from the 24.5 MHz oscillator divided by 8.

The clock control system offers the following features:

- Provides clock to core and peripherals.
- 24.5 MHz internal oscillator (HFOSC0), accurate to ±2% over supply and temperature corners.
- 49 MHz internal oscillator (HFOSC1), accurate to ±2% over supply and temperature corners.
- 80 kHz low-frequency oscillator (LFOSC0).
- External high frequency crystal, RC, and CMOS clock options (EXTCLK).
- Clock divider with eight settings for flexible clock scaling:
  - Divide the selected clock source by 1, 2, 4, 8, 16, 32, 64, or 128.
  - HFOSC0 and HFOSC1 include 1.5x pre-scalers for further flexibility.

1.5 Counters/Timers and PWM

Programmable Counter Array (PCA0)

The programmable counter array (PCA) provides multiple channels of enhanced timer and PWM functionality while requiring less CPU intervention than standard counter/timers. The PCA consists of a dedicated 16-bit counter/timer and one 16-bit capture/compare module for each channel. The counter/timer is driven by a programmable timebase that has flexible external and internal clocking options. Each capture/compare module may be configured to operate independently in one of five modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, or Pulse-Width Modulated (PWM) Output. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the crossbar to port I/O when enabled.

- 16-bit time base
- Programmable clock divisor and clock source selection
- Up to six independently-configurable channels
- 8, 9, 10, 11 and 16-bit PWM modes (center or edge-aligned operation)
- Output polarity control
- Frequency output mode
- Capture on rising, falling or any edge
- Compare function for arbitrary waveform generation
- Software timer (internal compare) mode
- Can accept hardware “kill” signal from comparator 0 or comparator 1

Timers (Timer 0, Timer 1, Timer 2, Timer 3, Timer 4, and Timer 5)

Several counter/timers are included in the device: two are 16-bit counter/timers compatible with those found in the standard 8051, and the rest are 16-bit auto-reload timers for timing peripherals or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. The other timers offer both 16-bit and split 8-bit timer functionality with auto-reload and capture capabilities.

Timer 0 and Timer 1 include the following features:

- Standard 8051 timers, supporting backwards-compatibility with firmware and hardware.
- Clock sources include SYSCLK, SYSCLK divided by 12, 4, or 48, the External Clock divided by 8, or an external pin.
- 8-bit auto-reload counter/timer mode
- 13-bit counter/timer mode
- 16-bit counter/timer mode
- Dual 8-bit counter/timer mode (Timer 0)

Timer 2, Timer 3, Timer 4, and Timer 5 are 16-bit timers including the following features:

- Clock sources for all timers include SYSCLK, SYSCLK divided by 12, or the External Clock divided by 8
- LFOSC0 divided by 8 may be used to clock Timer 3 and Timer 4 in active or suspend/snooze power modes
- Timer 4 is a low-power wake source, and can be chained together with Timer 3
- 16-bit auto-reload timer mode
- Dual 8-bit auto-reload timer mode
- External pin capture
- LFOSC0 capture
- Comparator 0 capture
- Configurable Logic output capture

Watchdog Timer (WDT0)

The device includes a programmable watchdog timer (WDT) running off the low-frequency oscillator. A WDT overflow forces the MCU into the reset state. To prevent the reset, the WDT must be restarted by application software before overflow. If the system experiences a software or hardware malfunction preventing the software from restarting the WDT, the WDT overflows and causes a reset. Following a reset, the WDT is automatically enabled and running with the default maximum time interval. If needed, the WDT can be disabled by system software or locked on to prevent accidental disabling. Once locked, the WDT cannot be disabled until the next system reset. The state of the RST pin is unaffected by this reset.

The Watchdog Timer has the following features:

- Programmable timeout interval
- Runs from the low-frequency oscillator
- Lock-out feature to prevent any modification until a system reset

1.6 Communications and Other Digital Peripherals

Universal Asynchronous Receiver/Transmitter (UART0)

UART0 is an asynchronous, full duplex serial port offering modes 1 and 3 of the standard 8051 UART. Enhanced baud rate support allows a wide range of clock sources to generate standard baud rates. Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

The UART module provides the following features:

- Asynchronous transmissions and receptions.
- Baud rates up to SYSCLK/2 (transmit) or SYSCLK/8 (receive).
- 8- or 9-bit data.
- Automatic start and stop generation.
- Single-byte FIFO on transmit and receive.

Universal Asynchronous Receiver/Transmitter (UART1)

UART1 is an asynchronous, full duplex serial port offering a variety of data formatting options. A dedicated baud rate generator with a 16-bit timer and selectable prescaler is included, which can generate a wide range of baud rates. A received data FIFO allows UART1 to receive multiple bytes before data is lost and an overflow occurs.

UART1 provides the following features:

- Asynchronous transmissions and receptions
- Dedicated baud rate generator supports baud rates up to SYSCLK/2 (transmit) or SYSCLK/8 (receive)
- 5, 6, 7, 8, or 9 bit data
- Automatic start and stop generation
- Automatic parity generation and checking
- Single-byte buffer on transmit and receive
- Auto-baud detection
- LIN break and sync field detection
- CTS / RTS hardware flow control

Serial Peripheral Interface (SPI0)

The serial peripheral interface (SPI) module provides access to a flexible, full-duplex synchronous serial bus. The SPI can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select the SPI in slave mode, or to disable master mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a firmware-controlled chip-select output in master mode, or disabled to reduce the number of pins required. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

- Supports 3- or 4-wire master or slave modes
- Supports external clock frequencies up to 12 Mbps in master or slave mode
- Support for all clock phase and polarity modes
- 8-bit programmable clock rate (master)
- Programmable receive timeout (slave)
- Two byte FIFO on transmit and receive
- Can operate in suspend or snooze modes and wake the CPU on reception of a byte
- Support for multiple masters on the same data lines

System Management Bus / I2C (SMB0)

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I^2C serial bus.

The SMBus module includes the following features:

- Standard (up to 100 kbps) and Fast (400 kbps) transfer speeds
- Support for master, slave, and multi-master modes
- Hardware synchronization and arbitration for multi-master mode
- Clock low extending (clock stretching) to interface with faster masters
- Hardware support for 7-bit slave and general call address recognition
- Firmware support for 10-bit slave address decoding
- Ability to inhibit all slave states
- Programmable data setup/hold times
- Transmit and receive FIFOs (one byte) to help increase throughput in faster applications

I2C Slave (I2CSLAVE0)

The I2C Slave interface is a 2-wire, bidirectional serial bus that is compatible with the I2C Bus Specification 3.0. It is capable of transferring in high-speed mode (HS-mode) at speeds of up to 3.4 Mbps. Firmware can write to the I2C interface, and the I2C interface can autonomously control the serial transfer of data. The interface also supports clock stretching for cases where the core may be temporarily prohibited from transmitting a byte or processing a received byte during an I2C transaction. This module operates only as an I2C slave device.

The I2C module includes the following features:

- Standard (up to 100 kbps), Fast (400 kbps), Fast Plus (1 Mbps), and High-speed (3.4 Mbps) transfer speeds
- Support for slave mode only
- Clock low extending (clock stretching) to interface with faster masters
- Hardware support for 7-bit slave address recognition
- Transmit and receive FIFOs (two byte) to help increase throughput in faster applications
- Hardware support for multiple slave addresses with the option to save the matching address in the receive FIFO

16-bit CRC (CRC0)

The cyclic redundancy check (CRC) module performs a CRC using a 16-bit polynomial. CRC0 accepts a stream of 8-bit data and posts the 16-bit result to an internal register. In addition to using the CRC block for data manipulation, hardware can automatically CRC the flash contents of the device.

The CRC module is designed to provide hardware calculations for flash memory verification and communications protocols. The CRC module supports the standard CCITT-16 16-bit polynomial (0x1021), and includes the following features:

- Support for CCITT-16 polynomial
- Byte-level bit reversal
- Automatic CRC of flash contents on one or more 256-byte blocks
- Initial seed selection of 0x0000 or 0xFFFF

Configurable Logic Units (CLU0, CLU1, CLU2, and CLU3)

The Configurable Logic block consists of multiple Configurable Logic Units (CLUs). CLUs are flexible logic functions which may be used for a variety of digital functions, such as replacing system glue logic, aiding in the generation of special waveforms, or synchronizing system event triggers.

- Four configurable logic units (CLUs), with direct-pin and internal logic connections
- Each unit supports 256 different combinatorial logic functions (AND, OR, XOR, muxing, etc.) and includes a clocked flip-flop for synchronous operations
- Units may be operated synchronously or asynchronously
- May be cascaded together to perform more complicated logic functions
- Can operate in conjunction with serial peripherals such as UART and SPI or timing peripherals such as timers and PCA channels
- Can be used to synchronize and trigger multiple on-chip resources (ADC, DAC, Timers, etc.)
- Asynchronous output may be used to wake from low-power states

1.7 Analog

12/10-Bit Analog-to-Digital Converter (ADC0)

The ADC is a successive-approximation-register (SAR) ADC with 12- and 10-bit modes, integrated track-and hold and a programmable window detector. The ADC is fully configurable under software control via several registers. The ADC may be configured to measure different signals using the analog multiplexer. The voltage reference for the ADC is selectable between internal and external reference sources.

- Up to 20 external inputs
- Single-ended 12-bit and 10-bit modes
- Supports an output update rate of up to 350 ksps in 12-bit mode
- Channel sequencer logic with direct-to-XDATA output transfers
- Operation in a low power mode at lower conversion speeds
- Asynchronous hardware conversion trigger, selectable between software, external I/O and internal timer and configurable logic sources
- Output data window comparator allows automatic range checking
- Support for output data accumulation
- Conversion complete and window compare interrupts supported
- Flexible output data formatting
- Includes a fully-internal fast-settling 1.65 V reference and an on-chip precision 2.4 / 1.2 V reference, with support for using the supply as the reference, an external reference and signal ground
- Integrated temperature sensor

12-Bit Digital-to-Analog Converters (DAC0, DAC1, DAC2, DAC3)

The DAC modules are 12-bit Digital-to-Analog Converters with the capability to synchronize multiple outputs together. The DACs are fully configurable under software control. The voltage reference for the DACs is selectable between internal and external reference sources.

- Voltage output with 12-bit performance
- Supports an update rate of 200 ksps
- Hardware conversion trigger, selectable between software, external I/O and internal timer and configurable logic sources
- Outputs may be configured to persist through reset and maintain output state to avoid system disruption
- Multiple DAC outputs can be synchronized together
- DAC pairs (DAC0 and 1 or DAC2 and 3) support complementary output waveform generation
- Outputs may be switched between two levels according to state of configurable logic / PWM input trigger
- Flexible input data formatting
- Supports references from internal supply, on-chip precision reference, or external VREF pin

Low Current Comparators (CMP0, CMP1)

An analog comparator is used to compare the voltage of two analog inputs, with a digital output indicating which input voltage is higher. External input connections to device I/O pins and internal connections are available through separate multiplexers on the positive and negative inputs. Hysteresis, response time, and current consumption may be programmed to suit the specific needs of the application.

The comparator includes the following features:

- Up to 10 (CMP0) or 9 (CMP1) external positive inputs
- Up to 10 (CMP0) or 9 (CMP1) external negative inputs
- Additional input options:
  - Internal connection to LDO output
  - Direct connection to GND
  - Direct connection to VDD
  - Dedicated 6-bit reference DAC
- Synchronous and asynchronous outputs can be routed to pins via crossbar
- Programmable hysteresis between 0 and ±20 mV
- Programmable response time
- Interrupts generated on rising, falling, or both edges
- PWM output kill feature

1.8 Reset Sources

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

- The core halts program execution.
- Module registers are initialized to their defined reset values unless the bits reset only with a power-on reset.
- External port pins are forced to a known state.
- Interrupts and timers are disabled.

All registers are reset to the predefined values noted in the register descriptions unless the bits only reset with a power-on reset. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost. By default, the Port I/O latches are reset to 1 in open-drain mode, with weak pullups enabled during and after the reset. Optionally, firmware may configure the port I/O, DAC outputs, and precision reference to maintain state through system resets other than power-on resets. For Supply Monitor and power-on resets, the RSTb pin is driven low until the device exits the reset state. On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to an internal oscillator. The Watchdog Timer is enabled, and program execution begins at location 0x0000.

Reset sources on the device include the following:

- Power-on reset
- External reset pin
- Comparator reset
- Software-triggered reset
- Supply monitor reset (monitors VDD supply)
- Watchdog timer reset
- Missing clock detector reset
- Flash error reset

1.9 Debugging

The EFM8BB3 devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

1.10 Bootloader

All devices come pre-programmed with a UART0 bootloader. This bootloader resides in the code security page, which is the last page of code flash; it can be erased if it is not needed.

The byte before the Lock Byte is the Bootloader Signature Byte. Setting this byte to a value of 0xA5 indicates the presence of the bootloader in the system. Any other value in this location indicates that the bootloader is not present in flash.

When a bootloader is present, the device will jump to the bootloader vector after any reset, allowing the bootloader to run. The bootloader then determines if the device should stay in bootload mode or jump to the reset vector located at 0x0000. When the bootloader is not present, the device will jump to the reset vector of 0x0000 after any reset.

Silicon Labs recommends the bootloader be disabled and the flash memory locked after the production programming step in applications where code security is a concern. More information about the factory bootloader protocol, usage, customization and best practices can be found in *AN945: EFM8 Factory Bootloader User Guide*. Application notes can be found on the Silicon Labs website (www.silabs.com/8bit-appnotes) or within Simplicity Studio by using the [**Application Notes**] tile.

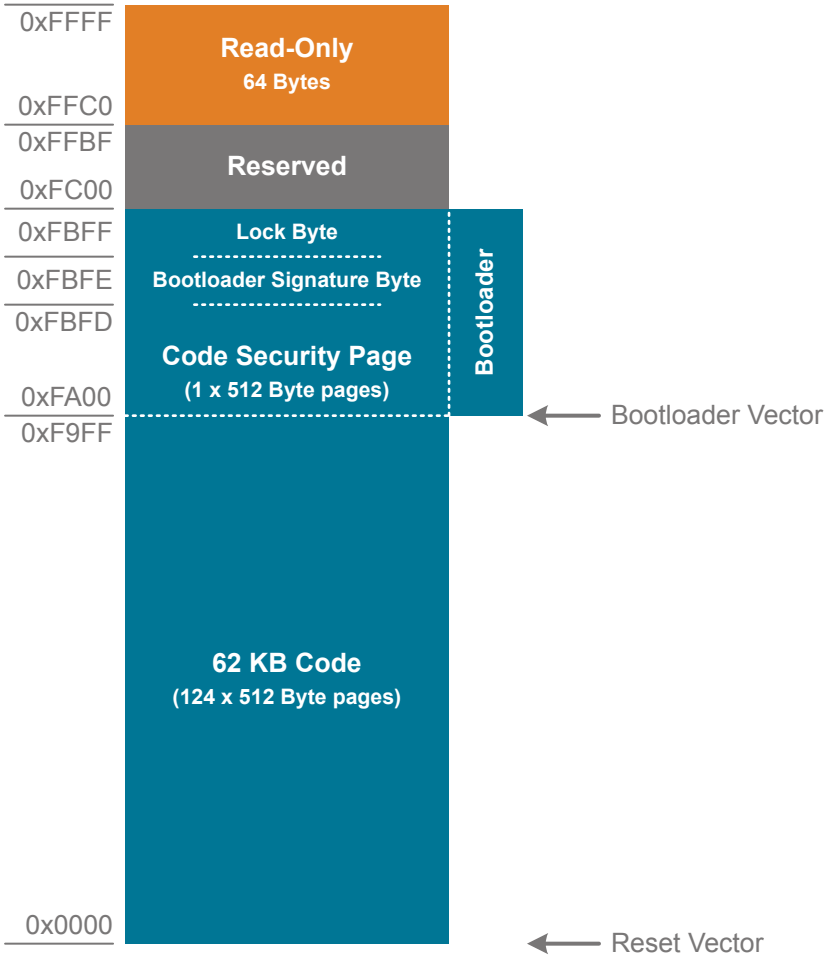


Figure 1.2. Flash Memory Map with Bootloader — 62.5 KB Devices

Table 1.2. Summary of Pins for Bootloader Communication

Bootloader	Pins for Bootload Communication
UART	TX – P0.4
	RX – P0.5

Table 1.3. Summary of Pins for Bootload Mode Entry

Device Package	Pin for Bootload Mode Entry
all QFN32 packages	P3.7 / C2D
QFP32	P3.7 / C2D
all QFN24 packages	P3.0 / C2D
QSOP24	P3.0 / C2D

2. Memory Organization

2.1 Memory Organization

The memory organization of the CIP-51 System Controller is similar to that of a standard 8051. There are two separate memory spaces: program memory and data memory. Program and data memory share the same address space but are accessed via different instruction types. Program memory consists of a non-volatile storage area that may be used for either program code or non-volatile data storage. The data memory, consisting of "internal" and "external" data space, is implemented as RAM, and may be used only for data storage. Program execution is not supported from the data memory space.

2.2 Program Memory

The CIP-51 core has a 64 KB program memory space. The product family implements some of this program memory space as in-system, re-programmable flash memory. Flash security is implemented by a user-programmable location in the flash block and provides read, write, and erase protection. All addresses not specified in the device memory map are reserved and may not be used for code or data storage.

MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip flash memory space. MOVC instructions are always used to read flash memory, while MOVX write instructions are used to erase and write flash. This flash access feature provides a mechanism for the product to update program code and use the program memory space for non-volatile data storage.

2.3 Data Memory

The RAM space on the chip includes both an "internal" RAM area which is accessed with MOV instructions, and an on-chip "external" RAM area which is accessed using MOVX instructions. Total RAM varies, based on the specific device. The device memory map has more details about the specific amount of RAM available in each area for the different device variants.

Internal RAM

There are 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight byte-wide registers. The next 16 bytes, locations 0x20 through 0x2F, may either be addressed as bytes or as 128 bit locations accessible with the direct addressing mode.

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFRs. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the upper 128 bytes of data memory.

General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word (PSW) register, RS0 and RS1, select the active register bank. This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

Bit Addressable Locations

In addition to direct access to data memory organized as bytes, the sixteen data memory locations at 0x20 through 0x2F are also accessible as 128 individually addressable bits. Each bit has a bit address from 0x00 to 0x7F. Bit 0 of the byte at 0x20 has bit address 0x00 while bit 7 of the byte at 0x20 has bit address 0x07. Bit 7 of the byte at 0x2F has bit address 0x7F. A bit access is distinguished from a full byte access by the type of instruction used (bit source or destination operands as opposed to a byte source or destination).

The MCS-51™ assembly language allows an alternate notation for bit addressing of the form XX.B where XX is the byte address and B is the bit position within the byte. For example, the instruction:

```
Mov  C, 22.3h
```

moves the Boolean value at 0x13 (bit 3 of the byte at location 0x22) into the Carry flag.

Stack

A programmer's stack can be located anywhere in the 256-byte data memory. The stack area is designated using the Stack Pointer (SP) SFR. The SP will point to the last location used. The next value pushed on the stack is placed at SP+1 and then SP is incremented. A reset initializes the stack pointer to location 0x07. Therefore, the first value pushed on the stack is placed at location 0x08, which is also the first register (R0) of register bank 1. Thus, if more than one register bank is to be used, the SP should be initialized to a location in the data memory not being used for data storage. The stack depth can extend up to 256 bytes.

External RAM

On devices with more than 256 bytes of on-chip RAM, the additional RAM is mapped into the external data memory space (XRAM). Addresses in XRAM area accessed using the external move (MOVX) instructions.

Note: The 16-bit MOVX write instruction is also used for writing and erasing the flash memory. More details may be found in the flash memory section.

2.4 Memory Map

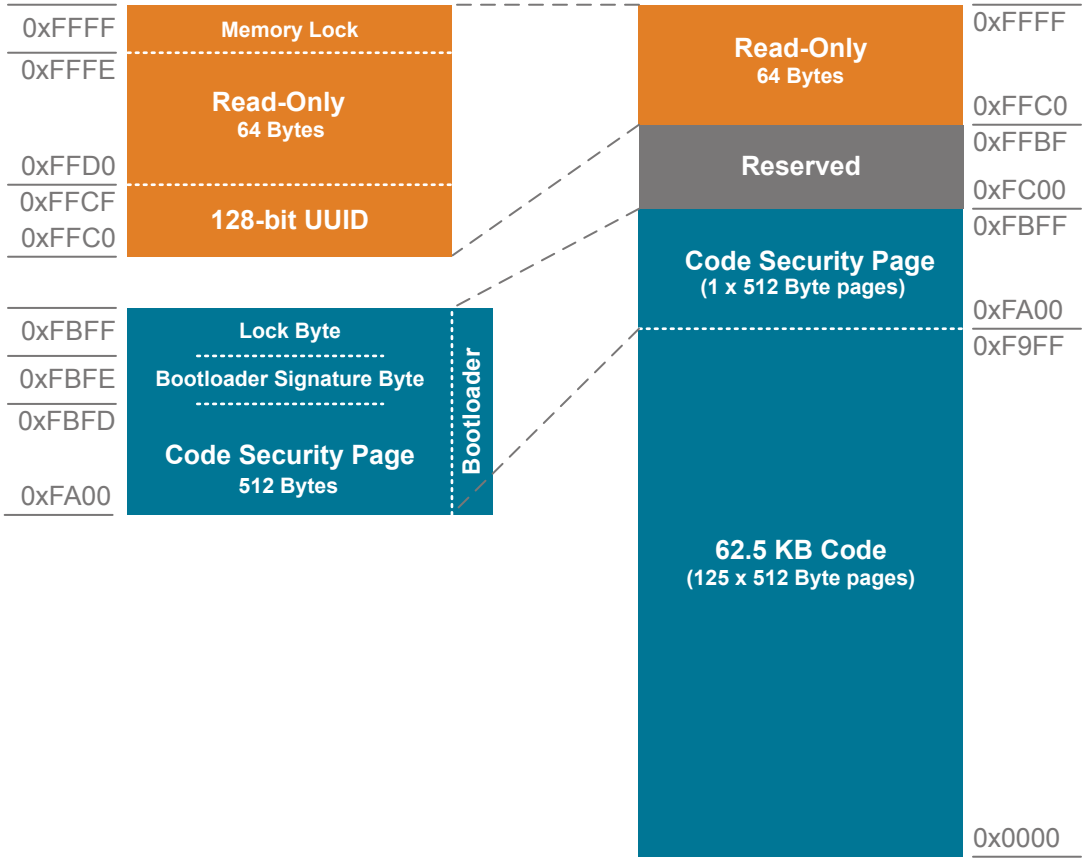


Figure 2.1. Flash Memory Map — 62.5 KB Devices

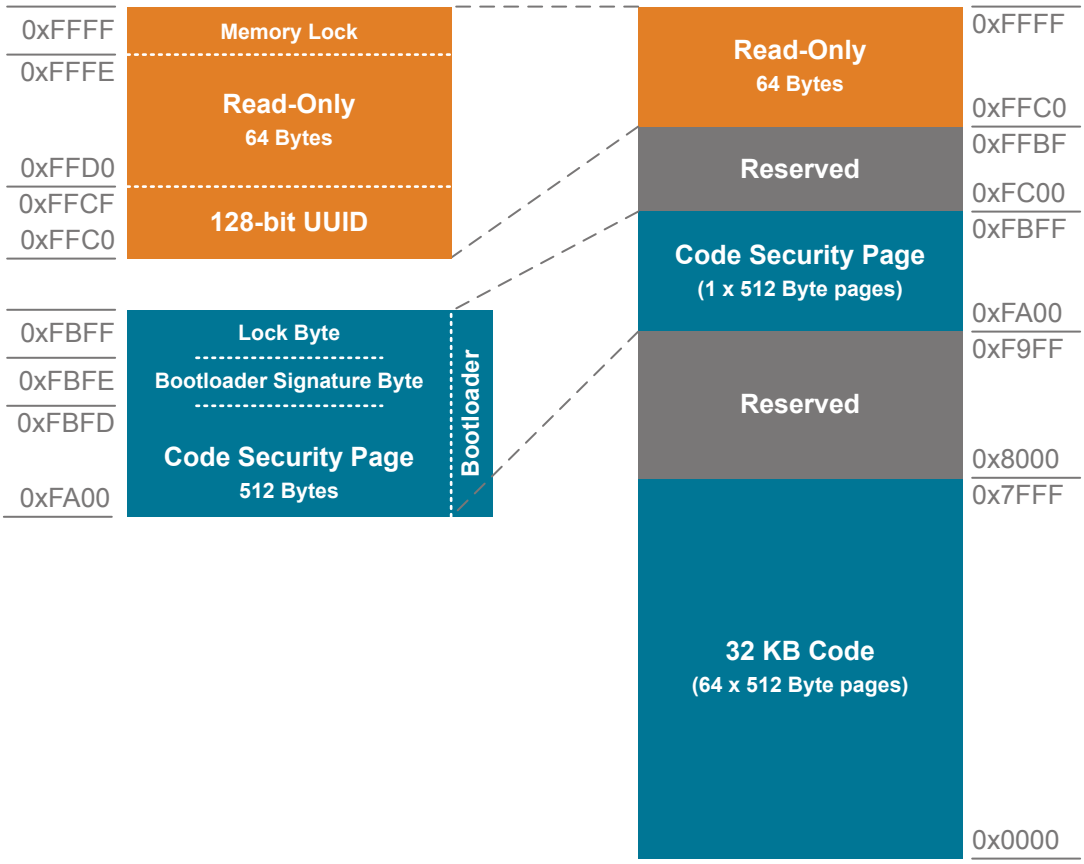


Figure 2.2. Flash Memory Map — 32 KB Devices

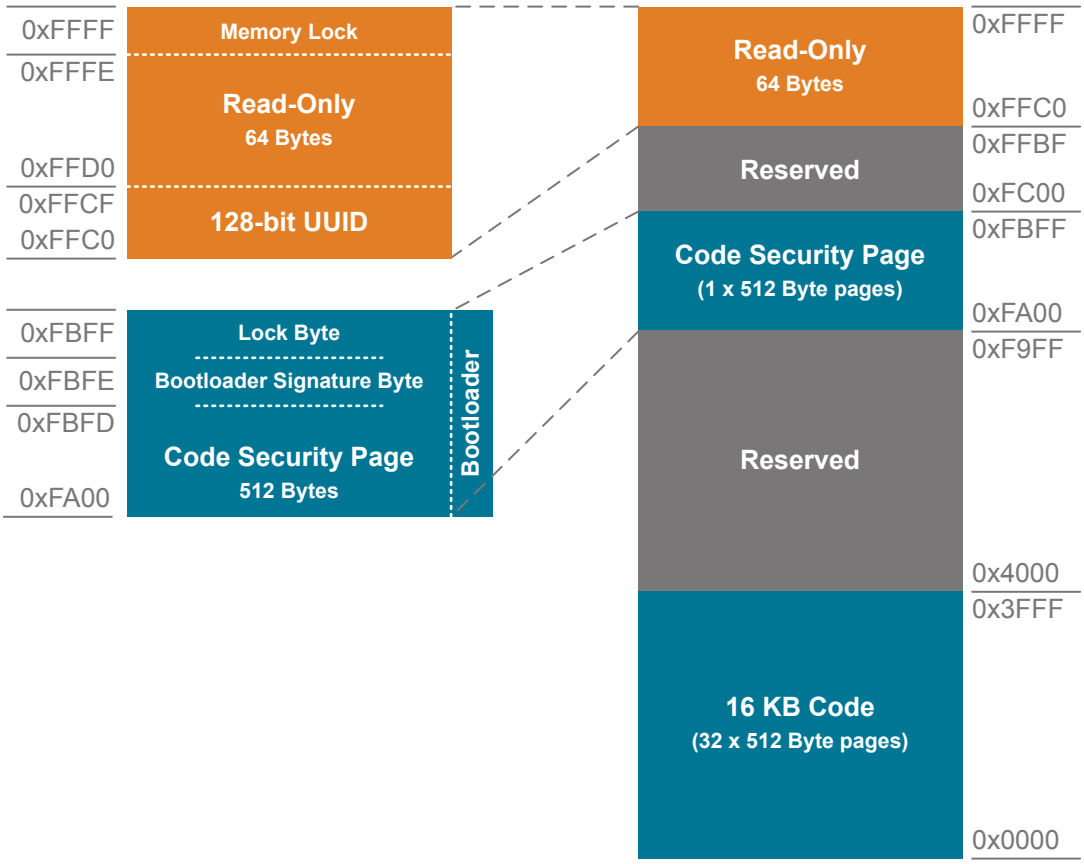


Figure 2.3. Flash Memory Map — 16 KB Devices

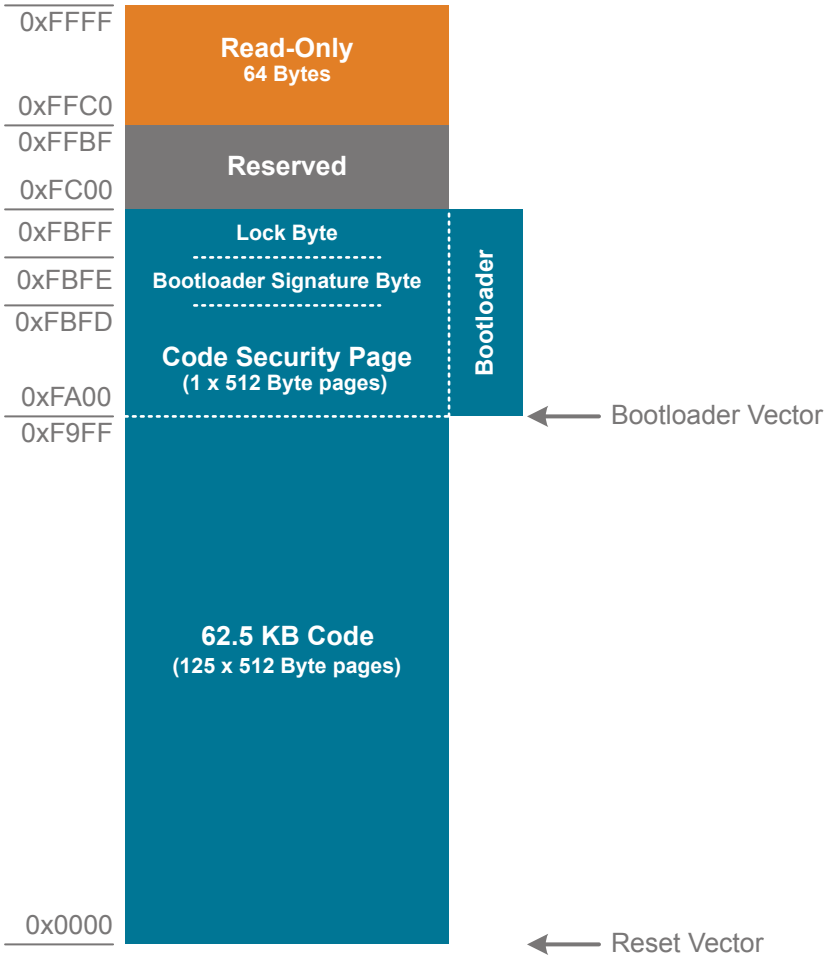


Figure 2.4. Bootloader Flash Memory Map — 62.5 KB Devices

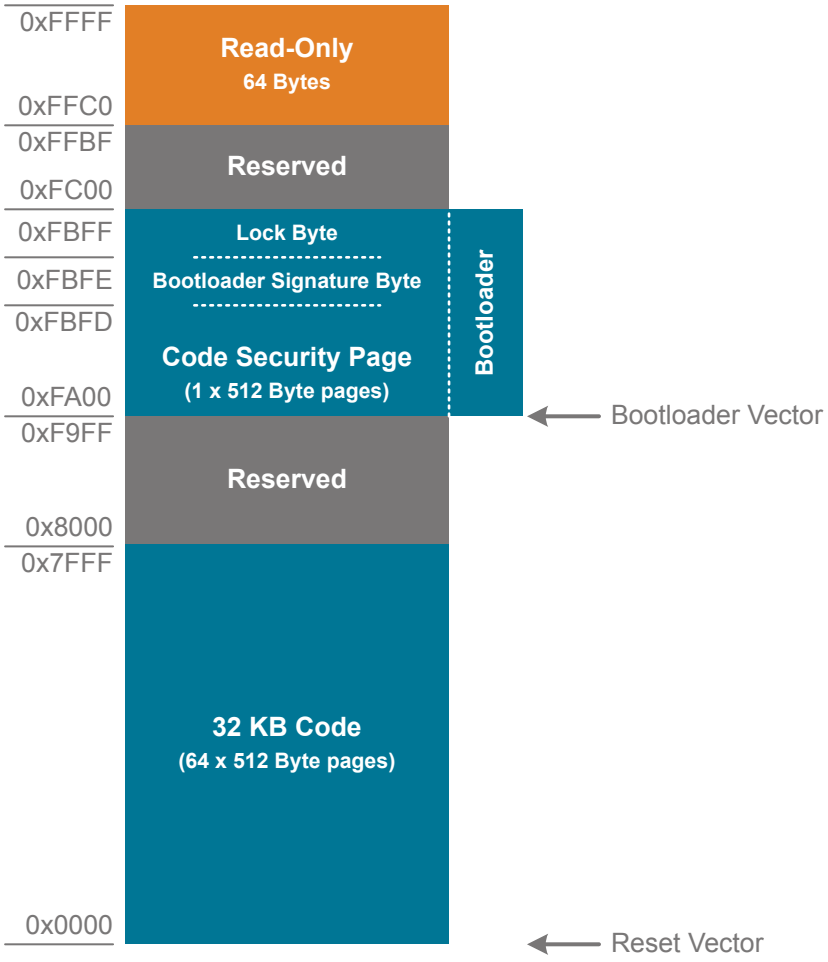


Figure 2.5. Bootloader Flash Memory Map — 32 KB Devices

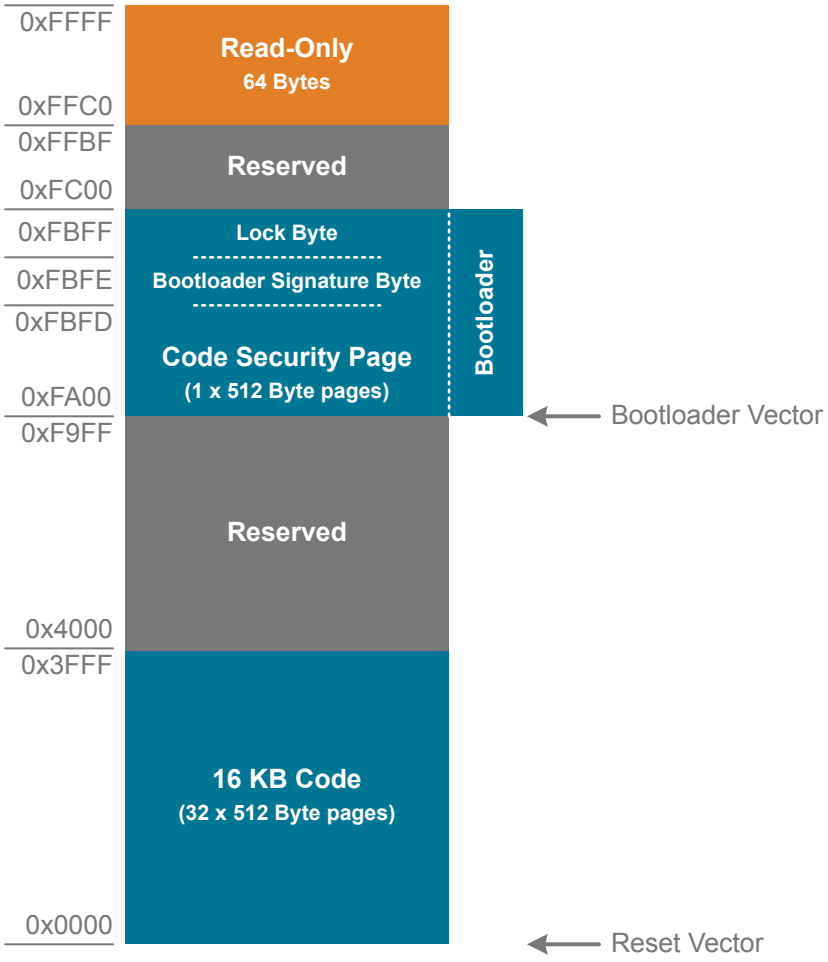


Figure 2.6. Bootloader Flash Memory Map — 16 KB Devices

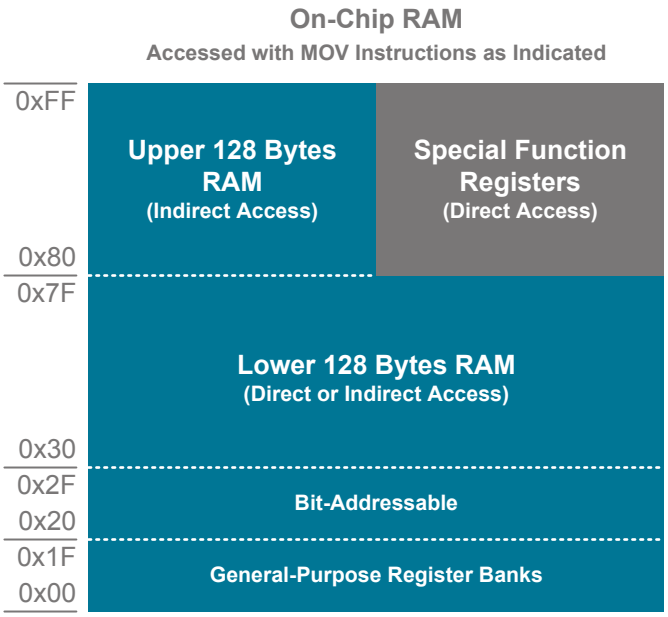


Figure 2.7. Direct / Indirect RAM Memory

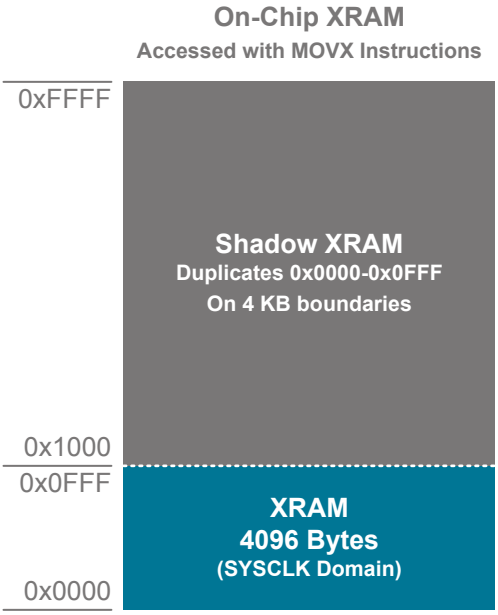


Figure 2.8. XRAM Memory

2.5 XRAM Control Registers

2.5.1 EMI0CN: External Memory Interface Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved				PGSEL			
Access	R				RW			
Reset	0x0				0x0			
SFR Page = ALL; SFR Address: 0xE7								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3:0	PGSEL	0x0	RW	**XRAM Page Select.** The XRAM Page Select field provides the high byte of the 16-bit data memory address when using 8-bit MOVX commands, effectively selecting a 256-byte page of RAM. Since the upper (unused) bits of the register are always zero, the PGSEL field determines which page of XRAM is accessed. For example, if PGSEL = 0x01, addresses 0x0100 to 0x01FF will be accessed by 8-bit MOVX instructions.

3. Special Function Registers

3.1 Special Function Register Access

The direct-access data memory locations from 0x80 to 0xFF constitute the special function registers (SFRs). The SFRs provide control and data exchange with the CIP-51's resources and peripherals. The CIP-51 duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the MCU. This allows the addition of new functionality while retaining compatibility with the MCS-51 ™ instruction set.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g., P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing these areas will have an indeterminate effect and should be avoided.

SFR Paging

The CIP-51 features SFR paging, allowing the device to map many SFRs into the 0x80 to 0xFF memory address space. The SFR memory space has 256 pages. In this way, each memory location from 0x80 to 0xFF can access up to 256 SFRs. The EFM8BB3 devices utilize multiple SFR pages. All of the common 8051 SFRs are available on all pages. Certain SFRs are only available on a subset of pages. SFR pages are selected using the SFRPAGE register. The procedure for reading and writing an SFR is as follows:

1. Select the appropriate SFR page using the SFRPAGE register.
2. Use direct accessing mode to read or write the special function register (MOV instruction).

The SFRPAGE register only needs to be changed in the case that the SFR to be accessed does not exist on the currently-selected page. See the SFR memory map for details on the locations of each SFR.

Interrupts and the SFR Page Stack

When an interrupt occurs, the current SFRPAGE is pushed onto an SFR page stack to preserve the current context of SFRPAGE. Upon execution of the RETI instruction, the SFRPAGE register is automatically restored to the SFR page that was in use prior to the interrupt. The stack is five elements deep to accomodate interrupts of different priority levels pre-empting lower priority interrupts. Firmware can read any element of the SFR page stack by setting the SFRPGIDX field in the SFRPGCN register and reading the SFRSTACK register.

Table 3.1. SFR Page Stack Access

SFRPGIDX Value	SFRSTACK Contains
0	Value of the first/top byte of the stack
1	Value of the second byte of the stack
2	Value of the third byte of the stack
3	Value of the fourth byte of the stack
4	Value of the fifth/bottom byte of the stack
Notes: 1. The top of the stack is the current SFRPAGE setting, and can also be directly accessed via the SFRPAGE register.	

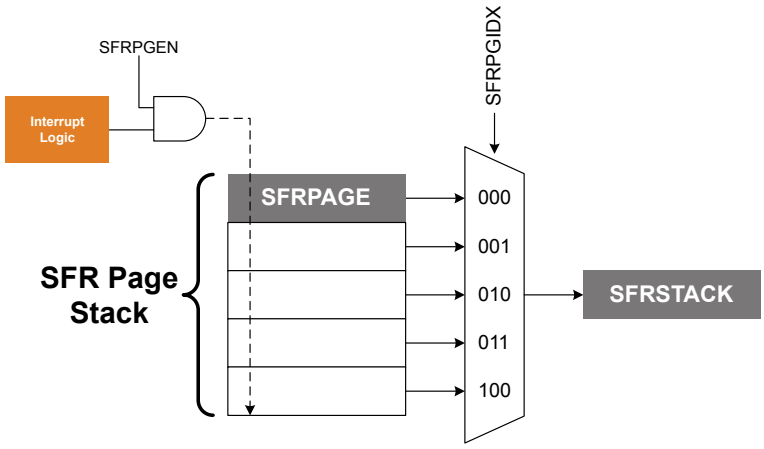


Figure 3.1. SFR Page Stack Block Diagram

When an interrupt occurs, hardware performs the following operations:

1. The value (if any) in the SFRPGIDX = 011b location is pushed to the SFRPAGE = 100b location.
2. The value (if any) in the SFRPGIDX = 010b location is pushed to the SFRPAGE = 011b location.
3. The value (if any) in the SFRPGIDX = 001b location is pushed to the SFRPAGE = 010b location.
4. The current SFRPAGE value is pushed to the SFRPGIDX = 001b location in the stack.
5. SFRPAGE is set to the page associated with the flag that generated the interrupt.

On a return from interrupt, hardware performs the following operations:

1. The SFR page stack is popped to the SFRPAGE register. This restores the SFR page context prior to the interrupt, without software intervention.
2. The value in the SFRPGIDX = 010b location of the stack is placed in the SFRPGIDX = 001b location.
3. The value in the SFRPGIDX = 011b location of the stack is placed in the SFRPGIDX = 010b location.
4. The value in the SFRPGIDX = 100b location of the stack is placed in the SFRPGIDX = 011b location.

Automatic hardware switching of the SFR page upon interrupt entries and exits may be enabled or disabled using the SFRPGEN located in SFRPGCN. Automatic SFR page switching is enabled after any reset.

3.2 Special Function Register Memory Map

Table 3.2. Special Function Registers by Address

Address	SFR Page			
	0x00	0x10	0x20	0x30
0x80*	P0			
0x81	SP			
0x82	DPL			
0x83	DPH			
0x84	PCA0CPL4		CLU0MX	DAC0L
0x85	PCA0CPH4		CLU1MX	DAC0H
0x86	XOSC0CN		CRC0CN1	-
0x87	PCON0			
0x88*	TCON			DACGCF0
0x89	TMOD			DAC1L
0x8A	TL0			DAC1H
0x8B	TL1			DAC2L
0x8C	TH0			DAC2H
0x8D	TH1			DAC3L
0x8E	CKCON0			DAC3H
0x8F	PSCTL			
0x90*	P1			
0x91	TMR3CN0		CLU2MX	DAC0CF0
0x92	TMR3RLL		SBUF1	DAC0CF1
0x93	TMR3RLH		SMOD1	DAC1CF0
0x94	TMR3L		SBCON1	DAC1CF1
0x95	TMR3H		SBRLL1	DAC2CF0
0x96	PCA0POL		SBRLH1	DAC2CF1
0x97	WDTCN			
0x98*	SCON0	TMR4CN0	SCON0	DACGCF1
0x99	SBUF0	-	SBUF0	CMP0CN1
0x9A	-		SPI0FCN0	DAC3CF0
0x9B	CMP0CN0	-	SPI0FCN1	CMP0CN0
0x9C	PCA0CLR		P3MDOUT	DAC3CF1
0x9D	CMP0MD	-	UART1FCN0	CMP0MD
0x9E	PCA0CENT		UART1LIN	-
0x9F	CMP0MX	-		CMP0MX
0xA0*	P2			

Address	SFR Page			
	0x00	0x10	0x20	0x30
0xA1	SPI0CFG	-	SPI0CFG	ADC0ASCF
0xA2	SPI0CKR	TMR4RLL	SPI0CKR	DACGCF2
0xA3	SPI0DAT	TMR4RLH	SPI0DAT	-
0xA4	P0MDOUT	TMR4L	P0MDOUT	-
0xA5	P1MDOUT	TMR4H	P1MDOUT	-
0xA6	P2MDOUT	CKCON1	P2MDOUT	-
0xA7	SFRPAGE			
0xA8*	IE			
0xA9	CLKSEL			
0xAA	CMP1MX	PSTAT0	-	CMP1MX
0xAB	CMP1MD	-	I2C0FCN1	CMP1MD
0xAC	SMB0TC	-	SMB0TC	CMP1CN1
0xAD	DERIVID	-	I2C0FCN0	-
0xAE	PCA0CPM3		CLU3MX	-
0xAF	PCA0CPM4		CLU0FN	-
0xB0*	P3			
0xB1	LFO0CN		CLU0CF	-
0xB2	ADC0CN1	-	CLU1FN	ADC0CN1
0xB3	ADC0CN2	-	CLU1CF	ADC0CN2
0xB4	-			
0xB5	DEVICEID	-	CLU2FN	ADC0ASAL
0xB6	REVID	-	CLU2CF	ADC0ASAH
0xB7	FLKEY			
0xB8*	IP			
0xB9	ADC0CF1	-	I2C0STAT	ADC0CF1
0xBA	-		I2C0CN0	-
0xBB	ADC0MX	EIP1	I2C0DOUT	ADC0MX
0xBC	ADC0CF0	SFRPGCN	I2C0DIN	ADC0CF0
0xBD	ADC0L	-	I2C0SLAD	ADC0L
0xBE	ADC0H	-	CLU3FN	ADC0H
0xBF	CMP1CN0	-	CLU3CF	CMP1CN0
0xC0*	SMB0CN0	TMR5CN0	SMB0CN0	-
0xC1	SMB0CF	PFE0CN	SMB0CF	-
0xC2	SMB0DAT	-	SMB0DAT	-
0xC3	ADC0GTL	-	SMB0FCN0	ADC0GTL
0xC4	ADC0GTH	-	SMB0FCN1	ADC0GTH

Address	SFR Page			
	0x00	0x10	0x20	0x30
0xC5	ADC0LTL	-	SMB0RXLN	ADC0LTL
0xC6	ADC0LTH	-	CLEN0	ADC0LTH
0xC7	HFO0CAL		CLIE0	ADC0ASCT
0xC8*	TMR2CN0		SCON1	-
0xC9	REG0CN	-	REG0CN	-
0xCA	TMR2RLL		CRC0IN	-
0xCB	TMR2RLH		CRC0DAT	-
0xCC	PCA0CPM5		P2SKIP	-
0xCD	PCON1			
0xCE	TMR2L		CRC0CN0	-
0xCF	TMR2H		CRC0FLIP	-
0xD0*	PSW			
0xD1	REF0CN	-	CLOUT0	REF0CN
0xD2	-	TMR5RLL	CRC0ST	-
0xD3	-	TMR5RLH	CRC0CNT	-
0xD4	P0SKIP	TMR5L	P0SKIP	-
0xD5	P1SKIP	TMR5H	P1SKIP	-
0xD6	SMB0ADM	HFO1CAL	SMB0ADM	-
0xD7	SMB0ADR	SFRSTACK	SMB0ADR	-
0xD8*	PCA0CN0		UART1FCN1	-
0xD9	PCA0MD		UART0PCF	-
0xDA	PCA0CPM0		UART1PCF	-
0xDB	PCA0CPM1		-	
0xDC	PCA0CPM2		-	
0xDD	PCA0CPL5		-	
0xDE	PCA0CPH5		-	
0xDF	ADC0CF2	-	SPI0PCF	ADC0CF2
0xE0*	ACC			
0xE1	XBR0	-	XBR0	-
0xE2	XBR1	-	XBR1	-
0xE3	XBR2	-	XBR2	-
0xE4	IT01CF		-	
0xE5	-			
0xE6	EIE1		-	
0xE7	EMI0CN			
0xE8*	ADC0CN0	-	CLIF0	ADC0CN0

Address	SFR Page			
	0x00	0x10	0x20	0x30
0xE9	PCA0CPL1		-	
0xEA	PCA0CPH1		-	
0xEB	PCA0CPL2		-	
0xEC	PCA0CPH2		-	
0xED	P1MAT	EIP2	P1MAT	-
0xEE	P1MASK	EIP1H	P1MASK	-
0xEF	RSTSRC	HFOCN	SMB0FCT	-
0xF0*	B			
0xF1	P0MDIN	TMR5CN1	P0MDIN	-
0xF2	P1MDIN	IPH	P1MDIN	-
0xF3	EIE2		P2MDIN	-
0xF4	PCA0CPL3		P3MDIN	-
0xF5	PCA0CPH3		I2C0FCT	-
0xF6	PRTDRV	EIP2H	PRTDRV	-
0xF7	PCA0PWM		SPI0FCT	-
0xF8*	SPI0CN0	-	SPI0CN0	-
0xF9	PCA0L		-	
0xFA	PCA0H		UART1FCT	-
0xFB	PCA0CPL0		P2MAT	-
0xFC	PCA0CPH0		P2MASK	-
0xFD	P0MAT	TMR2CN1	P0MAT	-
0xFE	P0MASK	TMR3CN1	P0MASK	-
0xFF	VDM0CN	TMR4CN1	I2C0ADM	-

Table 3.3. Special Function Registers by Name

Register	Address	SFR Pages	Description
ACC	0xE0	ALL	Accumulator
ADC0ASAH	0xB6	0x30	ADC0 Autoscan Start Address High Byte
ADC0ASAL	0xB5	0x30	ADC0 Autoscan Start Address Low Byte
ADC0ASCF	0xA1	0x30	ADC0 Autoscan Configuration
ADC0ASCT	0xC7	0x30	ADC0 Autoscan Output Count
ADC0CF0	0xBC	0x00, 0x30	ADC0 Configuration
ADC0CF1	0xB9	0x00, 0x30	ADC0 Configuration
ADC0CF2	0xDF	0x00, 0x30	ADC0 Power Control
ADC0CN0	0xE8	0x00, 0x30	ADC0 Control 0

Register	Address	SFR Pages	Description
ADC0CN1	0xB2	0x00, 0x30	ADC0 Control 1
ADC0CN2	0xB3	0x00, 0x30	ADC0 Control 2
ADC0GTH	0xC4	0x00, 0x30	ADC0 Greater-Than High Byte
ADC0GTL	0xC3	0x00, 0x30	ADC0 Greater-Than Low Byte
ADC0H	0xBE	0x00, 0x30	ADC0 Data Word High Byte
ADC0L	0xBD	0x00, 0x30	ADC0 Data Word Low Byte
ADC0LTH	0xC6	0x00, 0x30	ADC0 Less-Than High Byte
ADC0LTL	0xC5	0x00, 0x30	ADC0 Less-Than Low Byte
ADC0MX	0xBB	0x00, 0x30	ADC0 Multiplexer Selection
B	0xF0	ALL	B Register
CKCON0	0x8E	0x00, 0x10, 0x20	Clock Control 0
CKCON1	0xA6	0x10	Clock Control 1
CLEN0	0xC6	0x20	Configurable Logic Enable 0
CLIE0	0xC7	0x20	Configurable Logic Interrupt Enable 0
CLIF0	0xE8	0x20	Configurable Logic Interrupt Flag 0
CLKSEL	0xA9	ALL	Clock Select
CLOUT0	0xD1	0x20	Configurable Logic Output 0
CLU0CF	0xB1	0x20	Configurable Logic Unit 0 Configuration
CLU0FN	0xAF	0x20	Configurable Logic Unit 0 Function Select
CLU0MX	0x84	0x20	Configurable Logic Unit 0 Multiplexer
CLU1CF	0xB3	0x20	Configurable Logic Unit 1 Configuration
CLU1FN	0xB2	0x20	Configurable Logic Unit 1 Function Select
CLU1MX	0x85	0x20	Configurable Logic Unit 1 Multiplexer
CLU2CF	0xB6	0x20	Configurable Logic Unit 2 Configuration
CLU2FN	0xB5	0x20	Configurable Logic Unit 2 Function Select
CLU2MX	0x91	0x20	Configurable Logic Unit 2 Multiplexer
CLU3CF	0xBF	0x20	Configurable Logic Unit 3 Configuration
CLU3FN	0xBE	0x20	Configurable Logic Unit 3 Function Select
CLU3MX	0xAE	0x20	Configurable Logic Unit 3 Multiplexer
CMP0CN0	0x9B	0x00, 0x30	Comparator 0 Control 0
CMP0CN1	0x99	0x30	Comparator 0 Control 1
CMP0MD	0x9D	0x00, 0x30	Comparator 0 Mode
CMP0MX	0x9F	0x00, 0x30	Comparator 0 Multiplexer Selection
CMP1CN0	0xBF	0x00, 0x30	Comparator 1 Control 0
CMP1CN1	0xAC	0x30	Comparator 1 Control 1
CMP1MD	0xAB	0x00, 0x30	Comparator 1 Mode
CMP1MX	0xAA	0x00, 0x30	Comparator 1 Multiplexer Selection

Register	Address	SFR Pages	Description
CRC0CN0	0xCE	0x20	CRC0 Control 0
CRC0CN1	0x86	0x20	CRC0 Control 1
CRC0CNT	0xD3	0x20	CRC0 Automatic Flash Sector Count
CRC0DAT	0xCB	0x20	CRC0 Data Output
CRC0FLIP	0xCF	0x20	CRC0 Bit Flip
CRC0IN	0xCA	0x20	CRC0 Data Input
CRC0ST	0xD2	0x20	CRC0 Automatic Flash Sector Start
DAC0CF0	0x91	0x30	DAC0 Configuration 0
DAC0CF1	0x92	0x30	DAC0 Configuration 1
DAC0H	0x85	0x30	DAC0 Data Word High Byte
DAC0L	0x84	0x30	DAC0 Data Word Low Byte
DAC1CF0	0x93	0x30	DAC1 Configuration 0
DAC1CF1	0x94	0x30	DAC1 Configuration 1
DAC1H	0x8A	0x30	DAC1 Data Word High Byte
DAC1L	0x89	0x30	DAC1 Data Word Low Byte
DAC2CF0	0x95	0x30	DAC2 Configuration 0
DAC2CF1	0x96	0x30	DAC2 Configuration 1
DAC2H	0x8C	0x30	DAC2 Data Word High Byte
DAC2L	0x8B	0x30	DAC2 Data Word Low Byte
DAC3CF0	0x9A	0x30	DAC3 Configuration 0
DAC3CF1	0x9C	0x30	DAC3 Configuration 1
DAC3H	0x8E	0x30	DAC3 Data Word High Byte
DAC3L	0x8D	0x30	DAC3 Data Word Low Byte
DACGCF0	0x88	0x30	DAC Global Configuration 0
DACGCF1	0x98	0x30	DAC Global Configuration 1
DACGCF2	0xA2	0x30	DAC Global Configuration 2
DERIVID	0xAD	0x00	Derivative Identification
DEVICEID	0xB5	0x00	Device Identification
DPH	0x83	ALL	Data Pointer High
DPL	0x82	ALL	Data Pointer Low
EIE1	0xE6	0x00, 0x10	Extended Interrupt Enable 1
EIE2	0xF3	0x00, 0x10	Extended Interrupt Enable 2
EIP1	0xBB	0x10	Extended Interrupt Priority 1 Low
EIP1H	0xEE	0x10	Extended Interrupt Priority 1 High
EIP2	0xED	0x10	Extended Interrupt Priority 2
EIP2H	0xF6	0x10	Extended Interrupt Priority 2 High
EMI0CN	0xE7	ALL	External Memory Interface Control

Register	Address	SFR Pages	Description
FLKEY	0xB7	ALL	Flash Lock and Key
HFO0CAL	0xC7	0x00, 0x10	High Frequency Oscillator 0 Calibration
HFO1CAL	0xD6	0x10	High Frequency Oscillator 1 Calibration
HFOCN	0xEF	0x10	High Frequency Oscillator Control
I2C0ADM	0xFF	0x20	I2C0 Slave Address Mask
I2C0CN0	0xBA	0x20	I2C0 Control
I2C0DIN	0xBC	0x20	I2C0 Received Data
I2C0DOUT	0xBB	0x20	I2C0 Transmit Data
I2C0FCN0	0xAD	0x20	I2C0 FIFO Control 0
I2C0FCN1	0xAB	0x20	I2C0 FIFO Control 1
I2C0FCT	0xF5	0x20	I2C0 FIFO Count
I2C0SLAD	0xBD	0x20	I2C0 Slave Address
I2C0STAT	0xB9	0x20	I2C0 Status
IE	0xA8	ALL	Interrupt Enable
IP	0xB8	ALL	Interrupt Priority
IPH	0xF2	0x10	Interrupt Priority High
IT01CF	0xE4	0x00, 0x10	INT0/INT1 Configuration
LFO0CN	0xB1	0x00, 0x10	Low Frequency Oscillator Control
P0	0x80	ALL	Port 0 Pin Latch
P0MASK	0xFE	0x00, 0x20	Port 0 Mask
P0MAT	0xFD	0x00, 0x20	Port 0 Match
P0MDIN	0xF1	0x00, 0x20	Port 0 Input Mode
P0MDOUT	0xA4	0x00, 0x20	Port 0 Output Mode
P0SKIP	0xD4	0x00, 0x20	Port 0 Skip
P1	0x90	ALL	Port 1 Pin Latch
P1MASK	0xEE	0x00, 0x20	Port 1 Mask
P1MAT	0xED	0x00, 0x20	Port 1 Match
P1MDIN	0xF2	0x00, 0x20	Port 1 Input Mode
P1MDOUT	0xA5	0x00, 0x20	Port 1 Output Mode
P1SKIP	0xD5	0x00, 0x20	Port 1 Skip
P2	0xA0	ALL	Port 2 Pin Latch
P2MASK	0xFC	0x20	Port 2 Mask
P2MAT	0xFB	0x20	Port 2 Match
P2MDIN	0xF3	0x20	Port 2 Input Mode
P2MDOUT	0xA6	0x00, 0x20	Port 2 Output Mode
P2SKIP	0xCC	0x20	Port 2 Skip
P3	0xB0	ALL	Port 3 Pin Latch

Register	Address	SFR Pages	Description
P3MDIN	0xF4	0x20	Port 3 Input Mode
P3MDOUT	0x9C	0x20	Port 3 Output Mode
PCA0CENT	0x9E	0x00, 0x10	PCA Center Alignment Enable
PCA0CLR	0x9C	0x00, 0x10	PCA Comparator Clear Control
PCA0CN0	0xD8	0x00, 0x10	PCA Control
PCA0CPH0	0xFC	0x00, 0x10	PCA Channel 0 Capture Module High Byte
PCA0CPH1	0xEA	0x00, 0x10	PCA Channel 1 Capture Module High Byte
PCA0CPH2	0xEC	0x00, 0x10	PCA Channel 2 Capture Module High Byte
PCA0CPH3	0xF5	0x00, 0x10	PCA Channel 3 Capture Module High Byte
PCA0CPH4	0x85	0x00, 0x10	PCA Channel 4 Capture Module High Byte
PCA0CPH5	0xDE	0x00, 0x10	PCA Channel 5 Capture Module High Byte
PCA0CPL0	0xFB	0x00, 0x10	PCA Channel 0 Capture Module Low Byte
PCA0CPL1	0xE9	0x00, 0x10	PCA Channel 1 Capture Module Low Byte
PCA0CPL2	0xEB	0x00, 0x10	PCA Channel 2 Capture Module Low Byte
PCA0CPL3	0xF4	0x00, 0x10	PCA Channel 3 Capture Module Low Byte
PCA0CPL4	0x84	0x00, 0x10	PCA Channel 4 Capture Module Low Byte
PCA0CPL5	0xDD	0x00, 0x10	PCA Channel 5 Capture Module Low Byte
PCA0CPM0	0xDA	0x00, 0x10	PCA Channel 0 Capture/Compare Mode
PCA0CPM1	0xDB	0x00, 0x10	PCA Channel 1 Capture/Compare Mode
PCA0CPM2	0xDC	0x00, 0x10	PCA Channel 2 Capture/Compare Mode
PCA0CPM3	0xAE	0x00, 0x10	PCA Channel 3 Capture/Compare Mode
PCA0CPM4	0xAF	0x00, 0x10	PCA Channel 4 Capture/Compare Mode
PCA0CPM5	0xCC	0x00, 0x10	PCA Channel 5 Capture/Compare Mode
PCA0H	0xFA	0x00, 0x10	PCA Counter/Timer High Byte
PCA0L	0xF9	0x00, 0x10	PCA Counter/Timer Low Byte
PCA0MD	0xD9	0x00, 0x10	PCA Mode
PCA0POL	0x96	0x00, 0x10	PCA Output Polarity
PCA0PWM	0xF7	0x00, 0x10	PCA PWM Configuration
PCON0	0x87	ALL	Power Control
PCON1	0xCD	ALL	Power Control 1
PFE0CN	0xC1	0x10	Prefetch Engine Control
PRTDRV	0xF6	0x00, 0x20	Port Drive Strength
PSCTL	0x8F	ALL	Program Store Control
PSTAT0	0xAA	0x10	PMU Status 0
PSW	0xD0	ALL	Program Status Word
REF0CN	0xD1	0x00, 0x30	Voltage Reference Control
REG0CN	0xC9	0x00, 0x20	Voltage Regulator 0 Control

Register	Address	SFR Pages	Description
REVID	0xB6	0x00	Revision Identifcation
RSTSRC	0xEF	0x00	Reset Source
SBCON1	0x94	0x20	UART1 Baud Rate Generator Control
SBRLH1	0x96	0x20	UART1 Baud Rate Generator High Byte
SBRLL1	0x95	0x20	UART1 Baud Rate Generator Low Byte
SBUF0	0x99	0x00, 0x20	UART0 Serial Port Data Buffer
SBUF1	0x92	0x20	UART1 Serial Port Data Buffer
SCON0	0x98	0x00, 0x20	UART0 Serial Port Control
SCON1	0xC8	0x20	UART1 Serial Port Control
SFRPAGE	0xA7	ALL	SFR Page
SFRPGCN	0xBC	0x10	SFR Page Control
SFRSTACK	0xD7	0x10	SFR Page Stack
SMB0ADM	0xD6	0x00, 0x20	SMBus 0 Slave Address Mask
SMB0ADR	0xD7	0x00, 0x20	SMBus 0 Slave Address
SMB0CF	0xC1	0x00, 0x20	SMBus 0 Configuration
SMB0CN0	0xC0	0x00, 0x20	SMBus 0 Control
SMB0DAT	0xC2	0x00, 0x20	SMBus 0 Data
SMB0FCN0	0xC3	0x20	SMBus 0 FIFO Control 0
SMB0FCN1	0xC4	0x20	SMBus 0 FIFO Control 1
SMB0FCT	0xEF	0x20	SMBus 0 FIFO Count
SMB0RXLN	0xC5	0x20	SMBus 0 Receive Length Counter
SMB0TC	0xAC	0x00, 0x20	SMBus 0 Timing and Pin Control
SMOD1	0x93	0x20	UART1 Mode
SP	0x81	ALL	Stack Pointer
SPI0CFG	0xA1	0x00, 0x20	SPI0 Configuration
SPI0CKR	0xA2	0x00, 0x20	SPI0 Clock Rate
SPI0CN0	0xF8	0x00, 0x20	SPI0 Control
SPI0DAT	0xA3	0x00, 0x20	SPI0 Data
SPI0FCN0	0x9A	0x20	SPI0 FIFO Control 0
SPI0FCN1	0x9B	0x20	SPI0 FIFO Control 1
SPI0FCT	0xF7	0x20	SPI0 FIFO Count
SPI0PCF	0xDF	0x20	SPI0 Pin Configuration
TCON	0x88	0x00, 0x10, 0x20	Timer 0/1 Control
TH0	0x8C	0x00, 0x10, 0x20	Timer 0 High Byte
TH1	0x8D	0x00, 0x10, 0x20	Timer 1 High Byte
TL0	0x8A	0x00, 0x10, 0x20	Timer 0 Low Byte
TL1	0x8B	0x00, 0x10, 0x20	Timer 1 Low Byte

Register	Address	SFR Pages	Description
TMOD	0x89	0x00, 0x10, 0x20	Timer 0/1 Mode
TMR2CN0	0xC8	0x00, 0x10	Timer 2 Control 0
TMR2CN1	0xFD	0x10	Timer 2 Control 1
TMR2H	0xCF	0x00, 0x10	Timer 2 High Byte
TMR2L	0xCE	0x00, 0x10	Timer 2 Low Byte
TMR2RLH	0xCB	0x00, 0x10	Timer 2 Reload High Byte
TMR2RLL	0xCA	0x00, 0x10	Timer 2 Reload Low Byte
TMR3CN0	0x91	0x00, 0x10	Timer 3 Control 0
TMR3CN1	0xFE	0x10	Timer 3 Control 1
TMR3H	0x95	0x00, 0x10	Timer 3 High Byte
TMR3L	0x94	0x00, 0x10	Timer 3 Low Byte
TMR3RLH	0x93	0x00, 0x10	Timer 3 Reload High Byte
TMR3RLL	0x92	0x00, 0x10	Timer 3 Reload Low Byte
TMR4CN0	0x98	0x10	Timer 4 Control 0
TMR4CN1	0xFF	0x10	Timer 4 Control 1
TMR4H	0xA5	0x10	Timer 4 High Byte
TMR4L	0xA4	0x10	Timer 4 Low Byte
TMR4RLH	0xA3	0x10	Timer 4 Reload High Byte
TMR4RLL	0xA2	0x10	Timer 4 Reload Low Byte
TMR5CN0	0xC0	0x10	Timer 5 Control 0
TMR5CN1	0xF1	0x10	Timer 5 Control 1
TMR5H	0xD5	0x10	Timer 5 High Byte
TMR5L	0xD4	0x10	Timer 5 Low Byte
TMR5RLH	0xD3	0x10	Timer 5 Reload High Byte
TMR5RLL	0xD2	0x10	Timer 5 Reload Low Byte
UART0PCF	0xD9	0x20	UART0 Pin Configuration
UART1FCN0	0x9D	0x20	UART1 FIFO Control 0
UART1FCN1	0xD8	0x20	UART1 FIFO Control 1
UART1FCT	0xFA	0x20	UART1 FIFO Count
UART1LIN	0x9E	0x20	UART1 LIN Configuration
UART1PCF	0xDA	0x20	UART1 Pin Configuration
VDM0CN	0xFF	0x00	Supply Monitor Control
WDTCN	0x97	ALL	Watchdog Timer Control
XBR0	0xE1	0x00, 0x20	Port I/O Crossbar 0
XBR1	0xE2	0x00, 0x20	Port I/O Crossbar 1
XBR2	0xE3	0x00, 0x20	Port I/O Crossbar 2
XOSC0CN	0x86	0x00, 0x10	External Oscillator Control

3.3 SFR Access Control Registers

3.3.1 SFRPAGE: SFR Page

Bit	7	6	5	4	3	2	1	0
Name	SFRPAGE							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xA7								

Bit	Name	Reset	Access	Description
7:0	SFRPAGE	0x00	RW	**SFR Page.** Specifies the SFR Page used when reading, writing, or modifying special function registers.

3.3.2 SFRPGCN: SFR Page Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved	SFRPGIDX			Reserved			SFRPGEN
Access	RW	RW			RW			RW
Reset	0	0x0			0x0			1
SFR Page = 0x10; SFR Address: 0xBC								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	SFRPGIDX	0x0	RW	**SFR Page Stack Index.**

This field can be used to access the SFRPAGE values stored in the SFR page stack. It selects the level of the stack firmware can access when reading the SFRSTACK register.

Value	Name	Description
0x0	FIRST_BYTE	SFRSTACK contains the value of SFRPAGE, the first/top byte of the SFR page stack.
0x1	SECOND_BYTE	SFRSTACK contains the value of the second byte of the SFR page stack.
0x2	THIRD_BYTE	SFRSTACK contains the value of the third byte of the SFR page stack.
0x3	FOURTH_BYTE	SFRSTACK contains the value of the fourth byte of the SFR page stack.
0x4	FIFTH_BYTE	SFRSTACK contains the value of the fifth byte of the SFR page stack.

3:1	*Reserved*	*Must write reset value.*		
0	SFRPGEN	1	RW	**SFR Automatic Page Control Enable.**

This bit is used to enable automatic page switching on ISR entry/exit. When set to 1, the current SFRPAGE value will be pushed onto the SFR page stack and SFRPAGE will be set to the page corresponding to the flag which generated the interrupt; upon ISR exit, hardware will pop the value from the SFR page stack and restore SFRPAGE.

Value	Name	Description
0	DISABLED	Disable automatic SFR paging.
1	ENABLED	Enable automatic SFR paging.

3.3.3 SFRSTACK: SFR Page Stack

Bit	7	6	5	4	3	2	1	0
Name	SFRSTACK							
Access	R							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xD7								

Bit	Name	Reset	Access	Description
7:0	SFRSTACK	0x00	R	**SFR Page Stack.**

This register is used to read the contents of the SFR page stack. The SFRPGIDX field in the SFRPGCN register controls the level of the stack this register will access.

4. Flash Memory

4.1 Introduction

On-chip, re-programmable flash memory is included for program code and non-volatile data storage. The flash memory is organized in 512-byte pages. It can be erased and written through the C2 interface or from firmware by overloading the MOVX instruction. Any individual byte in flash memory must only be written once between page erase operations.

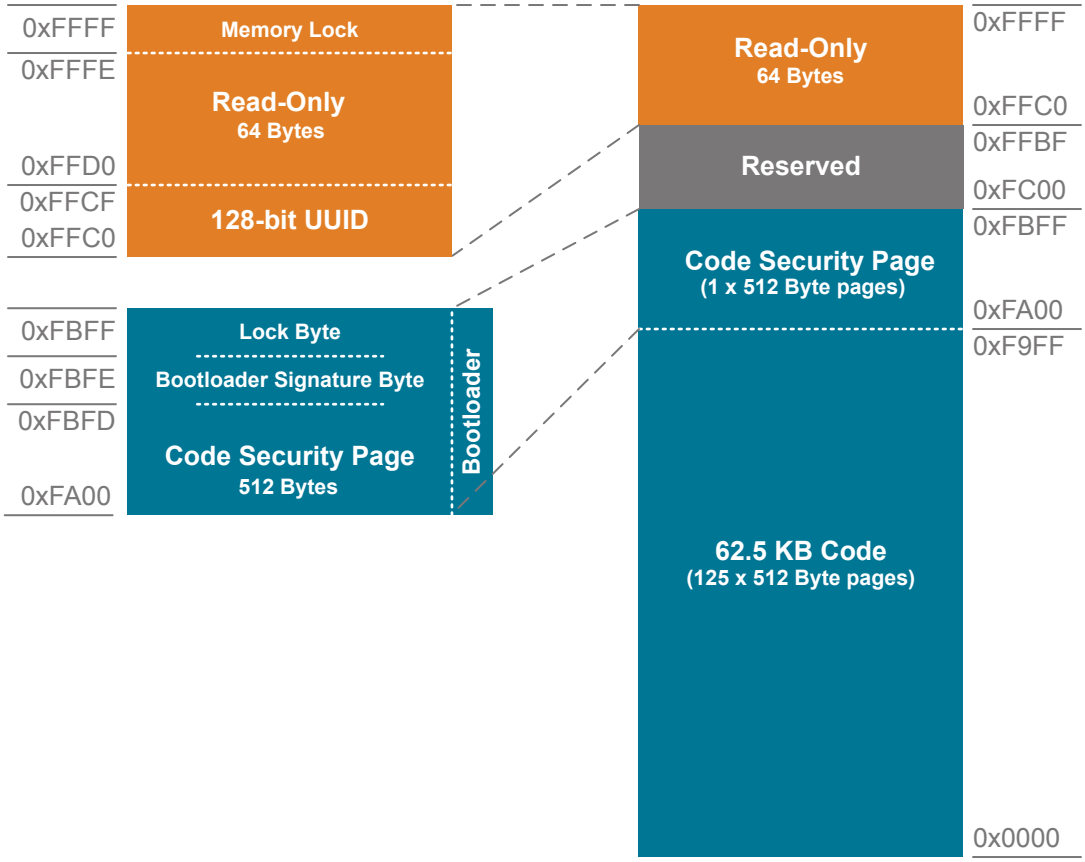


Figure 4.1. Flash Memory Map — 62.5 KB Devices

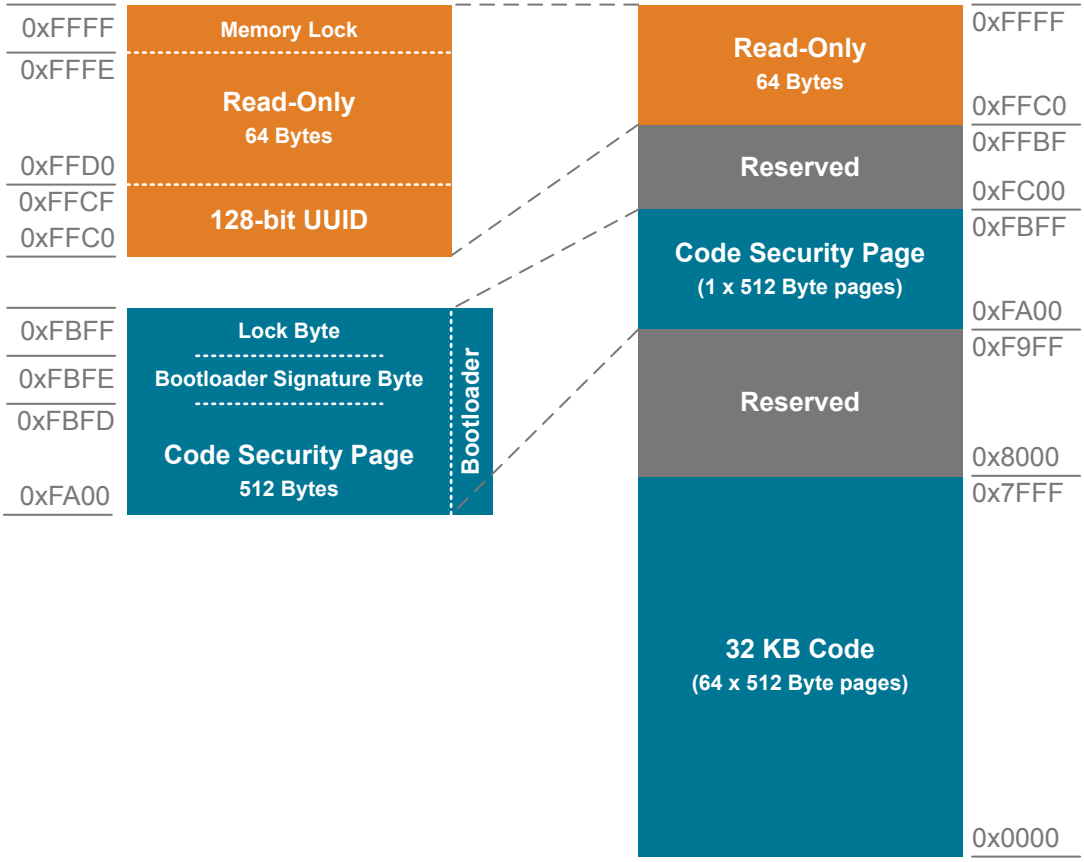


Figure 4.2. Flash Memory Map — 32 KB Devices

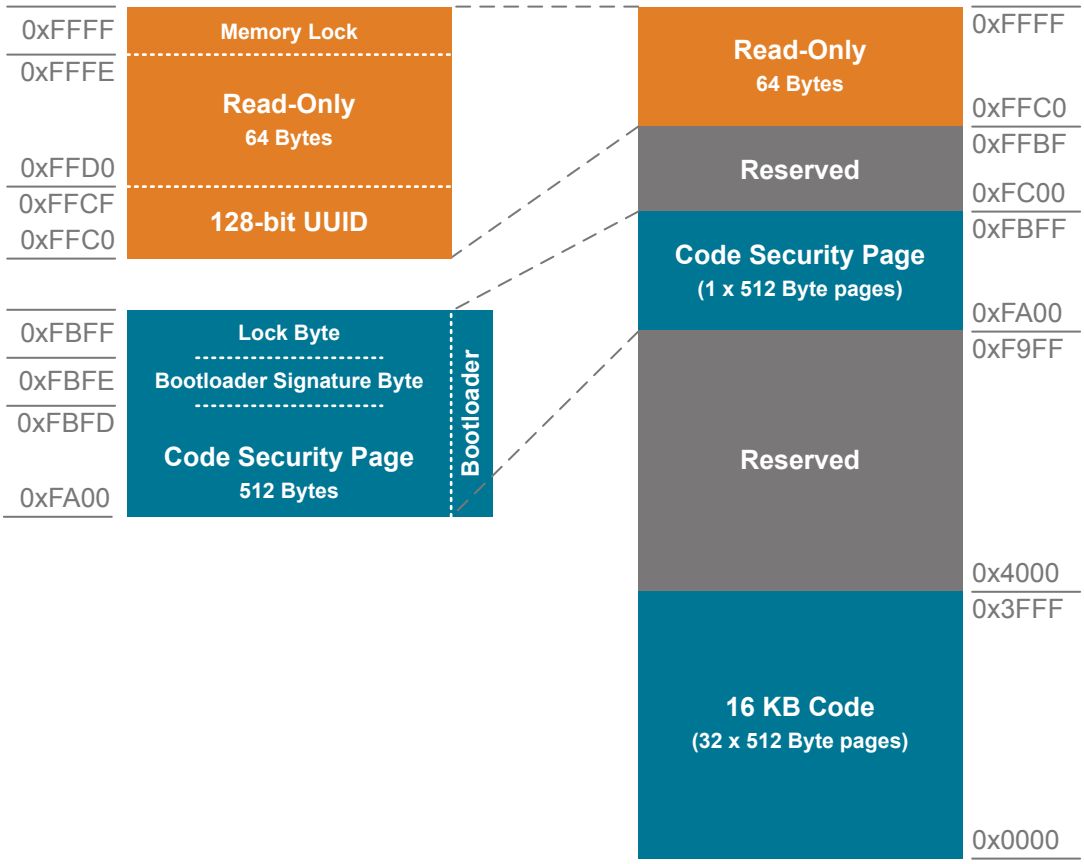


Figure 4.3. Flash Memory Map — 16 KB Devices

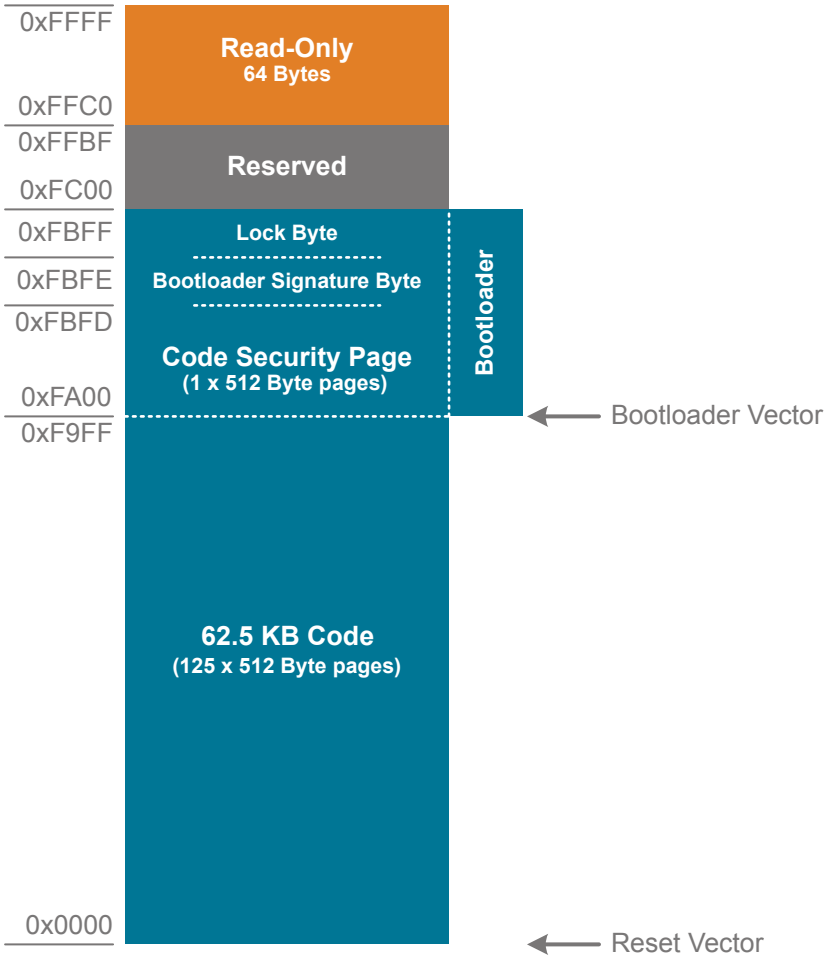


Figure 4.4. Bootloader Flash Memory Map — 62.5 KB Devices

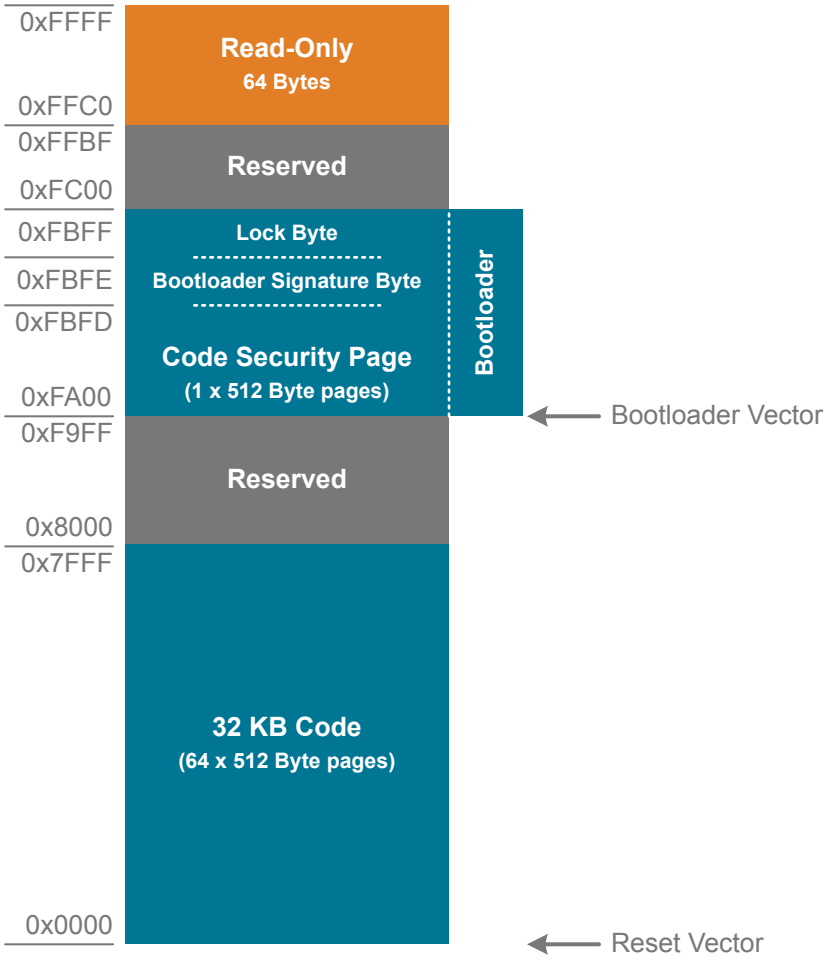


Figure 4.5. Bootloader Flash Memory Map — 32 KB Devices

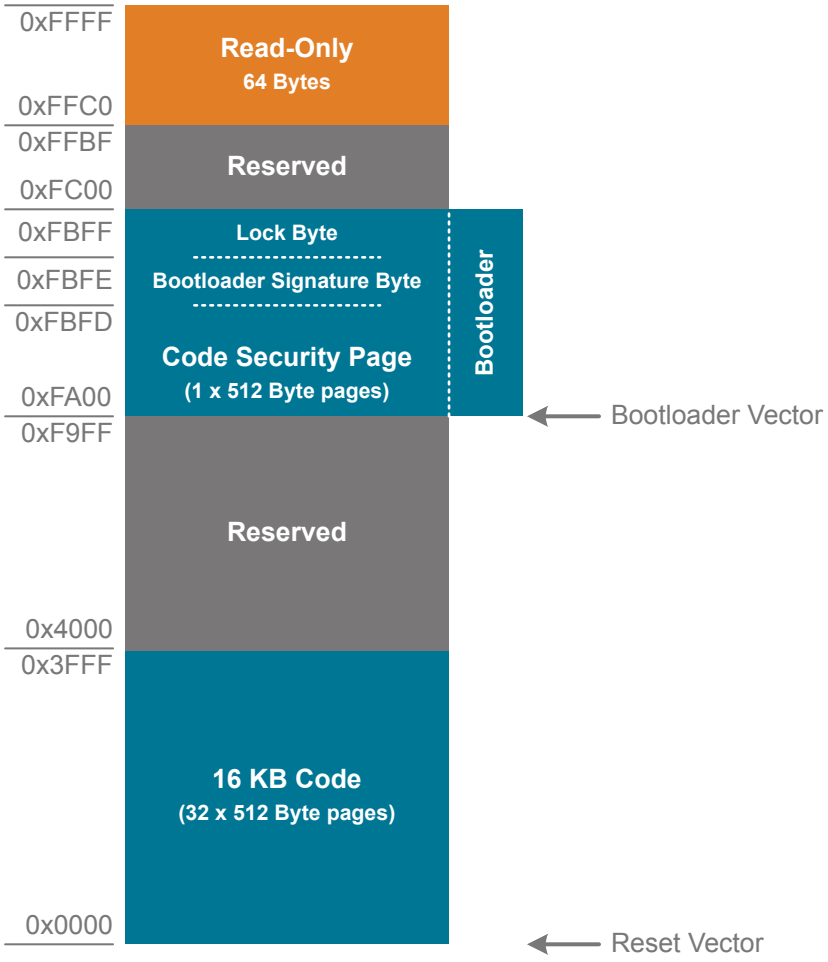


Figure 4.6. Bootloader Flash Memory Map — 16 KB Devices

4.2 Features

The flash memory has the following features:

- Up to 62.5 KB organized in 512-byte sectors.
- In-system programmable from user firmware.
- Security lock to prevent unwanted read/write/erase access.

4.3 Functional Description

4.3.1 Security Options

The CIP-51 provides security options to protect the flash memory from inadvertent modification by software as well as to prevent the viewing of proprietary program code and constants. The Program Store Write Enable (bit PSWE in register PSCTL) and the Program Store Erase Enable (bit PSEE in register PSCTL) bits protect the flash memory from accidental modification by software. PSWE must be explicitly set to 1 before software can modify the flash memory; both PSWE and PSEE must be set to 1 before software can erase flash memory. Additional security features prevent proprietary program code and data constants from being read or altered across the C2 interface.

A Security Lock Byte located in flash user space offers protection of the flash program memory from access (reads, writes, or erases) by unprotected code or the C2 interface. See the specific device memory map for the location of the security byte. The flash security mechanism allows the user to lock "n" flash pages, starting at page 0, where "n" is the 1s complement number represented by the Security Lock Byte. Silicon Labs recommends that the flash memory be locked after the production programming step in applications where code security is a concern. Some devices may also include a read-only area in the flash memory space for constants such as UID and calibration values.

Note: The page containing the flash Security Lock Byte is unlocked when no other flash pages are locked (all bits of the Lock Byte are 1) and locked when any other flash pages are locked (any bit of the Lock Byte is 0).

Table 4.1. Security Byte Decoding

Security Lock Byte	111111101b
1s Complement	00000010b
Flash Pages Locked	3 (First two flash pages + Lock Byte Page)

The level of flash security depends on the flash access method. The three flash access methods that can be restricted are reads, writes, and erases from the C2 debug interface, user firmware executing on unlocked pages, and user firmware executing on locked pages.

Table 4.2. Flash Security Summary—Firmware Permissions

	Permissions according to the area firmware is executing from:	
Target Area for Read / Write / Erase	**Unlocked Page**	**Locked Page**
Any Unlocked Page	[R] [W] [E]	[R] [W] [E]
Locked Page (except security page)	reset	[R] [W] [E]
Locked Security Page	reset	[R] [W]
Read-Only Area	[R]	[R]
Reserved Area	reset	reset
[R] = Read permitted [W] = Write permitted [E] = Erase permitted reset = Flash error reset triggered n/a = Not applicable		

Table 4.3. Flash Security Summary—C2 Permissions

Target Area for Read / Write / Erase	Permissions from C2 interface
Any Unlocked Page	[R] [W] [E]
Any Locked Page	Device Erase Only
Read-Only Area	[R]
Reserved Area	None
[R] = Read permitted [W] = Write permitted [E] = Erase permitted Device Erase Only = No read, write, or individual page erase is allowed. Must erase entire flash space. None = Read, write and erase are not permitted	

4.3.2 Programming the Flash Memory

Writes to flash memory clear bits from logic 1 to logic 0 and can be performed on single byte locations. Flash erasures set bits back to logic 1 and occur only on full pages. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operation is not required. Code execution is stalled during a flash write/erase operation.

The simplest means of programming the flash memory is through the C2 interface using programming tools provided by Silicon Labs or a third party vendor. Firmware may also be loaded into the device to implement code-loader functions or allow non-volatile data storage. To ensure the integrity of flash contents, it is strongly recommended that the on-chip supply monitor be enabled in any system that includes code that writes and/or erases flash memory from software.

4.3.2.1 Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The FLKEY register must be written with the correct key codes, in sequence, before flash operations may be performed. The key codes are 0xA5 and 0xF1. The timing does not matter, but the codes must be written in order. If the key codes are written out of order or the wrong codes are written, flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a flash write or erase is attempted before the key codes have been written properly. The flash lock resets after each write or erase; the key codes must be written again before another flash write or erase operation can be performed.

4.3.2.2 Flash Page Erase Procedure

The flash memory is erased one page at a time by firmware using the MOVX write instruction with the address targeted to any byte within the page. Before erasing a page of flash memory, flash write and erase operations must be enabled by setting the PSWE and PSEE bits in the PSCTL register to logic 1 (this directs the MOVX writes to target flash memory and enables page erasure) and writing the flash key codes in sequence to the FLKEY register. The PSWE and PSEE bits remain set until cleared by firmware.

Erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire page, perform the following steps:

1. Disable interrupts (recommended).
2. Write the first key code to FLKEY: 0xA5.
3. Write the second key code to FLKEY: 0xF1.
4. Set the PSEE bit (register PSCTL).
5. Set the PSWE bit (register PSCTL).
6. Using the MOVX instruction, write a data byte to any location within the page to be erased.
7. Clear the PSWE and PSEE bits.

4.3.2.3 Flash Byte Write Procedure

The flash memory is written by firmware using the MOVX write instruction with the address and data byte to be programmed provided as normal operands in DPTR and A. Before writing to flash memory using MOVX, flash write operations must be enabled by setting the PSWE bit in the PSCTL register to logic 1 (this directs the MOVX writes to target flash memory) and writing the flash key codes in sequence to the FLKEY register. The PSWE bit remains set until cleared by firmware. A write to flash memory can clear bits to logic 0 but cannot set them. A byte location to be programmed should be erased (already set to 0xFF) before a new value is written.

To write a byte of flash, perform the following steps:

1. Disable interrupts (recommended).
2. Write the first key code to FLKEY: 0xA5.
3. Write the second key code to FLKEY: 0xF1.
4. Set the PSWE bit (register PSCTL).
5. Clear the PSEE bit (register PSCTL).
6. Using the MOVX instruction, write a single data byte to the desired location within the desired page.
7. Clear the PSWE bit.

4.3.3 Flash Write and Erase Precautions

Any system which contains routines which write or erase flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of supply voltage, system clock frequency or temperature. This accidental execution of flash modifying code can result in alteration of flash memory contents causing a system failure that is only recoverable by re-flashing the code in the device.

To help prevent the accidental modification of flash by firmware, hardware restricts flash writes and erasures when the supply monitor is not active and selected as a reset source. As the monitor is enabled and selected as a reset source by default, it is recommended that systems writing or erasing flash simply maintain the default state.

The following sections provide general guidelines for any system which contains routines which write or erase flash from code. Additional flash recommendations and example code can be found in *AN201: Writing to Flash From Firmware*, available from the Silicon Laboratories website.

Voltage Supply Maintenance and the Supply Monitor

- If the system power supply is subject to voltage or current "spikes," add sufficient transient protection devices to the power supply to ensure that the supply voltages listed in the Absolute Maximum Ratings table are not exceeded.
- Make certain that the minimum supply rise time specification is met. If the system cannot meet this rise time specification, then add an external supply brownout circuit to the RSTb pin of the device that holds the device in reset until the voltage supply reaches the lower limit, and re-asserts RSTb if the supply drops below the low supply limit.
- Do not disable the supply monitor. If the supply monitor must be disabled in the system, firmware should be added to the startup routine to enable the on-chip supply monitor and enable the supply monitor as a reset source as early in code as possible. This should be the first set of instructions executed after the reset vector. For C-based systems, this may involve modifying the startup code added by the C compiler. See your compiler documentation for more details. Make certain that there are no delays in software between enabling the supply monitor and enabling the supply monitor as a reset source.

  Note: The supply monitor must be enabled and enabled as a reset source when writing or erasing flash memory. A flash error reset will occur if either condition is not met.

- As an added precaution if the supply monitor is ever disabled, explicitly enable the supply monitor and enable the supply monitor as a reset source inside the functions that write and erase flash memory. The supply monitor enable instructions should be placed just after the instruction to set PSWE to a 1, but before the flash write or erase operation instruction.
- Make certain that all writes to the RSTSRC (Reset Sources) register use direct assignment operators and explicitly do not use the bit-wise operators (such as AND or OR). For example, "RSTSRC = 0x02" is correct. "RSTSRC |= 0x02" is incorrect.
- Make certain that all writes to the RSTSRC register explicitly set the PORSF bit to a 1. Areas to check are initialization code which enables other reset sources, such as the Missing Clock Detector or Comparator, for example, and instructions which force a Software Reset. A global search on "RSTSRC" can quickly verify this.

PSWE Maintenance

- Reduce the number of places in code where the PSWE bit (in register PSCTL) is set to a 1. There should be exactly one routine in code that sets PSWE to a 1 to write flash bytes and one routine in code that sets PSWE and PSEE both to a 1 to erase flash pages.
- Minimize the number of variable accesses while PSWE is set to a 1. Handle pointer address updates and loop variable maintenance outside the "PSWE = 1;... PSWE = 0;" area.
- Disable interrupts prior to setting PSWE to a 1 and leave them disabled until after PSWE has been reset to 0. Any interrupts posted during the flash write or erase operation will be serviced in priority order after the flash operation has been completed and interrupts have been re-enabled by software.
- Make certain that the flash write and erase pointer variables are not located in XRAM. See your compiler documentation for instructions regarding how to explicitly locate variables in different memory areas.
- Add address bounds checking to the routines that write or erase flash memory to ensure that a routine called with an illegal address does not result in modification of the flash.

System Clock

- If operating from an external source, be advised that performance is susceptible to electrical interference and is sensitive to layout and to changes in temperature. If the system is operating in an electrically noisy environment, use the internal oscillator or use an external CMOS clock.
- If operating from the external oscillator, switch to the internal oscillator during flash write or erase operations. The external oscillator can continue to run, and the CPU can switch back to the external oscillator after the flash operation has completed.

4.4 Flash Control Registers

4.4.1 PSCTL: Program Store Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved						PSEE	PSWE
Access	R						RW	RW
Reset	0x00						0	0
SFR Page = ALL; SFR Address: 0x8F								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1	PSEE	0	RW	**Program Store Erase Enable.**

Setting this bit (in combination with PSWE) allows an entire page of flash program memory to be erased. If this bit is logic 1 and flash writes are enabled (PSWE is logic 1), a write to flash memory using the MOVX instruction will erase the entire page that contains the location addressed by the MOVX instruction. The value of the data byte written does not matter.

Value	Name	Description
0	ERASE_DISABLED	Flash program memory erasure disabled.
1	ERASE_ENABLED	Flash program memory erasure enabled.

Bit	Name	Reset	Access	Description
0	PSWE	0	RW	**Program Store Write Enable.**

Setting this bit allows writing a byte of data to the flash program memory using the MOVX write instruction. The flash location should be erased before writing data.

Value	Name	Description
0	WRITE_DISABLED	Writes to flash program memory disabled.
1	WRITE_ENABLED	Writes to flash program memory enabled; the MOVX write instruction targets flash memory.

4.4.2 FLKEY: Flash Lock and Key

Bit	7	6	5	4	3	2	1	0
Name	FLKEY							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xB7								

Bit	Name	Reset	Access	Description
7:0	FLKEY	0x00	RW	**Flash Lock and Key.** Write: This register provides a lock and key function for flash erasures and writes. Flash writes and erases are enabled by writing 0xA5 followed by 0xF1 to the FLKEY register. Flash writes and erases are automatically disabled after the next write or erase is complete. If any writes to FLKEY are performed incorrectly, or if a flash write or erase operation is attempted while these operations are disabled, the flash will be permanently locked from writes or erasures until the next device reset. If an application never writes to flash, it can intentionally lock the flash by writing a non-0xA5 value to FLKEY from firmware. Read: When read, bits 1-0 indicate the current flash lock state. 00: Flash is write/erase locked. 01: The first key code has been written (0xA5). 10: Flash is unlocked (writes/erases allowed). 11: Flash writes/erases are disabled until the next reset.

5. Device Identification

5.1 Device Identification

The SFR map includes registers that may be used to identify the device family (DEVICEID), derivative (DERIVID), and revision (REVID). These SFRs can be read by firmware at runtime to determine the capabilities of the MCU that is executing code. This allows the same firmware image to run on MCUs with different memory sizes and peripherals, and dynamically change functionality to suit the capabilities of that MCU.

5.2 Unique Identifier

A 128-bit universally unique identifier (UUID) is pre-programmed into all devices. The value assigned to a device is random and not sequential, but it is guaranteed unique. The UUID resides in the read-only area of flash memory which cannot be erased or written in the end application. The UUID can be read by firmware or through the debug interface at flash locations 0xFFC0-0xFFCF.

Table 5.1. UID Location in Memory

Device	Flash Addresses
EFM8BB31F64G, EFM8BB31F32G, EFM8BB31F16G	(MSB) 0xFFCF, 0xFFCE, 0xFFCD, 0xFFCC, 0xFFCB, 0xFFCA, 0xFFC9, 0xFFC8, 0xFFC7, 0xFFC6, 0xFFC5, 0xFFC4, 0xFFC3, 0xFFC2, 0xFFC1, 0xFFC0 (LSB)

5.3 Device Identification Registers

5.3.1 DEVICEID: Device Identification

Bit	7	6	5	4	3	2	1	0
Name	DEVICEID							
Access	R							
Reset	0x34							
SFR Page = 0x0; SFR Address: 0xB5								

Bit	Name	Reset	Access	Description
7:0	DEVICEID	0x34	R	**Device ID.** This read-only register returns the 8-bit device ID.

5.3.2 DERIVID: Derivative Identification

Bit	7	6	5	4	3	2	1	0
Name	DERIVID							
Access	R							
Reset	Varies							
SFR Page = 0x0; SFR Address: 0xAD								

Bit	Name	Reset	Access	Description
7:0	DERIVID	Varies	R	**Derivative ID.**

This read-only register returns the 8-bit derivative ID, which can be used by firmware to identify which device in the product family the code is executing on. The '{R}' tag in the part numbers indicates the device revision letter in the ordering code. The revision letter may be determined by decoding the REVID register.

Value	Name	Description
0x01	EFM8BB31F64G_QFN32	EFM8BB31F64G-{R}-QFN32
0x02	EFM8BB31F64G_QFP32	EFM8BB31F64G-{R}-QFP32
0x03	EFM8BB31F64G_QSOP24	EFM8BB31F64G-{R}-QSOP24
0x04	EFM8BB31F64G_QFN24	EFM8BB31F64G-{R}-QFN24
0x05	EFM8BB31F32G_QFN32	EFM8BB31F32G-{R}-QFN32
0x06	EFM8BB31F32G_QFP32	EFM8BB31F32G-{R}-QFP32
0x07	EFM8BB31F32G_QSOP24	EFM8BB31F32G-{R}-QSOP24
0x08	EFM8BB31F32G_QFN24	EFM8BB31F32G-{R}-QFN24
0x09	EFM8BB31F16G_QFN32	EFM8BB31F16G-{R}-QFN32
0x0A	EFM8BB31F16G_QFP32	EFM8BB31F16G-{R}-QFP32
0x0B	EFM8BB31F16G_QSOP24	EFM8BB31F16G-{R}-QSOP24
0x0C	EFM8BB31F16G_QFN24	EFM8BB31F16G-{R}-QFN24
0x11	EFM8BB31F64I_QFN32	EFM8BB31F64I-{R}-QFN32
0x12	EFM8BB31F64I_QFP32	EFM8BB31F64I-{R}-QFP32
0x13	EFM8BB31F64I_QSOP24	EFM8BB31F64I-{R}-QSOP24
0x14	EFM8BB31F64I_QFN24	EFM8BB31F64I-{R}-QFN24
0x15	EFM8BB31F32I_QFN32	EFM8BB31F32I-{R}-QFN32

Bit	Name	Reset	Access	Description
	0x16	EFM8BB31F32I_QFP32		EFM8BB31F32I-{R}-QFP32
	0x17	EFM8BB31F32I_QSOP24		EFM8BB31F32I-{R}-QSOP24
	0x18	EFM8BB31F32I_QFN24		EFM8BB31F32I-{R}-QFN24
	0x19	EFM8BB31F16I_QFN32		EFM8BB31F16I-{R}-QFN32
	0x1A	EFM8BB31F16I_QFP32		EFM8BB31F16I-{R}-QFP32
	0x1B	EFM8BB31F16I_QSOP24		EFM8BB31F16I-{R}-QSOP24
	0x1C	EFM8BB31F16I_QFN24		EFM8BB31F16I-{R}-QFN24
	0x21	EFM8BB31F64A_QFN32		EFM8BB31F64A-{R}-QFN32
	0x22	EFM8BB31F64A_QFP32		EFM8BB31F64A-{R}-QFP32
	0x23	EFM8BB31F64A_QSOP24		EFM8BB31F64A-{R}-QSOP24
	0x24	EFM8BB31F64A_QFN24		EFM8BB31F64A-{R}-QFN24
	0x25	EFM8BB31F32A_QFN32		EFM8BB31F32A-{R}-QFN32
	0x26	EFM8BB31F32A_QFP32		EFM8BB31F32A-{R}-QFP32
	0x27	EFM8BB31F32A_QSOP24		EFM8BB31F32A-{R}-QSOP24
	0x28	EFM8BB31F32A_QFN24		EFM8BB31F32A-{R}-QFN24
	0x29	EFM8BB31F16A_QFN32		EFM8BB31F16A-{R}-QFN32
	0x2A	EFM8BB31F16A_QFP32		EFM8BB31F16A-{R}-QFP32
	0x2B	EFM8BB31F16A_QSOP24		EFM8BB31F16A-{R}-QSOP24
	0x2C	EFM8BB31F16A_QFN24		EFM8BB31F16A-{R}-QFN24

5.3.3 REVID: Revision Identifcation

Bit	7	6	5	4	3	2	1	0
Name	REVID							
Access	R							
Reset	Varies							
SFR Page = 0x0; SFR Address: 0xB6								

Bit	Name	Reset	Access	Description
7:0	REVID	Varies	R	**Revision ID.**
	This read-only register returns the revision ID.			

Value	Name	Description
0x00	REV_A	Revision A.

6. Interrupts

6.1 Introduction

The MCU core includes an extended interrupt system supporting multiple interrupt sources and priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device.

Interrupt sources may have one or more associated interrupt-pending flag(s) located in an SFR local to the associated peripheral. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

If interrupts are enabled for the source, an interrupt request is generated when the interrupt-pending flag is set. As soon as execution of the current instruction is complete, the CPU generates an LCALL to a predetermined address to begin execution of an interrupt service routine (ISR). Each ISR must end with an return-from-interrupt (RETI) instruction, which returns program execution to the next instruction that would have been executed if the interrupt request had not occurred. If interrupts are not enabled, the source interrupt-pending flag is ignored by the hardware and program execution continues as normal. The source interrupt-pending flag is set to logic 1 regardless of whether the interrupt is enabled.

Each interrupt source can be individually enabled or disabled using an associated interrupt enable bit in the IE and EIEn registers. In order for individual interrupt enables to be recognized and trigger an interrupt, however, interrupts must first be globally enabled by setting the global interrupt enable bit (EA) in the IE register to logic 1. Setting the EA bit to logic 0 disables all interrupt sources regardless of the individual interrupt-enable settings.

Some interrupt-pending flags are automatically cleared by the hardware when the CPU vectors to the ISR or by other hardware conditions. Most, however, are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the RETI instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

6.2 Interrupt Sources and Vectors

The CIP51 core supports interrupt sources for each peripheral on the device. Software can simulate an interrupt for many peripherals by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag. Refer to the reference manual section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for that peripheral and the behavior of its interrupt-pending flag(s).

6.2.1 Interrupt Priorities

Each interrupt source can be individually programmed to one of four priority levels. This differs from the traditional two priority levels on the 8051 core. However, the implementation of the extra levels is backwards-compatible with legacy 8051 code.

An interrupt service routine can be preempted by any interrupt of higher priority. Interrupts at the highest priority level cannot be preempted. Each interrupt has two associated priority bits which are used to configure the priority level. For backwards compatibility, the bits are spread across two different registers. The LSBs of the priority setting are located in the IP and EIPn registers, while the MSBs are located in the IPH and EIPnH registers. Priority levels according to the MSB and LSB are decoded in Table 6.1 Configurable Interrupt Priority Decoding on page 62. The lowest priority setting is the default for all interrupts. If two or more interrupts are recognized simultaneously, the interrupt with the highest priority is serviced first. If both interrupts have the same priority level, a fixed order is used to arbitrate, based on the interrupt source's location in the interrupt vector table. Interrupts with a lower number in the vector table have priority. If legacy 8051 operation is desired, the bits of the “high” priority registers (IPH and EIPnH) should all be configured to 0.

Table 6.1. Configurable Interrupt Priority Decoding

Priority MSB (from IPH or EIPnH)	Priority LSB (from IP or EIPn)	Priority Level
0	0	Priority 0 (lowest priority, default)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3 (highest priority)

6.2.2 Interrupt Latency

Interrupt response time depends on the state of the CPU when the interrupt occurs. Pending interrupts are sampled and priority decoded on every system clock cycle. Therefore, the fastest possible response time is 5 system clock cycles: 1 clock cycle to detect the interrupt and 4 clock cycles to complete the LCALL to the ISR. If an interrupt is pending when a RETI is executed, a single instruction is executed before an LCALL is made to service the pending interrupt. Therefore, the maximum response time for an interrupt (when no other interrupt is currently being serviced or the new interrupt is of greater priority) occurs when the CPU is performing an RETI instruction followed by a DIV as the next instruction. In this case, the response time is 18 system clock cycles: 1 clock cycle to detect the interrupt, 5 clock cycles to execute the RETI, 8 clock cycles to complete the DIV instruction and 4 clock cycles to execute the LCALL to the ISR. If the CPU is executing an ISR for an interrupt with equal or higher priority, the new interrupt will not be serviced until the current ISR completes, including the RETI and following instruction. If more than one interrupt is pending when the CPU exits an ISR, the CPU will service the next highest priority interrupt that is pending.

6.2.3 Interrupt Summary

Table 6.2. Interrupt Priority Table

Interrupt Source	Vector	Priority	Primary Enable	Auxiliary Enable(s)	Pending Flag(s)
Reset	0x0000	Top	-	-	-
External Interrupt 0	0x0003	0	IE_EX0	-	TCON_IE0
Timer 0 Overflow	0x000B	1	IE_ET0	-	TCON_TF0
External Interrupt 1	0x0013	2	IE_EX1	-	TCON_IE1
Timer 1 Overflow	0x001B	3	IE_ET1	-	TCON_TF1
UART0	0x0023	4	IE_ES0	-	SCON0_RI SCON0_TI
Timer 2 Overflow / Capture	0x002B	5	IE_ET2	TMR2CN0_TF2CEN TMR2CN0_TF2LEN	TMR2CN0_TF2H TMR2CN0_TF2L
SPI0	0x0033	6	IE_ESPI0	SPI0FCN0_RFRQE SPI0FCN0_TFRQE SPI0FCN1_SPIFEN	SPI0CN0_MODF SPI0CN0_RXOVRN SPI0CN0_SPIF SPI0CN0_WCOL SPI0FCN1_RFRQ SPI0FCN1_TFRQ
SMBus 0	0x003B	7	EIE1_ESMB0	-	SMB0CN0_SI
Port Match	0x0043	8	EIE1_EMAT	-	-
ADC0 Window Compare	0x004B	9	EIE1_EWADC0	-	ADC0CN0_ADWINT
ADC0 End of Conversion	0x0053	10	EIE1_EADC0	-	ADC0CN0_ADINT
PCA0	0x005B	11	EIE1_EPCA0	PCA0CPM0_ECCF PCA0CPM1_ECCF PCA0CPM2_ECCF PCA0PWM_ECOV	PCA0CN0_CCF0 PCA0CN0_CCF1 PCA0CN0_CCF2 PCA0CN0_CF PCA0PWM_COVF
Comparator 0	0x0063	12	EIE1_ECP0	CMP0MD_CPRIE CMP0MD_CPFIE	CMP0CN0_CPFIF CMP0CN0_CPRIF
Comparator 1	0x006B	13	EIE1_ECP1	CMP1MD_CPFIE CMP1MD_CPRIE	CMP1CN0_CPFIF CMP1CN0_CPRIF
Timer 3 Overflow / Capture	0x0073	14	EIE1_ET3	TMR3CN0_TF3CEN TMR3CN0_TF3LEN	TMR3CN0_TF3H TMR3CN0_TF3L

Interrupt Source	Vector	Priority	Primary Enable	Auxiliary Enable(s)	Pending Flag(s)
UART1	0x007B	15	EIE2_ES1	UART1FCN0_RFRQE UART1FCN0_TFRQE UART1FCN1_RIE UART1FCN1_RXTO UART1FCN1_TIE	SCON1_RI SCON1_TI UART1FCN1_RFRQ UART1FCN1_TFRQ
I2C0 Slave	0x0083	16	EIE2_EI2C0	I2C0FCN0_RFRQE I2C0FCN0_TFRQE	I2C0STAT_I2C0INT I2C0FCN1_RFRQ I2C0FCN1_TFRQ
Timer 4 Overflow / Capture	0x008B	17	EIE2_ET4	TMR4CN0_TF4CEN TMR4CN0_TF4LEN	TMR4CN0_TF4H TMR4CN0_TF4L
Timer 5 Overflow / Capture	0x0093	18	EIE2_ET5	TMR5CN0_TF5CEN TMR5CN0_TF5LEN	TMR5CN0_TF5H TMR5CN0_TF5L
Configurable Logic	0x009B	19	EIE2_CL0	CLIE0_C0FIE CLIE0_C0RIE CLIE0_C1FIE CLIE0_C1RIE CLIE0_C2FIE CLIE0_C2RIE CLIE0_C3FIE CLIE0_C3RIE	CLIF0_C0FIF CLIF0_C0RIF CLIF0_C1FIF CLIF0_C1RIF CLIF0_C2FIF CLIF0_C2RIF CLIF0_C3FIF CLIF0_C3RIF

6.3 Interrupt Control Registers

6.3.1 IE: Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	EA	ESPI0	ET2	ES0	ET1	EX1	ET0	EX0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xA8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	EA	0	RW	**All Interrupts Enable.**
	Globally enables/disables all interrupts and overrides individual interrupt mask settings.			
	Value	Name		Description
	0	DISABLED		Disable all interrupt sources.
	1	ENABLED		Enable each interrupt according to its individual mask setting.
6	ESPI0	0	RW	**SPI0 Interrupt Enable.**
	This bit sets the masking of the SPI0 interrupts.			
	Value	Name		Description
	0	DISABLED		Disable all SPI0 interrupts.
	1	ENABLED		Enable interrupt requests generated by SPI0.
5	ET2	0	RW	**Timer 2 Interrupt Enable.**
	This bit sets the masking of the Timer 2 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable Timer 2 interrupt.
	1	ENABLED		Enable interrupt requests generated by the TF2L or TF2H flags.
4	ES0	0	RW	**UART0 Interrupt Enable.**
	This bit sets the masking of the UART0 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable UART0 interrupt.
	1	ENABLED		Enable UART0 interrupt.
3	ET1	0	RW	**Timer 1 Interrupt Enable.**
	This bit sets the masking of the Timer 1 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all Timer 1 interrupt.
	1	ENABLED		Enable interrupt requests generated by the TF1 flag.

Bit	Name	Reset	Access	Description
2	EX1	0	RW	**External Interrupt 1 Enable.**
	This bit sets the masking of External Interrupt 1.			
	Value	Name		Description
	0	DISABLED		Disable external interrupt 1.
	1	ENABLED		Enable interrupt requests generated by the INT1 input.
1	ET0	0	RW	**Timer 0 Interrupt Enable.**
	This bit sets the masking of the Timer 0 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all Timer 0 interrupt.
	1	ENABLED		Enable interrupt requests generated by the TF0 flag.
0	EX0	0	RW	**External Interrupt 0 Enable.**
	This bit sets the masking of External Interrupt 0.			
	Value	Name		Description
	0	DISABLED		Disable external interrupt 0.
	1	ENABLED		Enable interrupt requests generated by the INT0 input.

6.3.2 IP: Interrupt Priority

Bit	7	6	5	4	3	2	1	0
Name	Reserved	PSPI0	PT2	PS0	PT1	PX1	PT0	PX0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	1	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xB8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	PSPI0	0	RW	**Serial Peripheral Interface (SPI0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the SPI0 interrupt.
5	PT2	0	RW	**Timer 2 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 2 interrupt.
4	PS0	0	RW	**UART0 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the UART0 interrupt.
3	PT1	0	RW	**Timer 1 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 1 interrupt.
2	PX1	0	RW	**External Interrupt 1 Priority Control LSB.** This bit sets the LSB of the priority field for the External Interrupt 1 interrupt.
1	PT0	0	RW	**Timer 0 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 0 interrupt.
0	PX0	0	RW	**External Interrupt 0 Priority Control LSB.** This bit sets the LSB of the priority field for the External Interrupt 0 interrupt.

6.3.3 IPH: Interrupt Priority High

Bit	7	6	5	4	3	2	1	0
Name	Reserved	PHSPI0	PHT2	PHS0	PHT1	PHX1	PHT0	PHX0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	1	0	0	0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xF2								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	PHSPI0	0	RW	**Serial Peripheral Interface (SPI0) Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the SPI0 interrupt.			
5	PHT2	0	RW	**Timer 2 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Timer 2 interrupt.			
4	PHS0	0	RW	**UART0 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the UART0 interrupt.			
3	PHT1	0	RW	**Timer 1 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Timer 1 interrupt.			
2	PHX1	0	RW	**External Interrupt 1 Priority Control MSB.**
	This bit sets the MSB of the priority field for the External Interrupt 1 interrupt.			
1	PHT0	0	RW	**Timer 0 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Timer 0 interrupt.			
0	PHX0	0	RW	**External Interrupt 0 Priority Control MSB.**
	This bit sets the MSB of the priority field for the External Interrupt 0 interrupt.			

6.3.4 EIE1: Extended Interrupt Enable 1

Bit	7	6	5	4	3	2	1	0
Name	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	EMAT	ESMB0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xE6								

Bit	Name	Reset	Access	Description
7	ET3	0	RW	**Timer 3 Interrupt Enable.**
	This bit sets the masking of the Timer 3 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable Timer 3 interrupts.
	1	ENABLED		Enable interrupt requests generated by the TF3L or TF3H flags.
6	ECP1	0	RW	**Comparator1 (CP1) Interrupt Enable.**
	This bit sets the masking of the CP1 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CP1 interrupts.
	1	ENABLED		Enable interrupt requests generated by the comparator 1 CPRIF or CPFIF flags.
5	ECP0	0	RW	**Comparator0 (CP0) Interrupt Enable.**
	This bit sets the masking of the CP0 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CP0 interrupts.
	1	ENABLED		Enable interrupt requests generated by the comparator 0 CPRIF or CPFIF flags.
4	EPCA0	0	RW	**Programmable Counter Array (PCA0) Interrupt Enable.**
	This bit sets the masking of the PCA0 interrupts.			
	Value	Name		Description
	0	DISABLED		Disable all PCA0 interrupts.
	1	ENABLED		Enable interrupt requests generated by PCA0.
3	EADC0	0	RW	**ADC0 Conversion Complete Interrupt Enable.**
	This bit sets the masking of the ADC0 Conversion Complete interrupt.			
	Value	Name		Description
	0	DISABLED		Disable ADC0 Conversion Complete interrupt.
	1	ENABLED		Enable interrupt requests generated by the ADINT flag.

Bit	Name	Reset	Access	Description
2	EWADC0	0	RW	**ADC0 Window Comparison Interrupt Enable.**
	This bit sets the masking of ADC0 Window Comparison interrupt.			
	Value	Name		Description
	0	DISABLED		Disable ADC0 Window Comparison interrupt.
	1	ENABLED		Enable interrupt requests generated by ADC0 Window Compare flag (ADWINT).
1	EMAT	0	RW	**Port Match Interrupts Enable.**
	This bit sets the masking of the Port Match Event interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all Port Match interrupts.
	1	ENABLED		Enable interrupt requests generated by a Port Match.
0	ESMB0	0	RW	**SMBus (SMB0) Interrupt Enable.**
	This bit sets the masking of the SMB0 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all SMB0 interrupts.
	1	ENABLED		Enable interrupt requests generated by SMB0.

6.3.5 EIP1: Extended Interrupt Priority 1 Low

Bit	7	6	5	4	3	2	1	0
Name	PT3	PCP1	PCP0	PPCA0	PADC0	PWADC0	PMAT	PSMB0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xBB								

Bit	Name	Reset	Access	Description
7	PT3	0	RW	**Timer 3 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 3 interrupt.
6	PCP1	0	RW	**Comparator1 (CP1) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the CP1 interrupt.
5	PCP0	0	RW	**Comparator0 (CP0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the CP0 interrupt.
4	PPCA0	0	RW	**Programmable Counter Array (PCA0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the PCA0 interrupt.
3	PADC0	0	RW	**ADC0 Conversion Complete Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the ADC0 Conversion Complete interrupt.
2	PWADC0	0	RW	**ADC0 Window Comparator Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the ADC0 Window interrupt.
1	PMAT	0	RW	**Port Match Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Port Match Event interrupt.
0	PSMB0	0	RW	**SMBus (SMB0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the SMB0 interrupt.

6.3.6 EIP1H: Extended Interrupt Priority 1 High

Bit	7	6	5	4	3	2	1	0
Name	PHT3	PHCP1	PHCP0	PHPCA0	PHADC0	PHWADC0	PHMAT	PHSMB0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xEE								

Bit	Name	Reset	Access	Description
7	PHT3	0	RW	**Timer 3 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Timer 3 interrupt.			
6	PHCP1	0	RW	**Comparator1 (CP1) Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the CP1 interrupt.			
5	PHCP0	0	RW	**Comparator0 (CP0) Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the CP0 interrupt.			
4	PHPCA0	0	RW	**Programmable Counter Array (PCA0) Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the PCA0 interrupt.			
3	PHADC0	0	RW	**ADC0 Conversion Complete Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the ADC0 Conversion Complete interrupt.			
2	PHWADC0	0	RW	**ADC0 Window Comparator Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the ADC0 Window interrupt.			
1	PHMAT	0	RW	**Port Match Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Port Match Event interrupt.			
0	PHSMB0	0	RW	**SMBus (SMB0) Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the SMB0 interrupt.			

6.3.7 EIE2: Extended Interrupt Enable 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved			CL0	ET5	ET4	EI2C0	ES1
Access	RW			RW	RW	RW	RW	RW
Reset	0x0			0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xF3								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	CL0	0	RW	**Configurable Logic (CL0) Interrupt Enable.**
	This bit sets the masking of the CL0 interrupts.			
	Value	Name		Description
	0	DISABLED		Disable CL0 interrupts.
	1	ENABLED		Enable interrupt requests generated by CL0.
3	ET5	0	RW	**Timer 5 Interrupt Enable.**
	This bit sets the masking of the Timer 5 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable Timer 5 interrupts.
	1	ENABLED		Enable interrupt requests generated by the TF5L or TF5H flags.
2	ET4	0	RW	**Timer 4 Interrupt Enable.**
	This bit sets the masking of the Timer 4 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable Timer 4 interrupts.
	1	ENABLED		Enable interrupt requests generated by the TF4L or TF4H flags.
1	EI2C0	0	RW	**I2C0 Slave Interrupt Enable.**
	This bit sets the masking of the I2C0 slave interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all I2C0 slave interrupts.
	1	ENABLED		Enable interrupt requests generated by the I2C0 slave.
0	ES1	0	RW	**UART1 Interrupt Enable.**
	This bit sets the masking of the UART1 interrupts.			
	Value	Name		Description
	0	DISABLED		Disable UART1 interrupts.
	1	ENABLED		Enable UART1 interrupts.

6.3.8 EIP2: Extended Interrupt Priority 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved			PCL0	PT5	PT4	PI2C0	PS1
Access	RW			RW	RW	RW	RW	RW
Reset	0x0			0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xED								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	PCL0	0	RW	**Configurable Logic (CL0) Interrupt Priority Control LSB.**
	This bit sets the LSB of the priority field for the CL0 interrupt.			
3	PT5	0	RW	**Timer 5 Interrupt Priority Control LSB.**
	This bit sets the LSB of the priority field for the Timer 5 interrupt.			
2	PT4	0	RW	**Timer 4 Interrupt Priority Control LSB.**
	This bit sets the LSB of the priority field for the Timer 4 interrupt.			
1	PI2C0	0	RW	**I2C0 Slave Interrupt Priority Control LSB.**
	This bit sets the LSB of the priority field for the I2C0 Slave interrupt.			
0	PS1	0	RW	**UART1 Interrupt Priority Control LSB.**
	This bit sets the LSB of the priority field for the UART1 interrupt.			

6.3.9 EIP2H: Extended Interrupt Priority 2 High

Bit	7	6	5	4	3	2	1	0
Name	Reserved			PHCL0	PHT5	PHT4	PHI2C0	PHS1
Access	RW			RW	RW	RW	RW	RW
Reset	0x0			0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xF6								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	PHCL0	0	RW	**Configurable Logic (CL0) Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the CL0 interrupt.			
3	PHT5	0	RW	**Timer 5 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Timer 5 interrupt.			
2	PHT4	0	RW	**Timer 4 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the Timer 4 interrupt.			
1	PHI2C0	0	RW	**I2C0 Slave Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the I2C0 Slave interrupt.			
0	PHS1	0	RW	**UART1 Interrupt Priority Control MSB.**
	This bit sets the MSB of the priority field for the UART1 interrupt.			

7. Power Management and Internal Regulator

7.1 Introduction

All internal circuitry draws power from the VDD supply pin. External I/O pins are powered from the VIO supply voltage (or VDD on devices without a separate VIO connection), while most of the internal circuitry is supplied by an on-chip LDO regulator. Control over the device power can be achieved by enabling/disabling individual peripherals as needed. Each analog peripheral can be disabled when not in use and placed in low power mode. Digital peripherals, such as timers and serial buses, have their clocks gated off and draw little power when they are not in use.

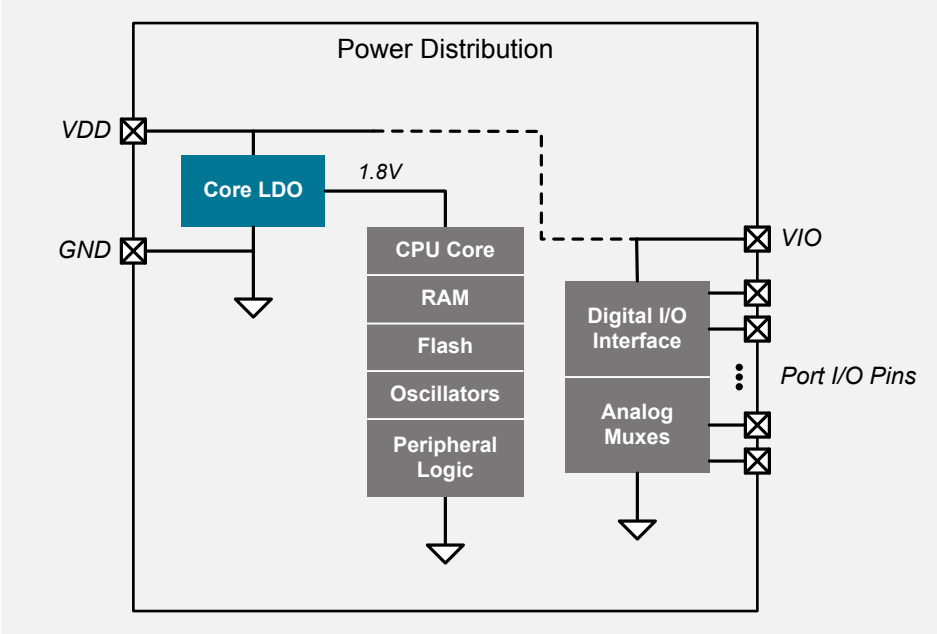


Figure 7.1. Power System Block Diagram

Table 7.1. Power Modes

Power Mode	Details	Mode Entry	Wake-Up Sources
Normal	Core and all peripherals clocked and fully operational		
Idle	• Core halted • All peripherals clocked and fully operational • Code resumes execution on wake event	Set IDLE bit in PCON0	Any interrupt
Suspend	• Core and peripheral clocks halted • HFOSC0 and HFOSC1 oscillators stopped • Regulator in normal bias mode for fast wake • Timer 3 and 4 may clock from LFOSC0 • Code resumes execution on wake event	1. Switch SYSCLK to HFOSC0 2. Set SUSPEND bit in PCON1	• Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Low Level • CLUn Interrupt-Enabled Event

Power Mode	Details	Mode Entry	Wake-Up Sources
Stop	• All internal power nets shut down • Pins retain state • Exit on any reset source	1. Clear STOPCF bit in REG0CN 2. Set STOP bit in PCON0	Any reset source
Snooze	• Core and peripheral clocks halted • HFOSC0 and HFOSC1 oscillators stopped • Regulator in low bias current mode for energy savings • Timer 3 and 4 may clock from LFOSC0 • Code resumes execution on wake event	1. Switch SYSCLK to HFOSC0 2. Set SNOOZE bit in PCON1	• Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Low Level • CLUn Interrupt-Enabled Event
Shutdown	• All internal power nets shut down • Pins retain state • Exit on pin or power-on reset	1. Set STOPCF bit in REG0CN 2. Set STOP bit in PCON0	• RSTb pin reset • Power-on reset

7.2 Features

The power management features of these devices include the following:

- Supports five power modes:
  1. Normal mode: Core and all peripherals fully operational.
  2. Idle mode: Core halted, peripherals fully operational, core waiting for interrupt to continue.
  3. Suspend mode: High-frequency internal clocks halted, select peripherals active, waiting for wake signal to continue.
  4. Snooze mode: High-frequency internal clocks halted, select peripherals active, regulators in low-power mode, waiting for wake signal to continue.
  5. Shutdown mode: All clocks stopped and internal LDO shut off, device waiting for POR or pin reset.

  Note: Legacy 8051 Stop mode is also supported, but Suspend and Snooze offer more functionality with better power consumption.
- Internal Core LDO:
  - Supplies power to majority of blocks.
  - Low power consumption in Snooze mode, can be shut down completely in Shutdown mode.

7.3 Idle Mode

In idle mode, CPU core execution is halted while any enabled peripherals and clocks remain active. Power consumption in idle mode is dependent upon the system clock frequency and any active peripherals.

Setting the IDLE bit in the PCON0 register causes the hardware to halt the CPU and enter idle mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. All analog and digital peripherals can remain active during idle mode.

Idle mode is terminated when an enabled interrupt is asserted or a reset occurs. The assertion of an enabled interrupt will cause the IDLE bit to be cleared and the CPU to resume operation. The pending interrupt will be serviced and the next instruction to be executed after the return from interrupt (RETI) will be the instruction immediately following the one that set the IDLE bit. If idle mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

Note: If the instruction following the write of the IDLE bit is a single-byte instruction and an interrupt occurs during the execution phase of the instruction that sets the IDLE bit, the CPU may not wake from idle mode when a future interrupt occurs. Therefore, instructions that set the IDLE bit should be followed by an instruction that has two or more opcode bytes. For example:

```
 // in 'C':
PCON0 |= 0x01; // set IDLE bit
PCON0 = PCON0; // ... followed by a 3-cycle dummy instruction

; in assembly:
ORL PCON0, #01h ; set IDLE bit
MOV PCON0, PCON0 ; ... followed by a 3-cycle dummy instruction
```

If enabled, the Watchdog Timer (WDT) will eventually cause an internal watchdog reset and thereby terminate idle mode. This feature protects the system from an unintended permanent shutdown in the event of an inadvertent write to the PCON0 register. If this behavior is not desired, the WDT may be disabled by software prior to entering the idle mode if the WDT was initially configured to allow this operation. This provides the opportunity for additional power savings, allowing the system to remain in the idle mode indefinitely, waiting for an external stimulus to wake up the system.

7.4 Suspend Mode

Suspend mode is entered by setting the SUSPEND bit while operating from the internal 24.5 MHz oscillator (HFOSC0). Upon entry into suspend mode, the hardware halts the high-frequency internal oscillator and goes into a low power state as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data.

Suspend mode is terminated by any enabled wake or reset source. When suspend mode is terminated, the device will continue execution on the instruction following the one that set the SUSPEND bit. If the wake event was configured to generate an interrupt, the interrupt will be serviced upon waking the device. If suspend mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

7.5 Stop Mode

In stop mode, the CPU is halted and peripheral clocks are stopped. Analog peripherals remain in their selected states.

Setting the STOP bit in the PCON0 register causes the controller core to enter stop mode as soon as the instruction that sets the bit completes execution. Before entering stop mode, the system clock must be sourced by HFOSC0. In stop mode, the CPU and internal clocks are stopped. Analog peripherals may remain enabled, but will not be provided a clock. Each analog peripheral may be shut down individually by firmware prior to entering stop mode. Stop mode can only be terminated by an internal or external reset. On reset, the device performs the normal reset sequence and begins program execution at address 0x0000.

If enabled as a reset source, the missing clock detector will cause an internal reset and thereby terminate the stop mode. If this reset is undesirable in the system, and the CPU is to be placed in stop mode for longer than the missing clock detector timeout, the missing clock detector should be disabled in firmware prior to setting the STOP bit.

7.6 Snooze Mode

Snooze mode is entered by setting the SNOOZE bit while operating from the internal 24.5 MHz oscillator (HFOSC0). Upon entry into snooze mode, the hardware halts both of the high-frequency internal oscillators and goes into a low power state as soon as the instruction that sets the bit completes execution. The internal LDO is then placed into a low-current standby mode. All internal registers and memory maintain their original data.

Snooze mode is terminated by any enabled wake or reset source. When snooze mode is terminated, the LDO is returned to normal operating conditions and the device will continue execution on the instruction following the one that set the SNOOZE bit. If the wake event was configured to generate an interrupt, the interrupt will be serviced upon waking the device. If snooze mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

7.7 Shutdown Mode

In shutdown mode, the CPU is halted and the internal LDO is powered down. External I/O will retain their configured states.

To enter shutdown mode, firmware should set the STOPCF bit in the regulator control register to 1, and then set the STOP bit in PCON0. In shutdown mode, the RSTb pin and a full power cycle of the device are the only methods of generating a reset and waking the device.

Note: In shutdown mode, all internal device circuitry is powered down, and no RAM nor registers are retained. The debug circuitry will not be able to connect to a device while it is in shutdown mode. Coming out of shutdown mode, whether by POR or pin reset, will appear as a power-on reset of the device.

7.8 Determining Wake Events (Suspend and Snooze Mode)

Upon exit from Suspend or Snooze mode, the wake-up flags in the PSTAT0 register can be read to determine the event(s) which caused the device to wake up. Wake-up flags in PSTAT0 should be cleared by firmware.

7.9 Power Management Control Registers

7.9.1 PCON0: Power Control

Bit	7	6	5	4	3	2	1	0
Name	GF5	GF4	GF3	GF2	GF1	GF0	STOP	IDLE
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0x87								

Bit	Name	Reset	Access	Description
7	GF5	0	RW	**General Purpose Flag 5.** This flag is a general purpose flag for use under firmware control.
6	GF4	0	RW	**General Purpose Flag 4.** This flag is a general purpose flag for use under firmware control.
5	GF3	0	RW	**General Purpose Flag 3.** This flag is a general purpose flag for use under firmware control.
4	GF2	0	RW	**General Purpose Flag 2.** This flag is a general purpose flag for use under firmware control.
3	GF1	0	RW	**General Purpose Flag 1.** This flag is a general purpose flag for use under firmware control.
2	GF0	0	RW	**General Purpose Flag 0.** This flag is a general purpose flag for use under firmware control.
1	STOP	0	RW	**Stop Mode Select.** Setting this bit will place the CIP-51 in Stop mode. This bit will always be read as 0.
0	IDLE	0	RW	**Idle Mode Select.** Setting this bit will place the CIP-51 in Idle mode. This bit will always be read as 0.

7.9.2 PCON1: Power Control 1

Bit	7	6	5	4	3	2	1	0
Name	SNOOZE	SUSPEND	Reserved					PINRSTMD
Access	RW	RW	R					RW
Reset	0	0	0x01					0
SFR Page = ALL; SFR Address: 0xCD								

Bit	Name	Reset	Access	Description
7	SNOOZE	0	RW	**Snooze Mode Select.** Setting this bit will place the device in snooze mode. High speed oscillators will be halted the SYSCLK signal will be gated off, and the internal regulator will be placed in a low power state.
6	SUSPEND	0	RW	**Suspend Mode Select.** Setting this bit will place the device in suspend mode. High speed oscillators will be halted and the SYSCLK signal will be gated off.
5:1	*Reserved*	*Must write reset value.*		
0	PINRSTMD	0	RW	**Pin Reset Mode.** This bit controls the behavior of the GPIO pin configuration (push-pull, PxMDIN, port latch, XBARE) when any reset event occurs. This bit is a sticky bit and will be retained across all reset events except POR. After a POR event, this bit is reset to 0.

Value	Name	Description
0	RESET	GPIO logic is reset when any reset event is asserted.
1	RETAIN	Pins will retain state across any reset except for power-on-reset events. Note that although pin configurations are maintained, the values of the pin control registers are reset. Registers PnMDIN, PnMDOUT, Pn, and the XBARE bit may not reflect the actual pin configuration at this time. New values written to these registers will take effect upon the write event.

7.9.3 PSTAT0: PMU Status 0

Bit	7	6	5	4	3	2	1	0
Name	Reserved		CL0WK	SPI0WK	I2C0WK	TMR4WK	PMATWK	CPT0WK
Access	R		RW	RW	RW	RW	RW	RW
Reset	0x0		0	0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xAA								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	CL0WK	0	RW	**Configurable Logic Wake Flag.** This bit is set to 1 if an interrupt-enabled configurable logic event occurred during suspend or snooze mode. This bit should be cleared by firmware.
4	SPI0WK	0	RW	**SPI0 Slave Wake Flag.** This bit is set to 1 if the SPI slave received a byte during suspend or snooze mode. This bit should be cleared by firmware.
3	I2C0WK	0	RW	**I2C0 Slave Wake Flag.** This bit is set to 1 if an I2C slave address match event occurred during suspend or snooze mode. This bit should be cleared by firmware.
2	TMR4WK	0	RW	**Timer 4 Wake Flag.** This bit is set to 1 if a Timer 4 event occurred during suspend or snooze mode. This bit should be cleared by firmware.
1	PMATWK	0	RW	**Port Match Wake Flag.** This bit is set to 1 if a Port Match event occurred during suspend or snooze mode. This bit should be cleared by firmware.
0	CPT0WK	0	RW	**Comparator 0 Wake Flag.** This bit is set to 1 if a comparator 0 output rising edge occurred during suspend or snooze mode. This bit should be cleared by firmware.

7.9.4 REG0CN: Voltage Regulator 0 Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved				STOPCF	Reserved		
Access	R				RW	R		
Reset	0x0				0	0x0		
SFR Page = 0x0, 0x20; SFR Address: 0xC9								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	STOPCF	0	RW	**Stop and Shutdown Mode Configuration.** This bit configures the regulator's behavior when the device enters stop mode.

Value	Name	Description
0	ACTIVE	Regulator is still active in stop mode. Any enabled reset source will reset the device.
1	SHUTDOWN	Regulator is shut down in stop mode (device enters Shutdown mode). Only the RSTb pin or power cycle can reset the device.

2:0	*Reserved*	*Must write reset value.*		

8. Clocking and Oscillators

8.1 Introduction

The CPU core and peripheral subsystem may be clocked by both internal and external oscillator resources. By default, the system clock comes up running from the 24.5 MHz oscillator divided by 8.

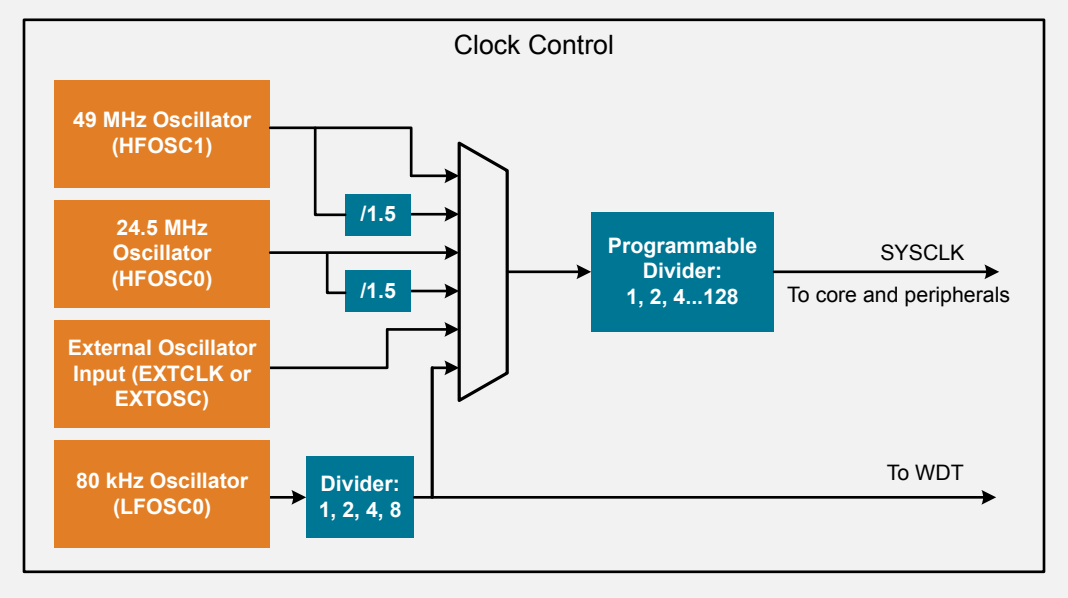


Figure 8.1. Clock Control Block Diagram

8.2 Features

The clock control system offers the following features:

- Provides clock to core and peripherals.
- 24.5 MHz internal oscillator (HFOSC0), accurate to ±2% over supply and temperature corners.
- 49 MHz internal oscillator (HFOSC1), accurate to ±2% over supply and temperature corners.
- 80 kHz low-frequency oscillator (LFOSC0).
- External high frequency crystal, RC, and CMOS clock options (EXTCLK).
- Clock divider with eight settings for flexible clock scaling:
  - Divide the selected clock source by 1, 2, 4, 8, 16, 32, 64, or 128.
  - HFOSC0 and HFOSC1 include 1.5x pre-scalers for further flexibility.

8.3 Functional Description

8.3.1 Clock Selection

The CLKSEL register is used to select the clock source for the system (SYSCLK). The CLKSL field selects which oscillator source is used as the system clock, while CLKDIV controls the programmable divider. When an internal oscillator source is selected as the SYSCLK, the external oscillator may still clock certain peripherals. In these cases, the external oscillator source is synchronized to the SYSCLK source. The system clock may be switched on-the-fly between any of the oscillator sources so long as the selected clock source is enabled and has settled, and CLKDIV may be changed at any time.

Note: Some device families do place restrictions on the difference in operating frequency when switching clock sources. Please see the CLKSEL register description for details.

8.3.2 HFOSC0 24.5 MHz Internal Oscillator

HFOSC0 is a programmable internal high-frequency oscillator that is factory-calibrated to 24.5 MHz. The oscillator is automatically enabled when it is requested. The oscillator period can be adjusted via the HFO0CAL register to obtain other frequencies.

Note: Changing the HFO0CAL register value from its default value may degrade the frequency stability of the oscillator across temperature and supply voltage.

8.3.3 HFOSC1 49 MHz Internal Oscillator

HFOSC1 is a programmable internal high-frequency oscillator that is factory-calibrated to 49 MHz. The oscillator is automatically enabled when it is requested. The oscillator period can be adjusted via the HFO1CAL register to obtain other frequencies.

Note: Changing the HFO1CAL register value from its default value may degrade the frequency stability of the oscillator across temperature and supply voltage.

Note: HFOSC0 consumes less current when enabled than HFOSC1.

8.3.4 LFOSC0 80 kHz Internal Oscillator

LFOSC0 is a progammable low-frequency oscillator, factory calibrated to a nominal frequency of 80 kHz. A dedicated divider at the oscillator output is capable of dividing the output clock by 1, 2, 4, or 8, using the OSCLD bits in the LFO0CN register. The OSCLF bits can be used to coarsely adjust the oscillator's output frequency.

The LFOSC0 circuit requires very little start-up time and may be selected as the system clock immediately following the register write which enables the oscillator.

Calibrating LFOSC0

On-chip calibration of the LFOSC0 can be performed using a timer to capture the oscillator period, when running from a known time base. When a timer is configured for L-F Oscillator capture mode, a falling edge of the low-frequency oscillator's output will cause a capture event on the corresponding timer. As a capture event occurs, the current timer value is copied into the timer reload registers. By recording the difference between two successive timer capture values, the low-frequency oscillator's period can be calculated. The OSCLF bits can then be adjusted to produce the desired oscillator frequency.

8.3.5 External Crystal

If a crystal or ceramic resonator is used as the external oscillator, the crystal/resonator and a 10 MΩ resistor must be wired across the XTAL1 and XTAL2 pins. Appropriate loading capacitors should be added to XTAL1 and XTAL2, and both pins should be configured for analog I/O with the digital output drivers disabled.

The capacitors shown in the external crystal configuration provide the load capacitance required by the crystal for correct oscillation. These capacitors are “in series” as seen by the crystal and “in parallel” with the stray capacitance of the XTAL1 and XTAL2 pins.

Note: The recommended load capacitance depends upon the crystal and the manufacturer. Refer to the crystal data sheet when completing these calculations.

The equation for determining the load capacitance for two capacitors is as follows:

$$C_L = \frac{C_A \times C_B}{C_A + C_B} + C_S$$

Figure 8.2. External Oscillator Load Capacitance

Where:

- C_A and C_B are the capacitors connected to the crystal leads.
- C_S is the total stray capacitance of the PCB.
- The stray capacitance for a typical layout where the crystal is as close as possible to the pins is 2-5 pF per pin.

If C_A and C_B are the same (C), then the equation becomes the following:

$$C_L = \frac{C}{2} + C_S$$

Figure 8.3. External Oscillator Load Capacitance with Equal Capacitors

For example, a tuning-fork crystal of 25 MHz has a recommended load capacitance of 12.5 pF. With a stray capacitance of 3 pF per pin (6 pF total), the 13 pF capacitors yield an equivalent capacitance of 12.5 pF across the crystal.

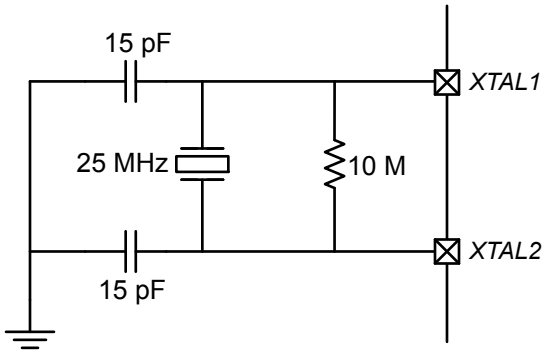


Figure 8.4. 25 MHz External Crystal Example

Crystal oscillator circuits are quite sensitive to PCB layout. The crystal should be placed as close as possible to the XTAL pins on the device. The traces should be as short as possible and shielded with ground plane from any other traces which could introduce noise or interference. When using an external crystal, the external oscillator drive circuit must be configured by firmware for Crystal Oscillator Mode or Crystal Oscillator Mode with divide by 2 stage. The divide by 2 stage ensures that the clock derived from the external oscillator has a duty cycle of 50%. The External Oscillator Frequency Control value (XFCN) must also be specified based on the crystal frequency. For example, a 25 MHz crystal requires an XFCN setting of 111b.

Table 8.1. Recommended XFCN Settings for Crystal Mode

XFCN Field Setting	Crystal Frequency	Approximate Bias Current
000	f ≤ 20 kHz	0.5 µA
001	20 kHz < f ≤ 58 kHz	1.5 µA
010	58 kHz < f ≤ 155 kHz	4.8 µA
011	155 kHz < f ≤ 415 kHz	14 µA
100	415 kHz < f ≤ 1.1 MHz	40 µA
101	1.1 MHz < f ≤ 3.1 MHz	120 µA
110	3.1 MHz < f ≤ 8.2 MHz	550 µA
111	8.2 MHz < f ≤ 25 MHz	2.6 mA

When the crystal oscillator is first enabled, the external oscillator valid detector allows software to determine when the external system clock has stabilized. Switching to the external oscillator before the crystal oscillator has stabilized can result in unpredictable behavior. The recommended procedure for starting the crystal is as follows:

1. Configure XTAL1 and XTAL2 for analog I/O and disable the digital output drivers.
2. Disable the XTAL1 and XTAL2 digital output drivers by writing 1's to the appropriate bits in the port latch register.
3. Configure and enable the external oscillator.
4. Wait at least 1 ms
5. Poll for XCLKVLD set to 1.
6. Switch the system clock to the external oscillator.

8.3.6 External RC Mode

External RC Example

An RC network connected to the EXTOSC pin can be used as a basic oscillator.

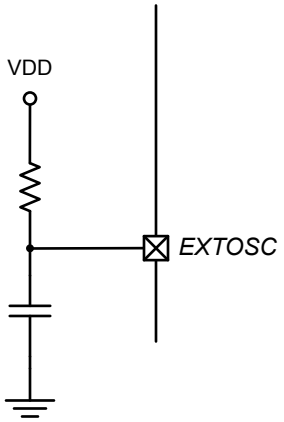


Figure 8.5. External RC Oscillator Configuration

The capacitor should be no greater than 100 pF; however, for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required XFCN field value, first select the RC network value to produce the desired frequency of oscillation, according to , where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and R = the pull-up resistor value in kΩ.

$$f = \frac{1.23 \times 10^3}{R \times C}$$

Figure 8.6. RC Mode Oscillator Frequency

For example, if the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

$$f = \frac{1.23 \times 10^3}{R \times C} = \frac{1.23 \times 10^3}{246 \times 50} = 100 \text{ kHz}$$

Figure 8.7. RC Mode Oscillator Example

Referencing , the recommended XFCN setting for 100 kHz is 010.

When the RC oscillator is first enabled, the external oscillator valid detector allows firmware to determine when oscillation has stabilized. The recommended procedure for starting the RC oscillator is as follows:

1. Configure EXTOSC for analog I/O and disable the digital output drivers.
2. Configure and enable the external oscillator.
3. Poll for XCLKVLD = 1.
4. Switch the system clock to the external oscillator.

Recommended XFCN Settings for RC Mode

Table 8.2. Recommended XFCN Settings for RC Mode

XFCN Field Setting	Approximate Frequency Range
000	f ≤ 25 kHz
001	25 kHz < f ≤ 50 kHz
010	50 kHz < f ≤ 100 kHz
011	100 kHz < f ≤ 200 kHz
100	200 kHz < f ≤ 400 kHz
101	400 kHz < f ≤ 800 kHz
110	800 kHz < f ≤ 1.6 MHz
111	1.6 MHz < f ≤ 3.2 MHz

8.3.7 External CMOS

An external CMOS clock source is also supported as a core clock source. The EXTOSC/EXTCLK pin on the device serves as the external clock input when running in this mode. When not selected as the SYSCLK source, the EXTCLK input is always re-synchronized to SYSCLK.

Note: When selecting the EXTCLK pin as a clock input source, the pin should be skipped in the crossbar and configured as a digital input. Firmware should ensure that the external clock source is present or enable the missing clock detector before switching the CLKSL field.

The external oscillator valid detector will always return zero when the external oscillator is configured to External CMOS Clock mode.

8.4 Clocking and Oscillator Control Registers

8.4.1 CLKSEL: Clock Select

Bit	7	6	5	4	3	2	1	0
Name	DIVRDY	CLKDIV			Reserved	CLKSL		
Access	R	RW			R	RW		
Reset	1	0x3			0	0x0		
SFR Page = ALL; SFR Address: 0xA9								

Bit	Name	Reset	Access	Description
7	DIVRDY	1	R	**Clock Divider Ready.**
	Indicates when the clock has propagated through the divider with the current CLKDIV setting.			
	Value	Name		Description
	0	NOT_READY		Clock has not propagated through divider yet.
	1	READY		Clock has propagated through divider.
6:4	CLKDIV	0x3	RW	**Clock Source Divider.**
	This field controls the divider applied to the clock source selected by CLKSL. The output of this divider is the system clock (SYSCLK).			
	Value	Name		Description
	0x0	SYSCLK_DIV_1		SYSCLK is equal to selected clock source divided by 1.
	0x1	SYSCLK_DIV_2		SYSCLK is equal to selected clock source divided by 2.
	0x2	SYSCLK_DIV_4		SYSCLK is equal to selected clock source divided by 4.
	0x3	SYSCLK_DIV_8		SYSCLK is equal to selected clock source divided by 8.
	0x4	SYSCLK_DIV_16		SYSCLK is equal to selected clock source divided by 16.
	0x5	SYSCLK_DIV_32		SYSCLK is equal to selected clock source divided by 32.
	0x6	SYSCLK_DIV_64		SYSCLK is equal to selected clock source divided by 64.
	0x7	SYSCLK_DIV_128		SYSCLK is equal to selected clock source divided by 128.
3	*Reserved*	*Must write reset value.*		
2:0	CLKSL	0x0	RW	**Clock Source Select.**
	Selects the system clock source.			
	Value	Name		Description
	0x0	HFOSC0		Clock derived from the Internal High Frequency Oscillator 0.
	0x1	EXTOSC		Clock derived from the External Oscillator circuit.
	0x2	LFOSC		Clock derived from the Internal Low-Frequency Oscillator.
	0x3	HFOSC1		Clock derived from the Internal High Frequency Oscillator 1.
	0x4	HFOSC0_DIV_1P5		Clock derived from the Internal High Frequency Oscillator 0, pre-scaled by 1.5.

Bit	Name	Reset	Access	Description
	0x7	HFOSC1_DIV_1P5		Clock derived from the Internal High Frequency Oscillator 1, pre-scaled by 1.5.

This device family has restrictions when switching to clock sources that are greater than 25 MHz. SYSCLK must be running at a frequency of 24 MHz or greater before switching the CLKSL field to HFOSC1. When transitioning from slower clock frequencies, firmware should make two writes to CLKSEL.

8.4.2 HFO0CAL: High Frequency Oscillator 0 Calibration

Bit	7	6	5	4	3	2	1	0
Name	HFO0CAL							
Access	RW							
Reset	Varies							
SFR Page = 0x0, 0x10; SFR Address: 0xC7								

Bit	Name	Reset	Access	Description
7:0	HFO0CAL	Varies	RW	**Oscillator Calibration.** These bits determine the period for high frequency oscillator 0. When set to 0x00, the oscillator operates at its fastest setting. When set to 0xFF, the oscillator operates at its slowest setting. The reset value is factory calibrated, and the oscillator will revert to the calibrated frequency upon reset.

8.4.3 HFO1CAL: High Frequency Oscillator 1 Calibration

Bit	7	6	5	4	3	2	1	0
Name	Reserved	HFO1CAL						
Access	R	RW						
Reset	0	Varies						
SFR Page = 0x10; SFR Address: 0xD6								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:0	HFO1CAL	Varies	RW	**Oscillator Calibration.** These bits determine the period for high frequency oscillator 1. When set to 0x00, the oscillator operates at its fastest setting. When set to 0x7F, the oscillator operates at its slowest setting. The reset value is factory calibrated, and the oscillator will revert to the calibrated frequency upon reset.

8.4.4 HFOCN: High Frequency Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	HFO1EN	Reserved			HFO0EN	Reserved		
Access	RW	R			RW	R		
Reset	0	0x0			0	0x0		
SFR Page = 0x10; SFR Address: 0xEF								

Bit	Name	Reset	Access	Description
7	HFO1EN	0	RW	**HFOSC1 Oscillator Enable.**
	Value	Name		Description
	0	DISABLED		Disable High Frequency Oscillator 1 (HFOSC1 will still turn on if requested by any block in the device or selected as the SYSCLK source).
	1	ENABLED		Force High Frequency Oscillator 1 to run.
6:4	*Reserved*	*Must write reset value.*		
3	HFO0EN	0	RW	**HFOSC0 Oscillator Enable.**
	Value	Name		Description
	0	DISABLED		Disable High Frequency Oscillator 0 (HFOSC0 will still turn on if requested by any block in the device or selected as the SYSCLK source).
	1	ENABLED		Force High Frequency Oscillator 0 to run.
2:0	*Reserved*	*Must write reset value.*		

8.4.5 LFO0CN: Low Frequency Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	OSCLEN	OSCLRDY	OSCLF				OSCLD	
Access	RW	R	RW				RW	
Reset	0	1	Varies				0x3	
SFR Page = 0x0, 0x10; SFR Address: 0xB1								

Bit	Name	Reset	Access	Description
7	OSCLEN	0	RW	**Internal L-F Oscillator Enable.**
	This bit enables the internal low-frequency oscillator. Note that the low-frequency oscillator is automatically enabled when the watchdog timer is active.			
	Value	Name		Description
	0	DISABLED		Internal L-F Oscillator Disabled.
	1	ENABLED		Internal L-F Oscillator Enabled.
6	OSCLRDY	1	R	**Internal L-F Oscillator Ready.**
	Value	Name		Description
	0	NOT_SET		Internal L-F Oscillator frequency not stabilized.
	1	SET		Internal L-F Oscillator frequency stabilized.
5:2	OSCLF	Varies	RW	**Internal L-F Oscillator Frequency Control.**
	Fine-tune control bits for the Internal L-F oscillator frequency. When set to 0000b, the L-F oscillator operates at its fastest setting. When set to 1111b, the L-F oscillator operates at its slowest setting. The OSCLF bits should only be changed by firmware when the L-F oscillator is disabled (OSCLEN = 0).			
1:0	OSCLD	0x3	RW	**Internal L-F Oscillator Divider Select.**
	Value	Name		Description
	0x0	DIVIDE_BY_8		Divide by 8 selected.
	0x1	DIVIDE_BY_4		Divide by 4 selected.
	0x2	DIVIDE_BY_2		Divide by 2 selected.
	0x3	DIVIDE_BY_1		Divide by 1 selected.
OSCLRDY is only set back to 0 in the event of a device reset or a change to the OSCLD bits.				

8.4.6 XOSC0CN: External Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	XCLKVLD	XOSCMD			Reserved	XFCN		
Access	R	RW			RW	RW		
Reset	0	0x0			0	0x0		
SFR Page = 0x0, 0x10; SFR Address: 0x86								

Bit	Name	Reset	Access	Description
7	XCLKVLD	0	R	**External Oscillator Valid Flag.**
	Provides External Oscillator status and is valid at all times for all modes of operation except External CMOS Clock Mode and External CMOS Clock Mode with divide by 2. In these modes, XCLKVLD always returns 0.			
	Value	Name		Description
	0	NOT_SET		External Oscillator is unused or not yet stable.
	1	SET		External Oscillator is running and stable.
6:4	XOSCMD	0x0	RW	**External Oscillator Mode.**
	Value	Name		Description
	0x0	DISABLED		External Oscillator circuit disabled.
	0x2	CMOS		External CMOS Clock Mode.
	0x3	CMOS_DIV_2		External CMOS Clock Mode with divide by 2 stage.
	0x4	RC		RC Oscillator Mode.
	0x6	CRYSTAL		Crystal Oscillator Mode
	0x7	CRYSTAL_DIV_2		Crystal Oscillator Mode with divide by 2 stage
3	*Reserved*	*Must write reset value.*		
2:0	XFCN	0x0	RW	**External Oscillator Frequency Control.**
	Controls the external oscillator bias current. The value selected for this field depends on the frequency range of the external oscillator.			

9. Reset Sources and Power Supply Monitor

9.1 Introduction

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

- The core halts program execution.
- Module registers are initialized to their defined reset values unless the bits reset only with a power-on reset.
- External port pins are forced to a known state.
- Interrupts and timers are disabled.

All registers are reset to the predefined values noted in the register descriptions unless the bits only reset with a power-on reset. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost. By default, the Port I/O latches are reset to 1 in open-drain mode, with weak pullups enabled during and after the reset. Optionally, firmware may configure the port I/O, DAC outputs, and precision reference to maintain state through system resets other than power-on resets. For Supply Monitor and power-on resets, the RSTb pin is driven low until the device exits the reset state. On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to an internal oscillator. The Watchdog Timer is enabled, and program execution begins at location 0x0000.

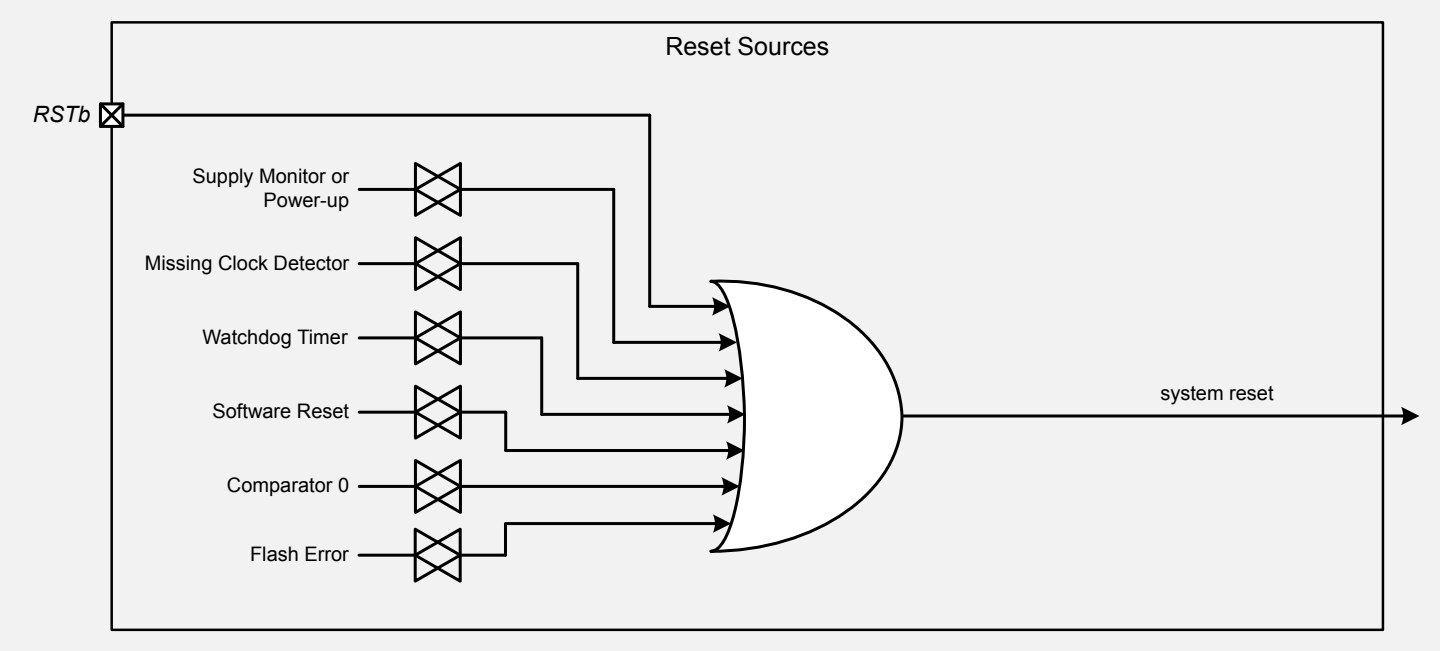


Figure 9.1. Reset Sources Block Diagram

9.2 Features

Reset sources on the device include the following:

- Power-on reset
- External reset pin
- Comparator reset
- Software-triggered reset
- Supply monitor reset (monitors VDD supply)
- Watchdog timer reset
- Missing clock detector reset
- Flash error reset

9.3 Functional Description

9.3.1 Device Reset

Upon entering a reset state from any source, the following events occur:

- The processor core halts program execution.
- Special Function Registers (SFRs) are initialized to their defined reset values.
- External port pins are placed in a known state.
- Interrupts and timers are disabled.

SFRs are reset to the predefined reset values noted in the detailed register descriptions. The contents of internal data memory are unaffected during a reset; any previously stored data is preserved. However, since the stack pointer SFR is reset, the stack is effectively lost, even though the data on the stack is not altered.

The port I/O latches are reset to 0xFF (all logic ones) in open-drain mode. Weak pullups are enabled during and after the reset. For Supply Monitor and power-on resets, the RSTb pin is driven low until the device exits the reset state.

Note: During a power-on event, there may be a short delay before the POR circuitry fires and the RSTb pin is driven low. During that time, the RSTb pin will be weakly pulled to the supply pin.

On exit from the reset state, the program counter (PC) is reset, the watchdog timer is enabled, and the system clock defaults to an internal oscillator. Program execution begins at location 0x0000.

Optionally, firmware may configure the port I/O, DAC outputs, and precision reference to maintain state through system resets other than power-on resets. Setting the RSTMD bits in the DACnCF0 registers will cause the DAC output voltage and precision reference to persist through all resets except for power-on resets. Setting the PINRSTMD bit in the PCON1 register will cause the port I/O state to persist through all resets except for power-on resets.

9.3.2 Power-On Reset

During power-up, the POR circuit fires. When POR fires, the device is held in a reset state and the RSTb pin is driven low until the supply voltage settles above V_{POR}. Two delays are present during the supply ramp time. First, a delay occurs before the POR circuitry fires and pulls the RSTb pin low. A second delay occurs before the device is released from reset; the delay decreases as the supply ramp time (T_{RMP}) increases (supply ramp time is defined as how fast the supply pin ramps from 0 V to V_{POR}). Additionally, the power supply must reach V_{POR} before the POR circuit releases the device from reset.

On exit from a power-on reset, the PORSF flag is set by hardware to logic 1. When PORSF is set, all of the other reset flags in the RSTSRC register are indeterminate. (PORSF is cleared by all other resets.) Since all resets cause program execution to begin at the same location (0x0000), software can read the PORSF flag to determine if a power-up was the cause of reset. The content of internal data memory should be assumed to be undefined after a power-on reset. The supply monitor is enabled following a power-on reset.

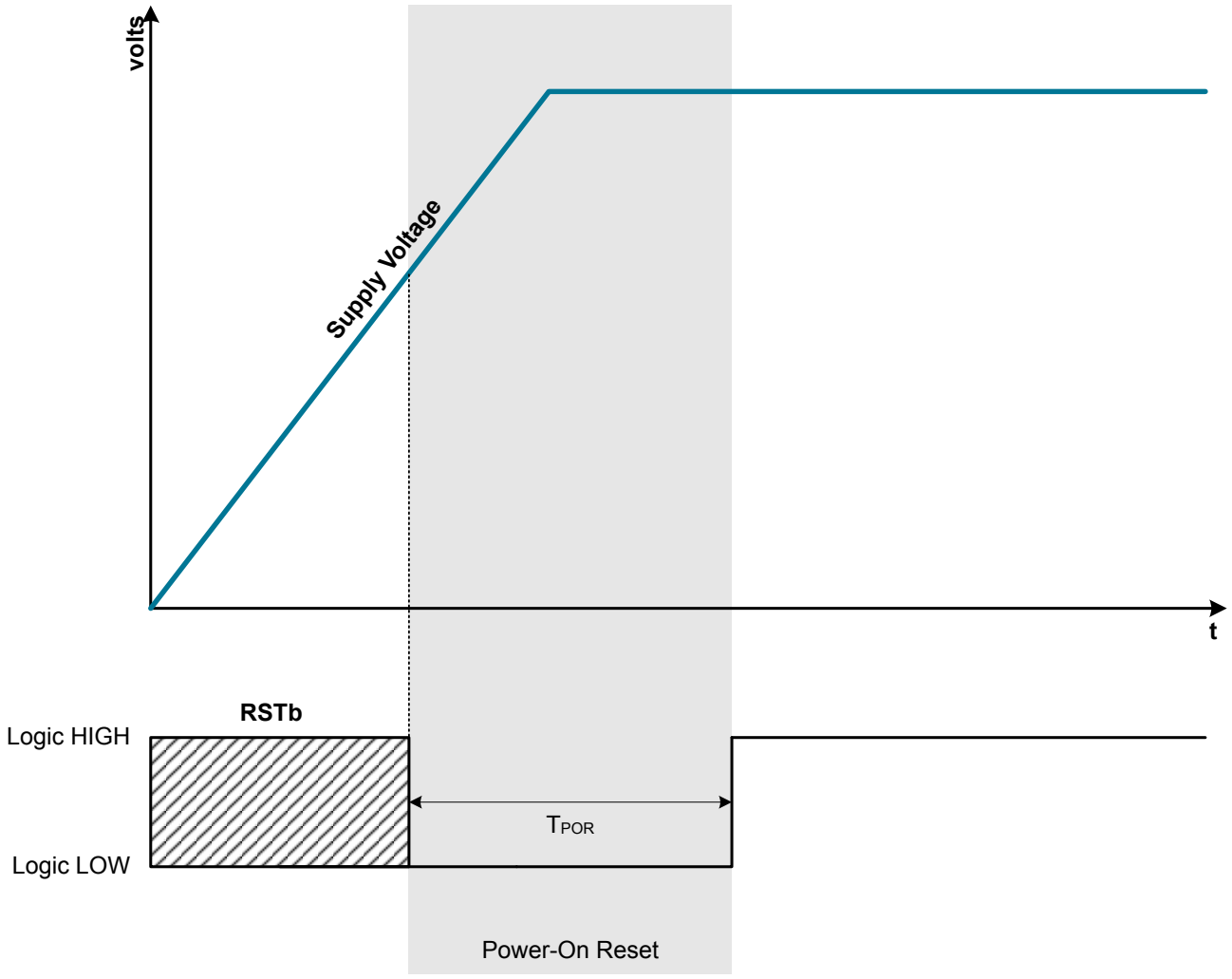


Figure 9.2. Power-On Reset Timing

9.3.3 Supply Monitor Reset

The supply monitor senses the voltage on the device's supply pin and can generate a reset if the supply drops below the corresponding threshold. This monitor is enabled and enabled as a reset source after initial power-on to protect the device until the supply is an adequate and stable voltage. When enabled and selected as a reset source, any power down transition or power irregularity that causes the supply to drop below the reset threshold will drive the RSTb pin low and hold the core in a reset state. When the supply returns to a level above the reset threshold, the monitor will release the core from the reset state. The reset status can then be read using the device reset sources module. After a power-fail reset, the PORF flag reads 1 and all of the other reset flags in the RSTSRC register are indeterminate. The power-on reset delay (t_{POR}) is not incurred after a supply monitor reset. The contents of RAM should be presumed invalid after a supply monitor reset. The enable state of the supply monitor and its selection as a reset source is not altered by device resets. For example, if the supply monitor is de-selected as a reset source and disabled by software using the VDMEN bit in the VDM0CN register, and then firmware performs a software reset, the supply monitor will remain disabled and de-selected after the reset. To protect the integrity of flash contents, the supply monitor must be enabled and selected as a reset source if software contains routines that erase or write flash memory. If the supply monitor is not enabled, any erase or write performed on flash memory will be ignored.

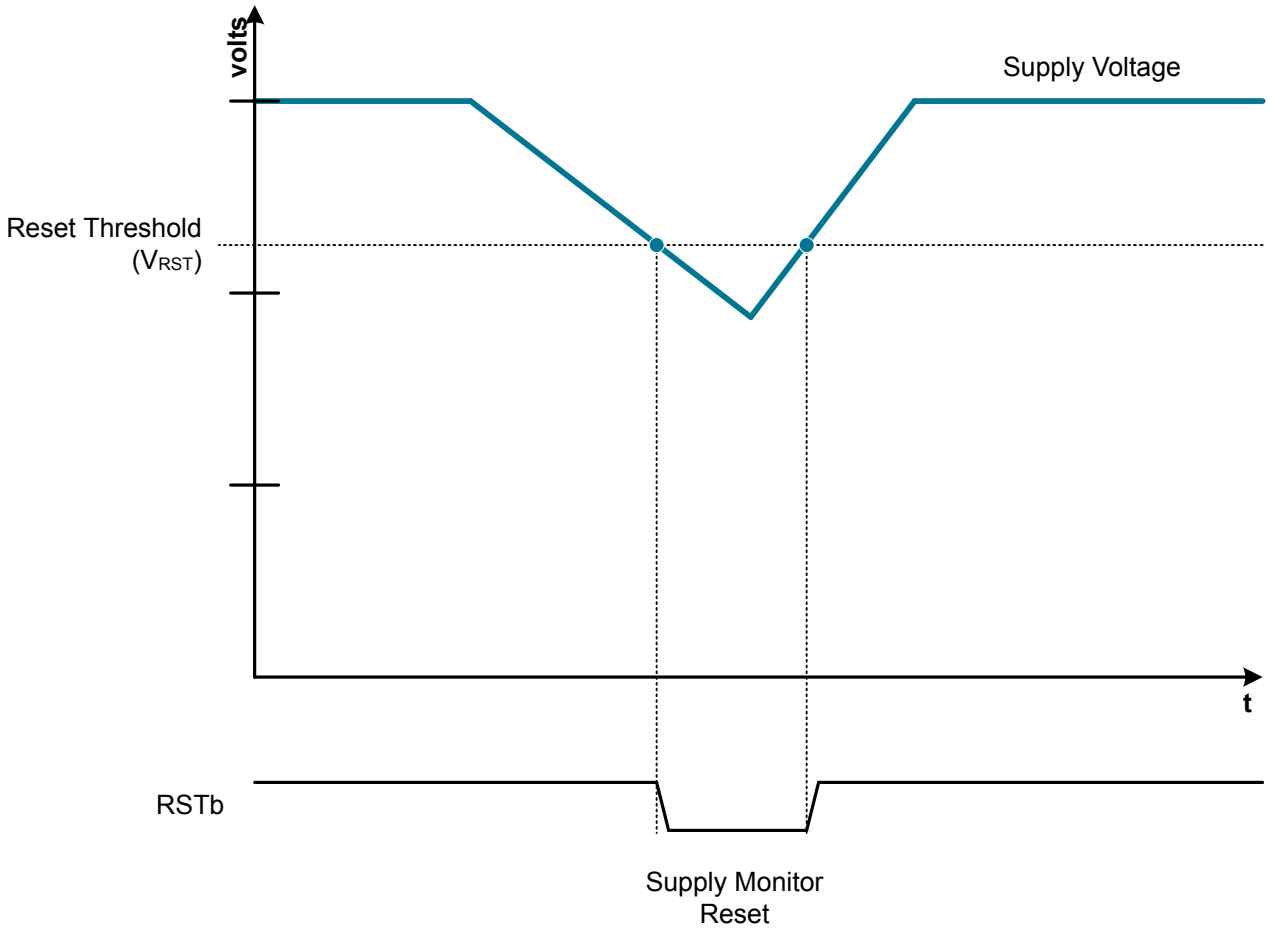


Figure 9.3. Reset Sources

9.3.4 External Reset

The external RSTb pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the RSTb pin generates a reset; an external pullup and/or decoupling of the RSTb pin may be necessary to avoid erroneous noise-induced resets. The PINRSF flag is set on exit from an external reset.

9.3.5 Missing Clock Detector Reset

The Missing Clock Detector (MCD) is a one-shot circuit that is triggered by the system clock. If the system clock remains high or low for more than the MCD time window, the one-shot will time out and generate a reset. After a MCD reset, the MCDRSF flag will read 1, signifying the MCD as the reset source; otherwise, this bit reads 0. Writing a 1 to the MCDRSF bit enables the Missing Clock Detector; writing a 0 disables it. The state of the RSTb pin is unaffected by this reset.

9.3.6 Comparator (CMP0) Reset

Comparator0 can be configured as a reset source by writing a 1 to the C0RSEF flag. Comparator0 should be enabled and allowed to settle prior to writing to C0RSEF to prevent any turn-on chatter on the output from generating an unwanted reset. The Comparator0 reset is active-low: if the non-inverting input voltage (on CP0+) is less than the inverting input voltage (on CP0–), the device is put into the reset state. After a Comparator0 reset, the C0RSEF flag will read 1 signifying Comparator0 as the reset source; otherwise, this bit reads 0. The state of the RSTb pin is unaffected by this reset.

9.3.7 Watchdog Timer Reset

The programmable Watchdog Timer (WDT) can be used to prevent software from running out of control during a system malfunction. The WDT function can be enabled or disabled by software as described in the watchdog timer section. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit is set to 1. The state of the RSTb pin is unaffected by this reset.

9.3.8 Flash Error Reset

If a flash read/write/erase or program read targets an illegal address, a system reset is generated. This may occur due to any of the following:

- A flash write or erase is attempted above user code space.
- A flash read is attempted above user code space.
- A program read is attempted above user code space (i.e., a branch instruction to the reserved area).
- A flash read, write or erase attempt is restricted due to a flash security setting.

The FERROR bit is set following a flash error reset. The state of the RSTb pin is unaffected by this reset.

9.3.9 Software Reset

Software may force a reset by writing a 1 to the SWRSF bit. The SWRSF bit will read 1 following a software forced reset. The state of the RSTb pin is unaffected by this reset.

9.4 Reset Sources and Supply Monitor Control Registers

9.4.1 RSTSRC: Reset Source

Bit	7	6	5	4	3	2	1	0
Name	Reserved	FERROR	C0RSEF	SWRSF	WDTRSF	MCDRSF	PORSF	PINRSF
Access	RW	R	RW	RW	R	RW	RW	R
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies
SFR Page = 0x0; SFR Address: 0xEF								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	FERROR	Varies	R	**Flash Error Reset Flag.** This read-only bit is set to '1' if a flash read/write/erase error caused the last reset.
5	C0RSEF	Varies	RW	**Comparator0 Reset Enable and Flag.** Read: This bit reads 1 if Comparator 0 caused the last reset. Write: Writing a 1 to this bit enables Comparator 0 (active-low) as a reset source.
4	SWRSF	Varies	RW	**Software Reset Force and Flag.** Read: This bit reads 1 if last reset was caused by a write to SWRSF. Write: Writing a 1 to this bit forces a system reset.
3	WDTRSF	Varies	R	**Watchdog Timer Reset Flag.** This read-only bit is set to '1' if a watchdog timer overflow caused the last reset.
2	MCDRSF	Varies	RW	**Missing Clock Detector Enable and Flag.** Read: This bit reads 1 if a missing clock detector timeout caused the last reset. Write: Writing a 1 to this bit enables the missing clock detector. The MCD triggers a reset if a missing clock condition is detected.
1	PORSF	Varies	RW	**Power-On / Supply Monitor Reset Flag, and Supply Monitor Reset Enable.** Read: This bit reads 1 anytime a power-on or supply monitor reset has occurred. Write: Writing a 1 to this bit enables the supply monitor as a reset source.
0	PINRSF	Varies	R	**HW Pin Reset Flag.** This read-only bit is set to '1' if the RSTb pin caused the last reset.

Reads and writes of the RSTSRC register access different logic in the device. Reading the register always returns status information to indicate the source of the most recent reset. Writing to the register activates certain options as reset sources. It is recommended to not use any kind of read-modify-write operation on this register.

When the PORSF bit reads back '1' all other RSTSRC flags are indeterminate.

Writing '1' to the PORSF bit when the supply monitor is not enabled and stabilized may cause a system reset.

9.4.2 VDM0CN: Supply Monitor Control

Bit	7	6	5	4	3	2	1	0
Name	VDMEN	VDDSTAT	Reserved					
Access	RW	R	R					
Reset	Varies	Varies	Varies					
SFR Page = 0x0; SFR Address: 0xFF								

Bit	Name	Reset	Access	Description
7	VDMEN	Varies	RW	**Supply Monitor Enable.** This bit turns the supply monitor circuit on/off. The supply monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC. Selecting the supply monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the supply monitor and selecting it as a reset source.
	Value	Name		Description
	0	DISABLED		Supply Monitor Disabled.
	1	ENABLED		Supply Monitor Enabled.
6	VDDSTAT	Varies	R	**Supply Status.** This bit indicates the current power supply status (supply monitor output).
	Value	Name		Description
	0	BELOW		V_{DD} is at or below the supply monitor threshold.
	1	ABOVE		V_{DD} is above the supply monitor threshold.
5:0	*Reserved*	*Must write reset value.*		

10. CIP-51 Microcontroller Core

10.1 Introduction

The CIP-51 microcontroller core is a high-speed, pipelined, 8-bit core utilizing the standard MCS-51™ instruction set. Any standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 includes on-chip debug hardware and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control system solution.

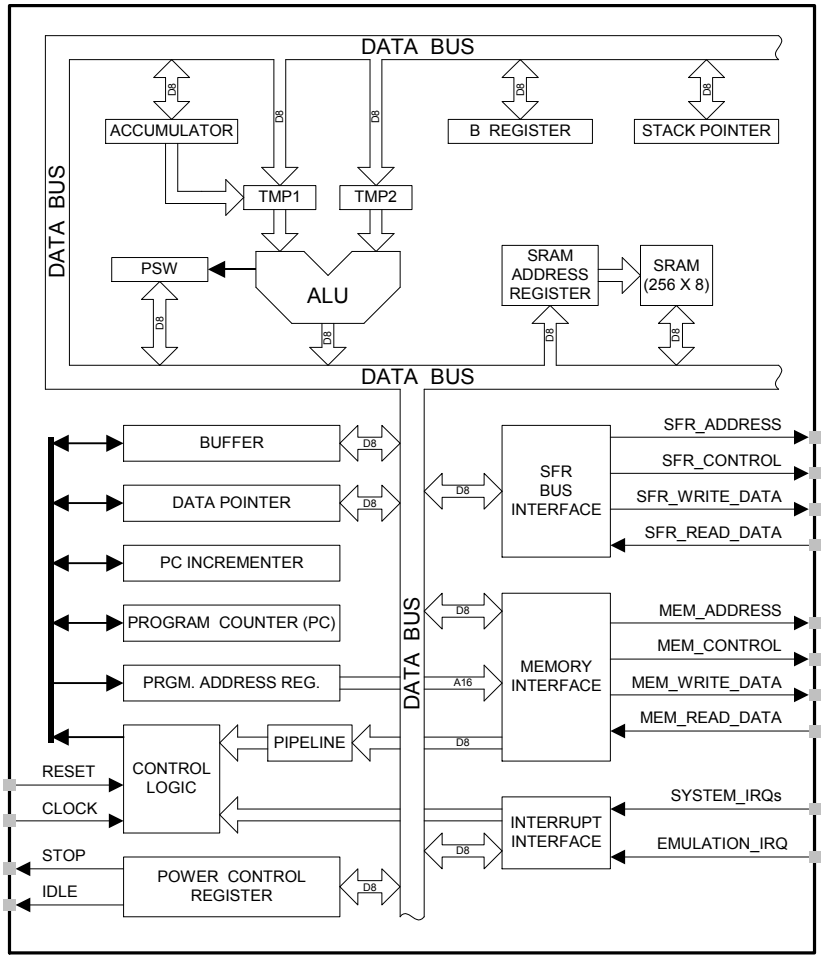


Figure 10.1. CIP-51 Block Diagram

Performance

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. The CIP-51 core executes 76 of its 109 instructions in one or two clock cycles, with no instructions taking more than eight clock cycles. The table below shows the distribution of instructions vs. the number of clock cycles required for execution.

Table 10.1. Instruction Execution Timing

Clocks to Execute	1	2	2 or 3*	3	3 or 4*	4	4 or 5*	5	8
Number of Instructions	26	50	5	14	7	3	1	2	1

Notes:
1. Conditional branch instructions (indicated by "2 or 3*", "3 or 4*" and "4 or 5*") require extra clock cycles if the branch is taken. See the instruction table for more information.

10.2 Features

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability. The CIP-51 includes the following features:

- Fast, efficient, pipelined architecture.
- Fully compatible with MCS-51 instruction set.
- 0 to 50 MHz operating clock frequency.
- 50 MIPS peak throughput with 50 MHz clock.
- Extended interrupt handler.
- Power management modes.
- On-chip debug logic.
- Program and data memory security.

10.3 Functional Description

10.3.1 Programming and Debugging Support

In-system programming of the flash program memory and communication with on-chip debug support logic is accomplished via the Silicon Labs 2-Wire development interface (C2).

The on-chip debug support logic facilitates full speed in-circuit debugging, allowing the setting of hardware breakpoints, starting, stopping and single stepping through program execution (including interrupt service routines), examination of the program's call stack, and reading/writing the contents of registers and memory. This method of on-chip debugging is completely non-intrusive, requiring no RAM, stack, timers, or other on-chip resources.

The CIP-51 is supported by development tools from Silicon Labs and third party vendors. Silicon Labs provides an integrated development environment (IDE) including editor, debugger and programmer. The IDE's debugger and programmer interface to the CIP-51 via the C2 interface to provide fast and efficient in-system device programming and debugging. Third party macro assemblers and C compilers are also available.

10.3.2 Prefetch Engine

The CIP-51 core incorporates a multi-byte prefetch engine to enable faster core clock speeds. Because the access time of the flash memory is 40 ns, and the minimum instruction time is 13.6 ns, the prefetch engine is necessary for full-speed code execution. Multiple instruction bytes are read from flash memory by the prefetch engine and given to the CIP-51 processor core to execute. When running linear code (code without any jumps or branches), the prefetch engine allows instructions to be executed at full speed. When a code branch occurs, the processor may be stalled for up to five clock cycles (FLRT = 2) or three clock cycles (FLRT = 1) while the next set of code bytes is retrieved from flash memory.

When operating at speeds greater than 25 MHz, the prefetch engine must be used. To enable the prefetch engine, the FLRT bit field should be configured to the desired speed setting. For example, if running between 25 and 50 MHz, FLRT should be set to 1, and when operating between 50 and 73.5 MHz, FLRT should be set to 2. When changing clocks, the FLRT field should be set to the higher number during the clock change, to ensure that flash is never read too quickly.

10.3.3 Instruction Set

The instruction set of the CIP-51 System Controller is fully compatible with the standard MCS-51™ instruction set. Standard 8051 development tools can be used to develop software for the CIP-51. All CIP-51 instructions are the binary and functional equivalent of their MCS-51™ counterparts, including opcodes, addressing modes and effect on PSW flags. However, instruction timing is much faster than that of the standard 8051.

All instruction timing on the CIP-51 controller is based directly on the core clock timing. This is in contrast to many other 8-bit architectures, where a distinction is made between machine cycles and clock cycles, with machine cycles taking multiple core clock cycles.

Due to the pipelined architecture of the CIP-51, most instructions execute in the same number of clock cycles as there are program bytes in the instruction. Conditional branch instructions take one less clock cycle to complete when the branch is not taken as opposed to when the branch is taken. The following table summarizes the instruction set, including the mnemonic, number of bytes, and number of clock cycles for each instruction.

Table 10.2. CIP-51 Instruction Set Summary

Mnemonic	Description	Bytes	Clock Cycles		
			FLRT = 0	FLRT = 1	FLRT = 2
Arithmetic Operations					
ADD A, Rn	Add register to A	1	1	1	1
ADD A, direct	Add direct byte to A	2	2	2	2
ADD A, @Ri	Add indirect RAM to A	1	2	2	2
ADD A, #data	Add immediate to A	2	2	2	2
ADDC A, Rn	Add register to A with carry	1	1	1	1
ADDC A, direct	Add direct byte to A with carry	2	2	2	2
ADDC A, @Ri	Add indirect RAM to A with carry	1	2	2	2
ADDC A, #data	Add immediate to A with carry	2	2	2	2
SUBB A, Rn	Subtract register from A with borrow	1	1	1	1
SUBB A, direct	Subtract direct byte from A with borrow	2	2	2	2
SUBB A, @Ri	Subtract indirect RAM from A with borrow	1	2	2	2
SUBB A, #data	Subtract immediate from A with borrow	2	2	2	2
INC A	Increment A	1	1	1	1
INC Rn	Increment register	1	1	1	1
INC direct	Increment direct byte	2	2	2	2
INC @Ri	Increment indirect RAM	1	2	2	2
DEC A	Decrement A	1	1	1	1
DEC Rn	Decrement register	1	1	1	1
DEC direct	Decrement direct byte	2	2	2	2
DEC @Ri	Decrement indirect RAM	1	2	2	2
INC DPTR	Increment Data Pointer	1	1	1	1
MUL AB	Multiply A and B	1	4	4	4
DIV AB	Divide A by B	1	8	8	8
DA A	Decimal adjust A	1	1	1	1
Logical Operations					

Mnemonic	Description	Bytes	Clock Cycles		
			FLRT = 0	FLRT = 1	FLRT = 2
ANL A, Rn	AND Register to A	1	1	1	1
ANL A, direct	AND direct byte to A	2	2	2	2
ANL A, @Ri	AND indirect RAM to A	1	2	2	2
ANL A, #data	AND immediate to A	2	2	2	2
ANL direct, A	AND A to direct byte	2	2	2	2
ANL direct, #data	AND immediate to direct byte	3	3	3	3
ORL A, Rn	OR Register to A	1	1	1	1
ORL A, direct	OR direct byte to A	2	2	2	2
ORL A, @Ri	OR indirect RAM to A	1	2	2	2
ORL A, #data	OR immediate to A	2	2	2	2
ORL direct, A	OR A to direct byte	2	2	2	2
ORL direct, #data	OR immediate to direct byte	3	3	3	3
XRL A, Rn	Exclusive-OR Register to A	1	1	1	1
XRL A, direct	Exclusive-OR direct byte to A	2	2	2	2
XRL A, @Ri	Exclusive-OR indirect RAM to A	1	2	2	2
XRL A, #data	Exclusive-OR immediate to A	2	2	2	2
XRL direct, A	Exclusive-OR A to direct byte	2	2	2	2
XRL direct, #data	Exclusive-OR immediate to direct byte	3	3	3	3
CLR A	Clear A	1	1	1	1
CPL A	Complement A	1	1	1	1
RL A	Rotate A left	1	1	1	1
RLC A	Rotate A left through Carry	1	1	1	1
RR A	Rotate A right	1	1	1	1
RRC A	Rotate A right through Carry	1	1	1	1
SWAP A	Swap nibbles of A	1	1	1	1
Data Transfer					
MOV A, Rn	Move Register to A	1	1	1	1
MOV A, direct	Move direct byte to A	2	2	2	2
MOV A, @Ri	Move indirect RAM to A	1	2	2	2
MOV A, #data	Move immediate to A	2	2	2	2
MOV Rn, A	Move A to Register	1	1	1	1
MOV Rn, direct	Move direct byte to Register	2	2	2	2
MOV Rn, #data	Move immediate to Register	2	2	2	2
MOV direct, A	Move A to direct byte	2	2	2	2
MOV direct, Rn	Move Register to direct byte	2	2	2	2
MOV direct, direct	Move direct byte to direct byte	3	3	3	3

Mnemonic	Description	Bytes	Clock Cycles		
			FLRT = 0	FLRT = 1	FLRT = 2
MOV direct, @Ri	Move indirect RAM to direct byte	2	2	2	2
MOV direct, #data	Move immediate to direct byte	3	3	3	3
MOV @Ri, A	Move A to indirect RAM	1	2	2	2
MOV @Ri, direct	Move direct byte to indirect RAM	2	2	2	2
MOV @Ri, #data	Move immediate to indirect RAM	2	2	2	2
MOV DPTR, #data16	Load DPTR with 16-bit constant	3	3	3	3
MOVC A, @A+DPTR	Move code byte relative DPTR to A	1	3	7	9
MOVC A, @A+PC	Move code byte relative PC to A	1	3	7	9
MOVX A, @Ri	Move external data (8-bit address) to A	1	3	3	3
MOVX @Ri, A	Move A to external data (8-bit address)	1	3	3	3
MOVX A, @DPTR	Move external data (16-bit address) to A	1	3	3	3
MOVX @DPTR, A	Move A to external data (16-bit address)	1	3	3	3
PUSH direct	Push direct byte onto stack	2	2	2	2
POP direct	Pop direct byte from stack	2	2	2	2
XCH A, Rn	Exchange Register with A	1	1	1	1
XCH A, direct	Exchange direct byte with A	2	2	2	2
XCH A, @Ri	Exchange indirect RAM with A	1	2	2	2
XCHD A, @Ri	Exchange low nibble of indirect RAM with A	1	2	2	2
Boolean Manipulation					
CLR C	Clear Carry	1	1	1	1
CLR bit	Clear direct bit	2	2	2	2
SETB C	Set Carry	1	1	1	1
SETB bit	Set direct bit	2	2	2	2
CPL C	Complement Carry	1	1	1	1
CPL bit	Complement direct bit	2	2	2	2
ANL C, bit	AND direct bit to Carry	2	2	2	2
ANL C, /bit	AND complement of direct bit to Carry	2	2	2	2
ORL C, bit	OR direct bit to carry	2	2	2	2
ORL C, /bit	OR complement of direct bit to Carry	2	2	2	2
MOV C, bit	Move direct bit to Carry	2	2	2	2
MOV bit, C	Move Carry to direct bit	2	2	2	2
JC rel	Jump if Carry is set	2	2 or 3	2 or 6	2 or 8
JNC rel	Jump if Carry is not set	2	2 or 3	2 or 6	2 or 8
JB bit, rel	Jump if direct bit is set	3	3 or 4	3 or 7	3 or 9
JNB bit, rel	Jump if direct bit is not set	3	3 or 4	3 or 7	3 or 9
JBC bit, rel	Jump if direct bit is set and clear bit	3	3 or 4	3 or 7	3 or 9

Mnemonic	Description	Bytes	Clock Cycles		
			FLRT = 0	FLRT = 1	FLRT = 2
Program Branching					
ACALL addr11	Absolute subroutine call	2	3	6	8
LCALL addr16	Long subroutine call	3	4	7	9
RET	Return from subroutine	1	5	8	10
RETI	Return from interrupt	1	5	8	10
AJMP addr11	Absolute jump	2	3	6	8
LJMP addr16	Long jump	3	4	7	9
SJMP rel	Short jump (relative address)	2	3	6	8
JMP @A+DPTR	Jump indirect relative to DPTR	1	3	6	8
JZ rel	Jump if A equals zero	2	2 or 3	2 or 6	2 or 8
JNZ rel	Jump if A does not equal zero	2	2 or 3	2 or 6	2 or 8
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	4 or 5	4 or 8	4 or 10
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3 or 4	3 or 7	3 or 9
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3 or 4	3 or 7	3 or 9
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4 or 5	4 or 8	4 or 10
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2 or 3	2 or 6	2 or 8
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3 or 4	3 or 7	3 or 9
NOP	No operation	1	1	1	1

Notes:

- **Rn**: Register R0–R7 of the currently selected register bank.
- **@Ri**: Data RAM location addressed indirectly through R0 or R1.
- **rel**: 8-bit, signed (twos complement) offset relative to the first byte of the following instruction. Used by SJMP and all conditional jumps.
- **direct**: 8-bit internal data location's address. This could be a direct-access Data RAM location (0x00–0x7F) or an SFR (0x80–0xFF).
- **#data**: 8-bit constant.
- **#data16**: 16-bit constant.
- **bit**: Direct-accessed bit in Data RAM or SFR.
- **addr11**: 11-bit destination address used by ACALL and AJMP. The destination must be within the same 2 KB page of program memory as the first byte of the following instruction.
- **addr16**: 16-bit destination address used by LCALL and LJMP. The destination may be anywhere within the 8 KB program memory space.
- There is one unused opcode (0xA5) that performs the same function as NOP. All mnemonics copyrighted © Intel Corporation 1980.

10.4 CPU Core Registers

10.4.1 DPL: Data Pointer Low

Bit	7	6	5	4	3	2	1	0
Name	DPL							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0x82								

Bit	Name	Reset	Access	Description
7:0	DPL	0x00	RW	**Data Pointer Low.** The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed flash memory or XRAM.

10.4.2 DPH: Data Pointer High

Bit	7	6	5	4	3	2	1	0
Name	DPH							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0x83								

Bit	Name	Reset	Access	Description
7:0	DPH	0x00	RW	**Data Pointer High.** The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed flash memory or XRAM.

10.4.3 SP: Stack Pointer

Bit	7	6	5	4	3	2	1	0
Name	SP							
Access	RW							
Reset	0x07							
SFR Page = ALL; SFR Address: 0x81								

Bit	Name	Reset	Access	Description
7:0	SP	0x07	RW	**Stack Pointer.** The Stack Pointer holds the location of the top of the stack. The stack pointer is incremented before every PUSH operation. The SP register defaults to 0x07 after reset.

10.4.4 ACC: Accumulator

Bit	7	6	5	4	3	2	1	0
Name	ACC							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xE0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7:0	ACC	0x00	RW	**Accumulator.** This register is the accumulator for arithmetic operations.

10.4.5 B: B Register

Bit	7	6	5	4	3	2	1	0
Name	B							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xF0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7:0	B	0x00	RW	**B Register.** This register serves as a second accumulator for certain arithmetic operations.

10.4.6 PSW: Program Status Word

Bit	7	6	5	4	3	2	1	0
Name	CY	AC	F0	RS		OV	F1	PARITY
Access	RW	RW	RW	RW		RW	RW	R
Reset	0	0	0	0x0		0	0	0
SFR Page = ALL; SFR Address: 0xD0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	CY	0	RW	**Carry Flag.** This bit is set when the last arithmetic operation resulted in a carry (addition) or a borrow (subtraction). It is cleared to logic 0 by all other arithmetic operations.
6	AC	0	RW	**Auxiliary Carry Flag.** This bit is set when the last arithmetic operation resulted in a carry into (addition) or a borrow from (subtraction) the high order nibble. It is cleared to logic 0 by all other arithmetic operations.
5	F0	0	RW	**User Flag 0.** This is a bit-addressable, general purpose flag for use under firmware control.
4:3	RS	0x0	RW	**Register Bank Select.** These bits select which register bank is used during register accesses.

Value	Name	Description
0x0	BANK0	Bank 0, Addresses 0x00-0x07
0x1	BANK1	Bank 1, Addresses 0x08-0x0F
0x2	BANK2	Bank 2, Addresses 0x10-0x17
0x3	BANK3	Bank 3, Addresses 0x18-0x1F

Bit	Name	Reset	Access	Description
2	OV	0	RW	**Overflow Flag.** This bit is set to 1 under the following circumstances: 1. An ADD, ADDC, or SUBB instruction causes a sign-change overflow. 2. A MUL instruction results in an overflow (result is greater than 255). 3. A DIV instruction causes a divide-by-zero condition. The OV bit is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases.
1	F1	0	RW	**User Flag 1.** This is a bit-addressable, general purpose flag for use under firmware control.
0	PARITY	0	R	**Parity Flag.** This bit is set to logic 1 if the sum of the eight bits in the accumulator is odd and cleared if the sum is even.

10.4.7 PFE0CN: Prefetch Engine Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved		FLRT		Reserved			
Access	R		RW		R			
Reset	0x0		0x0		0x0			
SFR Page = 0x10; SFR Address: 0xC1								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5:4	FLRT	0x0	RW	**Flash Read Timing.** This field should be programmed to the smallest allowed value, according to the system clock speed. When transitioning to a faster clock speed, program FLRT before changing the clock. When changing to a slower clock speed, change the clock before changing FLRT.

Value	Name	Description
0x0	SYSCLK_BELOW_25_MHZ	SYSCLK < 25 MHz.
0x1	SYSCLK_BELOW_50_MHZ	SYSCLK < 50 MHz.

3:0	*Reserved*	*Must write reset value.*		

11. Port I/O, Crossbar, External Interrupts, and Port Match

11.1 Introduction

Digital and analog resources are externally available on the device's multi-purpose I/O pins. Port pins P0.0-P2.3 can be defined as general-purpose I/O (GPIO), assigned to one of the internal digital resources through the crossbar or dedicated channels, or assigned to an analog function. Port pins P2.4 to P3.7 can be used as GPIO. Additionally, the C2 Interface Data signal (C2D) is shared with P3.0 or P3.7, depending on the package option.

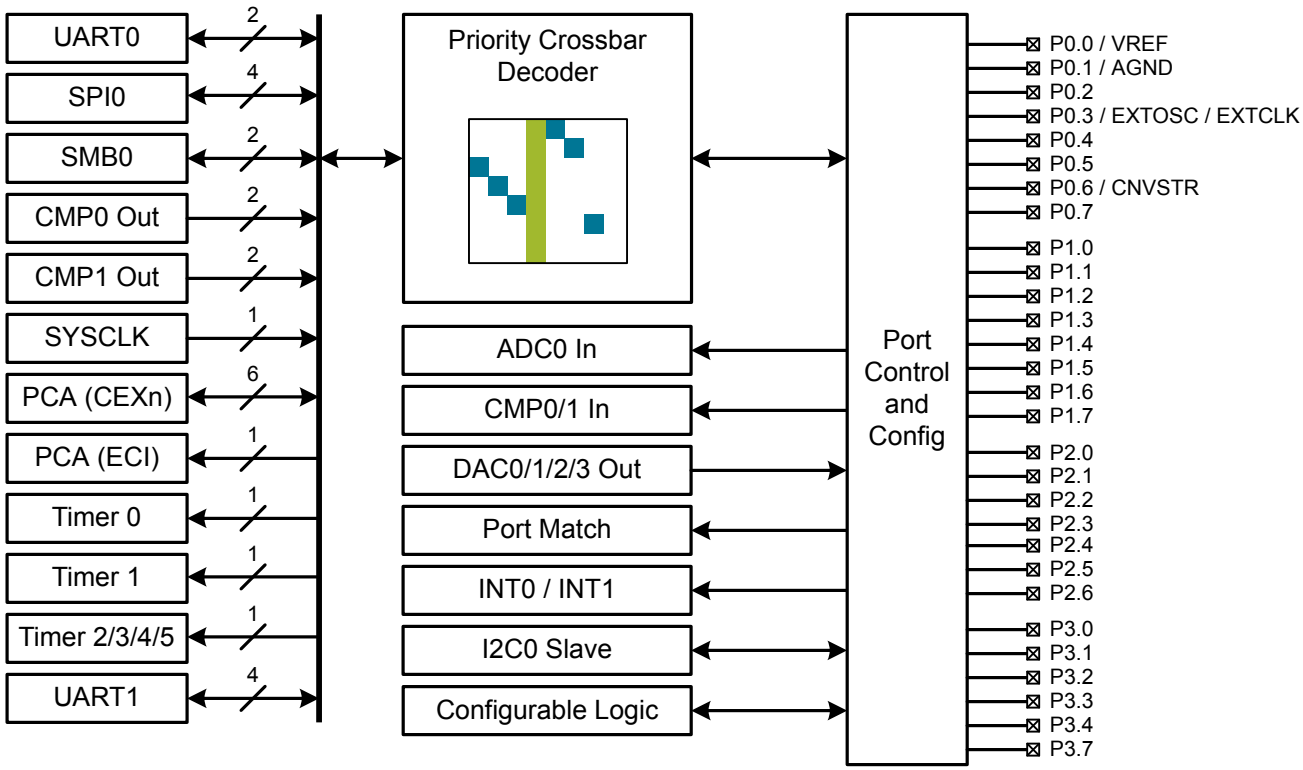


Figure 11.1. Port I/O Block Diagram

11.2 Features

The port control block offers the following features:

- Up to 29 multi-function I/O pins, supporting digital and analog functions.
- Flexible priority crossbar decoder for digital peripheral assignment.
- Two drive strength settings for each port.
- State retention feature allows pins to retain configuration through most reset sources.
- Two direct-pin interrupt sources with dedicated interrupt vectors (INT0 and INT1).
- Up to 24 direct-pin interrupt sources with shared interrupt vector (Port Match).

11.3 Functional Description

11.3.1 Port I/O Modes of Operation

Port pins are configured by firmware as digital or analog I/O using the special function registers. Port I/O initialization consists of the following general steps:

1. Select the input mode (analog or digital) for all port pins, using the Port Input Mode register (PnMDIN).
2. Select the output mode (open-drain or push-pull) for all port pins, using the Port Output Mode register (PnMDOUT).
3. Select any pins to be skipped by the I/O crossbar using the Port Skip registers (PnSKIP).
4. Assign port pins to desired peripherals.
5. Enable the crossbar (XBARE = 1).

A diagram of the port I/O cell is shown in the following figure.

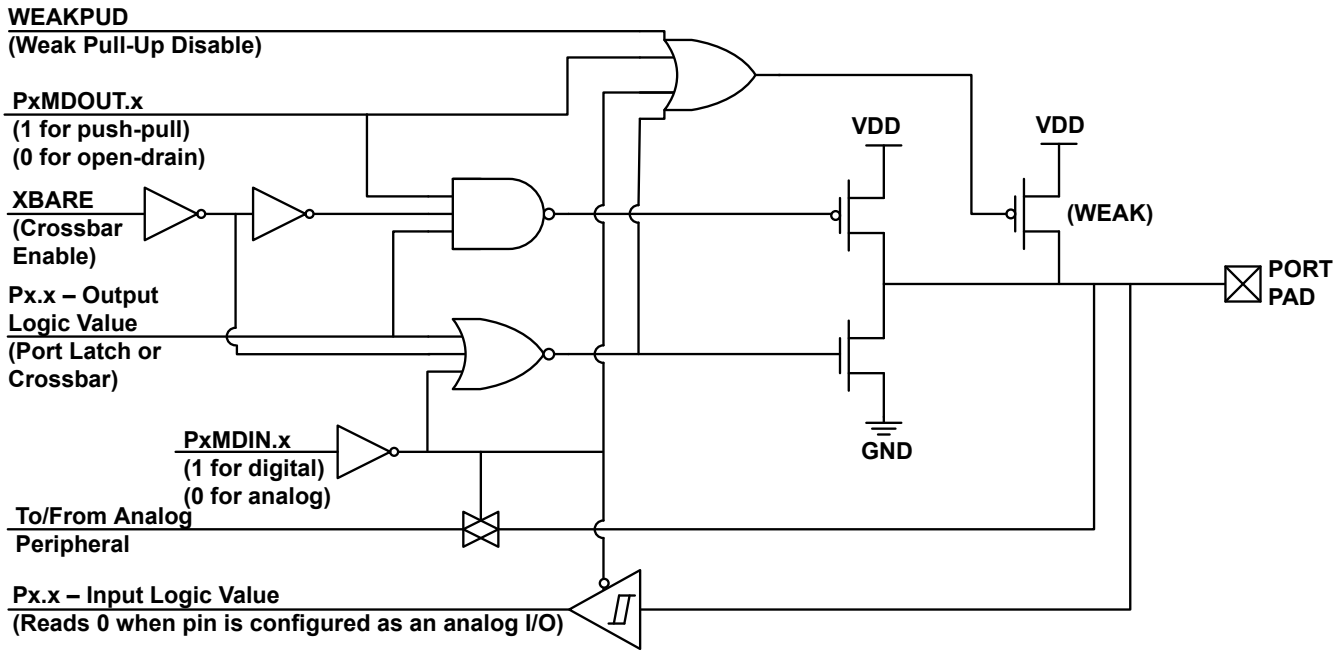


Figure 11.2. Port I/O Cell Block Diagram

Configuring Port Pins For Analog Modes

Any pins to be used for analog functions should be configured for analog mode. When a pin is configured for analog I/O, its weak pull-up, digital driver, and digital receiver are disabled. This saves power by eliminating crowbar current, and reduces noise on the analog input. Pins configured as digital inputs may still be used by analog peripherals; however this practice is not recommended. Port pins configured for analog functions will always read back a value of 0 in the corresponding Pn Port Latch register. To configure a pin as analog, the following steps should be taken:

1. Clear the bit associated with the pin in the PnMDIN register to 0. This selects analog mode for the pin.
2. Set the bit associated with the pin in the Pn register to 1.
3. Skip the bit associated with the pin in the PnSKIP register to ensure the crossbar does not attempt to assign a function to the pin.

Configuring Port Pins For Digital Modes

Any pins to be used by digital peripherals or as GPIO should be configured as digital I/O (PnMDIN.n = 1). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PnMDOUT registers.

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

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the port pad to the high side rail to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven low to minimize power consumption, and they may be globally disabled by setting WEAKPUD to 1. The user should ensure that digital I/O are always internally or externally pulled or driven to a valid logic state to minimize power consumption. Port pins configured for digital I/O always read back the logic state of the port pad, regardless of the output logic value of the port pin.

To configure a pin as a digital input:

1. Set the bit associated with the pin in the PnMDIN register to 1. This selects digital mode for the pin.
2. Clear the bit associated with the pin in the PnMDOUT register to 0. This configures the pin as open-drain.
3. Set the bit associated with the pin in the Pn register to 1. This tells the output driver to “drive” logic high. Because the pin is configured as open-drain, the high-side driver is disabled, and the pin may be used as an input.

Open-drain outputs are configured exactly as digital inputs. The pin may be driven low by an assigned peripheral, or by writing 0 to the associated bit in the Pn register if the signal is a GPIO.

To configure a pin as a digital, push-pull output:

1. Set the bit associated with the pin in the PnMDIN register to 1. This selects digital mode for the pin.
2. Set the bit associated with the pin in the PnMDOUT register to 1. This configures the pin as push-pull.

If a digital pin is to be used as a general-purpose I/O, or with a digital function that is not part of the crossbar, the bit associated with the pin in the PnSKIP register can be set to 1 to ensure the crossbar does not attempt to assign a function to the pin. The crossbar must be enabled to use port pins as standard port I/O in output mode. Port output drivers of all I/O pins are disabled whenever the crossbar is disabled.

11.3.1.1 Port Drive Strength

Port drive strength can be controlled on a port-by-port basis using the PRTDRV register. Each port has a bit in PRTDRV to select the high or low drive strength setting for all pins on that port. By default, all ports are configured for high drive strength.

11.3.2 Analog and Digital Functions

11.3.2.1 Port I/O Analog Assignments

The following table displays the potential mapping of port I/O to each analog function.

Table 11.1. Port I/O Assignment for Analog Functions

Analog Function	Potentially Assignable Port Pins (32-pin packages)	Potentially Assignable Port Pins (24-pin packages)	SFR(s) Used For Assignment
ADC Input	P0.[1-2, 4-7], P1.[0-7], P2.[1-6]	P0.[1-2, 4-7] P1.[0-2,4-7]	ADC0MX, PnSKIP, PnMDIN
High Quality ADC Input	P1.[3-6]	P1.[0-2]	ADC0MX, PnSKIP, PnMDIN
Comparator 0 Input	P0.[1-2, 4-7], P1.[0-2,7]	P0.[1-2, 4-7]	CMP0MX, PnSKIP, PnMDIN
Comparator 1 Input	P0.7, P1.0, P2[0-6]	P0.[6-7], P1.[3-7]	CMP1MX, PnSKIP, PnMDIN
Voltage Reference (VREF)	P0.0	P0.0	REF0CN, PnSKIP, PnMDIN
Reference Ground (AGND)	P0.1	P0.1	REF0CN, PnSKIP, PnMDIN
External Oscillator (EXTOSC)	P0.3	P0.3	HFO0CN, PnSKIP, PnMDIN
DAC0 Output	P3.0	P2.0	DAC0CF0, PnSKIP, PnMDIN
DAC1 Output	P3.1	P2.1	DAC1CF0, PnSKIP, PnMDIN
DAC2 Output	P3.2	P2.2	DAC2CF0, PnSKIP, PnMDIN
DAC3 Output	P3.3	P2.3	DAC3CF0, PnSKIP, PnMDIN

Note: When utilizing analog peripherals with source voltage below fixed 5.0 V, enabling the built-in analog multiplexer charge pump circuit is recommended for improved performance. See analog multiplexer charge pump (AMUXCP) chapter for additional details.

11.3.2.2 Port I/O Digital Assignments

The following table displays the potential mapping of port I/O to each digital function.

Table 11.2. Port I/O Assignment for Digital Functions

Digital Function	Potentially Assignable Port Pins	SFR(s) Used For Assignment
UART0, UART1, SPI0, SMB0, CP0, CP0A, CP1, CP1A, SYSCLK, PCA0 (CEX0-5 and ECI), T0, T1, T2/3/4/5	Any port pin available for assignment by the crossbar. This includes P0.0 – P2.3 pins which have their PnSKIP bit set to 0. The crossbar will always assign UART0 pins to P0.4 and P0.5.	XBR0, XBR1, XBR2
I2C0 Slave	P2.0 - 2.1 (QFN32, QFP32) P1.3 – P1.4 (QFN24, QSOP24)	I2C0CN0
External Interrupt 0, External Interrupt 1	P0.0 – P0.7	IT01CF
Conversion Start (CNVSTR)	P0.6	ADC0CN2
External Clock Input (EXTCLK)	P0.3	CLKSEL
Port Match	P0.0 – P2.6	P0MASK, P0MAT, P1MASK, P1MAT, P2MASK, P2MAT
Configurable Logic Inputs A and B (Assignable pins vary across CLUs)	P0.0 – P2.3	CLUnMX
Configurable Logic Unit 0 Output (CLU0OUT)	P0.2	CLU0CF
Configurable Logic Unit 1 Output (CLU1OUT)	P1.0 (QFN32, QFP32) P0.7 (QFN24, QSOP24)	CLU1CF
Configurable Logic Unit 2 Output (CLU2OUT)	P2.2 (QFN32, QFP32) P1.5 (QFN24, QSOP24)	CLU2CF
Configurable Logic Unit 3 Output (CLU3OUT)	P2.5 (QFN32, QFP32) P1.6 (QFN24, QSOP24)	CLU3CF
Any pin used for GPIO	P0.0 – P3.7	P0SKIP, P1SKIP, P2SKIP

11.3.3 Priority Crossbar Decoder

The priority crossbar decoder assigns a priority to each I/O function, starting at the top with UART0. The XBRn registers are used to control which crossbar resources are assigned to physical I/O port pins.

When a digital resource is selected, the least-significant unassigned port pin is assigned to that resource (excluding UART0, which is always assigned to dedicated pins). If a port pin is assigned, the crossbar skips that pin when assigning the next selected resource. Additionally, the the PnSKIP registers allow software to skip port pins that are to be used for analog functions, dedicated digital functions, or GPIO. If a port pin is to be used by a function which is not assigned through the crossbar, its corresponding PnSKIP bit should be set to 1 in most cases. The crossbar skips these pins as if they were already assigned, and moves to the next unassigned pin.

It is possible for crossbar-assigned peripherals and dedicated functions to coexist on the same pin. For example, the port match function could be configured to watch for a falling edge on a UART RX line and generate an interrupt or wake up the device from a low-power state. However, if two functions share the same pin, the crossbar will have control over the output characteristics of that pin and the dedicated function will only have input access. Likewise, it is possible for firmware to read the logic state of any digital I/O pin assigned to a crossbar peripheral, but the output state cannot be directly modified.

Figure 11.3 Crossbar Priority Decoder Example Assignments on page 117 shows an example of the resulting pin assignments of the device with UART0 and SPI0 enabled and P0.3 skipped (P0SKIP = 0x08). UART0 is the highest priority and it will be assigned first. The UART0 pins can only appear at fixed locations (in this example, P0.4 and P0.5), so it occupies those pins. The next-highest enabled peripheral is SPI0. P0.0, P0.1 and P0.2 are free, so SPI0 takes these three pins. The fourth pin, NSS, is routed to P0.6 because P0.3 is skipped and P0.4 and P0.5 are already occupied by the UART. Any other pins on the device are available for use as general-purpose digital I/O or analog functions.

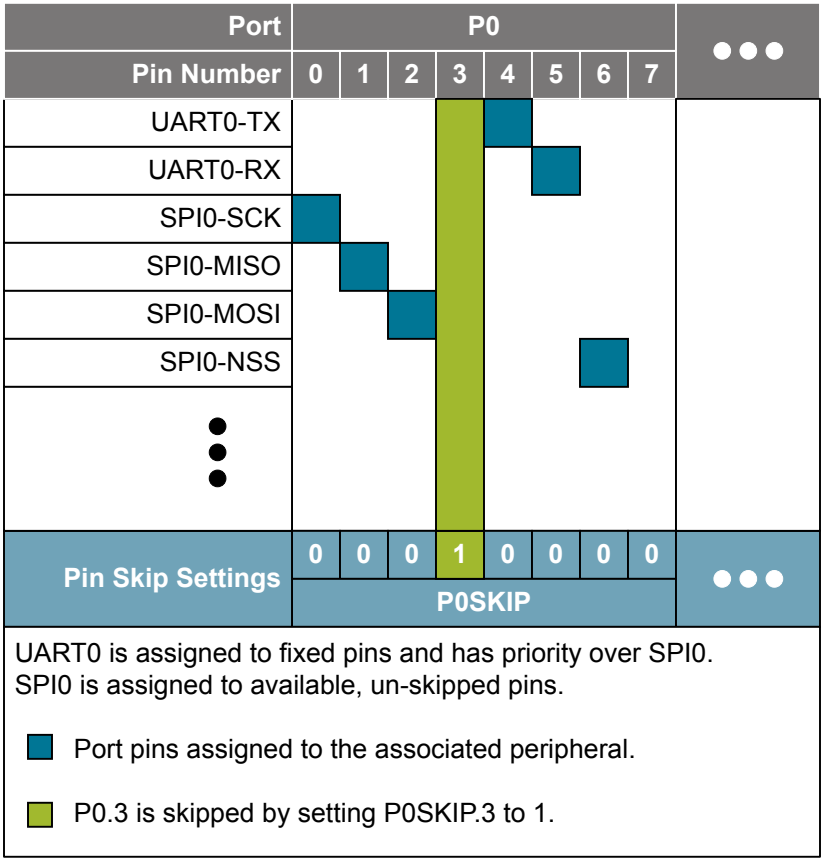
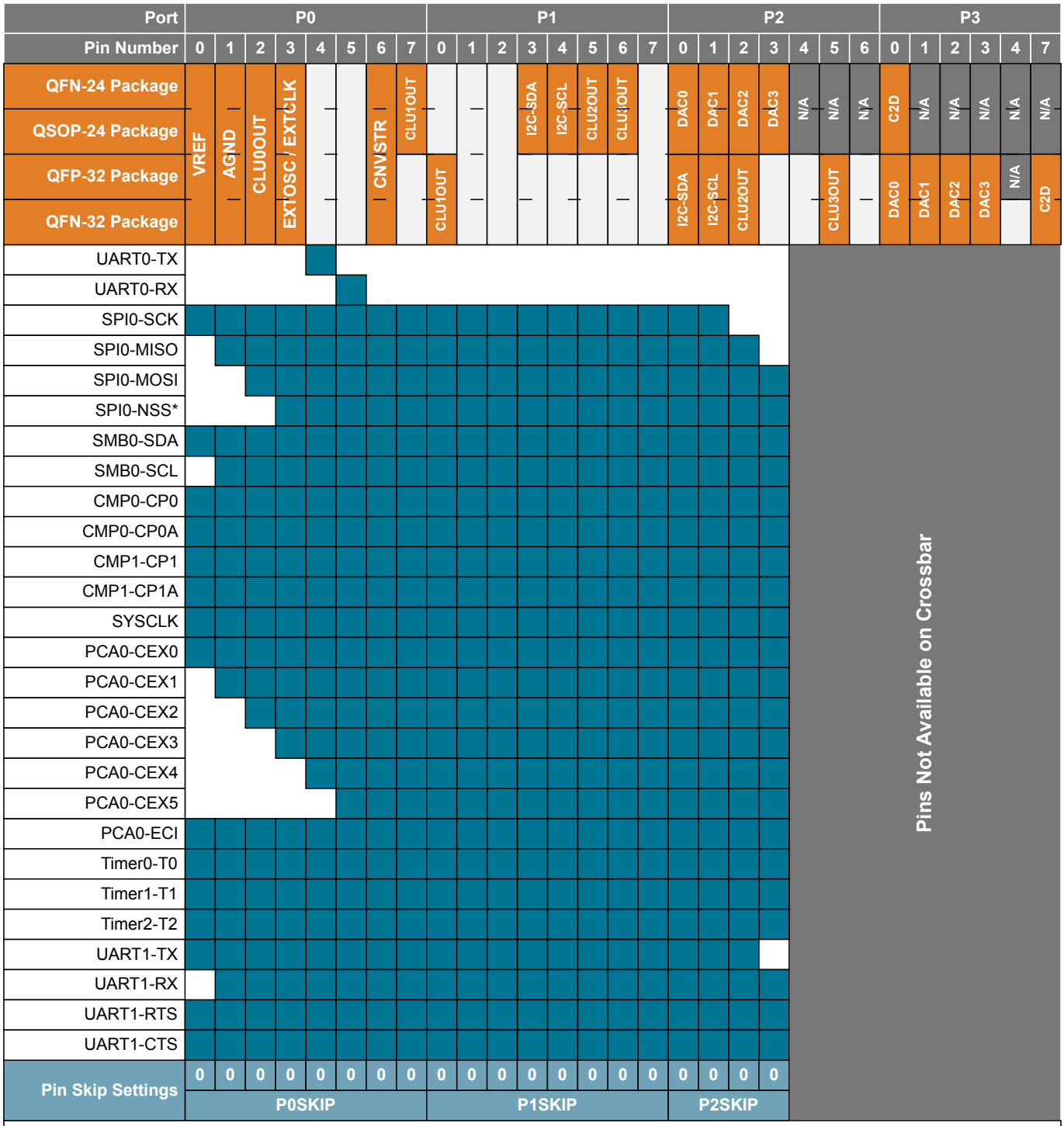


Figure 11.3. Crossbar Priority Decoder Example Assignments

11.3.3.1 Crossbar Functional Map

The figure below shows all of the potential peripheral-to-pin assignments available to the crossbar. Note that this does not mean any peripheral can always be assigned to the highlighted pins. The actual pin assignments are determined by the priority of the enabled peripherals.



The crossbar peripherals are assigned in priority order from top to bottom.

These boxes represent Port pins which can potentially be assigned to a peripheral.

Special Function Signals are not assigned by the crossbar. When these signals are enabled, the Crossbar should be manually configured to skip the corresponding port pins.

Pins can be “skipped” by setting the corresponding bit in PnSKIP to 1.

* NSS is only pinned out when the SPI is in 4-wire mode.

Figure 11.4. Full Crossbar Map

11.3.4 INT0 and INT1

Two direct-pin digital interrupt sources (INT0 and INT1) are included, which can be routed to port 0 pins. Additional I/O interrupts are available through the port match function. As is the case on a standard 8051 architecture, certain controls for these two interrupt sources are available in the Timer0/1 registers. Extensions to these controls which provide additional functionality are available in the IT01CF register. INT0 and INT1 are configurable as active high or low, edge- or level-sensitive. The IN0PL and IN1PL bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON select level- or edge-sensitive. The table below lists the possible configurations.

Table 11.3. INT0/INT1 configuration

IT0 or IT1	IN0PL or IN1PL	INT0 or INT1 Interrupt
1	0	Interrupt on falling edge
1	1	Interrupt on rising edge
0	0	Interrupt on low level
0	1	Interrupt on high level

INT0 and INT1 are assigned to port pins as defined in the IT01CF register. INT0 and INT1 port pin assignments are independent of any crossbar assignments, and may be assigned to pins used by crossbar peripherals. INT0 and INT1 will monitor their assigned port pins without disturbing the peripheral that was assigned the port pin via the crossbar. To assign a port pin only to INT0 and/or INT1, configure the crossbar to skip the selected pin(s).

IE0 and IE1 in the TCON register serve as the interrupt-pending flags for the INT0 and INT1 external interrupts, respectively. If an INT0 or INT1 external interrupt is configured as edge-sensitive, the corresponding interrupt pending flag is automatically cleared by the hardware when the CPU vectors to the ISR. When configured as level sensitive, the interrupt-pending flag remains logic 1 while the input is active as defined by the corresponding polarity bit (IN0PL or IN1PL); the flag remains logic 0 while the input is inactive. The external interrupt source must hold the input active until the interrupt request is recognized. It must then deactivate the interrupt request before execution of the ISR completes or another interrupt request will be generated.

11.3.5 Port Match

Port match functionality allows system events to be triggered by a logic value change on one or more port I/O pins. A software controlled value stored in the PnMATCH registers specifies the expected or normal logic values of the associated port pins (for example, P0MATCH.0 would correspond to P0.0). A port mismatch event occurs if the logic levels of the port's input pins no longer match the software controlled value. This allows software to be notified if a certain change or pattern occurs on the input pins regardless of the XBRn settings.

The PnMASK registers can be used to individually select which pins should be compared against the PnMATCH registers. A port mismatch event is generated if (Pn & PnMASK) does not equal (PnMATCH & PnMASK) for all ports with a PnMAT and PnMASK register.

If more than one port pin mask is enabled, firmware will need to read the pin values on a port match interrupt in order to determine the source pin that caused the interrupt. In some cases, the interrupt source may change or be removed before firmware has a chance to interrogate the port.

A port mismatch event may be used to generate an interrupt or wake the device from low power modes. See the interrupts and power options chapters for more details on interrupt and wake-up sources.

11.3.6 Direct Port I/O Access (Read/Write)

All port I/O are accessed through corresponding special function registers. When writing to a port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the crossbar, the port register can always read its corresponding port I/O pin). The exception to this is the execution of the read-modify-write instructions that target a Port Latch register as the destination. The read-modify-write instructions when operating on a port SFR are the following: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ and MOV, CLR or SETB, when the destination is an individual bit in a port SFR. For these instructions, the value of the latch register (not the pin) is read, modified, and written back to the SFR.

11.4 Port I/O Control Registers

11.4.1 XBR0: Port I/O Crossbar 0

Bit	7	6	5	4	3	2	1	0
Name	SYSCKE	CP1AE	CP1E	CP0AE	CP0E	SMB0E	SPI0E	URT0E
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xE1								

Bit	Name	Reset	Access	Description
7	SYSCKE	0	RW	**SYSCLK Output Enable.**
	Value	Name		Description
	0	DISABLED		SYSCLK unavailable at Port pin.
	1	ENABLED		SYSCLK output routed to Port pin.
6	CP1AE	0	RW	**Comparator1 Asynchronous Output Enable.**
	Value	Name		Description
	0	DISABLED		Asynchronous CP1 unavailable at Port pin.
	1	ENABLED		Asynchronous CP1 routed to Port pin.
5	CP1E	0	RW	**Comparator1 Output Enable.**
	Value	Name		Description
	0	DISABLED		CP1 unavailable at Port pin.
	1	ENABLED		CP1 routed to Port pin.
4	CP0AE	0	RW	**Comparator0 Asynchronous Output Enable.**
	Value	Name		Description
	0	DISABLED		Asynchronous CP0 unavailable at Port pin.
	1	ENABLED		Asynchronous CP0 routed to Port pin.
3	CP0E	0	RW	**Comparator0 Output Enable.**
	Value	Name		Description
	0	DISABLED		CP0 unavailable at Port pin.
	1	ENABLED		CP0 routed to Port pin.
2	SMB0E	0	RW	**SMB0 I/O Enable.**
	Value	Name		Description
	0	DISABLED		SMBus 0 I/O unavailable at Port pins.
	1	ENABLED		SMBus 0 I/O routed to Port pins.
1	SPI0E	0	RW	**SPI I/O Enable.**

Bit	Name	Reset	Access	Description
	Value	Name		Description
	0	DISABLED		SPI I/O unavailable at Port pins.
	1	ENABLED		SPI I/O routed to Port pins. The SPI can be assigned either 3 or 4 GPIO pins.
0	URT0E	0	RW	**UART0 I/O Enable.**
	Value	Name		Description
	0	DISABLED		UART0 I/O unavailable at Port pin.
	1	ENABLED		UART0 TX0, RX0 routed to Port pins P0.4 and P0.5.

11.4.2 XBR1: Port I/O Crossbar 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved	T2E	T1E	T0E	ECIE	PCA0ME		
Access	R	RW	RW	RW	RW	RW		
Reset	0	0	0	0	0	0x0		
SFR Page = 0x0, 0x20; SFR Address: 0xE2								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	T2E	0	RW	**T2 Enable.**
	Value	Name		Description
	0	DISABLED		T2 unavailable at Port pin.
	1	ENABLED		T2 routed to Port pin.
5	T1E	0	RW	**T1 Enable.**
	Value	Name		Description
	0	DISABLED		T1 unavailable at Port pin.
	1	ENABLED		T1 routed to Port pin.
4	T0E	0	RW	**T0 Enable.**
	Value	Name		Description
	0	DISABLED		T0 unavailable at Port pin.
	1	ENABLED		T0 routed to Port pin.
3	ECIE	0	RW	**PCA0 External Counter Input Enable.**
	Value	Name		Description
	0	DISABLED		ECI unavailable at Port pin.
	1	ENABLED		ECI routed to Port pin.
2:0	PCA0ME	0x0	RW	**PCA Module I/O Enable.**
	Value	Name		Description
	0x0	DISABLED		All PCA I/O unavailable at Port pins.
	0x1	CEX0		CEX0 routed to Port pin.
	0x2	CEX0_TO_CEX1		CEX0, CEX1 routed to Port pins.
	0x3	CEX0_TO_CEX2		CEX0, CEX1, CEX2 routed to Port pins.
	0x4	CEX0_TO_CEX3		CEX0, CEX1, CEX2, CEX3 routed to Port pins.
	0x5	CEX0_TO_CEX4		CEX0, CEX1, CEX2, CEX3, CEX4 routed to Port pins.
	0x6	CEX0_TO_CEX5		CEX0, CEX1, CEX2, CEX3, CEX4, CEX5 routed to Port pins.

11.4.3 XBR2: Port I/O Crossbar 2

Bit	7	6	5	4	3	2	1	0
Name	WEAKPUD	XBARE	Reserved			URT1CTSE	URT1RTSE	URT1E
Access	RW	RW	R			RW	RW	RW
Reset	0	0	0x0			0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xE3								

Bit	Name	Reset	Access	Description
7	WEAKPUD	0	RW	**Port I/O Weak Pullup Disable.**
	Value	Name		Description
	0	PULL_UPS_ENABLED		Weak Pullups enabled (except for Ports whose I/O are configured for analog mode).
	1	PULL_UPS_DISABLED		Weak Pullups disabled.
6	XBARE	0	RW	**Crossbar Enable.**
	Value	Name		Description
	0	DISABLED		Crossbar disabled.
	1	ENABLED		Crossbar enabled.
5:3	*Reserved*	*Must write reset value.*		
2	URT1CTSE	0	RW	**UART1 CTS Input Enable.**
	Value	Name		Description
	0	DISABLED		UART1 CTS1 unavailable at Port pin.
	1	ENABLED		UART1 CTS1 routed to Port pin.
1	URT1RTSE	0	RW	**UART1 RTS Output Enable.**
	Value	Name		Description
	0	DISABLED		UART1 RTS1 unavailable at Port pin.
	1	ENABLED		UART1 RTS1 routed to Port pin.
0	URT1E	0	RW	**UART1 I/O Enable.**
	Value	Name		Description
	0	DISABLED		UART1 I/O unavailable at Port pin.
	1	ENABLED		UART1 TX1 RX1 routed to Port pins.

11.4.4 PRTDRV: Port Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	Reserved				P3DRV	P2DRV	P1DRV	P0DRV
Access	R				RW	RW	RW	RW
Reset	0x0				1	1	1	1
SFR Page = 0x0, 0x20; SFR Address: 0xF6								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	P3DRV	1	RW	**Port 3 Drive Strength.**
	Value	Name		Description
	0	LOW_DRIVE		All pins on P3 use low drive strength.
	1	HIGH_DRIVE		All pins on P3 use high drive strength.
2	P2DRV	1	RW	**Port 2 Drive Strength.**
	Value	Name		Description
	0	LOW_DRIVE		All pins on P2 use low drive strength.
	1	HIGH_DRIVE		All pins on P2 use high drive strength.
1	P1DRV	1	RW	**Port 1 Drive Strength.**
	Value	Name		Description
	0	LOW_DRIVE		All pins on P1 use low drive strength.
	1	HIGH_DRIVE		All pins on P1 use high drive strength.
0	P0DRV	1	RW	**Port 0 Drive Strength.**
	Value	Name		Description
	0	LOW_DRIVE		All pins on P0 use low drive strength.
	1	HIGH_DRIVE		All pins on P0 use high drive strength.

11.4.5 P0MASK: Port 0 Mask

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xFE								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 0 Bit 7 Mask Value.**
	Value	Name		Description
	0	IGNORED		P0.7 pin logic value is ignored and will not cause a port mismatch event.
	1	COMPARED		P0.7 pin logic value is compared to P0MAT.7.
6	B6	0	RW	**Port 0 Bit 6 Mask Value.**
	See bit 7 description			
5	B5	0	RW	**Port 0 Bit 5 Mask Value.**
	See bit 7 description			
4	B4	0	RW	**Port 0 Bit 4 Mask Value.**
	See bit 7 description			
3	B3	0	RW	**Port 0 Bit 3 Mask Value.**
	See bit 7 description			
2	B2	0	RW	**Port 0 Bit 2 Mask Value.**
	See bit 7 description			
1	B1	0	RW	**Port 0 Bit 1 Mask Value.**
	See bit 7 description			
0	B0	0	RW	**Port 0 Bit 0 Mask Value.**
	See bit 7 description			

11.4.6 P0MAT: Port 0 Match

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = 0x0, 0x20; SFR Address: 0xFD								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 0 Bit 7 Match Value.**
	Value	Name		Description
	0	LOW		P0.7 pin logic value is compared with logic LOW.
	1	HIGH		P0.7 pin logic value is compared with logic HIGH.
6	B6 See bit 7 description	1	RW	**Port 0 Bit 6 Match Value.**
5	B5 See bit 7 description	1	RW	**Port 0 Bit 5 Match Value.**
4	B4 See bit 7 description	1	RW	**Port 0 Bit 4 Match Value.**
3	B3 See bit 7 description	1	RW	**Port 0 Bit 3 Match Value.**
2	B2 See bit 7 description	1	RW	**Port 0 Bit 2 Match Value.**
1	B1 See bit 7 description	1	RW	**Port 0 Bit 1 Match Value.**
0	B0 See bit 7 description	1	RW	**Port 0 Bit 0 Match Value.**

11.4.7 P0: Port 0 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = ALL; SFR Address: 0x80 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 0 Bit 7 Latch.**
	Value	Name		Description
	0	LOW		P0.7 is low. Set P0.7 to drive low.
	1	HIGH		P0.7 is high. Set P0.7 to drive or float high.
6	B6 See bit 7 description	1	RW	**Port 0 Bit 6 Latch.**
5	B5 See bit 7 description	1	RW	**Port 0 Bit 5 Latch.**
4	B4 See bit 7 description	1	RW	**Port 0 Bit 4 Latch.**
3	B3 See bit 7 description	1	RW	**Port 0 Bit 3 Latch.**
2	B2 See bit 7 description	1	RW	**Port 0 Bit 2 Latch.**
1	B1 See bit 7 description	1	RW	**Port 0 Bit 1 Latch.**
0	B0 See bit 7 description	1	RW	**Port 0 Bit 0 Latch.**

Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.

Reading this register returns the logic value at the pin, regardless if it is configured as output or input.

11.4.8 P0MDIN: Port 0 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = 0x0, 0x20; SFR Address: 0xF1								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 0 Bit 7 Input Mode.**
	Value	Name		Description
	0	ANALOG		P0.7 pin is configured for analog mode.
	1	DIGITAL		P0.7 pin is configured for digital mode.
6	B6 See bit 7 description	1	RW	**Port 0 Bit 6 Input Mode.**
5	B5 See bit 7 description	1	RW	**Port 0 Bit 5 Input Mode.**
4	B4 See bit 7 description	1	RW	**Port 0 Bit 4 Input Mode.**
3	B3 See bit 7 description	1	RW	**Port 0 Bit 3 Input Mode.**
2	B2 See bit 7 description	1	RW	**Port 0 Bit 2 Input Mode.**
1	B1 See bit 7 description	1	RW	**Port 0 Bit 1 Input Mode.**
0	B0 See bit 7 description	1	RW	**Port 0 Bit 0 Input Mode.**
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.9 P0MDOUT: Port 0 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xA4								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 0 Bit 7 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P0.7 output is open-drain.
	1	PUSH_PULL		P0.7 output is push-pull.
6	B6	0	RW	**Port 0 Bit 6 Output Mode.**
	See bit 7 description			
5	B5	0	RW	**Port 0 Bit 5 Output Mode.**
	See bit 7 description			
4	B4	0	RW	**Port 0 Bit 4 Output Mode.**
	See bit 7 description			
3	B3	0	RW	**Port 0 Bit 3 Output Mode.**
	See bit 7 description			
2	B2	0	RW	**Port 0 Bit 2 Output Mode.**
	See bit 7 description			
1	B1	0	RW	**Port 0 Bit 1 Output Mode.**
	See bit 7 description			
0	B0	0	RW	**Port 0 Bit 0 Output Mode.**
	See bit 7 description			

11.4.10 P0SKIP: Port 0 Skip

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xD4								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 0 Bit 7 Skip.**
	Value	Name		Description
	0	NOT_SKIPPED		P0.7 pin is not skipped by the crossbar.
	1	SKIPPED		P0.7 pin is skipped by the crossbar.
6	B6	0	RW	**Port 0 Bit 6 Skip.**
	See bit 7 description			
5	B5	0	RW	**Port 0 Bit 5 Skip.**
	See bit 7 description			
4	B4	0	RW	**Port 0 Bit 4 Skip.**
	See bit 7 description			
3	B3	0	RW	**Port 0 Bit 3 Skip.**
	See bit 7 description			
2	B2	0	RW	**Port 0 Bit 2 Skip.**
	See bit 7 description			
1	B1	0	RW	**Port 0 Bit 1 Skip.**
	See bit 7 description			
0	B0	0	RW	**Port 0 Bit 0 Skip.**
	See bit 7 description			

11.4.11 P1MASK: Port 1 Mask

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xEE								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 1 Bit 7 Mask Value.**
	Value	Name		Description
	0	IGNORED		P1.7 pin logic value is ignored and will not cause a port mismatch event.
	1	COMPARED		P1.7 pin logic value is compared to P1MAT.7.
6	B6	0	RW	**Port 1 Bit 6 Mask Value.**
	See bit 7 description			
5	B5	0	RW	**Port 1 Bit 5 Mask Value.**
	See bit 7 description			
4	B4	0	RW	**Port 1 Bit 4 Mask Value.**
	See bit 7 description			
3	B3	0	RW	**Port 1 Bit 3 Mask Value.**
	See bit 7 description			
2	B2	0	RW	**Port 1 Bit 2 Mask Value.**
	See bit 7 description			
1	B1	0	RW	**Port 1 Bit 1 Mask Value.**
	See bit 7 description			
0	B0	0	RW	**Port 1 Bit 0 Mask Value.**
	See bit 7 description			

11.4.12 P1MAT: Port 1 Match

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = 0x0, 0x20; SFR Address: 0xED								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 1 Bit 7 Match Value.**
	Value	Name		Description
	0	LOW		P1.7 pin logic value is compared with logic LOW.
	1	HIGH		P1.7 pin logic value is compared with logic HIGH.
6	B6 See bit 7 description	1	RW	**Port 1 Bit 6 Match Value.**
5	B5 See bit 7 description	1	RW	**Port 1 Bit 5 Match Value.**
4	B4 See bit 7 description	1	RW	**Port 1 Bit 4 Match Value.**
3	B3 See bit 7 description	1	RW	**Port 1 Bit 3 Match Value.**
2	B2 See bit 7 description	1	RW	**Port 1 Bit 2 Match Value.**
1	B1 See bit 7 description	1	RW	**Port 1 Bit 1 Match Value.**
0	B0 See bit 7 description	1	RW	**Port 1 Bit 0 Match Value.**

11.4.13 P1: Port 1 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1

SFR Page = ALL; SFR Address: 0x90 (bit-addressable)

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 1 Bit 7 Latch.**
	Value	Name		Description
	0	LOW		P1.7 is low. Set P1.7 to drive low.
	1	HIGH		P1.7 is high. Set P1.7 to drive or float high.
6	B6 See bit 7 description	1	RW	**Port 1 Bit 6 Latch.**
5	B5 See bit 7 description	1	RW	**Port 1 Bit 5 Latch.**
4	B4 See bit 7 description	1	RW	**Port 1 Bit 4 Latch.**
3	B3 See bit 7 description	1	RW	**Port 1 Bit 3 Latch.**
2	B2 See bit 7 description	1	RW	**Port 1 Bit 2 Latch.**
1	B1 See bit 7 description	1	RW	**Port 1 Bit 1 Latch.**
0	B0 See bit 7 description	1	RW	**Port 1 Bit 0 Latch.**

Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.

Reading this register returns the logic value at the pin, regardless if it is configured as output or input.

11.4.14 P1MDIN: Port 1 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = 0x0, 0x20; SFR Address: 0xF2								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 1 Bit 7 Input Mode.**
	Value	Name		Description
	0	ANALOG		P1.7 pin is configured for analog mode.
	1	DIGITAL		P1.7 pin is configured for digital mode.
6	B6 See bit 7 description	1	RW	**Port 1 Bit 6 Input Mode.**
5	B5 See bit 7 description	1	RW	**Port 1 Bit 5 Input Mode.**
4	B4 See bit 7 description	1	RW	**Port 1 Bit 4 Input Mode.**
3	B3 See bit 7 description	1	RW	**Port 1 Bit 3 Input Mode.**
2	B2 See bit 7 description	1	RW	**Port 1 Bit 2 Input Mode.**
1	B1 See bit 7 description	1	RW	**Port 1 Bit 1 Input Mode.**
0	B0 See bit 7 description	1	RW	**Port 1 Bit 0 Input Mode.**
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.15 P1MDOUT: Port 1 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xA5								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 1 Bit 7 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P1.7 output is open-drain.
	1	PUSH_PULL		P1.7 output is push-pull.
6	B6	0	RW	**Port 1 Bit 6 Output Mode.**
	See bit 7 description			
5	B5	0	RW	**Port 1 Bit 5 Output Mode.**
	See bit 7 description			
4	B4	0	RW	**Port 1 Bit 4 Output Mode.**
	See bit 7 description			
3	B3	0	RW	**Port 1 Bit 3 Output Mode.**
	See bit 7 description			
2	B2	0	RW	**Port 1 Bit 2 Output Mode.**
	See bit 7 description			
1	B1	0	RW	**Port 1 Bit 1 Output Mode.**
	See bit 7 description			
0	B0	0	RW	**Port 1 Bit 0 Output Mode.**
	See bit 7 description			

11.4.16 P1SKIP: Port 1 Skip

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xD5								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 1 Bit 7 Skip.**
	Value	Name		Description
	0	NOT_SKIPPED		P1.7 pin is not skipped by the crossbar.
	1	SKIPPED		P1.7 pin is skipped by the crossbar.
6	B6 See bit 7 description	0	RW	**Port 1 Bit 6 Skip.**
5	B5 See bit 7 description	0	RW	**Port 1 Bit 5 Skip.**
4	B4 See bit 7 description	0	RW	**Port 1 Bit 4 Skip.**
3	B3 See bit 7 description	0	RW	**Port 1 Bit 3 Skip.**
2	B2 See bit 7 description	0	RW	**Port 1 Bit 2 Skip.**
1	B1 See bit 7 description	0	RW	**Port 1 Bit 1 Skip.**
0	B0 See bit 7 description	0	RW	**Port 1 Bit 0 Skip.**

11.4.17 P2MASK: Port 2 Mask

Bit	7	6	5	4	3	2	1	0
Name	Reserved	B6	B5	B4	B3	B2	B1	B0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x20; SFR Address: 0xFC								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	B6	0	RW	**Port 2 Bit 6 Mask Value.**
	Value	Name		Description
	0	IGNORED		P2.6 pin logic value is ignored and will not cause a port mismatch event.
	1	COMPARED		P2.6 pin logic value is compared to P2MAT.6.
5	B5 See bit 6 description	0	RW	**Port 2 Bit 5 Mask Value.**
4	B4 See bit 6 description	0	RW	**Port 2 Bit 4 Mask Value.**
3	B3 See bit 6 description	0	RW	**Port 2 Bit 3 Mask Value.**
2	B2 See bit 6 description	0	RW	**Port 2 Bit 2 Mask Value.**
1	B1 See bit 6 description	0	RW	**Port 2 Bit 1 Mask Value.**
0	B0 See bit 6 description	0	RW	**Port 2 Bit 0 Mask Value.**

11.4.18 P2MAT: Port 2 Match

Bit	7	6	5	4	3	2	1	0
Name	Reserved	B6	B5	B4	B3	B2	B1	B0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = 0x20; SFR Address: 0xFB								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	B6	1	RW	**Port 2 Bit 6 Match Value.**
	Value	Name		Description
	0	LOW		P2.6 pin logic value is compared with logic LOW.
	1	HIGH		P2.6 pin logic value is compared with logic HIGH.
5	B5	1	RW	**Port 2 Bit 5 Match Value.**
	See bit 6 description			
4	B4	1	RW	**Port 2 Bit 4 Match Value.**
	See bit 6 description			
3	B3	1	RW	**Port 2 Bit 3 Match Value.**
	See bit 6 description			
2	B2	1	RW	**Port 2 Bit 2 Match Value.**
	See bit 6 description			
1	B1	1	RW	**Port 2 Bit 1 Match Value.**
	See bit 6 description			
0	B0	1	RW	**Port 2 Bit 0 Match Value.**
	See bit 6 description			

11.4.19 P2: Port 2 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	Reserved	B6	B5	B4	B3	B2	B1	B0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	0	1	1	1	1	1	1	1

SFR Page = ALL; SFR Address: 0xA0 (bit-addressable)

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	B6	1	RW	**Port 2 Bit 6 Latch.**
	Value	Name		Description
	0	LOW		P2.6 is low. Set P2.6 to drive low.
	1	HIGH		P2.6 is high. Set P2.6 to drive or float high.
5	B5	1	RW	**Port 2 Bit 5 Latch.**
	See bit 6 description			
4	B4	1	RW	**Port 2 Bit 4 Latch.**
	See bit 6 description			
3	B3	1	RW	**Port 2 Bit 3 Latch.**
	See bit 6 description			
2	B2	1	RW	**Port 2 Bit 2 Latch.**
	See bit 6 description			
1	B1	1	RW	**Port 2 Bit 1 Latch.**
	See bit 6 description			
0	B0	1	RW	**Port 2 Bit 0 Latch.**
	See bit 6 description			

Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.

Reading this register returns the logic value at the pin, regardless if it is configured as output or input.

11.4.20 P2MDIN: Port 2 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	Reserved	B6	B5	B4	B3	B2	B1	B0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = 0x20; SFR Address: 0xF3								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	B6	1	RW	**Port 2 Bit 6 Input Mode.**
	Value	Name		Description
	0	ANALOG		P2.6 pin is configured for analog mode.
	1	DIGITAL		P2.6 pin is configured for digital mode.
5	B5 See bit 6 description	1	RW	**Port 2 Bit 5 Input Mode.**
4	B4 See bit 6 description	1	RW	**Port 2 Bit 4 Input Mode.**
3	B3 See bit 6 description	1	RW	**Port 2 Bit 3 Input Mode.**
2	B2 See bit 6 description	1	RW	**Port 2 Bit 2 Input Mode.**
1	B1 See bit 6 description	1	RW	**Port 2 Bit 1 Input Mode.**
0	B0 See bit 6 description	1	RW	**Port 2 Bit 0 Input Mode.**
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.21 P2MDOUT: Port 2 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	Reserved	B6	B5	B4	B3	B2	B1	B0
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xA6								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	B6	0	RW	**Port 2 Bit 6 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P2.6 output is open-drain.
	1	PUSH_PULL		P2.6 output is push-pull.
5	B5	0	RW	**Port 2 Bit 5 Output Mode.**
	See bit 6 description			
4	B4	0	RW	**Port 2 Bit 4 Output Mode.**
	See bit 6 description			
3	B3	0	RW	**Port 2 Bit 3 Output Mode.**
	See bit 6 description			
2	B2	0	RW	**Port 2 Bit 2 Output Mode.**
	See bit 6 description			
1	B1	0	RW	**Port 2 Bit 1 Output Mode.**
	See bit 6 description			
0	B0	0	RW	**Port 2 Bit 0 Output Mode.**
	See bit 6 description			

11.4.22 P2SKIP: Port 2 Skip

Bit	7	6	5	4	3	2	1	0
Name	Reserved				B3	B2	B1	B0
Access	R				RW	RW	RW	RW
Reset	0x0				0	0	0	0
SFR Page = 0x20; SFR Address: 0xCC								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	B3	0	RW	**Port 2 Bit 3 Skip.**
	Value	Name		Description
	0	NOT_SKIPPED		P2.3 pin is not skipped by the crossbar.
	1	SKIPPED		P2.3 pin is skipped by the crossbar.
2	B2 See bit 3 description	0	RW	**Port 2 Bit 2 Skip.**
1	B1 See bit 3 description	0	RW	**Port 2 Bit 1 Skip.**
0	B0 See bit 3 description	0	RW	**Port 2 Bit 0 Skip.**

11.4.23 P3: Port 3 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	B7	Reserved		B4	B3	B2	B1	B0
Access	RW	R		RW	RW	RW	RW	RW
Reset	1	0x0		1	1	1	1	1
SFR Page = ALL; SFR Address: 0xB0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 3 Bit 7 Latch.**
	Value	Name		Description
	0	LOW		P3.7 is low. Set P3.7 to drive low.
	1	HIGH		P3.7 is high. Set P3.7 to drive or float high.
6:5	*Reserved*	*Must write reset value.*		
4	B4	1	RW	**Port 3 Bit 4 Latch.**
	See bit 7 description			
3	B3	1	RW	**Port 3 Bit 3 Latch.**
	See bit 7 description			
2	B2	1	RW	**Port 3 Bit 2 Latch.**
	See bit 7 description			
1	B1	1	RW	**Port 3 Bit 1 Latch.**
	See bit 7 description			
0	B0	1	RW	**Port 3 Bit 0 Latch.**
	See bit 7 description			

Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.

Reading this register returns the logic value at the pin, regardless if it is configured as output or input.

11.4.24 P3MDIN: Port 3 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	B7	Reserved		B4	B3	B2	B1	B0
Access	RW	R		RW	RW	RW	RW	RW
Reset	1	0x3		1	1	1	1	1
SFR Page = 0x20; SFR Address: 0xF4								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 3 Bit 7 Input Mode.**
	Value	Name		Description
	0	ANALOG		P3.7 pin is configured for analog mode.
	1	DIGITAL		P3.7 pin is configured for digital mode.
6:5	*Reserved*	*Must write reset value.*		
4	B4	1	RW	**Port 3 Bit 4 Input Mode.**
	See bit 7 description			
3	B3	1	RW	**Port 3 Bit 3 Input Mode.**
	See bit 7 description			
2	B2	1	RW	**Port 3 Bit 2 Input Mode.**
	See bit 7 description			
1	B1	1	RW	**Port 3 Bit 1 Input Mode.**
	See bit 7 description			
0	B0	1	RW	**Port 3 Bit 0 Input Mode.**
	See bit 7 description			
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.25 P3MDOUT: Port 3 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	B7	Reserved		B4	B3	B2	B1	B0
Access	RW	R		RW	RW	RW	RW	RW
Reset	0	0x0		0	0	0	0	0
SFR Page = 0x20; SFR Address: 0x9C								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 3 Bit 7 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P3.7 output is open-drain.
	1	PUSH_PULL		P3.7 output is push-pull.
6:5	*Reserved*	*Must write reset value.*		
4	B4	0	RW	**Port 3 Bit 4 Output Mode.**
	See bit 7 description			
3	B3	0	RW	**Port 3 Bit 3 Output Mode.**
	See bit 7 description			
2	B2	0	RW	**Port 3 Bit 2 Output Mode.**
	See bit 7 description			
1	B1	0	RW	**Port 3 Bit 1 Output Mode.**
	See bit 7 description			
0	B0	0	RW	**Port 3 Bit 0 Output Mode.**
	See bit 7 description			

11.5 INT0 and INT1 Control Registers

11.5.1 IT01CF: INT0/INT1 Configuration

Bit	7	6	5	4	3	2	1	0
Name	IN1PL	IN1SL			IN0PL	IN0SL		
Access	RW	RW			RW	RW		
Reset	0	0x0			0	0x1		
SFR Page = 0x0, 0x10; SFR Address: 0xE4								

Bit	Name	Reset	Access	Description
7	IN1PL	0	RW	**INT1 Polarity.**

Value	Name	Description
0	ACTIVE_LOW	INT1 input is active low.
1	ACTIVE_HIGH	INT1 input is active high.

Bit	Name	Reset	Access	Description
6:4	IN1SL	0x0	RW	**INT1 Port Pin Selection.** These bits select which port pin is assigned to INT1. This pin assignment is independent of the Crossbar; INT1 will monitor the assigned port pin without disturbing the peripheral that has been assigned the port pin via the Crossbar. The Crossbar will not assign the port pin to a peripheral if it is configured to skip the selected pin.

Value	Name	Description
0x0	P0_0	Select P0.0.
0x1	P0_1	Select P0.1.
0x2	P0_2	Select P0.2.
0x3	P0_3	Select P0.3.
0x4	P0_4	Select P0.4.
0x5	P0_5	Select P0.5.
0x6	P0_6	Select P0.6.
0x7	P0_7	Select P0.7.

Bit	Name	Reset	Access	Description
3	IN0PL	0	RW	**INT0 Polarity.**

Value	Name	Description
0	ACTIVE_LOW	INT0 input is active low.
1	ACTIVE_HIGH	INT0 input is active high.

Bit	Name	Reset	Access	Description
2:0	IN0SL	0x1	RW	**INT0 Port Pin Selection.** These bits select which port pin is assigned to INT0. This pin assignment is independent of the Crossbar; INT0 will monitor the assigned port pin without disturbing the peripheral that has been assigned the port pin via the Crossbar. The Crossbar will not assign the port pin to a peripheral if it is configured to skip the selected pin.

Value	Name	Description
0x0	P0_0	Select P0.0.
0x1	P0_1	Select P0.1.
0x2	P0_2	Select P0.2.

Bit	Name	Reset	Access	Description
	0x3	P0_3		Select P0.3.
	0x4	P0_4		Select P0.4.
	0x5	P0_5		Select P0.5.
	0x6	P0_6		Select P0.6.
	0x7	P0_7		Select P0.7.

12. Analog to Digital Converter (ADC0)

12.1 Introduction

The ADC is a successive-approximation-register (SAR) ADC with 12- and 10-bit modes, integrated track-and hold and a programmable window detector. The ADC is fully configurable under software control via several registers. The ADC may be configured to measure different signals using the analog multiplexer. The voltage reference for the ADC is selectable between internal and external reference sources.

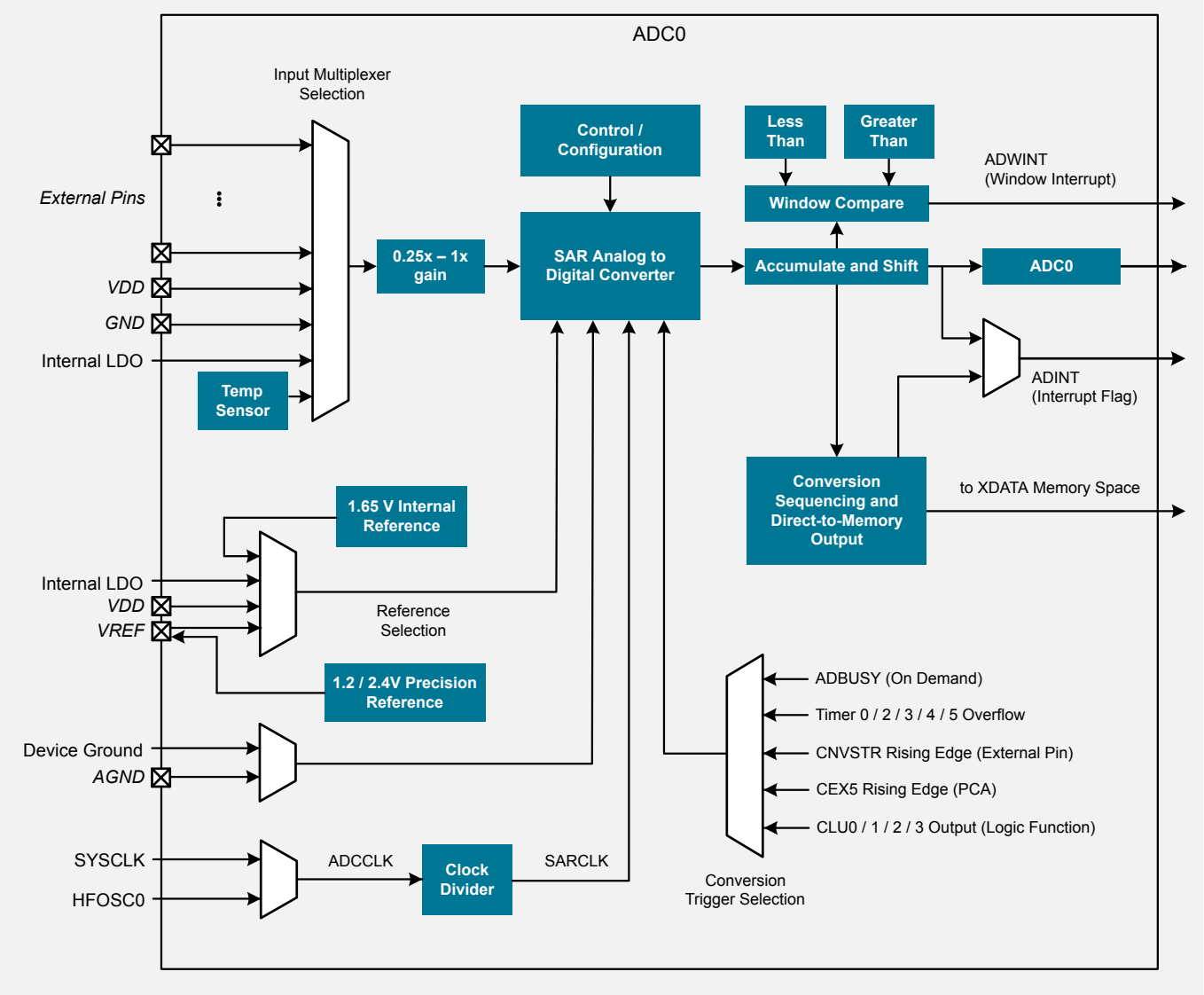


Figure 12.1. ADC Block Diagram

12.2 Features

- Up to 20 external inputs
- Single-ended 12-bit and 10-bit modes
- Supports an output update rate of up to 350 ksps in 12-bit mode
- Channel sequencer logic with direct-to-XDATA output transfers
- Operation in a low power mode at lower conversion speeds
- Asynchronous hardware conversion trigger, selectable between software, external I/O and internal timer and configurable logic sources
- Output data window comparator allows automatic range checking
- Support for output data accumulation
- Conversion complete and window compare interrupts supported
- Flexible output data formatting
- Includes a fully-internal fast-settling 1.65 V reference and an on-chip precision 2.4 / 1.2 V reference, with support for using the supply as the reference, an external reference and signal ground
- Integrated temperature sensor

12.3 Functional Description

12.3.1 Input Selection

The ADC has an analog multiplexer which allows selection of external pins, the on-chip temperature sensor, the internal regulated supply, the VDD supply, or GND. ADC input channels are selected using the ADC0MX register.

Note: Any port pins selected as ADC inputs should be configured as analog inputs in their associated port configuration register, and configured to be skipped by the crossbar.

12.3.1.1 Multiplexer Channel Selection

Table 12.1. ADC0 Input Multiplexer Channels

ADC0MX setting	Signal Name	Enumeration Name	QFN32 / QFP32 Pin Name	QSOP24 Pin Name	QFN24 Pin Name
00000	ADC0.0	ADC0P0	P0.1	P0.1	P0.1
00001	ADC0.1	ADC0P1	P0.2	P0.2	P0.2
00010	ADC0.2	ADC0P2	P0.4	P0.4	P0.4
00011	ADC0.3	ADC0P3	P0.5	P0.5	P0.5
00100	ADC0.4	ADC0P4	P0.6	P0.6	P0.6
00101	ADC0.5	ADC0P5	P0.7	P0.7	P0.7
00110	ADC0.6	ADC0P6	P1.0	P1.0	P1.0
00111	ADC0.7	ADC0P7	P1.1	P1.1	P1.1
01000	ADC0.8	ADC0P8	P1.2	P1.2	P1.2
01001	ADC0.9	ADC0P9	P1.3	P1.4	P1.4
01010	ADC0.10	ADC0P10	P1.4	P1.5	P1.5
01011	ADC0.11	ADC0P11	P1.5	P1.6	P1.6
01100	ADC0.12	ADC0P12	P1.6	P1.7	Reserved
01101	ADC0.13	ADC0P13	P1.7	Reserved	Reserved
01110	ADC0.14	ADC0P14	P2.1	Reserved	Reserved

ADC0MX setting	Signal Name	Enumeration Name	QFN32 / QFP32 Pin Name	QSOP24 Pin Name	QFN24 Pin Name
01111	ADC0.15	ADC0P15	P2.2	Reserved	Reserved
10000	ADC0.16	ADC0P16	P2.3	Reserved	Reserved
10001	ADC0.17	ADC0P17	P2.4	Reserved	Reserved
10010	ADC0.18	ADC0P18	P2.5	Reserved	Reserved
10011	ADC0.19	ADC0P19	P2.6	Reserved	Reserved
10100	ADC0.20	TEMP	Internal Temperature Sensor		
10101	ADC0.21	LDO_OUT	Internal 1.8 V LDO Output		
10110	ADC0.22	VDD	VDD Supply Pin		
10111	ADC0.23	GND	GND Supply Pin		
11000 - 11110	ADC0.24 - ADC0.30		Reserved	Reserved	Reserved
11111	ADC0.31	NONE	No connection		

12.3.2 Gain Setting

The ADC has gain settings of 1x, 0.75x, 0.5x and 0.25x. In 1x mode, the full scale reading of the ADC is determined directly by VREF. In the other modes, the full-scale reading of the ADC occurs when the input voltage is equal to VREF divided by the selected gain. For example, in 0.5x mode, the full scale input voltage is VREF / 0.5 = VREF x 2. The lower gain settings can be useful to obtain a higher input voltage range when using a small VREF voltage, or to measure input voltages that are between VREF and the supply voltage. Gain settings for the ADC are controlled by the ADGN field in register ADC0CN0. Note that even with the lower gain settings, voltages above the supply rail cannot be measured directly by the ADC.

12.3.3 Voltage Reference Options

The voltage reference multiplexer is configurable to use a number of different internal and external reference sources. The ground reference mux allows the ground reference for ADC0 to be selected between the ground pin (GND) or a port pin dedicated to analog ground (AGND). The voltage and ground reference options are configured using the REF0CN register. The REFSL field selects between the different reference options, while GNDSL configures the ground connection.

12.3.3.1 Internal Voltage Reference

The high-speed internal reference is self-contained and stabilized. It is not routed to an external pin and requires no external decoupling. When selected, the internal reference will be automatically enabled/disabled on an as-needed basis by the ADC. The reference is nominally 1.65 V. The electrical specification tables in the datasheet have more information about the accuracy of this reference source.

12.3.3.2 Supply or LDO Voltage Reference

For applications with a non-varying power supply voltage, using the power supply as the voltage reference can provide the ADC with added dynamic range at the cost of reduced power supply noise rejection. Additionally, the internal LDO supply to the core may be used as a reference. Neither of these reference sources are routed to the VREF pin, and do not require additional external decoupling.

12.3.3.3 External Voltage Reference

An external reference may be applied to the VREF pin. Bypass capacitors should be added as recommended by the manufacturer of the external voltage reference. If the manufacturer does not provide recommendations, a 4.7 µF in parallel with a 0.1 µF capacitor is recommended.

Note: The VREF pin is a multi-function GPIO pin. When using an external voltage reference, VREF should be configured as an analog input and skipped by the crossbar.

12.3.3.4 Precision Voltage Reference

The precision voltage reference source is an on-chip block which requires external bypass (see the VREF chapter for details). The precision reference is routed to the VREF pin. To use the precision reference with the ADC, it should be enabled and settled, and the ADC's REFSL field should be set to the VREF pin setting.

12.3.3.5 Ground Reference

To prevent ground noise generated by switching digital logic from affecting sensitive analog measurements, a separate analog ground reference option is available. When enabled, the ground reference for the ADC during both the tracking/sampling and the conversion periods is taken from the AGND pin. Any external sensors sampled by the ADC should be referenced to the AGND pin. If an external voltage reference is used, the AGND pin should be connected to the ground of the external reference and its associated decoupling capacitor. The separate analog ground reference option is enabled by setting GNDSL to 1. Note that when sampling the internal temperature sensor, the internal chip ground is always used for the sampling operation, regardless of the setting of the GNDSL bit. Similarly, whenever the internal high-speed reference is selected, the internal chip ground is always used during the conversion period, regardless of the setting of the GNDSL bit.

Note: The AGND pin is a multi-function GPIO pin. When using AGND as the ground reference to the ADC, AGND should be configured as an analog input and skipped by the crossbar.

12.3.4 Clocking

The ADC clock (ADCCLK) can be selected from one of two sources using the ADCLKSEL field in ADC0CF0. The default selection is the system clock (SYSCLK). For applications requiring faster conversions but using a slower system clock, the HFOSC0 oscillator may be selected as the ADC clock source. ADCCLK is used to clock registers and other logic in the ADC.

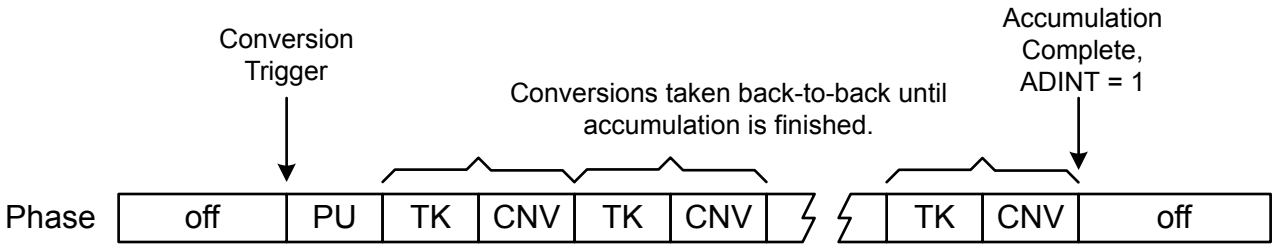
The conversion process is driven by the SAR clock (SARCLK). SARCLK is a divided version of the ADCCLK. The ADSC field in ADC0CF0 determines the divide ratio for SARCLK. In most applications, SARCLK should be adjusted to operate as fast as possible, without exceeding the maximum SAR clock frequency of 18 MHz.

12.3.5 Timing

Each ADC conversion may consist of multiple phases: power-up, tracking, and conversion. The power-up phase allows time for the ADC and internal reference circuitry to power on before sampling the input and performing a conversion. The power-up phase is part of the conversion process only when the ADC is configured to power off after the conversion is complete (IPOEN = 1). When IPOEN = 1, the ADC will power up, accumulate the requested number of conversions, and then power back off. The power-up phase is only present before the first conversion. When IPOEN = 0, the power up phase will only happen when the ADC is enabled.

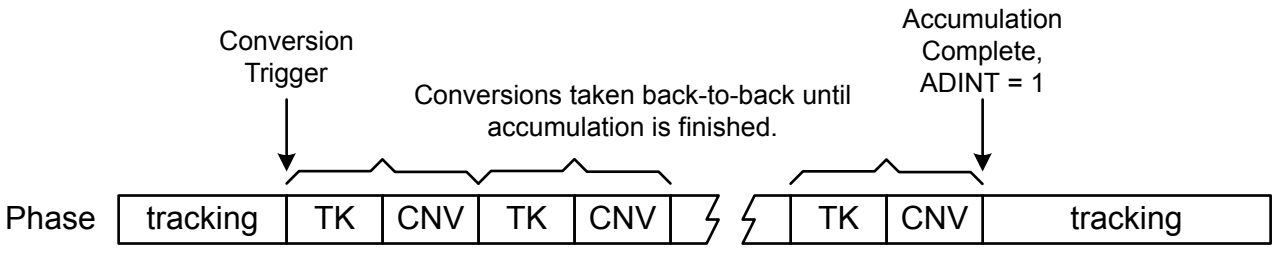
The tracking phase is the time period when the ADC multiplexer is connected to the selected input and sampled. Tracking can be defined to occur whenever a conversion is not in progress, or the ADC may be configured to track the input for a specific time prior to each conversion. When accumulating multiple conversions, it is important that the ADTK field be programmed for sufficient tracking between each conversion.

At the end of the tracking phase, the sample/hold circuit disconnects the input from the selected channel, and the sampled voltage is then converted to a digital value during the conversion phase.



off = ADC shut down.
PU = Power-Up Phase. Timing Defined by ADPWR field.
TK = Tracking Phase. Timing Defined by ADTK field.
CNV = Conversion Phase. Timing depends on resolution and SARCLK.

Figure 12.2. ADC Timing With IPOEN = 1



tracking = Converter tracking selected input any time conversion is not in progress.
TK = Tracking Phase. Timing Defined by ADTK field.
CNV = Conversion Phase. Timing depends on resolution and SARCLK.

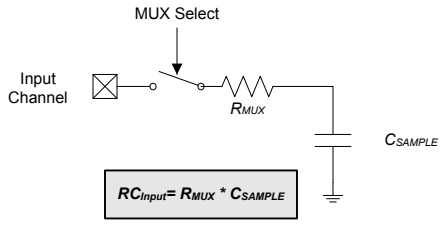
Figure 12.3. ADC Timing With IPOEN = 0

12.3.5.1 Input Tracking

Each ADC conversion must be preceded by a minimum tracking time to allow the voltage on the sampling capacitor to settle, and for the converted result to be accurate.

Settling Time Requirements

The absolute minimum tracking time is given in the electrical specifications tables, and will vary based on whether the ADC is in low power mode. It may be necessary to track for longer than the minimum tracking time specification, depending on the application. For example, if the ADC input is presented with a large series impedance, it will take longer for the sampling cap to settle on the final value during the tracking phase. The exact amount of tracking time required is a function of all series impedance (including the internal mux impedance and any external impedance sources), the sampling capacitance, and the desired accuracy.



Note: The value of C_{SAMPLE} depends on the PGA gain. See the electrical specifications for details.

Figure 12.4. ADC Equivalent Input Circuit

The required ADC0 settling time for a given settling accuracy (SA) may be approximated as follows:

$$t = \ln\left(\frac{2^n}{SA}\right) \times R_{TOTAL} \times C_{SAMPLE}$$

Where: SA is the settling accuracy, given as a fraction of an LSB (for example, 0.25 to settle within 1/4 LSB)

t is the required settling time in seconds

R_{TOTAL} is the sum of the ADC mux resistance and any external source resistance.

C_{SAMPLE} is the size of the ADC sampling capacitor.

n is the ADC resolution in bits.

When measuring any internal source, R_{TOTAL} reduces to R_{MUX}. See the electrical specification tables in the datasheet for ADC minimum settling time requirements as well as the mux impedance and sampling capacitor values.

Configuring the Tracking Time

The ADTK field configures the amount of time which will be allocated for input tracking by the ADC conversion logic.

When IPOEN is set to 1, firmware must always configure the ADTK field to allow adequate tracking and settling of the selected input. The tracking time will be applied after the power-up phase is complete, and before the conversion begins.

When IPOEN is cleared to 0, the ADC-timed tracking phase will still be applied before every conversion. If ADRPT is configured to accumulate multiple conversions, firmware must configure the ADTK bits to ensure that adequate tracking is given to every conversion. However, the ADC will continue to track the input whenever it is not actively performing a conversion. ADTK may be set to zero, provided that ADRPT is configured for single conversions, and adequate tracking time is allowed for in-between every conversion.

12.3.5.2 Power-Up Timing

The ADC requires up to 1.2 µs to power up and settle all internal circuitry. When IPOEN is set to 1, the ADC will power down between conversions to save energy. Firmware must configure the ADPWR field to allow adequate time for the ADC and internal reference circuitry to power up before each conversion.

When IPOEN is cleared to 0, the ADPWR time is not applied. This is primarily useful when operating the ADC in faster data acquisition systems. When firmware enables the ADC from a powered-down state, it must take the required power time into account before initiating a conversion. Once the ADC is powered on in this mode, it will remain powered up and the power-up time is not needed between subsequent conversions.

12.3.5.3 Conversion Resolution and Timing

The conversion resolution is adjusted using the ADBITS field in ADC0CN1, and selectable between 12- and 10-bit modes. The total amount of time required for a conversion is equal to:

Total Conversion Time = [RPT × (ADTK + NUMBITS + 1) × T(SARCLK)] + (T(ADCCLK) × 4)

where RPT is the number of conversions represented by the ADRPT field and ADCCLK is the clock selected for the ADC. Up to one SYSCLK of synchronization time is also required when triggering from the external CNVSTR pin source.

12.3.6 Initiating Conversions

Conversions may be initiated in many ways, depending on the programmed state of the ADCM bitfield. The following options are available as conversion trigger sources:

1. Software-triggered—Writing a 1 to the ADBUSY bit initiates conversions.
2. Hardware-triggered—An automatic internal event such as a timer overflow initiates conversions.
3. External pin-triggered—A rising edge on the CNVSTR input signal initiates conversions.

   Note: The CNVSTR pin is a multi-function GPIO pin. When the CNVSTR input is used as the ADC conversion source, the associated port pin should be skipped in the crossbar settings.

Basic converter operation is straightforward. The selected conversion trigger will begin the conversion cycle. Writing a 1 to ADBUSY provides software control of ADC0 whereby conversions are performed "on-demand". All other trigger sources occur autonomous to code execution. Each conversion cycle may consist of one or more conversions, as determined by the ADRPT setting. Individual conversions from the ADC will be accumulated until the requested number of conversions has been accumulated. When the converter is finished accumulating conversions, the ADINT flag will be posted and firmware may read the output results from the ADC data registers (ADC0H:ADC0L). Note that the first conversion in an accumulation sequence is triggered from the selected trigger source, while all subsequent conversions in an accumulation sequence will be self-triggered upon completing the previous conversion.

During any conversion, the ADBUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. However, the ADBUSY bit should not be used to poll for ADC conversion completion. It will read back 0 whenever the converter is not in the conversion phase, and results may not yet be available in the ADC data registers. The ADC interrupt flag (ADINT) should be polled instead, when writing polled-mode firmware.

12.3.7 Autoscan Mode

In addition to basic conversions, the ADC includes a flexible autoscan mode, which offloads much of the firmware tasks required to collect information from the ADC. Autoscan allows multiple output words from the ADC to be collected on up to four contiguous ADC channels without firmware intervention. ADC outputs are written to a firmware-designated area of XDATA space in the order they are received. The firmware specifies the number of desired output words (up to 64) before a scan begins. When active, the scanner will collect the requested number of output words from the ADC. At the end of a scan sequence, the autoscan hardware stores the current state of select register fields, generates an interrupt, and optionally continues with a new scan.

Trigger Configuration

In autoscan mode, the ADC may be triggered by any of the trigger source options selected by ADCM. The STEN bit in ADC0ASCF controls whether multiple triggers or a single trigger is required to complete the scan operation. When STEN is cleared to 0 (MULTIPLE_TRIGGERS), each (accumulated) conversion in the scan requires a new conversion trigger event. For example, if Timer 3 is the selected ADC trigger source and the autoscan hardware is configured to accumulate 20 sets of 4 conversions, Timer 3 would need to overflow 20 times to generate a trigger event for each conversion. When STEN is set to 1 (SINGLE_TRIGGER), an entire scan will be performed using a single trigger. In the preceding example, the first conversion would be triggered from a Timer 3 overflow event, and then the rest of the conversions would be automatically triggered by the scan hardware as each conversion completes.

Note: The converter must not be in the process of a normal conversion when entering autoscan mode. For this reason, firmware should ensure that the desired trigger source will not trigger the ADC before ASEN is set to 1. The simplest way to do this is to leave ADCM configured for software triggers until after ASEN is set to 1, and then select the desired trigger source.

Channel Configuration

The scanner hardware is capable of collecting data from up to four contiguous ADC channels in sequence. The ADC0MX register defines the first channel to be converted, and the NASCH field in ADC0ASCF defines the number of channels (1, 2, 3, or 4) to be converted. Channels are converted in circular fashion, one at a time. For example, if ADC0MX is configured to 0x02, NASCH is configured to convert three channels, and nine conversions are requested, the autoscan hardware will collect a conversion from ADC0MX = 0x02, then ADC0MX = 0x03, then ADC0MX = 0x04, then repeat at ADC0MX = 0x02, and so on until nine conversions are collected (three conversions on each of the three channels).

The ADRPT setting is valid in autoscan mode, and each accumulated sample counts as one conversion output from the autoscan hardware. If ADRPT is configured to accumulate 4 conversions and the scanner is configured to collect 9 samples, a total of 9 x 4, or 36 conversions will be performed. When scanning through multiple channels, the ADC will accumulate the requested number of conversions on each channel before proceeding to the next channel.

Output Data Configuration

Data from the autoscanner is written directly into XDATA space, starting at an address defined by the 16-bit ADC0ASA register (the combination of the two 8-bit registers ADC0ASAH and ADC0ASAL). ADC0ASA[11:1] correspond directly to bits 11:1 of the XRAM starting address. This means that the starting address must occur on an even-numbered address location. The ENDIAN bit in ADC0ASAL defines the endian-ness of the output data.

Each output word from the ADC will require two bytes of XDATA space. For a single scan consisting of 10 conversions, 20 XDATA bytes are required to hold the output.

Note: The toolchain used for firmware development will not be automatically aware of the location for the scanner output. When using the autoscan function, it is very important for the firmware developer to reserve the area intended for scanner output, to avoid contention with other variables.

Autoscan Operation

When ADC configuration is complete, firmware may place the ADC in scan mode by setting the ASEN bit in ADC0ASCF to 1. Note that the scan does not immediately begin when the ASEN bit is set. ASEN places the ADC into autoscan mode, waiting for the first trigger to occur. When ASEN is set, hardware will copy the contents of the ADC0ASAH, ADC0ASAL, AD0ASCNT and ADC0MX registers, as well as the NASCH field in ADC0ASCF into local registers for the scanner to use. This allows firmware to immediately set up the parameters for the following scan.

If only one scan is desired, firmware can immediately clear ASEN back to 0. Just as setting ASEN does not immediately begin a scan, clearing ASEN does not immediately take the converter out of autoscan mode. Autoscan mode will only be halted if ASEN is 0 at the completion of a scan operation. To terminate a scan in progress, firmware must disable the ADC completely with the ADEN bit.

When the ADC first enters autoscan mode, it waits for the selected conversion trigger to occur. In the case of software-triggered operation, firmware can begin the scan by setting the ADBUSY bit to 1. For timer-triggered conversions, firmware should enable the selected timer.

The scan will proceed according to the configuration options until all of the operations specified by AD0ASCNT have been completed. At the end of a scan operation, the scanner will set the AD0INT bit to 1, and check the status of ASEN. If ASEN is 0, autoscan mode is terminated, and the converter will return to normal mode. If ASEN is 1 however, a new scan is immediately begun, scan settings are loaded into the scanner's local registers, and the ADC waits for the next trigger to occur.

Autoscan Example: Circular Buffer

This example shows the steps necessary to use autoscan mode to implement a 128-word ping-pong buffer for a single ADC channel in XDATA. The buffer will consist of two 64-word (128-byte) areas in XDATA, beginning at 0x0000 and 0x0080, and the firmware is responsible for changing the scanner hardware at the appropriate intervals to keep a continual flow of data into memory. This example assumes that the ADC will be triggered in multiple-trigger mode from a hardware source, such as a timer.

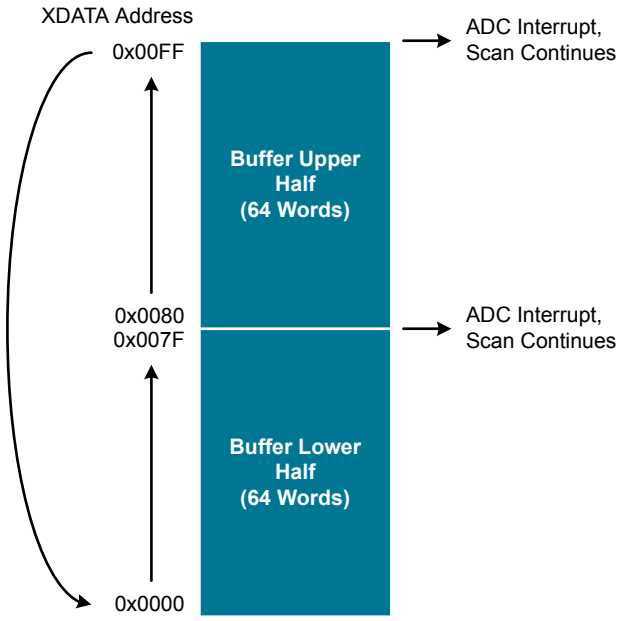


Figure 12.5. Circular Buffer Example

Initialization sequence:

1. Configure the ADC for no accumulation: Write ADRPT to 0.
2. Configure the input mux settings: Write ADC0MX to the desired channel, and write NASCH to 0.
3. Configure the starting address for the first half of the buffer: Write ADC0ASA[H:L] to 0x0000.
4. Configure to collect 64 samples: Write ADC0ASCNT to 63.
5. Initiate autoscan mode: Write ASEN to 1.
6. Configure the starting address for the second half of the buffer: Write ADC0ASA[H:L] to 0x0080.
7. Begin ADC conversions: Either start the conversion trigger source, or if the trigger source is already running, switch the ADC to use it).

Interrupt Service Routine:

1. Clear AD0INT.
2. Configure the starting address for the opposite buffer: Write ADC0ASA[H:L] to 0x0000 if it is 0x0080, or vice-versa.
3. Process the data in the most recent buffer, or optionally signal to the main thread that data is ready to be processed.

Autoscan Example: Single Scan of Two Channels

This example shows the steps necessary to use autoscan mode to implement a single scan of two adjacent mux channels into a 64-word buffer (32 conversions per channel). In this example, a single software trigger is used to initiate the entire scan sequence.

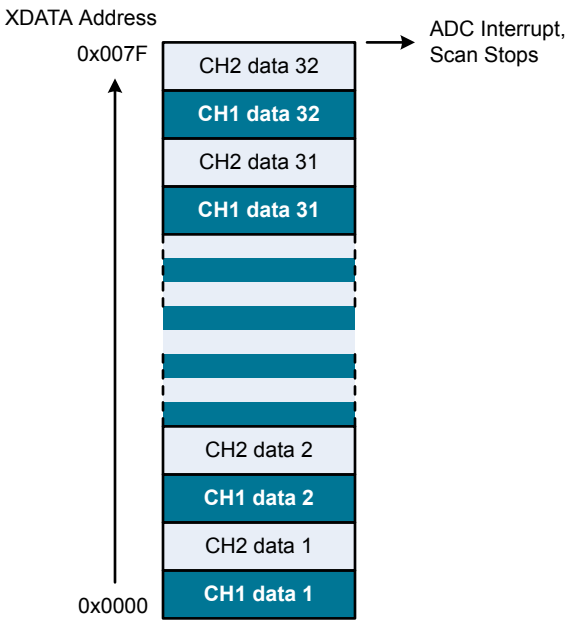


Figure 12.6. Circular Buffer Example

Initialization sequence:

1. Configure the ADC for no accumulation: Write ADRPT to 0.
2. Configure the ADC trigger source: Write ADCM to 0 for software triggers, and write STEN to 1 to enable a single-trigger autoscan.
3. Configure the input mux settings: Write ADC0MX to the starting (lowest-numbered) channel, and write NASCH to 1 (for two channels).
4. Configure the starting address for the memory output: Write ADC0ASA[H:L] to 0x0000.
5. Configure to collect 64 samples: Write ADC0ASCNT to 63.
6. Initiate autoscan mode: Write ASEN to 1.
7. Write ASEN to 0. This will instruct the scanner to stop upon scan completion.
8. Begin ADC conversions: Write ADBUSY to 1.

Interrupt Service Routine:

1. Clear AD0INT.
2. Process the data, or optionally signal to the main thread that data is ready to be processed.

12.3.8 Output Formatting and Accumulation

The registers ADC0H and ADC0L contain the high and low bytes of the output conversion code from the ADC at the completion of each conversion. Data may be accumulated over multiple conversions and the final output may be shifted right by a selectable amount, effectively providing an "accumulate and average" function. In the following examples, 1 LSBn refers to the voltage of one LSB of the converter at the specified resolution, calculated as VREF x 1 / 2^n. An LSB12 would be calculated as VREF x 1/4096.

When the repeat count ADRPT is configured for a single conversion and the ADSJST field is configured for no shifting, output conversion codes are represented in the selected resolution of the converter. Example codes are shown below for the different data formats with ADRPT = 0, ADSJST = 0, and a gain setting of 1x (ADGN = 0). Unused bits in the ADC0H and ADC0L registers are set to 0.

Table 12.2. Output Coding, ADRPT = 0, ADSJST = 0

Input Voltage	10-bit ADC0H:L	12-bit ADC0H:L
VREF - 1 LSBn	0x03FF	0x0FFF
VREF / 2	0x0200	0x0800
VREF / 4	0x0100	0x0400
0	0x0000	0x0000

When the repeat count is greater than 1, the output conversion code represents the accumulated result of the conversions performed and is updated after the last conversion in the series is finished. Sets of 4, 8, 16, or 32 consecutive samples can be accumulated and represented in unsigned integer format. The repeat count can be selected using the ADRPT bit field. Unused bits in the ADC0H and ADC0L registers are set to 0. The example below shows the right-justified result for various input voltages and repeat counts for 12-bit conversions. Notice that accumulating 2^n samples is equivalent to left-shifting by n bit positions when all samples returned from the ADC have the same value.

Table 12.3. Effects of ADRPT on Output Code (12-bit conversions, ADSJST = 0)

Input Voltage	Repeat Count = 4	Repeat Count = 8	Repeat Count = 16
VREF - 1 LSB12	0x3FFC	0x7FF8	0xFFF0
VREF / 2	0x2000	0x4000	0x8000
(VREF / 2) - 1 LSB12	0x1FFC	0x3FF8	0x7FF0
0	0x0000	0x0000	0x0000

Additionally, the ADSJST bit field can be used to format the contents of the 16-bit accumulator. The accumulated result can be shifted right by 1, 2, or 3 bit positions, effectively dividing the output by 2, 4, or 8. The example below shows the effects of using ADSJST on a 12-bit sample.

Table 12.4. Using ADSJST for Output Formatting (12-bit conversions, ADRPT = 8)

Input Voltage	ADSJST = 0 (no shift)	ADSJST = 1 (shift right 1 bit)	ADSJST = 3 (shift right 3 bits)
VREF - 1 LSB12	0x7FF8	0x3FFC	0x0FFF
VREF / 2	0x4000	0x2000	0x0800
(VREF / 2) - 1 LSB12	0x3FF8	0x1FFC	0x07FF
0	0x0000	0x0000	0x0000

Integration (Preserving the Accumulator)

Some applications do not require accumulation for a defined period, but instead need to integrate samples until a specific threshold is reached or a certain event occurs. For these applications, the accumulator clear function can be disabled by setting PACEN to 1. The ADC will always add the latest result to the value present in the accumulator, and the accumulator will never be reset to 0 by hardware. Firmware my over-write the accumulator output as needed by writing to ADC0H and ADC0L. ADRPT should be set to 0 by firmware (single conversions) any time PACEN is set to 1.

12.3.9 Window Comparator

The ADC's programmable window detector compares the ADC output registers to user-programmed limits, and notifies the system when a desired condition is detected. This is especially effective in an interrupt driven system, saving code space and CPU bandwidth while delivering faster system response times. The window detector interrupt flag (ADWINT) can also be used in polled mode. The ADC Greater-Than (ADC0GTH, ADC0GTL) and Less-Than (ADC0LTH, ADC0LTL) registers hold the comparison values. The window detector flag can be programmed to indicate when measured data is inside or outside of the user-programmed limits, depending on the contents of the ADC0GT and ADC0LT registers. The following tables show how the ADC0GT and ADC0LT registers may be configured to set the ADWINT flag when the ADC output code is above, below, between, or outside of specific values.

Table 12.5. ADC Window Comparator Example (10-bit codes, Above 0x0080)

Comparison Register Settings	Output Code (ADC0H:L)	ADWINT Effects
	0x03FF	ADWINT = 1
	...	
	0x0081	
ADC0GTH:L = 0x0080	0x0080	ADWINT Not Affected
	0x007F	
	...	
	0x0001	
ADC0LTH:L = 0x0000	0x0000	

Table 12.6. ADC Window Comparator Example (10-bit codes, Below 0x0040)

Comparison Register Settings	Output Code (ADC0H:L)	ADWINT Effects
ADC0GTH:L = 0x03FF	0x03FF	ADWINT Not Affected
	0x03FE	
	...	
	0x0041	
ADC0LTH:L = 0x0040	0x0040	
	0x003F	ADWINT = 1
	...	
	0x0000	

Table 12.7. ADC Window Comparator Example (10-bit codes, Between 0x0040 and 0x0080)

Comparison Register Settings	Output Code (ADC0H:L)	ADWINT Effects
	0x03FF	ADWINT Not Affected
	...	
	0x0081	
ADC0LTH:L = 0x0080	0x0080	
	0x007F	ADWINT = 1
	...	
	0x0041	

Comparison Register Settings	Output Code (ADC0H:L)	ADWINT Effects
ADC0GTH:L = 0x0040	0x0040	ADWINT Not Affected
	0x003F	
	...	
	0x0000	

Table 12.8. ADC Window Comparator Example (10-bit codes, Outside the 0x0040 to 0x0080 range)

Comparison Register Settings	Output Code (ADC0H:L)	ADWINT Effects
	0x03FF	ADWINT = 1
	...	
	0x0081	
ADC0GTH:L = 0x0080	0x0080	ADWINT Not Affected
	0x007F	
	...	
	0x0041	
ADC0LTH:L = 0x0040	0x0040	
	0x003F	ADWINT = 1
	...	
	0x0000	

12.3.10 Temperature Sensor

An on-chip analog temperature sensor is available to the ADC multiplexer input. To use the ADC to measure the temperature sensor, the ADC mux channel should select the temperature sensor. The temperature sensor transfer function is shown in Figure 12.7 Temperature Sensor Transfer Function on page 163. The output voltage (V_{TEMP}) is the positive ADC input when the ADC multiplexer is set correctly. The TEMPE bit in register ADC0CN0 enables/disables the temperature sensor. While disabled, the temperature sensor defaults to a high impedance state and any ADC measurements performed on the sensor will result in meaningless data. Refer to the electrical specification tables for the slope and offset parameters of the temperature sensor.

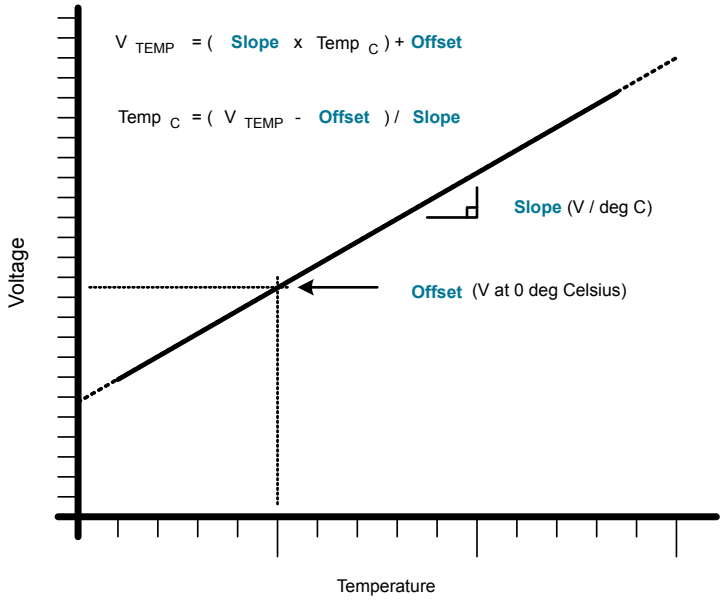


Figure 12.7. Temperature Sensor Transfer Function

12.3.10.1 Temperature Sensor Calibration

For greater precision on absolute temperature measurements, offset and/or gain calibration may be performed in-system. Typically a 1-point (offset) calibration includes the following steps:

1. Control/measure the ambient temperature (this temperature must be known).
2. Power the device, and delay for a few seconds to allow for self-heating.
3. Perform an ADC conversion with the temperature sensor selected as the ADC input.
4. Calculate the offset characteristics, and store this value in non-volatile memory for use with subsequent temperature sensor measurements.

12.4 ADC Control Registers

12.4.1 ADC0CN0: ADC0 Control 0

Bit	7	6	5	4	3	2	1	0
Name	ADEN	IPOEN	ADINT	ADBUSY	ADWINT	ADGN		TEMPE
Access	RW	RW	RW	RW	RW	RW		RW
Reset	0	0	0	0	0	0x0		0

SFR Page = 0x0, 0x30; SFR Address: 0xE8 (bit-addressable)

Bit	Name	Reset	Access	Description
7	ADEN	0	RW	**ADC Enable.**
	Value	Name		Description
	0	DISABLED		Disable ADC0 (low-power shutdown).
	1	ENABLED		Enable ADC0 (active and ready for data conversions).
6	IPOEN	0	RW	**Idle Powered-off Enable.**
	Value	Name		Description
	0	ALWAYS_ON		Keep ADC powered on when ADEN is 1.
	1	POWER_DOWN		Power down when ADC is idle (not converting).
5	ADINT	0	RW	**Conversion Complete Interrupt Flag.**
	Set by hardware upon completion of a data conversion (ADBMEN=0), or a burst of conversions (ADBMEN=1). Can trigger an interrupt. Must be cleared by firmware.			
4	ADBUSY	0	RW	**ADC Busy.**
	Writing 1 to this bit initiates an ADC conversion when ADCM = 000. This bit should not be polled to indicate when a conversion is complete. Instead, the ADINT bit should be used when polling for conversion completion.			
3	ADWINT	0	RW	**Window Compare Interrupt Flag.**
	Set by hardware when the contents of ADC0H:ADC0L fall within the window specified by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL. Can trigger an interrupt. Must be cleared by firmware.			
2:1	ADGN	0x0	RW	**Gain Control.**
	Value	Name		Description
	0x0	GAIN_1		The on-chip PGA gain is 1.
	0x1	GAIN_0P75		The on-chip PGA gain is 0.75.
	0x2	GAIN_0P5		The on-chip PGA gain is 0.5.
	0x3	GAIN_0P25		The on-chip PGA gain is 0.25.
0	TEMPE	0	RW	**Temperature Sensor Enable.**
	Enables/Disables the internal temperature sensor.			
	Value	Name		Description
	0	TEMP_DISABLED		Disable the Temperature Sensor.
	1	TEMP_ENABLED		Enable the Temperature Sensor.

12.4.2 ADC0CN1: ADC0 Control 1

Bit	7	6	5	4	3	2	1	0
Name	ADBITS		ADSJST			ADRPT		
Access	RW		RW			RW		
Reset	0x1		0x0			0x0		
SFR Page = 0x0, 0x30; SFR Address: 0xB2								

Bit	Name	Reset	Access	Description
7:6	ADBITS	0x1	RW	**Resolution Control.**
	Value	Name		Description
	0x0	10_BIT		ADC0 operates in 10-bit mode.
	0x1	12_BIT		ADC0 operates in 12-bit mode.
5:3	ADSJST	0x0	RW	**Accumulator Shift and Justify.**
	Specifies the format of data read from ADC0H:ADC0L. All remaining bit combinations are reserved.			
	Value	Name		Description
	0x0	RIGHT_NO_SHIFT		Right justified. No shifting applied.
	0x1	RIGHT_SHIFT_1		Right justified. Shifted right by 1 bit.
	0x2	RIGHT_SHIFT_2		Right justified. Shifted right by 2 bits.
	0x3	RIGHT_SHIFT_3		Right justified. Shifted right by 3 bits.
2:0	ADRPT	0x0	RW	**Repeat Count.**
	Selects the number of conversions to perform and accumulate per ADC conversion trigger.			
	Value	Name		Description
	0x0	ACC_1		Perform and Accumulate 1 conversion.
	0x1	ACC_4		Perform and Accumulate 4 conversions.
	0x2	ACC_8		Perform and Accumulate 8 conversions.
	0x3	ACC_16		Perform and Accumulate 16 conversions.
	0x4	ACC_32		Perform and Accumulate 32 conversions.

12.4.3 ADC0CN2: ADC0 Control 2

Bit	7	6	5	4	3	2	1	0
Name	PACEN	Reserved			ADCM			
Access	RW	R			RW			
Reset	0	0x0			0x0			
SFR Page = 0x0, 0x30; SFR Address: 0xB3								

Bit	Name	Reset	Access	Description
7	PACEN	0	RW	**Preserve Accumulator Enable.**
	This bit controls whether the ADC accumulator is preserved for further accumulation from subsequent ADC conversions.			
	Value	Name		Description
	0	PAC_DISABLED		The ADC accumulator is over-written with the results of any conversion (or set of conversions as specified by ADRPT).
	1	PAC_ENABLED		The ADC accumulator always adds new results to the existing output. The accumulator is never cleared in this mode.
6:4	*Reserved*	*Must write reset value.*		
3:0	ADCM	0x0	RW	**Start of Conversion Mode Select.**
	Specifies the ADC0 start of conversion source. All remaining bit combinations are reserved.			
	Value	Name		Description
	0x0	ADBUSY		ADC0 conversion initiated on write of 1 to ADBUSY.
	0x1	TIMER0		ADC0 conversion initiated on overflow of Timer 0.
	0x2	TIMER2		ADC0 conversion initiated on overflow of Timer 2.
	0x3	TIMER3		ADC0 conversion initiated on overflow of Timer 3.
	0x4	CNVSTR		ADC0 conversion initiated on rising edge of CNVSTR.
	0x5	CEX5		ADC0 conversion initiated on rising edge of CEX5.
	0x6	TIMER4		ADC0 conversion initiated on overflow of Timer 4.
	0x7	TIMER5		ADC0 conversion initiated on overflow of Timer 5.
	0x8	CLU0		ADC0 conversion initiated on CLU0 Output.
	0x9	CLU1		ADC0 conversion initiated on CLU1 Output.
	0xA	CLU2		ADC0 conversion initiated on CLU2 Output.
	0xB	CLU3		ADC0 conversion initiated on CLU3 Output.

12.4.4 ADC0CF0: ADC0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	ADSC					ADCLKSEL	Reserved	
Access	RW					RW	R	
Reset	0x1F					0	0x0	
SFR Page = 0x0, 0x30; SFR Address: 0xBC								

Bit	Name	Reset	Access	Description
7:3	ADSC	0x1F	RW	**SAR Clock Divider.** This field sets the ADC clock divider value. It should normally be configured to be as close to the maximum SAR clock speed as the datasheet will allow. The SAR clock frequency is given by the following equation: Fsarclk = (Fadcclk) / (ADSC + 1)
2	ADCLKSEL	0	RW	**ADC Clock Select.**
	Value	Name		Description
	0	SYSCLK		ADCCLK = SYSCLK.
	1	HFOSC0		ADCCLK = HFOSC0.
1:0	*Reserved*	*Must write reset value.*		

12.4.5 ADC0CF1: ADC0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	Reserved		ADTK					
Access	R		RW					
Reset	0x2		0x1E					
SFR Page = 0x0, 0x30; SFR Address: 0xB9								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5:0	ADTK	0x1E	RW	**Conversion Tracking Time.** This field sets the time delay before an conversion. This field should be set to the minimum settling time required for the sampling capacitor voltage to settle. Tadtk = ADTK / (Fsarclk)

12.4.6 ADC0CF2: ADC0 Power Control

Bit	7	6	5	4	3	2	1	0
Name	GNDSL	REFSL		ADPWR				
Access	RW	RW		RW				
Reset	0	0x3		0x1F				
SFR Page = 0x0, 0x30; SFR Address: 0xDF								

Bit	Name	Reset	Access	Description
7	GNDSL	0	RW	**Analog Ground Reference.**
	Selects the ADC0 ground reference.			
	Value	Name		Description
	0	GND_PIN		The ADC0 ground reference is the GND pin.
	1	AGND_PIN		The ADC0 ground reference is the AGND pin.
6:5	REFSL	0x3	RW	**Voltage Reference Select.**
	Selects the ADC0 voltage reference.			
	Value	Name		Description
	0x0	VREF_PIN		The ADC0 voltage reference is the VREF pin (external or from the on-chip reference).
	0x1	VDD_PIN		The ADC0 voltage reference is the VDD pin.
	0x2	INTERNAL_LDO		The ADC0 voltage reference is the internal 1.8 V digital supply voltage.
	0x3	INTERNAL_VREF		The ADC0 voltage reference is the internal voltage reference.
4:0	ADPWR	0x1F	RW	**Power Up Delay Time.**
	This field sets the time delay allowed for the ADC to power up when IPOEN is set to 1. Power-up time is not applied if IPOEN is 0.			
	Tpwrtime = ((4 * (ADPWR + 1)) + 2) / (Fadcclk)			

12.4.7 ADC0L: ADC0 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0L							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x30; SFR Address: 0xBD								

Bit	Name	Reset	Access	Description
7:0	ADC0L	0x00	RW	**Data Word Low Byte.**
	When read, this register returns the least significant byte of the 16-bit ADC0 accumulator, formatted according to the settings in ADSJST. The register may also be written, to set the lower byte of the 16-bit ADC0 accumulator.			
If Accumulator shifting is enabled, the most significant bits of the value read will be zeros.				

12.4.8 ADC0H: ADC0 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0H							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x30; SFR Address: 0xBE								

Bit	Name	Reset	Access	Description
7:0	ADC0H	0x00	RW	**Data Word High Byte.** When read, this register returns the most significant byte of the 16-bit ADC0 accumulator, formatted according to the settings in ADSJST. The register may also be written, to set the upper byte of the 16-bit ADC0 accumulator.
If Accumulator shifting is enabled, the most significant bits of the value read will be zeros.				

12.4.9 ADC0GTH: ADC0 Greater-Than High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTH							
Access	RW							
Reset	0xFF							
SFR Page = 0x0, 0x30; SFR Address: 0xC4								

Bit	Name	Reset	Access	Description
7:0	ADC0GTH	0xFF	RW	**Greater-Than High Byte.** Most significant byte of the 16-bit greater-than window compare register.

12.4.10 ADC0GTL: ADC0 Greater-Than Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTL							
Access	RW							
Reset	0xFF							
SFR Page = 0x0, 0x30; SFR Address: 0xC3								

Bit	Name	Reset	Access	Description
7:0	ADC0GTL	0xFF	RW	**Greater-Than Low Byte.** Least significant byte of the 16-bit greater-than window compare register.
In 8-bit mode, this register should be set to 0x00.				

12.4.11 ADC0LTH: ADC0 Less-Than High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTH							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x30; SFR Address: 0xC6								

Bit	Name	Reset	Access	Description
7:0	ADC0LTH	0x00	RW	**Less-Than High Byte.** Most significant byte of the 16-bit less-than window compare register.

12.4.12 ADC0LTL: ADC0 Less-Than Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTL							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x30; SFR Address: 0xC5								

Bit	Name	Reset	Access	Description
7:0	ADC0LTL	0x00	RW	**Less-Than Low Byte.** Least significant byte of the 16-bit less-than window compare register.
In 8-bit mode, this register should be set to 0x00.				

12.4.13 ADC0MX: ADC0 Multiplexer Selection

Bit	7	6	5	4	3	2	1	0
Name	Reserved			ADC0MX				
Access	R			RW				
Reset	0x0			0x1F				
SFR Page = 0x0, 0x30; SFR Address: 0xBB								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4:0	ADC0MX	0x1F	RW	**AMUX0 Positive Input Selection.** Selects the positive input channel for ADC0. For reserved bit combinations, no input is selected.

12.4.14 ADC0ASCF: ADC0 Autoscan Configuration

Bit	7	6	5	4	3	2	1	0
Name	ASEN	STEN	ASACT	Reserved			NASCH	
Access	RW	RW	R	R			RW	
Reset	0	0	0	0x0			0x0	

SFR Page = 0x30; SFR Address: 0xA1

Bit	Name	Reset	Access	Description
7	ASEN	0	RW	**Autoscan Enable.**
	Value	Name		Description
	0	HALT_SCAN		Clearing to 0 will halt scan operations once any pending scan is complete.
	1	START_SCAN		Setting to 1 will initialize a scan operation. If set to 1 at the end of a scan, a new scan will begin.
6	STEN	0	RW	**Autoscan Single Trigger Enable.**
	Value	Name		Description
	0	MULTIPLE_TRIGGERS		Each conversion in a scan requires a new conversion trigger from the selected conversion trigger source.
	1	SINGLE_TRIGGER		The selected conversion trigger source will begin each scan cycle. All conversions within a scan cycle are performed automatically when the previous conversion is complete.
5	ASACT	0	R	**Autoscan Active.**
	This bit indicates that the ADC is in scan mode. When in scan mode, the AD0INT flag will only be set on the completion of a scan cycle.			
4:2	*Reserved*	*Must write reset value.*		
1:0	NASCH	0x0	RW	**Number of Autoscan Channels.**
	Specifies the number of contiguous mux channels to cycle through in autoscan mode. This field may be changed during a scan cycle to set up a new value for the next scan cycle.			
	Value	Name		Description
	0x0	ONE		Autoscan will only use the ADC0MX setting directly.
	0x1	TWO		Autoscan will alternate between ADC0MX and ADC0MX+1.
	0x2	THREE		Autoscan will cycle through ADC0MX, ADC0MX+1 and ADC0MX+2.
	0x3	FOUR		Autoscan will cycle through ADC0MX, ADC0MX+1, ADC0MX+2, and ADC0MX+3.

12.4.15 ADC0ASAH: ADC0 Autoscan Start Address High Byte

Bit	7	6	5	4	3	2	1	0
Name	Reserved				STADDRH			
Access	R				RW			
Reset	0x0				0x0			
SFR Page = 0x30; SFR Address: 0xB6								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3:0	STADDRH	0x0	RW	**Start Address High.** This field contains the upper 4 bits of the XRAM starting address to use during a scan operation. This field may be changed during a scan cycle to set up a new value for the next scan cycle.

12.4.16 ADC0ASAL: ADC0 Autoscan Start Address Low Byte

Bit	7	6	5	4	3	2	1	0
Name	STADDRL							ENDIAN
Access	RW							RW
Reset	0x00							0
SFR Page = 0x30; SFR Address: 0xB5								

Bit	Name	Reset	Access	Description
7:1	STADDRL	0x00	RW	**Start Address Low.** This field contains bits 7-1 of the XRAM starting address to use during a scan operation. This field may be changed during a scan cycle to set up a new value for the next scan cycle.
0	ENDIAN	0	RW	**Endianness Control.** This bit controls the byte order of the ADC results written to XRAM.

Value	Name	Description
0	BIG_ENDIAN	ADC results in XRAM are stored in big-endian order. This will result in the most significant byte stored in the even-numbered address.
1	LITTLE_ENDIAN	ADC results in XRAM are stored in little-endian order. This will result in the most significant byte stored in the odd-numbered address.

12.4.17 ADC0ASCT: ADC0 Autoscan Output Count

Bit	7	6	5	4	3	2	1	0
Name	Reserved		ASCNT					
Access	R		RW					
Reset	0x0		0x00					
SFR Page = 0x30; SFR Address: 0xC7								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5:0	ASCNT	0x00	RW	**Autoscan Output Count.** This field specifies how many ADC outputs to collect per scan cycle. The number of outputs collected on each scan cycle will be equal to ASCNT+1. Note that each conversion requires two bytes of XRAM, and the number of bytes written to XRAM during a scan will be equal to (ASCNT+1)*2. This field may be changed during a scan cycle to set up a new value for the next scan cycle.

13. Comparators (CMP0 and CMP1)

13.1 Introduction

An analog comparator is used to compare the voltage of two analog inputs, with a digital output indicating which input voltage is higher. External input connections to device I/O pins and internal connections are available through separate multiplexers on the positive and negative inputs. Hysteresis, response time, and current consumption may be programmed to suit the specific needs of the application.

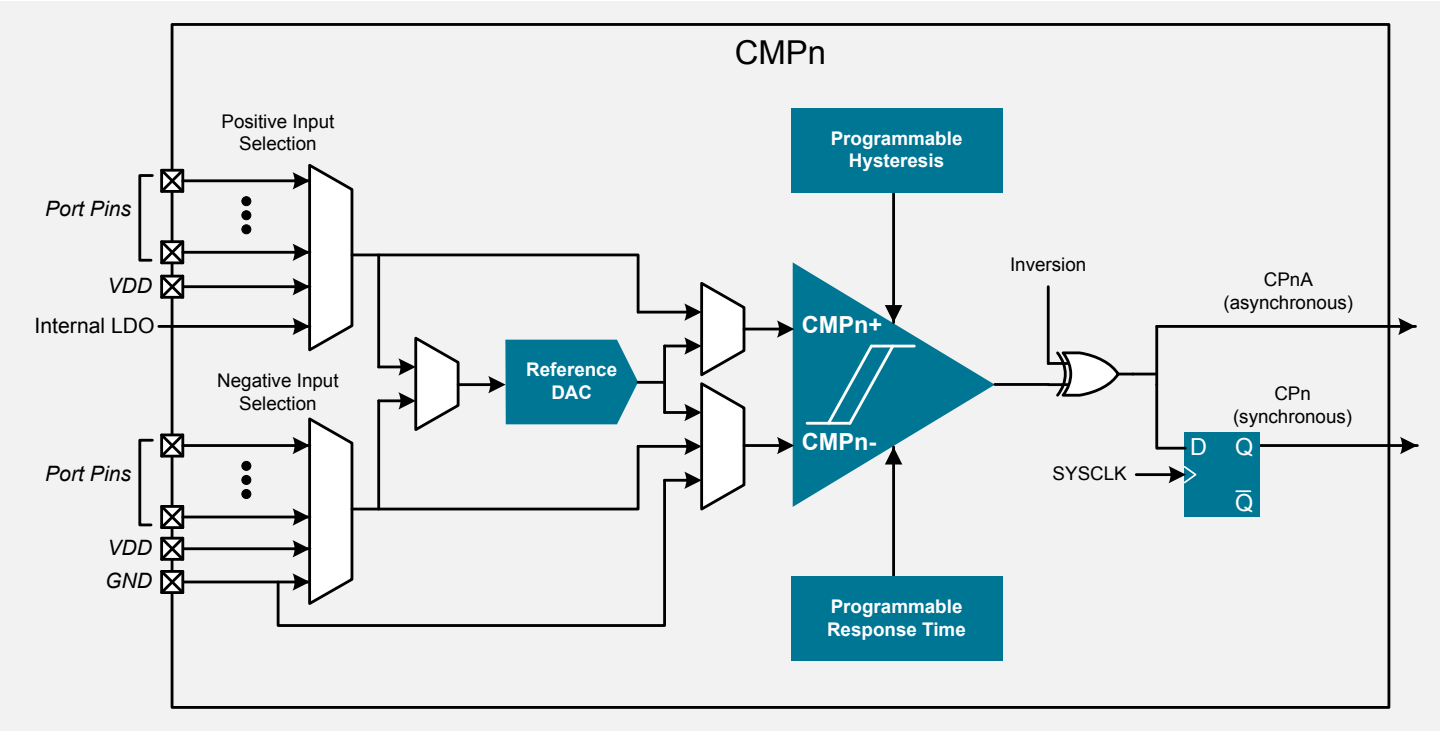


Figure 13.1. Comparator Block Diagram

13.2 Features

The comparator includes the following features:

- Up to 10 (CMP0) or 9 (CMP1) external positive inputs
- Up to 10 (CMP0) or 9 (CMP1) external negative inputs
- Additional input options:
  - Internal connection to LDO output
  - Direct connection to GND
  - Direct connection to VDD
  - Dedicated 6-bit reference DAC
- Synchronous and asynchronous outputs can be routed to pins via crossbar
- Programmable hysteresis between 0 and ±20 mV
- Programmable response time
- Interrupts generated on rising, falling, or both edges
- PWM output kill feature

13.3 Functional Description

13.3.1 Response Time and Supply Current

Response time is the amount of time delay between a change at the comparator inputs and the comparator's reaction at the output. The comparator response time may be configured in software via the CPMD field in the CMPnMD register. Selecting a longer response time reduces the comparator supply current, while shorter response times require more supply current.

13.3.2 Hysteresis

The comparator hysteresis is software-programmable via its Comparator Control register CMPnCN. The user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage.

The comparator hysteresis is programmable using the CPHYN and CPHYP fields in the Comparator Control Register CMPnCN. The amount of negative hysteresis voltage is determined by the settings of the CPHYN bits. Settings of 20, 10, or 5 mV (nominal) of negative hysteresis can be programmed, or negative hysteresis can be disabled. In a similar way, the amount of positive hysteresis is determined by the setting the CPHYP bits.

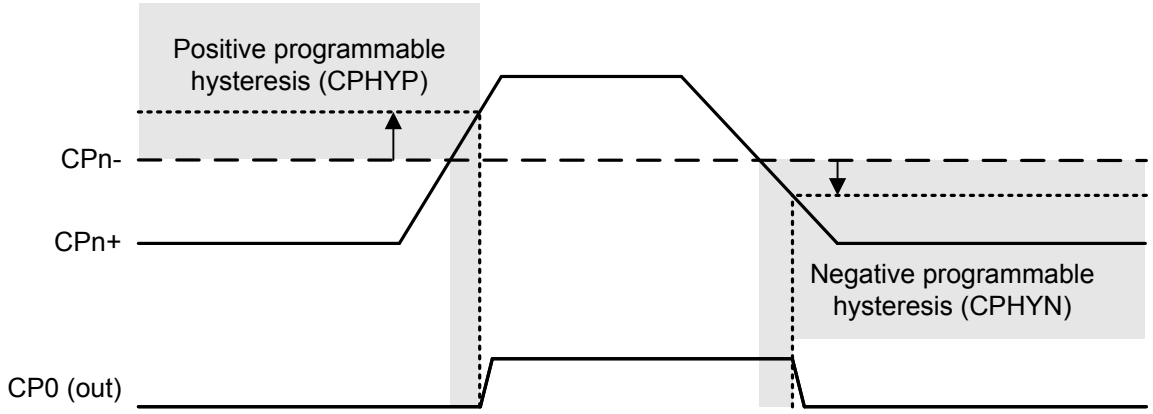


Figure 13.2. Comparator Hysteresis Plot

13.3.3 Input Selection

Comparator inputs may be routed to port I/O pins or internal signals. The CMPnMX register selects the inputs for the associated comparator. The CMXP field selects the comparator's positive input (CPnP.x) and the CMXN field selects the comparator's negative input (CPnN.x).

Note: Any port pins selected as comparator inputs should be configured as analog inputs in their associated port configuration register, and configured to be skipped by the crossbar.

13.3.3.1 Multiplexer Channel Selection

Table 13.1. CMP0 Positive Input Multiplexer Channels

CMXP Setting in Register CMP0MX	Signal Name	Enumeration Name	QFN32 / QFP32 Pin Name	QSOP24 Pin Name	QFN24 Pin Name
0000	CMP0P.0	CMP0P0	P0.1	P0.1	P0.1
0001	CMP0P.1	CMP0P1	P0.2	P0.2	P0.2
0010	CMP0P.2	CMP0P2	P0.4	P0.4	P0.4
0011	CMP0P.3	CMP0P3	P0.5	P0.5	P0.5
0100	CMP0P.4	CMP0P4	P0.6	P0.6	P0.6
0101	CMP0P.5	CMP0P5	P0.7	P0.7	P0.7
0110	CMP0P.6	CMP0P6	P1.0	Reserved	Reserved
0111	CMP0P.7	CMP0P7	P1.1	Reserved	Reserved
1000	CMP0P.8	CMP0P8	P1.2	Reserved	Reserved
1001	CMP0P.9	CMP0P9	P1.7	Reserved	Reserved
1010	CMP0P.10	LDO_OUT	Internal 1.8 V LDO output		
1011	CMP0P.11	VDD	VDD Supply Pin		
1100-1111	CMP0P.12 - CMP0P.15		No connection / Reserved		

Table 13.2. CMP0 Negative Input Multiplexer Channels

CMXN Setting in Register CMP0MX	Signal Name	Enumeration Name	QFN32 / QFP32 Pin Name	QSOP24 Pin Name	QFN24 Pin Name
0000	CMP0N.0	CMP0N0	P0.1	P0.1	P0.1
0001	CMP0N.1	CMP0N1	P0.2	P0.2	P0.2
0010	CMP0N.2	CMP0N2	P0.4	P0.4	P0.4
0011	CMP0N.3	CMP0N3	P0.5	P0.5	P0.5
0100	CMP0N.4	CMP0N4	P0.6	P0.6	P0.6
0101	CMP0N.5	CMP0N5	P0.7	P0.7	P0.7
0110	CMP0N.6	CMP0N6	P1.0	Reserved	Reserved
0111	CMP0N.7	CMP0N7	P1.1	Reserved	Reserved
1000	CMP0N.8	CMP0N8	P1.2	Reserved	Reserved
1001	CMP0N.9	CMP0N9	P1.7	Reserved	Reserved
1010	CMP0N.10	GND	GND Supply Pin		
1111	CMP0N.11	VDD	VDD Supply Pin		
1100-1111	CMP0N.12 - CMP0N.15		No connection / Reserved		

Table 13.3. CMP1 Positive Input Multiplexer Channels

CMXP Setting in Register CMP1MX	Signal Name	Enumeration Name	QFN32 / QFP32 Pin Name	QSOP24 Pin Name	QFN24 Pin Name
0000	CMP1P.0	CMP1P0	P0.7	P0.6	P0.6
0001	CMP1P.1	CMP1P1	P1.0	P0.7	P0.7
0010	CMP1P.2	CMP1P2	P2.0	P1.3	P1.3
0011	CMP1P.3	CMP1P3	P2.1	P1.4	P1.4
0100	CMP1P.4	CMP1P4	P2.2	P1.5	P1.5
0101	CMP1P.5	CMP1P5	P2.3	P1.6	P1.6
0110	CMP1P.6	CMP1P6	P2.4	P1.7	Reserved
0111	CMP1P.7	CMP1P7	P2.5	Reserved	Reserved
1000	CMP1P.8	CMP1P8	P2.6	Reserved	Reserved
1001	CMP1P.9		No connection / Reserved		
1010	CMP1P.10	LDO_OUT	Internal 1.8 V LDO output		
1011	CMP1P.11	VDD	VDD Supply Pin		
1100-1111	CMP1P.12 - CMP1P.15		No connection / Reserved		

Table 13.4. CMP1 Negative Input Multiplexer Channels

CMXN Setting in Register CMP1MX	Signal Name	Enumeration Name	QFN32 / QFP32 Pin Name	QSOP24 Pin Name	QFN24 Pin Name
0000	CMP1N.0	CMP1N0	P0.7	P0.6	P0.6
0001	CMP1N.1	CMP1N1	P1.0	P0.7	P0.7
0010	CMP1N.2	CMP1N2	P2.0	P1.3	P1.3
0011	CMP1N.3	CMP1N3	P2.1	P1.4	P1.4
0100	CMP1N.4	CMP1N4	P2.2	P1.5	P1.5
0101	CMP1N.5	CMP1N5	P2.3	P1.6	P1.6
0110	CMP1N.6	CMP1N6	P2.4	P1.7	Reserved
0111	CMP1N.7	CMP1N7	P2.5	Reserved	Reserved
1000	CMP1N.8	CMP1N8	P2.6	Reserved	Reserved
1001	CMP1N.9		No connection / Reserved		
1010	CMP1N.10	GND	GND Supply Pin		
1011	CMP1N.11	VDD	VDD Supply Pin		
1100-1111	CMP1N.12 - CMP1N.15		No connection / Reserved		

13.3.3.2 Reference DAC

The comparator module includes a dedicated reference DAC, which can be inserted between the selected mux channel and the comparator on either the positive or negative inputs. The INSL field in the CMPnMD register determines the connections between the selected mux inputs, the reference DAC, and the comparator inputs. There are four possible configurations.

When INSL is configured for direct input connection, the comparator mux channels are directly connected to the comparator inputs. The reference DAC is not used in this configuration.

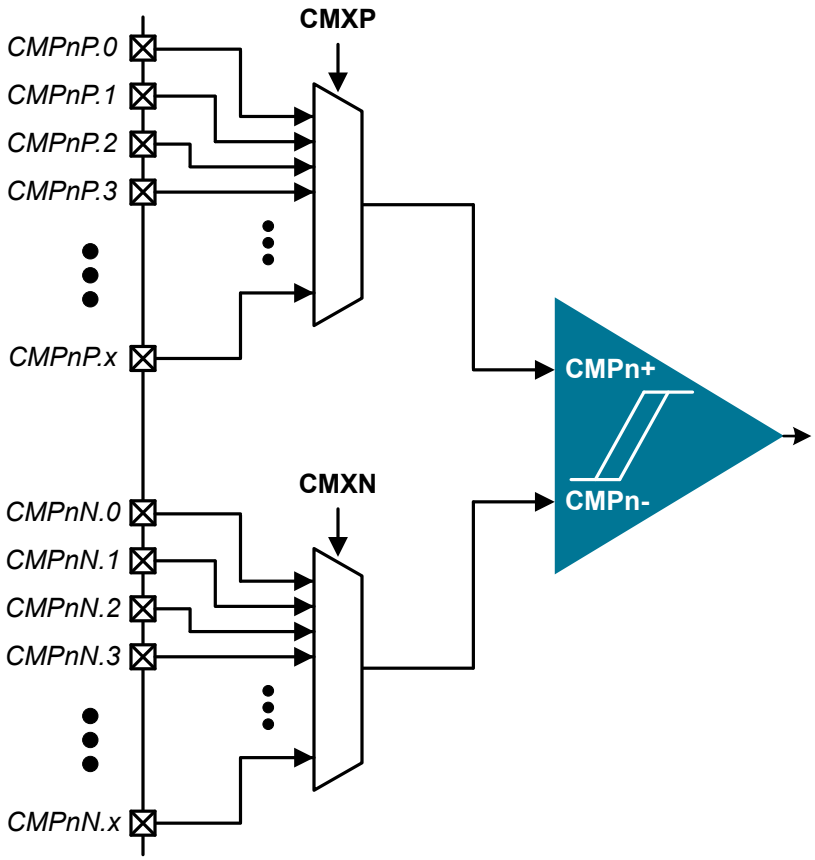


Figure 13.3. Direct Input Connection

When INSL is configured to ground the negative input, the positive comparator mux selection is directly connected to the positive comparator input, and the negative comparator input is connected to GND. The reference DAC is not used in this configuration.

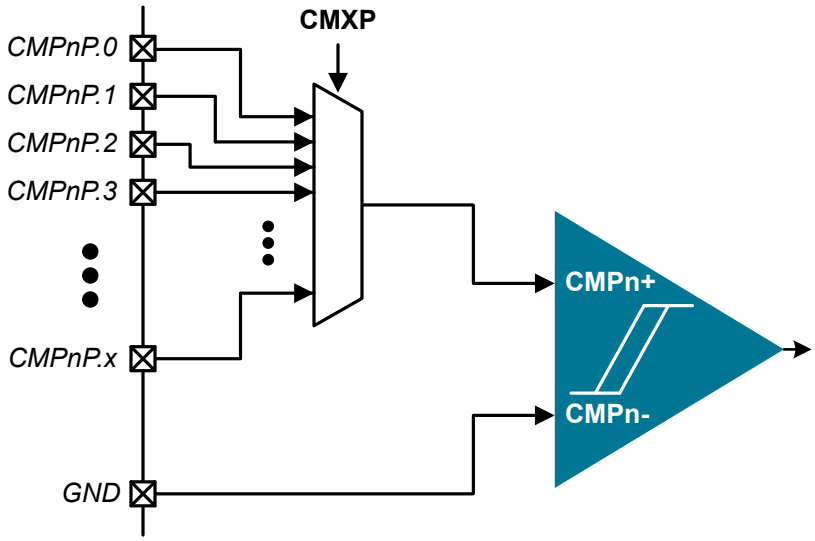


Figure 13.4. Negative Input Ground Connection

When INSL is configured to use the reference DAC on the negative channel, the positive comparator mux selection is directly connected to the positive comparator input. The negative mux selection becomes the full scale voltage reference for the DAC, and the DAC output is connected to the negative comparator input.

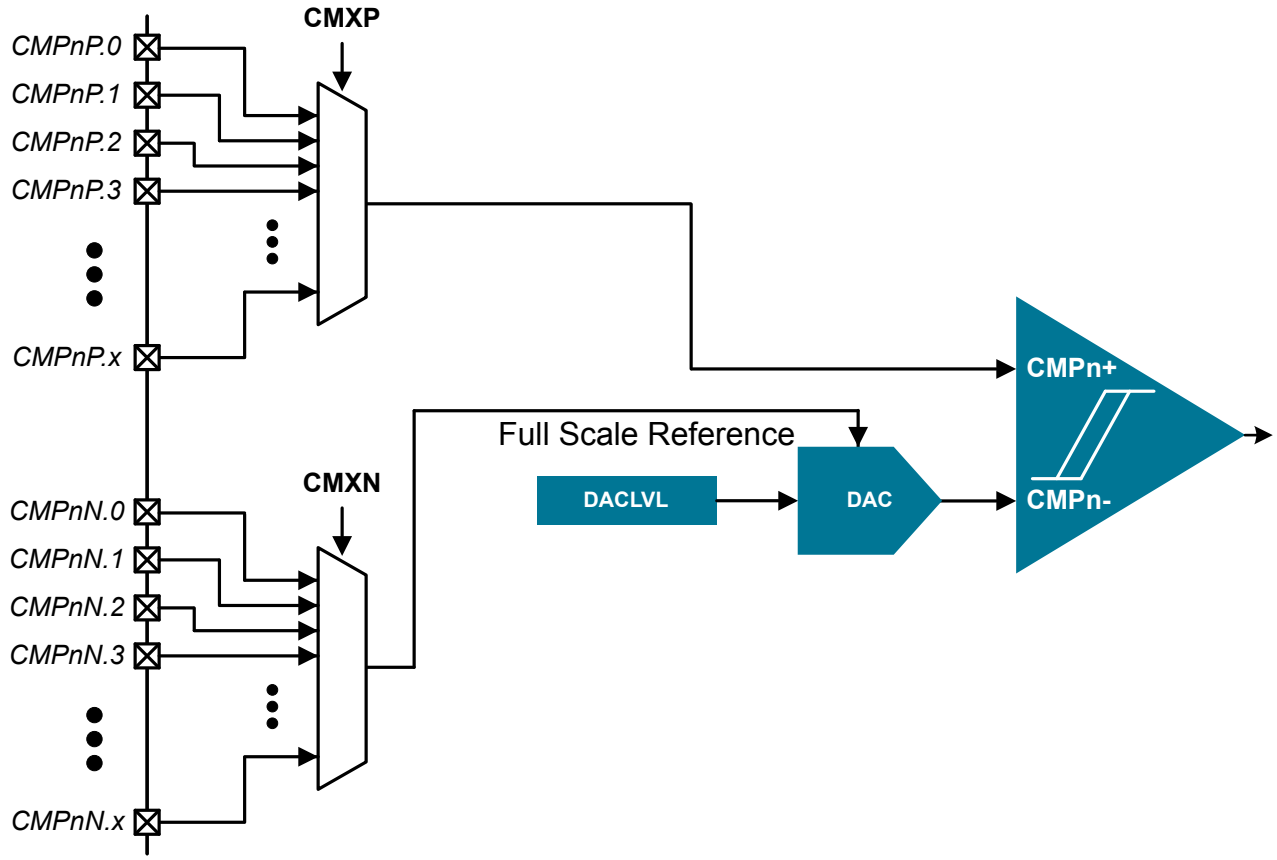


Figure 13.5. Negative Input DAC Connection

When INSL is configured to use the reference DAC on the positive channel, the negative comparator mux selection is directly connected to the negative comparator input. The positive mux selection becomes the full scale voltage reference for the DAC, and the DAC output is connected to the positive comparator input.

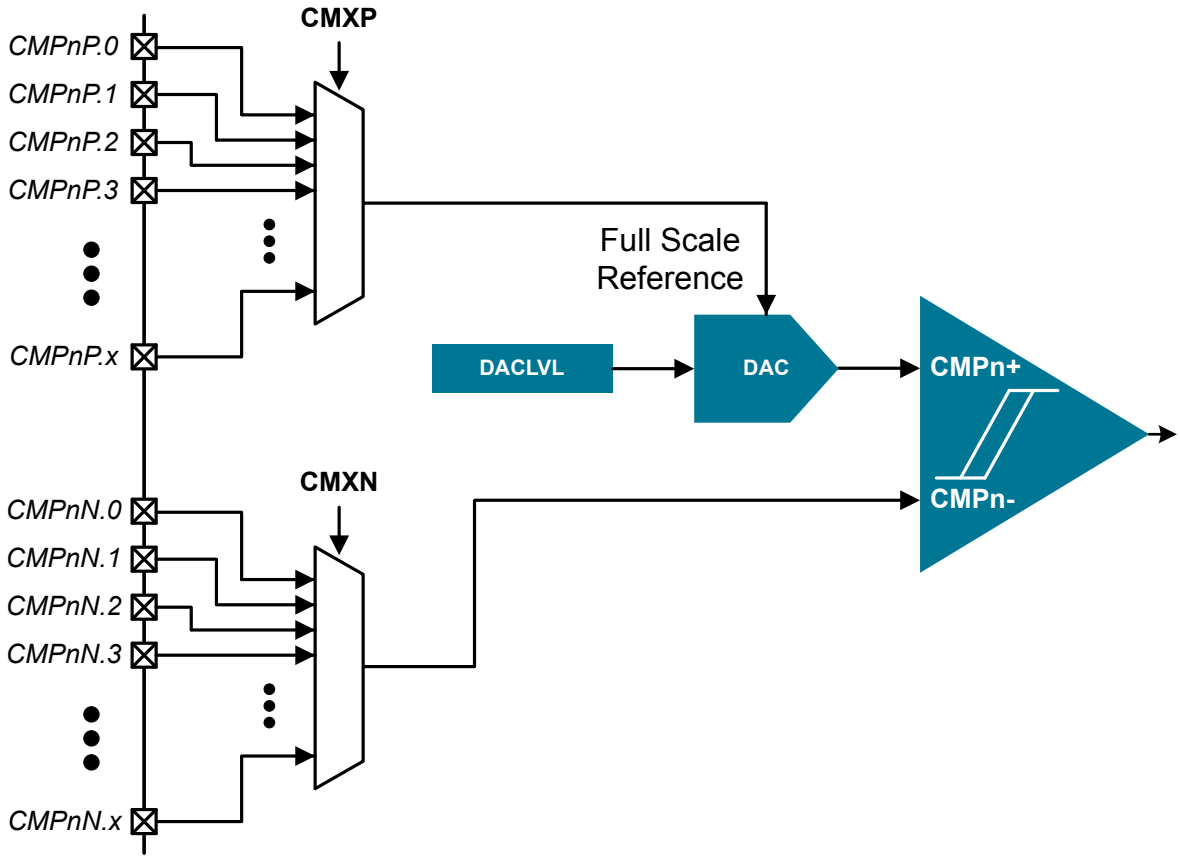


Figure 13.6. Positive Input DAC Connection

13.3.4 Output Routing

The comparator's synchronous and asynchronous outputs can optionally be routed to port I/O pins through the port I/O crossbar. The output of either comparator may be configured to generate a system interrupt on rising, falling, or both edges. CMPn may also be used as a reset source or as a trigger to kill a PCA output channel.

The output state of the comparator can be obtained at any time by reading the CPOUT bit. The comparator is enabled by setting the CPEN bit to logic 1, and is disabled by clearing this bit to logic 0. When disabled, the comparator output (if assigned to a port I/O pin via the crossbar) defaults to the logic low state, and the power supply to the comparator is turned off.

Comparator interrupts can be generated on both rising-edge and falling-edge output transitions. The CPFIF flag is set to logic 1 upon a comparator falling-edge occurrence, and the CPRIF flag is set to logic 1 upon the comparator rising-edge occurrence. Once set, these bits remain set until cleared by software. The comparator rising-edge interrupt mask is enabled by setting CPRIE to a logic 1. The comparator falling-edge interrupt mask is enabled by setting CPFIE to a logic 1.

False rising edges and falling edges may be detected when the comparator is first powered on or if changes are made to the hysteresis or response time control bits. Therefore, it is recommended that the rising-edge and falling-edge flags be explicitly cleared to logic 0 a short time after the comparator is enabled or its mode bits have been changed, before enabling comparator interrupts.

13.3.4.1 Output Inversion

The output state of the comparator may be inverted using the CPINV bit in register CMPnMD. When CPINV is 0, the output reflects the non-inverted state: CPOUT will be 1 when CP+ > CP- and 0 when CP+ < CP-. When CPINV is set to 1, the output reflects the inverted state: CPOUT will be 0 when CP+ > CP- and 1 when CP+ < CP-. Output inversion is applied directly at the comparator module output and affects the signal anywhere else it is used in the system.

13.3.4.2 Output Inhibit

The comparator module includes a feature to inhibit output changes whenever the PCA's CEX2 channel is logic low. This can be used to prevent undersirable glitches during known noise events, such as power FET switching. The CPINH bit in register CMPnCN1 enables this option. When CPINH is set to 1, the comparator output will hold its current state any time the CEX2 channel is logic low.

13.4 CMP0 Control Registers

13.4.1 CMP0CN0: Comparator 0 Control 0

Bit	7	6	5	4	3	2	1	0
Name	CPEN	CPOUT	CPRIF	CPFIF	CPHYP		CPHYN	
Access	RW	R	RW	RW	RW		RW	
Reset	0	0	0	0	0x0		0x0	

SFR Page = 0x0, 0x30; SFR Address: 0x9B

Bit	Name	Reset	Access	Description
7	CPEN	0	RW	**Comparator Enable.**
	Value	Name		Description
	0	DISABLED		Comparator disabled.
	1	ENABLED		Comparator enabled.
6	CPOUT	0	R	**Comparator Output State Flag.**
	Value	Name		Description
	0	POS_LESS_THAN_NEG		Voltage on CP0P < CP0N.
	1	POS_GREATER_THAN_NEG		Voltage on CP0P > CP0N.
5	CPRIF	0	RW	**Comparator Rising-Edge Flag.**
	Must be cleared by firmware.			
	Value	Name		Description
	0	NOT_SET		No comparator rising edge has occurred since this flag was last cleared.
	1	RISING_EDGE		Comparator rising edge has occurred.
4	CPFIF	0	RW	**Comparator Falling-Edge Flag.**
	Must be cleared by firmware.			
	Value	Name		Description
	0	NOT_SET		No comparator falling edge has occurred since this flag was last cleared.
	1	FALLING_EDGE		Comparator falling edge has occurred.
3:2	CPHYP	0x0	RW	**Comparator Positive Hysteresis Control.**
	Value	Name		Description
	0x0	DISABLED		Positive Hysteresis disabled.
	0x1	ENABLED_MODE1		Positive Hysteresis = Hysteresis 1.
	0x2	ENABLED_MODE2		Positive Hysteresis = Hysteresis 2.
	0x3	ENABLED_MODE3		Positive Hysteresis = Hysteresis 3 (Maximum).

Bit	Name	Reset	Access	Description
1:0	CPHYN	0x0	RW	**Comparator Negative Hysteresis Control.**

Value	Name	Description
0x0	DISABLED	Negative Hysteresis disabled.
0x1	ENABLED_MODE1	Negative Hysteresis = Hysteresis 1.
0x2	ENABLED_MODE2	Negative Hysteresis = Hysteresis 2.
0x3	ENABLED_MODE3	Negative Hysteresis = Hysteresis 3 (Maximum).

13.4.2 CMP0MD: Comparator 0 Mode

Bit	7	6	5	4	3	2	1	0
Name	CPLOUT	CPINV	CPRIE	CPFIE	INSL		CPMD	
Access	RW	RW	RW	RW	RW		RW	
Reset	0	0	0	0	0x0		0x2	
SFR Page = 0x0, 0x30; SFR Address: 0x9D								

Bit	Name	Reset	Access	Description
7	CPLOUT	0	RW	**Comparator Latched Output Flag.**
	This bit represents the comparator output value at the most recent PCA counter overflow.			
	Value	Name		Description
	0	LOW		Comparator output was logic low at last PCA overflow.
	1	HIGH		Comparator output was logic high at last PCA overflow.
6	CPINV	0	RW	**Output Inversion.**
	This bit inverts the polarity of the comparator output when set.			
	Value	Name		Description
	0	NORMAL		Output is not inverted.
	1	INVERT		Output is inverted.
5	CPRIE	0	RW	**Comparator Rising-Edge Interrupt Enable.**
	Value	Name		Description
	0	RISE_INT_DISABLED		Comparator rising-edge interrupt disabled.
	1	RISE_INT_ENABLED		Comparator rising-edge interrupt enabled.
4	CPFIE	0	RW	**Comparator Falling-Edge Interrupt Enable.**
	Value	Name		Description
	0	FALL_INT_DISABLED		Comparator falling-edge interrupt disabled.
	1	FALL_INT_ENABLED		Comparator falling-edge interrupt enabled.
3:2	INSL	0x0	RW	**Comparator Input Selection.**
	These bits control how the comparator input pins (CMP+ and CMP-) are connected internally.			
	Value	Name		Description
	0x0	CMXP_CMXN		Connect the comparator inputs directly to the signals selected in the CMP0MX register. CMP+ is selected by CMXP and CMP- is selected by CMXN. The internal DAC is not active.
	0x1	CMXP_GND		Connect the CMP+ input to the signal selected by CMXP, and CMP- is connected to GND. The internal DAC is not active.
	0x2	DAC_CMXN		Connect the CMP+ input to the internal DAC output, and CMP- is selected by CMXN. The internal DAC uses the signal specified by CMXP as its full-scale reference.

Bit	Name	Reset	Access	Description
	0x3	CMXP_DAC		Connect the CMP- input to the internal DAC output, and CMP+ is selected by CMXP. The internal DAC uses the signal specified by CMXN as its full-scale reference.
1:0	CPMD	0x2	RW	**Comparator Mode Select.** These bits affect the response time and power consumption of the comparator.

Value	Name	Description
0x0	MODE0	Mode 0 (Fastest Response Time, Highest Power Consumption)
0x1	MODE1	Mode 1
0x2	MODE2	Mode 2
0x3	MODE3	Mode 3 (Slowest Response Time, Lowest Power Consumption)

13.4.3 CMP0MX: Comparator 0 Multiplexer Selection

Bit	7	6	5	4	3	2	1	0
Name	CMXN				CMXP			
Access	RW				RW			
Reset	0xF				0xF			
SFR Page = 0x0, 0x30; SFR Address: 0x9F								

Bit	Name	Reset	Access	Description
7:4	CMXN	0xF	RW	**Comparator Negative Input MUX Selection.** This field selects the negative input for the comparator.
3:0	CMXP	0xF	RW	**Comparator Positive Input MUX Selection.** This field selects the positive input for the comparator.

13.4.4 CMP0CN1: Comparator 0 Control 1

Bit	7	6	5	4	3	2	1	0
Name	CPINH	Reserved	DACLVL					
Access	RW	R	RW					
Reset	0	0	0x00					
SFR Page = 0x30; SFR Address: 0x99								

Bit	Name	Reset	Access	Description
7	CPINH	0	RW	**Output Inhibit.**
	This bit is used to inhibit the comparator output during CEX2 low times.			
	Value	Name		Description
	0	DISABLED		The comparator output will always reflect the input conditions.
	1	ENABLED		The comparator output will hold state any time the PCA CEX2 channel is low.
6	*Reserved*	*Must write reset value.*		
5:0	DACLVL	0x00	RW	**Internal Comparator DAC Reference Level.**
	These bits control the output of the comparator reference DAC. The voltage is given by:			
	DAC Output = CMPREF * (DACLVL / 64)			
	CMPREF is the selected input reference for the DAC according to INSL, CMXP and CMXN.			

13.5 CMP1 Control Registers

13.5.1 CMP1CN0: Comparator 1 Control 0

Bit	7	6	5	4	3	2	1	0
Name	CPEN	CPOUT	CPRIF	CPFIF	CPHYP		CPHYN	
Access	RW	R	RW	RW	RW		RW	
Reset	0	0	0	0	0x0		0x0	
SFR Page = 0x0, 0x30; SFR Address: 0xBF								

Bit	Name	Reset	Access	Description
7	CPEN	0	RW	**Comparator Enable.**
	Value	Name		Description
	0	DISABLED		Comparator disabled.
	1	ENABLED		Comparator enabled.
6	CPOUT	0	R	**Comparator Output State Flag.**
	Value	Name		Description
	0	POS_LESS_THAN_NEG		Voltage on CP1P < CP1N.
	1	POS_GREATER_THAN_NEG		Voltage on CP1P > CP1N.
5	CPRIF	0	RW	**Comparator Rising-Edge Flag.**
	Must be cleared by firmware.			
	Value	Name		Description
	0	NOT_SET		No comparator rising edge has occurred since this flag was last cleared.
	1	RISING_EDGE		Comparator rising edge has occurred.
4	CPFIF	0	RW	**Comparator Falling-Edge Flag.**
	Must be cleared by firmware.			
	Value	Name		Description
	0	NOT_SET		No comparator falling edge has occurred since this flag was last cleared.
	1	FALLING_EDGE		Comparator falling edge has occurred.
3:2	CPHYP	0x0	RW	**Comparator Positive Hysteresis Control.**
	Value	Name		Description
	0x0	DISABLED		Positive Hysteresis disabled.
	0x1	ENABLED_MODE1		Positive Hysteresis = Hysteresis 1.
	0x2	ENABLED_MODE2		Positive Hysteresis = Hysteresis 2.
	0x3	ENABLED_MODE3		Positive Hysteresis = Hysteresis 3 (Maximum).

Bit	Name	Reset	Access	Description
1:0	CPHYN	0x0	RW	**Comparator Negative Hysteresis Control.**

Value	Name	Description
0x0	DISABLED	Negative Hysteresis disabled.
0x1	ENABLED_MODE1	Negative Hysteresis = Hysteresis 1.
0x2	ENABLED_MODE2	Negative Hysteresis = Hysteresis 2.
0x3	ENABLED_MODE3	Negative Hysteresis = Hysteresis 3 (Maximum).

13.5.2 CMP1MD: Comparator 1 Mode

Bit	7	6	5	4	3	2	1	0
Name	CPLOUT	CPINV	CPRIE	CPFIE	INSL		CPMD	
Access	RW	RW	RW	RW	RW		RW	
Reset	0	0	0	0	0x0		0x2	
SFR Page = 0x0, 0x30; SFR Address: 0xAB								

Bit	Name	Reset	Access	Description
7	CPLOUT	0	RW	**Comparator Latched Output Flag.**

This bit represents the comparator output value at the most recent PCA counter overflow.

Value	Name	Description
0	LOW	Comparator output was logic low at last PCA overflow.
1	HIGH	Comparator output was logic high at last PCA overflow.

Bit	Name	Reset	Access	Description
6	CPINV	0	RW	**Output Inversion.**

This bit inverts the polarity of the comparator output when set.

Value	Name	Description
0	NORMAL	Output is not inverted.
1	INVERT	Output is inverted.

Bit	Name	Reset	Access	Description
5	CPRIE	0	RW	**Comparator Rising-Edge Interrupt Enable.**

Value	Name	Description
0	RISE_INT_DISABLED	Comparator rising-edge interrupt disabled.
1	RISE_INT_ENABLED	Comparator rising-edge interrupt enabled.

Bit	Name	Reset	Access	Description
4	CPFIE	0	RW	**Comparator Falling-Edge Interrupt Enable.**

Value	Name	Description
0	FALL_INT_DISABLED	Comparator falling-edge interrupt disabled.
1	FALL_INT_ENABLED	Comparator falling-edge interrupt enabled.

Bit	Name	Reset	Access	Description
3:2	INSL	0x0	RW	**Comparator Input Selection.**

These bits control how the comparator input pins (CMP+ and CMP-) are connected internally.

Value	Name	Description
0x0	CMXP_CMXN	Connect the comparator inputs directly to the signals selected in the CMP1MX register. CMP+ is selected by CMXP and CMP- is selected by CMXN. The internal DAC is not active.
0x1	CMXP_GND	Connect the CMP+ input to the signal selected by CMXP, and CMP- is connected to GND. The internal DAC is not active.
0x2	DAC_CMXN	Connect the CMP+ input to the internal DAC output, and CMP- is selected by CMXN. The internal DAC uses the signal specified by CMXP as its full-scale reference.

Bit	Name	Reset	Access	Description
	0x3	CMXP_DAC		Connect the CMP- input to the internal DAC output, and CMP+ is selected by CMXP. The internal DAC uses the signal specified by CMXN as its full-scale reference.
1:0	CPMD	0x2	RW	**Comparator Mode Select.** These bits affect the response time and power consumption of the comparator.

Value	Name	Description
0x0	MODE0	Mode 0 (Fastest Response Time, Highest Power Consumption)
0x1	MODE1	Mode 1
0x2	MODE2	Mode 2
0x3	MODE3	Mode 3 (Slowest Response Time, Lowest Power Consumption)

13.5.3 CMP1MX: Comparator 1 Multiplexer Selection

Bit	7	6	5	4	3	2	1	0
Name	CMXN				CMXP			
Access	RW				RW			
Reset	0xF				0xF			
SFR Page = 0x0, 0x30; SFR Address: 0xAA								

Bit	Name	Reset	Access	Description
7:4	CMXN	0xF	RW	**Comparator Negative Input MUX Selection.** This field selects the negative input for the comparator.
3:0	CMXP	0xF	RW	**Comparator Positive Input MUX Selection.** This field selects the positive input for the comparator.

13.5.4 CMP1CN1: Comparator 1 Control 1

Bit	7	6	5	4	3	2	1	0
Name	CPINH	Reserved	DACLVL					
Access	RW	R	RW					
Reset	0	0	0x00					
SFR Page = 0x30; SFR Address: 0xAC								

Bit	Name	Reset	Access	Description
7	CPINH	0	RW	**Output Inhibit.**
	This bit is used to inhibit the comparator output during CEX2 low times.			
	Value	Name		Description
	0	DISABLED		The comparator output will always reflect the input conditions.
	1	ENABLED		The comparator output will hold state any time the PCA CEX2 channel is low.
6	*Reserved*	*Must write reset value.*		
5:0	DACLVL	0x00	RW	**Internal Comparator DAC Reference Level.**
	These bits control the output of the comparator reference DAC. The voltage is given by:			
	DAC Output = CMPREF * (DACLVL / 64)			
	CMPREF is the selected input reference for the DAC according to INSL, CMXP and CMXN.			

14. Configurable Logic Units (CLU0, CLU1, CLU2, CLU3)

14.1 Introduction

The configurable logic (CL) module provides multiple blocks of user-programmed digital logic that operates without CPU intervention. It consists of four dedicated independent configurable logic units (CLUs) which support user programmable asynchronous and synchronous boolean logic operations. A number of internal and external signals may be used as inputs to each CLU, and the outputs may be routed out to port I/O pins or directly to select peripheral inputs.

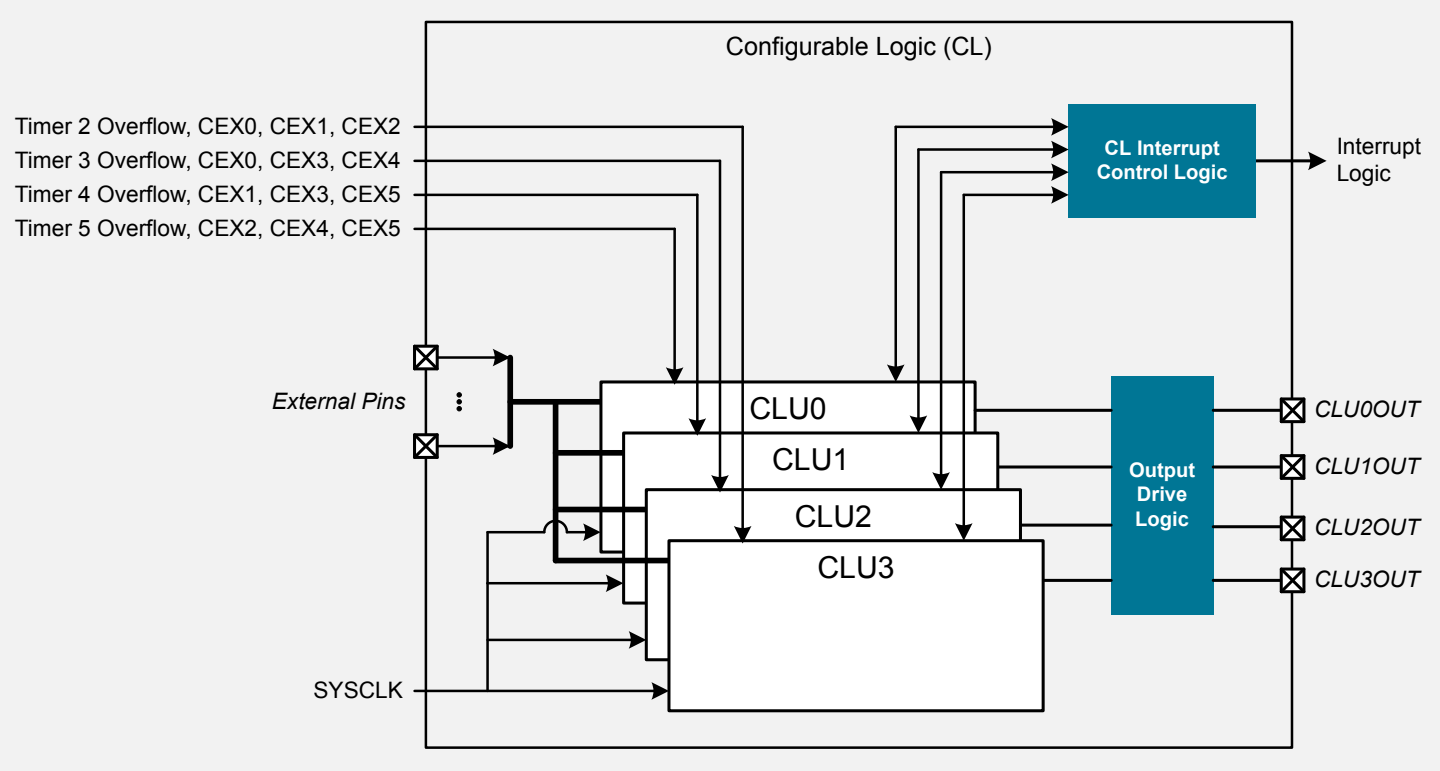


Figure 14.1. Configurable Logic Top-Level Block Diagram

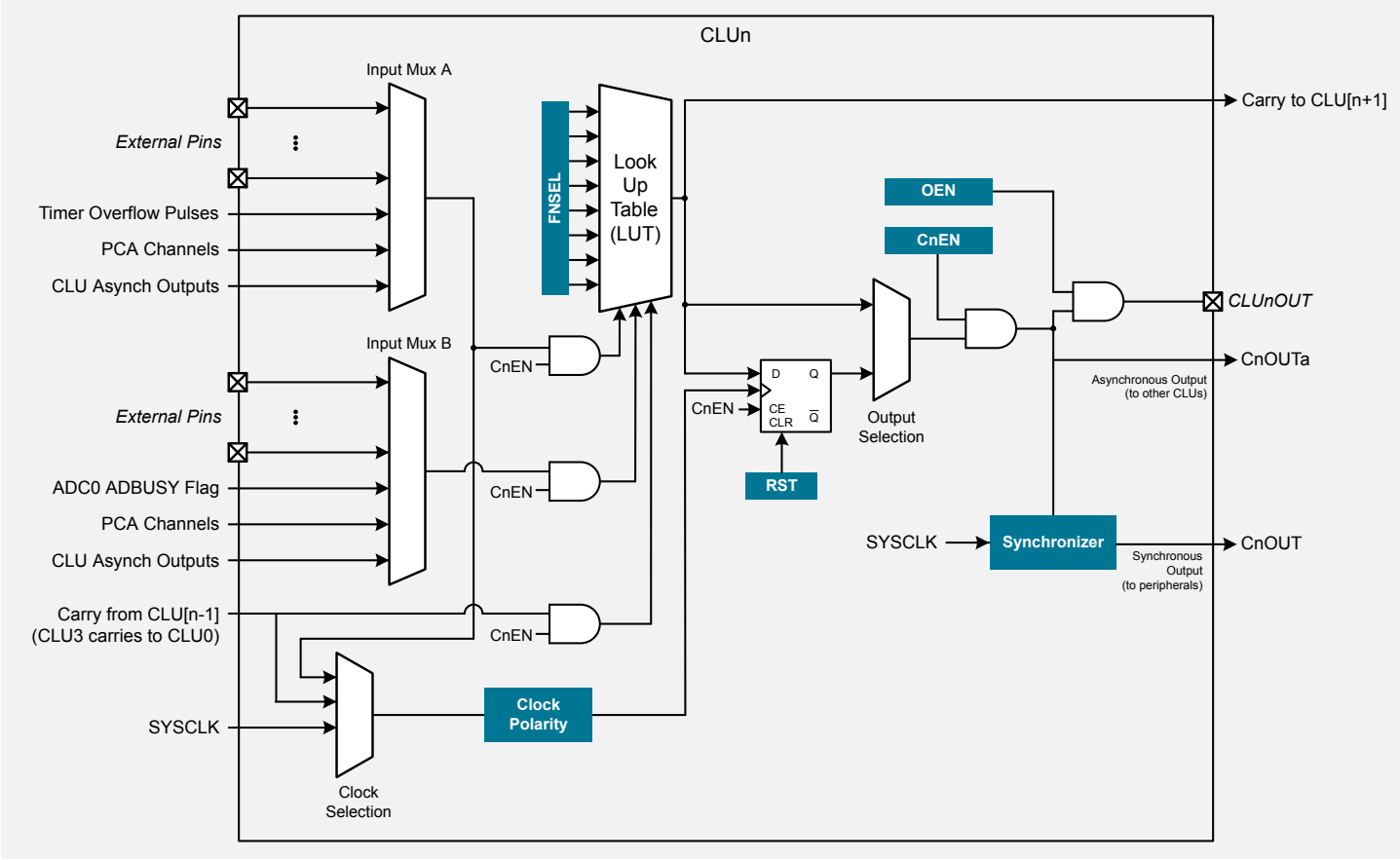


Figure 14.2. Individual CLU Block Diagram

14.2 Features

The key features of the Configurable Logic block are as follows:

- Four configurable logic units (CLUs), with direct-pin and internal logic connections
- Each unit supports 256 different combinatorial logic functions (AND, OR, XOR, muxing, etc.) and includes a clocked flip-flop for synchronous operations
- Units may be operated synchronously or asynchronously
- May be cascaded together to perform more complicated logic functions
- Can operate in conjunction with serial peripherals such as UART and SPI or timing peripherals such as timers and PCA channels
- Can be used to synchronize and trigger multiple on-chip resources (ADC, DAC, Timers, etc.)
- Asynchronous output may be used to wake from low-power states

14.3 Functional Description

14.3.1 Configuration Sequence

Firmware should configure the function select, mux inputs and output functionality before enabling individual CLUs. CLU initialization consists of the following general steps:

1. Select the A and B inputs to the LUT in CLUnMX
2. Select the LUT function using CLUnFN
3. Configure the CLU via CLUnCF.
4. If the D flip-flop output is selected (OUTSEL=1) for the CLU, it is advised to also set RST=1 to reset the flop output to 0.
5. Setup any interrupt required in CLIE0. Falling and rising edge interrupts for each module are enabled using the CnFIE and CnRIE bits, respectively.
6. Enable the CLU by setting the CnEN bit in CLEN0. Firmware may enable multiple CLUs at the same time by setting more than one bit in CLEN0.
7. If direct pin output is required, firmware may enable the output by setting the OEN bit in CLUnCF

14.3.2 Input Multiplexer Selection

Each CLU has two primary logic inputs (A and B) and a carry input (C). The A and B inputs are selected by the MXA and MXB fields in the CLUnMX register, and may be one of many different internal and external signals. When another CLU output is selected as an input, the asynchronous output from that CLU is used, enabling more complex boolean logic functions to be implemented.

Note: When using timer overflow events as an input, the timer overflow event is a pulse which will be logic high for one SYSCLK cycle, and logic low for the rest of the timer period.

The carry input, C, is the LUT output of the previous CLU. For example, the carry input on CLU1 is CLU0's LUT output. The carry input for CLU0 is CLU3's LUT output.

Pin inputs to CLU inputs are not SYSCLK-synchronized. Other internal peripherals (such as Timers) to CLU inputs are SYSCLK-synchronized since these peripherals are SYSCLK-synchronized. A pulse needs to be at least 1 SYSCLK wide for a timer in capture mode to be guaranteed to capture the edge. However, it is still possible for a narrower pulse to be captured. So, firmware incorporating a CLU cannot depend on the timer not capturing a pulse that is less than 1 SYSCLK period wide.

14.3.2.1 CLU Multiplexer Input Selection

Table 14.1. CLUnA Input Selection

CLUnMX.MXA	CLU0A	CLU1A	CLU2A	CLU3A
0000	C0OUTa	C0OUTa	C0OUTa	C0OUTa
0001	C1OUTa	C1OUTa	C1OUTa	C1OUTa
0010	C2OUTa	C2OUTa	C2OUTa	C2OUTa
0011	C3OUTa	C3OUTa	C3OUTa	C3OUTa
0100	Timer2 Overflow	Timer3 Overflow	Timer4 Overflow	Timer5 Overflow
0101	CEX0	CEX0	CEX1	CEX2
0110	CEX1	CEX3	CEX3	CEX4
0111	CEX2	CEX4	CEX5	CEX5
1000	P0.0	P0.4	P0.0	P0.2
1001	P0.2	P0.5	P0.1	P0.3
1010	P0.4	P1.0	P1.0	P0.6
1011	P0.6	P1.2	P1.1	P0.7
1100	P1.0	P1.4	P1.6	P1.2
1101	P1.2	P1.5	P1.7	P1.3
1110	P1.4	P2.0	P2.0	P2.2
1111	P1.6	P2.2	P2.1	P2.3

Table 14.2. CLUnB Input Selection

CLUnMX.MXB	CLU0B	CLU1B	CLU2B	CLU3B
0000	C0OUTa	C0OUTa	C0OUTa	C0OUTa
0001	C1OUTa	C1OUTa	C1OUTa	C1OUTa
0010	C2OUTa	C2OUTa	C2OUTa	C2OUTa
0011	C3OUTa	C3OUTa	C3OUTa	C3OUTa
0100	ADBUSY	ADBUSY	ADBUSY	ADBUSY
0101	CEX3	CEX1	CEX0	CEX0
0110	CEX4	CEX2	CEX2	CEX1
0111	CEX5	CEX5	CEX4	CEX3
1000	P0.1	P0.6	P0.2	P0.0
1001	P0.3	P0.7	P0.3	P0.1
1010	P0.5	P1.1	P1.2	P0.4
1011	P0.7	P1.3	P1.3	P0.5
1100	P1.1	P1.6	P1.4	P1.0
1101	P1.3	P1.7	P1.5	P1.1
1110	P1.5	P2.1	P2.2	P2.0

CLUnMX.MXB	CLU0B	CLU1B	CLU2B	CLU3B
1111	P1.7	P2.3	P2.3	P2.1

14.3.3 Output Configuration

Each CLU presents an asynchronous and a synchronous (synchronized to SYSCLK) output to the system. The synchronous output may be read by firmware at any time by reading the CLOUT0 register. CLU outputs may be derived directly from the LUT, or from a latched D-type flip-flop output, as controlled by the OUTSEL bit in CLUnCF. When a CLU is disabled (CnEN in CLEN0 is 0), both of its outputs will be held at logic 0.

The D flip-flop clock may be configured from one of four sources, selected by the CLKSEL field in CLUnCF. The flip-flop clock may optionally be inverted, using the CLKINV bit. Each CLU has the following options for clocking its flip-flop:

- CARRY_IN: The carry (C) input from the previous CLU. The first CLU uses the carry from the last CLU.
- MXA_INPUT: The A input to the CLU, as defined by the MXA register field.
- SYSCLK: The system clock.

When using the D flip-flop output, the flip-flop may be reset to logic 0 at any time by writing 1 to the RST bit in CLUnCF. The output will not be held in this reset state (RST returns to 0 after the reset occurs).

The CLU outputs may also be present on selected pins.

CLU output signals to internal peripherals (except another CLU input) are SYSCLK-synchronized. CLU output signals to any CLU input are not SYSCLK-synchronized.

14.3.4 LUT Configuration

The boolean logic function in each CLU is determined by the LUT, and may be changed by programming the FNSEL field in register CLUnFN. The LUT is implemented as an 8-input multiplexer. The bits of FNSEL map to the 8 multiplexer inputs, and the output of the LUT is selected by the combination of the A, B, and C inputs.

Table 14.3. LUT Truth Table

A Input	B Input	C Input	LUT Output
0	0	0	FNSEL.0
0	0	1	FNSEL.1
0	1	0	FNSEL.2
0	1	1	FNSEL.3
1	0	0	FNSEL.4
1	0	1	FNSEL.5
1	1	0	FNSEL.6
1	1	1	FNSEL.7

It is possible to realize any 3-input boolean logic function using the LUT. To determine the value to be programmed into FNSEL for a given logic function, the truth table in Table 14.3 LUT Truth Table on page 196 may be used. For example, to implement the boolean function (A AND B), the LUT output should be 1 for any combination where A and B are 1, and 0 for all other combinations. The last two rows in the table (corresponding to FNSEL.7 and FNSEL.6) meet this criteria, so FNSEL should be programmed to 11000000b, or 0xC0.

As a second example, if the function (A XOR B) is required, the rows corresponding to FNSEL.2, FNSEL.3, FNSEL.4 and FNSEL.5 would be logic 1, and logic 0 for FNSEL.0, FNSEL.1, FNSEL.6 and FNSEL.7. Therefore, FNSEL should be programmed to 00111100b, or 0x3C to realize this function.

14.4 Configurable Logic Control Registers

14.4.1 CLEN0: Configurable Logic Enable 0

Bit	7	6	5	4	3	2	1	0
Name	Reserved				C3EN	C2EN	C1EN	C0EN
Access	R				RW	RW	RW	RW
Reset	0x0				0	0	0	0
SFR Page = 0x20; SFR Address: 0xC6								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	C3EN	0	RW	**CLU3 Enable.**
	Value	Name		Description
	0	DISABLE		CLU3 is disabled. The output of the block will be logic low.
	1	ENABLE		CLU3 is enabled.
2	C2EN	0	RW	**CLU2 Enable.**
	Value	Name		Description
	0	DISABLE		CLU2 is disabled. The output of the block will be logic low.
	1	ENABLE		CLU2 is enabled.
1	C1EN	0	RW	**CLU1 Enable.**
	Value	Name		Description
	0	DISABLE		CLU1 is disabled. The output of the block will be logic low.
	1	ENABLE		CLU1 is enabled.
0	C0EN	0	RW	**CLU0 Enable.**
	Value	Name		Description
	0	DISABLE		CLU0 is disabled. The output of the block will be logic low.
	1	ENABLE		CLU0 is enabled.

14.4.2 CLIE0: Configurable Logic Interrupt Enable 0

Bit	7	6	5	4	3	2	1	0
Name	C3RIE	C3FIE	C2RIE	C2FIE	C1RIE	C1FIE	C0RIE	C0FIE
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x20; SFR Address: 0xC7								

Bit	Name	Reset	Access	Description
7	C3RIE	0	RW	**CLU3 Rising Edge Interrupt Enable.**
	Enables interrupts generated by CLU3 rising edges (synchronized with SYSCLK).			
	Value	Name		Description
	0	DISABLE		Interrupts will not be generated for CLU3 rising-edge events.
	1	ENABLE		Interrupts will be generated for CLU3 rising-edge events.
6	C3FIE	0	RW	**CLU3 Falling Edge Interrupt Enable.**
	Enables interrupts generated by CLU3 falling edges (synchronized with SYSCLK).			
	Value	Name		Description
	0	DISABLE		Interrupts will not be generated for CLU3 falling-edge events.
	1	ENABLE		Interrupts will be generated for CLU3 falling-edge events.
5	C2RIE	0	RW	**CLU2 Rising Edge Interrupt Enable.**
	See bit 7 description			
4	C2FIE	0	RW	**CLU2 Falling Edge Interrupt Enable.**
	See bit 6 description			
3	C1RIE	0	RW	**CLU1 Rising Edge Interrupt Enable.**
	See bit 7 description			
2	C1FIE	0	RW	**CLU1 Falling Edge Interrupt Enable.**
	See bit 6 description			
1	C0RIE	0	RW	**CLU0 Rising Edge Interrupt Enable.**
	See bit 7 description			
0	C0FIE	0	RW	**CLU0 Falling Edge Interrupt Enable.**
	See bit 6 description			

14.4.3 CLIF0: Configurable Logic Interrupt Flag 0

Bit	7	6	5	4	3	2	1	0
Name	C3RIF	C3FIF	C2RIF	C2FIF	C1RIF	C1FIF	C0RIF	C0FIF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x20; SFR Address: 0xE8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	C3RIF	0	RW	**CLU3 Rising Edge Flag.**
	Value	Name		Description
	0	NOT_SET		A CLU3 rising edge has not been detected since this flag was last cleared.
	1	SET		A CLU3 rising edge (synchronized with SYSCLK) has occurred. This bit must be cleared by firmware.
6	C3FIF	0	RW	**CLU3 Falling Edge Flag.**
	Value	Name		Description
	0	NOT_SET		A CLU3 falling edge has not been detected since this flag was last cleared.
	1	SET		A CLU3 falling edge (synchronized with SYSCLK) has occurred. This bit must be cleared by firmware.
5	C2RIF	0	RW	**CLU2 Rising Edge Flag.**
	See bit 7 description			
4	C2FIF	0	RW	**CLU2 Falling Edge Flag.**
	See bit 6 description			
3	C1RIF	0	RW	**CLU1 Rising Edge Flag.**
	See bit 7 description			
2	C1FIF	0	RW	**CLU1 Falling Edge Flag.**
	See bit 6 description			
1	C0RIF	0	RW	**CLU0 Rising Edge Flag.**
	See bit 7 description			
0	C0FIF	0	RW	**CLU0 Falling Edge Flag.**
	See bit 6 description			

14.4.4 CLOUT0: Configurable Logic Output 0

Bit	7	6	5	4	3	2	1	0
Name	Reserved				C3OUT	C2OUT	C1OUT	C0OUT
Access	R				R	R	R	R
Reset	0x0				0	0	0	0
SFR Page = 0x20; SFR Address: 0xD1								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	C3OUT	0	R	**CLU3 Output State.** This bit represents the logic level of the CLU3 output, synchronized with SYSCLK.
2	C2OUT	0	R	**CLU2 Output State.** This bit represents the logic level of the CLU2 output, synchronized with SYSCLK.
1	C1OUT	0	R	**CLU1 Output State.** This bit represents the logic level of the CLU1 output, synchronized with SYSCLK.
0	C0OUT	0	R	**CLU0 Output State.** This bit represents the logic level of the CLU0 output, synchronized with SYSCLK.

14.4.5 CLU0MX: Configurable Logic Unit 0 Multiplexer

Bit	7	6	5	4	3	2	1	0
Name	MXA				MXB			
Access	RW				RW			
Reset	0x0				0x0			
SFR Page = 0x20; SFR Address: 0x84								

Bit	Name	Reset	Access	Description
7:4	MXA	0x0	RW	**CLU0 A Input Multiplexer Selection.** Selects the A input to CLU0.
3:0	MXB	0x0	RW	**CLU0 B Input Multiplexer Selection.** Selects the B input to CLU0.

14.4.6 CLU0FN: Configurable Logic Unit 0 Function Select

Bit	7	6	5	4	3	2	1	0
Name	FNSEL							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xAF								

Bit	Name	Reset	Access	Description
7:0	FNSEL	0x00	RW	**CLU Look-Up-Table function select.** Function select for the CLU0 LUT. The LUT is an 8-input multiplexer where the inputs are the bits of FNSEL. The multiplexer selection signals are (MS bit first): MXA, MXB, Carry-in Examples: FNSEL = 0xC0 implements: MXA & MXB. FNSEL = 0xE4 implements: (Carry & MXA) | ((not Carry) & MXB) The second example is a multiplexer where Carry is used to select MXA or MXB.

14.4.7 CLU0CF: Configurable Logic Unit 0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	OUTSEL	OEN	Reserved		RST	CLKINV	CLKSEL	
Access	RW	RW	R		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x20; SFR Address: 0xB1								

Bit	Name	Reset	Access	Description
7	OUTSEL	0	RW	**CLU Output Select.**
	Value	Name		Description
	0	D_FF		Select D flip-flop output of CLU
	1	LUT		Select LUT output.
6	OEN	0	RW	**CLU Port Output Enable.**
	This bit enables the asynchronous output of CLU0 to CLU0OUT.			
	Value	Name		Description
	0	DISABLE		Disables asynchronous output to the selected GPIO pin
	1	ENABLE		Enables asynchronous output to the selected GPIO pin
5:4	*Reserved*	*Must write reset value.*		
3	RST	0	RW	**CLU D flip-flop Reset.**
	Writing this bit to 1 resets the D flip flop for CLU0. The bit will immediately return to 0.			
	Value	Name		Description
	1	RESET		Reset the flip flop.
2	CLKINV	0	RW	**CLU D flip-flop Clock Invert.**
	Value	Name		Description
	0	NORMAL		Clock signal is not inverted.
	1	INVERT		Clock signal will be inverted.
1:0	CLKSEL	0x0	RW	**CLU D flip-flop Clock Selection.**
	Value	Name		Description
	0x0	CARRY_IN		The carry-in signal.
	0x1	MXA_INPUT		The MXA input.
	0x2	SYSCLK		SYSCLK.

14.4.8 CLU1MX: Configurable Logic Unit 1 Multiplexer

Bit	7	6	5	4	3	2	1	0
Name	MXA				MXB			
Access	RW				RW			
Reset	0x0				0x0			
SFR Page = 0x20; SFR Address: 0x85								

Bit	Name	Reset	Access	Description
7:4	MXA	0x0	RW	**CLU1 A Input Multiplexer Selection.** Selects the A input to CLU1.
3:0	MXB	0x0	RW	**CLU1 B Input Multiplexer Selection.** Selects the B input to CLU1.

14.4.9 CLU1FN: Configurable Logic Unit 1 Function Select

Bit	7	6	5	4	3	2	1	0
Name	FNSEL							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xB2								

Bit	Name	Reset	Access	Description
7:0	FNSEL	0x00	RW	**CLU Look-Up-Table function select.** Function select for the CLU1 LUT. The LUT is an 8-input multiplexer where the inputs are the bits of FNSEL. The multiplexer selection signals are (MS bit first): MXA, MXB, Carry-in Examples: FNSEL = 0xC0 implements: MXA & MXB. FNSEL = 0xE4 implements: (Carry & MXA) | ((not Carry) & MXB) The second example is a multiplexer where Carry is used to select MXA or MXB.

14.4.10 CLU1CF: Configurable Logic Unit 1 Configuration

Bit	7	6	5	4	3	2	1	0
Name	OUTSEL	OEN	Reserved		RST	CLKINV	CLKSEL	
Access	RW	RW	R		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	

SFR Page = 0x20; SFR Address: 0xB3

Bit	Name	Reset	Access	Description
7	OUTSEL	0	RW	**CLU Output Select.**
	Value	Name		Description
	0	D_FF		Select D flip-flop output of CLU
	1	LUT		Select LUT output.
6	OEN	0	RW	**CLU Port Output Enable.**
	This bit enables the asynchronous output of CLU1 to CLU1OUT.			
	Value	Name		Description
	0	DISABLE		Disables asynchronous output to the selected GPIO pin
	1	ENABLE		Enables asynchronous output to the selected GPIO pin
5:4	*Reserved*	*Must write reset value.*		
3	RST	0	RW	**CLU D flip-flop Reset.**
	Writing this bit to 1 resets the D flip flop for CLU1. The bit will immediately return to 0.			
	Value	Name		Description
	1	RESET		Reset the flip flop.
2	CLKINV	0	RW	**CLU D flip-flop Clock Invert.**
	Value	Name		Description
	0	NORMAL		Clock signal is not inverted.
	1	INVERT		Clock signal will be inverted.
1:0	CLKSEL	0x0	RW	**CLU D flip-flop Clock Selection.**
	Value	Name		Description
	0x0	CARRY_IN		The carry-in signal.
	0x1	MXA_INPUT		The MXA input.
	0x2	SYSCLK		SYSCLK.

14.4.11 CLU2MX: Configurable Logic Unit 2 Multiplexer

Bit	7	6	5	4	3	2	1	0
Name	MXA				MXB			
Access	RW				RW			
Reset	0x0				0x0			
SFR Page = 0x20; SFR Address: 0x91								

Bit	Name	Reset	Access	Description
7:4	MXA	0x0	RW	**CLU2 A Input Multiplexer Selection.** Selects the A input to CLU2.
3:0	MXB	0x0	RW	**CLU2 B Input Multiplexer Selection.** Selects the B input to CLU2.

14.4.12 CLU2FN: Configurable Logic Unit 2 Function Select

Bit	7	6	5	4	3	2	1	0
Name	FNSEL							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xB5								

Bit	Name	Reset	Access	Description
7:0	FNSEL	0x00	RW	**CLU Look-Up-Table function select.** Function select for the CLU2 LUT. The LUT is an 8-input multiplexer where the inputs are the bits of FNSEL. The multiplexer selection signals are (MS bit first): MXA, MXB, Carry-in Examples: FNSEL = 0xC0 implements: MXA & MXB. FNSEL = 0xE4 implements: (Carry & MXA) | ((not Carry) & MXB) The second example is a multiplexer where Carry is used to select MXA or MXB.

14.4.13 CLU2CF: Configurable Logic Unit 2 Configuration

Bit	7	6	5	4	3	2	1	0
Name	OUTSEL	OEN	Reserved		RST	CLKINV	CLKSEL	
Access	RW	RW	R		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	

SFR Page = 0x20; SFR Address: 0xB6

Bit	Name	Reset	Access	Description
7	OUTSEL	0	RW	**CLU Output Select.**
	Value	Name		Description
	0	D_FF		Select D flip-flop output of CLU
	1	LUT		Select LUT output.
6	OEN	0	RW	**CLU Port Output Enable.**
	This bit enables the asynchronous output of CLU2 to CLU2OUT.			
	Value	Name		Description
	0	DISABLE		Disables asynchronous output to the selected GPIO pin
	1	ENABLE		Enables asynchronous output to the selected GPIO pin
5:4	*Reserved*	*Must write reset value.*		
3	RST	0	RW	**CLU D flip-flop Reset.**
	Writing this bit to 1 resets the D flip flop for CLU2. The bit will immediately return to 0.			
	Value	Name		Description
	1	RESET		Reset the flip flop.
2	CLKINV	0	RW	**CLU D flip-flop Clock Invert.**
	Value	Name		Description
	0	NORMAL		Clock signal is not inverted.
	1	INVERT		Clock signal will be inverted.
1:0	CLKSEL	0x0	RW	**CLU D flip-flop Clock Selection.**
	Value	Name		Description
	0x0	CARRY_IN		The carry-in signal.
	0x1	MXA_INPUT		The MXA input.
	0x2	SYSCLK		SYSCLK.

14.4.14 CLU3MX: Configurable Logic Unit 3 Multiplexer

Bit	7	6	5	4	3	2	1	0
Name	MXA				MXB			
Access	RW				RW			
Reset	0x0				0x0			
SFR Page = 0x20; SFR Address: 0xAE								

Bit	Name	Reset	Access	Description
7:4	MXA	0x0	RW	**CLU3 A Input Multiplexer Selection.** Selects the A input to CLU3.
3:0	MXB	0x0	RW	**CLU3 B Input Multiplexer Selection.** Selects the B input to CLU3.

14.4.15 CLU3FN: Configurable Logic Unit 3 Function Select

Bit	7	6	5	4	3	2	1	0
Name	FNSEL							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xBE								

Bit	Name	Reset	Access	Description
7:0	FNSEL	0x00	RW	**CLU Look-Up-Table function select.** Function select for the CLU3 LUT. The LUT is an 8-input multiplexer where the inputs are the bits of FNSEL. The multiplexer selection signals are (MS bit first): MXA, MXB, Carry-in Examples: FNSEL = 0xC0 implements: MXA & MXB. FNSEL = 0xE4 implements: (Carry & MXA) | ((not Carry) & MXB) The second example is a multiplexer where Carry is used to select MXA or MXB.

14.4.16 CLU3CF: Configurable Logic Unit 3 Configuration

Bit	7	6	5	4	3	2	1	0
Name	OUTSEL	OEN	Reserved		RST	CLKINV	CLKSEL	
Access	RW	RW	R		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x20; SFR Address: 0xBF								

Bit	Name	Reset	Access	Description
7	OUTSEL	0	RW	**CLU Output Select.**
	Value	Name		Description
	0	D_FF		Select D flip-flop output of CLU
	1	LUT		Select LUT output.
6	OEN	0	RW	**CLU Port Output Enable.**
	This bit enables the asynchronous output of CLU3 to CLU3OUT.			
	Value	Name		Description
	0	DISABLE		Disables asynchronous output to the selected GPIO pin
	1	ENABLE		Enables asynchronous output to the selected GPIO pin
5:4	*Reserved*	*Must write reset value.*		
3	RST	0	RW	**CLU D flip-flop Reset.**
	Writing this bit to 1 resets the D flip flop for CLU3. The bit will immediately return to 0.			
	Value	Name		Description
	1	RESET		Reset the flip flop.
2	CLKINV	0	RW	**CLU D flip-flop Clock Invert.**
	Value	Name		Description
	0	NORMAL		Clock signal is not inverted.
	1	INVERT		Clock signal will be inverted.
1:0	CLKSEL	0x0	RW	**CLU D flip-flop Clock Selection.**
	Value	Name		Description
	0x0	CARRY_IN		The carry-in signal.
	0x1	MXA_INPUT		The MXA input.
	0x2	SYSCLK		SYSCLK.

15. Cyclic Redundancy Check (CRC0)

15.1 Introduction

The cyclic redundancy check (CRC) module performs a CRC using a 16-bit polynomial. CRC0 accepts a stream of 8-bit data and posts the 16-bit result to an internal register. In addition to using the CRC block for data manipulation, hardware can automatically CRC the flash contents of the device.

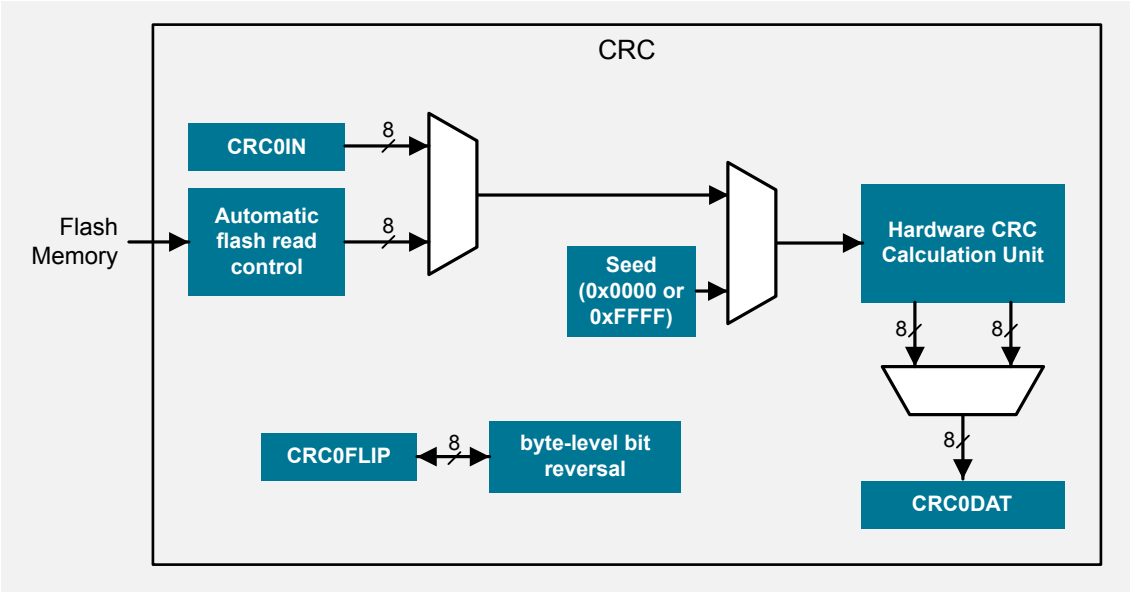


Figure 15.1. CRC Functional Block Diagram

15.2 Features

The CRC module is designed to provide hardware calculations for flash memory verification and communications protocols. The CRC module supports the standard CCITT-16 16-bit polynomial (0x1021), and includes the following features:

- Support for CCITT-16 polynomial
- Byte-level bit reversal
- Automatic CRC of flash contents on one or more 256-byte blocks
- Initial seed selection of 0x0000 or 0xFFFF

15.3 Functional Description

15.3.1 16-bit CRC Algorithm

The CRC unit generates a 16-bit CRC result equivalent to the following algorithm:

1. XOR the input with the most-significant bits of the current CRC result. If this is the first iteration of the CRC unit, the current CRC result will be the set initial value (0x0000 or 0xFFFF).
2. If the MSB of the CRC result is set, shift the CRC result and XOR the result with the polynomial.
3. If the MSB of the CRC result is not set, shift the CRC result.
4. Repeat steps 2 and 3 for all 8 bits.

The algorithm is also described in the following example.

```
unsigned short UpdateCRC (unsigned short CRC_acc, unsigned char CRC_input)
{
    unsigned char i; // loop counter
    #define POLY 0x1021

    // Create the CRC "dividend" for polynomial arithmetic (binary arithmetic
    // with no carries)
    CRC_acc = CRC_acc ^ (CRC_input << 8);

    // "Divide" the poly into the dividend using CRC XOR subtraction
    // CRC_acc holds the "remainder" of each divide
    //
    // Only complete this division for 8 bits since input is 1 byte
    for (i = 0; i < 8; i++)
    {
        // Check if the MSB is set (if MSB is 1, then the POLY can "divide"
        // into the "dividend")
        if ((CRC_acc & 0x8000) == 0x8000)
        {
            // if so, shift the CRC value, and XOR "subtract" the poly
            CRC_acc = CRC_acc << 1;
            CRC_acc ^= POLY;
        }
        else
        {
            // if not, just shift the CRC value
            CRC_acc = CRC_acc << 1;
        }
    }

    // Return the final remainder (CRC value)
    return CRC_acc;
}
```

The following table lists several input values and the associated outputs using the 16-bit CRC algorithm:

Table 15.1. Example 16-bit CRC Outputs

Input	Output
0x63	0xBD35
0x8C	0xB1F4
0x7D	0x4ECA
0xAA, 0xBB, 0xCC	0x6CF6
0x00, 0x00, 0xAA, 0xBB, 0xCC	0xB166

15.3.2 Using the CRC on a Data Stream

The CRC module may be used to perform CRC calculations on any data set available to the firmware. To perform a CRC on an arbitrary data stream:

1. Select the initial result value using CRCVAL.
2. Set the result to its initial value (write 1 to CRCINIT).
3. Write the data to CRC0IN one byte at a time. The CRC result registers are automatically updated after each byte is written.
4. Write the CRCPNT bit to 0 to target the low byte of the result.
5. Read CRC0DAT multiple times to access each byte of the CRC result. CRCPNT will automatically point to the next value after each read.

15.3.3 Using the CRC to Check Code Memory

The CRC module may be configured to automatically perform a CRC on one or more blocks of code memory. To perform a CRC on code contents:

1. Select the initial result value using CRCVAL.
2. Set the result to its initial value (write 1 to CRCINIT).
3. Write the high byte of the starting address to the CRCST bit field.
4. Set the AUTOEN bit to 1.
5. Write the number of byte blocks to perform in the CRC calculation to CRCCNT.
6. Write any value to CRC0CN0 (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes.

   Note: Upon initiation of an automatic CRC calculation, the three cycles following a write to CRC0CN0 that initiate a CRC operation must only contain instructions which execute in the same number of cycles as the number of bytes in the instruction. An example of such an instruction is a 3-byte MOV that targets the CRC0FLIP register. When programming in C, the dummy value written to CRC0FLIP should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.

7. Clear the AUTOEN.
8. Write the CRCPNT bit to 0 to target the low byte of the result.
9. Read CRC0DAT multiple times to access each byte of the CRC result. CRCPNT will automatically point to the next value after each read.

15.3.4 Bit Reversal

CRC0 includes hardware to reverse the bit order of each bit in a byte. Writing a byte to CRC0FLIP initiates the bit reversal operation, and the result may be read back from CRC0FLIP on the next instruction. For example, if 0xC0 is written to CRC0FLIP, the data read back is 0x03. Bit reversal can be used to change the order of information passing through the CRC engine and is also used in algorithms such as FFT.

15.4 CRC0 Control Registers

15.4.1 CRC0CN0: CRC0 Control 0

Bit	7	6	5	4	3	2	1	0
Name	Reserved				CRCINIT	CRCVAL	Reserved	CRCPNT
Access	R				W	RW	R	RW
Reset	0x1				0	0	0	0
SFR Page = 0x20; SFR Address: 0xCE								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	CRCINIT	0	W	**CRC Initialization Enable.** Writing a 1 to this bit initializes the entire CRC result based on CRCVAL. Always reads 0.
2	CRCVAL	0	RW	**CRC Initialization Value.** This bit selects the set value of the CRC result.

Value	Name	Description
0	SET_ZEROES	CRC result is set to 0x0000 on write of 1 to CRCINIT.
1	SET_ONES	CRC result is set to 0xFFFF on write of 1 to CRCINIT.

Bit	Name	Reset	Access	Description
1	*Reserved*	*Must write reset value.*		
0	CRCPNT	0	RW	**CRC Result Pointer.** Specifies the byte of the CRC result to be read/written on the next access to CRC0DAT. This bit will automatically toggle upon each read or write.

Value	Name	Description
0	ACCESS_LOWER	CRC0DAT accesses bits 7-0 of the 16-bit CRC result.
1	ACCESS_UPPER	CRC0DAT accesses bits 15-8 of the 16-bit CRC result.

Upon initiation of an automatic CRC calculation, the three cycles following a write to CRC0CN0 that initiate a CRC operation must only contain instructions which execute in the same number of cycles as the number of bytes in the instruction. An example of such an instruction is a 3-byte MOV that targets the CRC0FLIP register. When programming in C, the dummy value written to CRC0FLIP should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.

15.4.2 CRC0IN: CRC0 Data Input

Bit	7	6	5	4	3	2	1	0
Name	CRC0IN							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xCA								

Bit	Name	Reset	Access	Description
7:0	CRC0IN	0x00	RW	**CRC Data Input.** Each write to CRC0IN results in the written data being computed into the existing CRC result according to the CRC algorithm.

15.4.3 CRC0DAT: CRC0 Data Output

Bit	7	6	5	4	3	2	1	0
Name	CRC0DAT							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xCB								

Bit	Name	Reset	Access	Description
7:0	CRC0DAT	0x00	RW	**CRC Data Output.** Each read or write performed on CRC0DAT targets the CRC result bits pointed to by the CRC0 Result Pointer (CRCPNT bits in CRC0CN0).
CRC0DAT may not be valid for one cycle after setting the CRCINIT bit in the CRC0CN0 register to 1. Any time CRCINIT is written to 1 by firmware, at least one instruction should be performed before reading CRC0DAT.				

15.4.4 CRC0ST: CRC0 Automatic Flash Sector Start

Bit	7	6	5	4	3	2	1	0
Name	CRCST							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xD2								

Bit	Name	Reset	Access	Description
7:0	CRCST	0x00	RW	**Automatic CRC Calculation Starting Block.** These bits specify the flash block to start the automatic CRC calculation. The starting address of the first flash block included in the automatic CRC calculation is CRCST x block_size, where block_size is 256 bytes.

15.4.5 CRC0CNT: CRC0 Automatic Flash Sector Count

Bit	7	6	5	4	3	2	1	0
Name	CRCCNT							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xD3								

Bit	Name	Reset	Access	Description
7:0	CRCCNT	0x00	RW	**Automatic CRC Calculation Block Count.** These bits specify the number of flash blocks to include in an automatic CRC calculation. The last address of the last flash block included in the automatic CRC calculation is (CRCST+CRCCNT) x Block Size - 1. The block size is 256 bytes.

15.4.6 CRC0FLIP: CRC0 Bit Flip

Bit	7	6	5	4	3	2	1	0
Name	CRC0FLIP							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xCF								

Bit	Name	Reset	Access	Description
7:0	CRC0FLIP	0x00	RW	**CRC0 Bit Flip.** Any byte written to CRC0FLIP is read back in a bit-reversed order, i.e., the written LSB becomes the MSB. For example: If 0xC0 is written to CRC0FLIP, the data read back will be 0x03. If 0x05 is written to CRC0FLIP, the data read back will be 0xA0.

15.4.7 CRC0CN1: CRC0 Control 1

Bit	7	6	5	4	3	2	1	0
Name	AUTOEN	CRCDN	Reserved					
Access	RW	R	R					
Reset	0	1	0x00					
SFR Page = 0x20; SFR Address: 0x86								

Bit	Name	Reset	Access	Description
7	AUTOEN	0	RW	**Automatic CRC Calculation Enable.** When AUTOEN is set to 1, any write to CRC0CN0 will initiate an automatic CRC starting at flash sector CRCST and continuing for CRCCNT sectors.
6	CRCDN	1	R	**Automatic CRC Calculation Complete.** Set to 0 when a CRC calculation is in progress. Code execution is stopped during a CRC calculation; therefore, reads from firmware will always return 1.
5:0	*Reserved*	*Must write reset value.*		

16. Digital to Analog Converters (DAC0, DAC1, DAC2, DAC3)

16.1 Introduction

Up to four 12-bit voltage-output DACs are included. The DACs are fully static, requiring no clocking to maintain their output voltage. The DAC outputs may be updated by a variety of hardware and firmware trigger sources. The DAC output buffers are capable of producing near rail-to-rail output voltages when driving high load resistances, and they can accommodate load resistances as low as 1 kΩ across a narrower output voltage range. Each DAC may be configured to maintain its output state during a system reset.

The DACs are arranged in two identical pairs, with DAC0 and DAC1 comprising one pair and DAC2 and DAC3 comprising the other. The two DACs within a pair share the same reference buffer but are otherwise independent, with individual data inputs, trigger sources, and driver gains. The DAC pairs include special modes which link the two DACs together, enabling features such as complementary output waveform generation and resolution-enhancing interpolation to be performed in hardware.

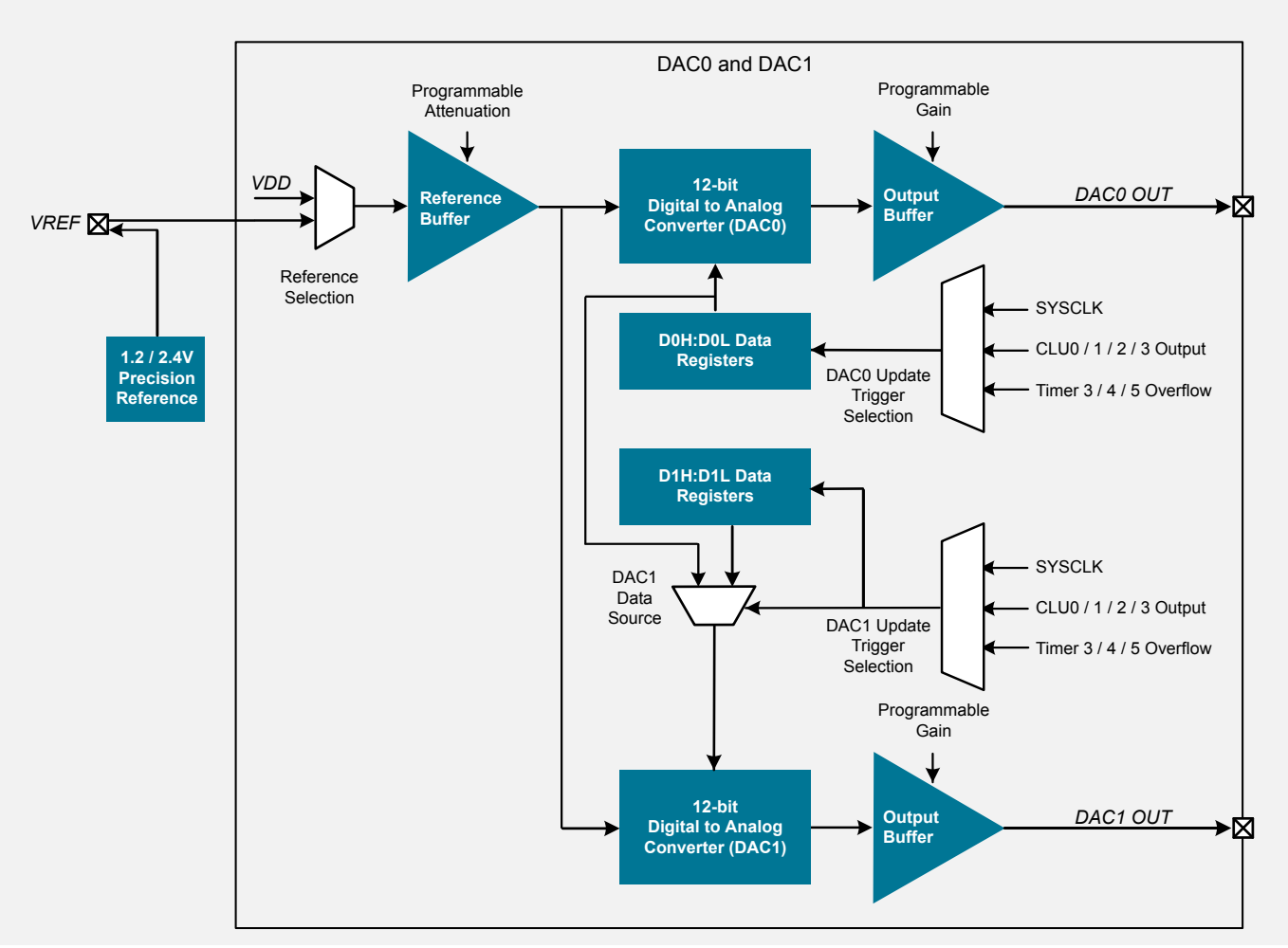


Figure 16.1. DAC Pair Block Diagram

16.2 Features

- Voltage output with 12-bit performance
- Supports an update rate of 200 ksps
- Hardware conversion trigger, selectable between software, external I/O and internal timer and configurable logic sources
- Outputs may be configured to persist through reset and maintain output state to avoid system disruption
- Multiple DAC outputs can be synchronized together
- DAC pairs (DAC0 and 1 or DAC2 and 3) support complementary output waveform generation
- Outputs may be switched between two levels according to state of configurable logic / PWM input trigger
- Flexible input data formatting
- Supports references from internal supply, on-chip precision reference, or external VREF pin

16.3 Functional Description

16.3.1 Enabling the DACs

Each of the DACs has an enable bit (bit EN in register DACnCF0) which turns on the DAC and allows its output buffer to drive its respective pin. The enable bit only affects the analog circuitry and pin interface associated with the DAC; the registers may be accessed by firmware even if the associated DAC is disabled. In the paired operating modes, the second DAC of the pair can make use of both DACs data registers while the first DAC is left disabled.

By default, any system reset will disable the DACs and reset all associated registers. By setting the Reset Mode bit (RSTMD) in register DACnCF0, the output of the DAC will persist through any system reset except for a power-on reset.

16.3.2 Reference and Output Configuration

The full-scale output voltage of each DAC is determined by the reference voltage, the gain of the reference buffer, and the gain of the output driver. The overall gain of the DAC is the full-scale output voltage divided by the reference voltage. In most cases this overall gain will be unity, but other gain values are possible with appropriate choice of reference voltage and register settings.

16.3.2.1 Reference Selection

Each DAC pair is supplied by a voltage reference buffer having selectable input sources. The DAC01REFSL bit selects the reference input source for DAC0 and DAC1, and the DAC23REFSL bit in DACGCF0 selects the reference input source for DAC2 and DAC3. The input source may be the VDD supply or the VREF pin. The VREF pin may be driven from an external reference or from the internal Precision Reference (VREF0), which can provide references of 1.2V or 2.4V. Details on the available options for driving the VREF pin may be found in the Precision Reference chapter.

16.3.2.2 Reference and Output Buffer Gains

For best operation, the DACs require reference buffer output voltages between 0.6V and 1.2V. The reference buffers each have three programmable attenuation values that allow a wide range of reference voltages to map to this range of output voltages. The attenuation value for the DAC0 and DAC1 reference buffer is selected by the D01REFGN field in DACGCF2, while the value for the DAC2 and DAC3 reference buffer is selected by the D23REFGN field in DACGCF2.

The DAC output buffers each have three programmable gain values that are the reciprocals of the three reference buffer attenuation values, meaning that an overall gain of unity may be programmed with any of the reference buffer attenuation settings. The output buffer gain for DACn is selected using the DRVGAIN field in the DACnCF1 register. Non-unity gain values may also be realized by programming the buffer and reference gains to values which are not reciprocals of one another. Note that the minimum and maximum DAC output voltages are limited by the supply rails.

Table 16.1 DAC Reference and Gain Settings on page 217 provides a summary of the reference voltage ranges, reference buffer attenuation settings, output buffer gain settings, and the resulting overall gain.

Table 16.1. DAC Reference and Gain Settings

Reference Range	DxxREFGN setting	Reference Buffer Attenuation	DRVGAIN setting	Output Buffer Gain	Overall Gain
1.15V to 1.8V	0x0	Low (Gain = 1/2)	0x0	Low (Gain = 2)	1
			0x1	Medium (Gain = 2.4)	1.2
			0x2	High (Gain = 3)	1.5
1.6V to 2.55V	0x1	Medium (Gain = 1/2.4)	0x0	Low (Gain = 2)	0.8
			0x1	Medium (Gain = 2.4)	1
			0x2	High (Gain = 3)	1.2
2.2V to 3.6V	0x2	High (Gain = 1/3)	0x0	Low (Gain = 2)	0.667
			0x1	Medium (Gain = 2.4)	0.833
			0x2	High (Gain = 3)	1

16.3.3 Input Data and Update Triggers

Each DAC includes a low byte data register (DACnL) and a high byte data register (DACnH). By default, the 12 data bits are right-justified, meaning that the four MSBs are located in the lower four bits of DACnH and the eight LSBs are located in DACnL. By setting the LJST bit in the DACnCF0 register, the input data is left-justified, meaning that the eight MSBs are in DACnH and the four LSBs are in the upper four bits of DACnL. When updating the DACn input, DACnL must always be written first. Writing to DACnL will inhibit trigger events in order to prevent a DAC update when only part of the data is present. A write to DACnH will uninhibit trigger events and allow an update.

Each DAC may be updated by a variety of trigger sources specified in the UPDATE field in register DACnCF0. The trigger source options are the same for all of the DACs, but the trigger source for each DAC is specified independently. A trigger source of SYSCLK means that the DACn output will update on the SYSCLK cycle following a write to DACnH. Other trigger source options are the high byte overflows of TIMER3, TIMER4, and TIMER5, and a rising edge on the output of any of the four Configurable Logic Units.

The DACGCF1 register provides additional control over the updating of the DAC outputs. This register includes four DACn Update Disable (DnUDIS) bits that can independently disable all update triggers for each of the four DACs. These bits allow firmware to update two or more DAC outputs on the same clock edge without configuring timers or Configurable Logic Units to perform the triggering. Firmware can set DnUDIS bits to disable DAC updates, write values to multiple DAC input data registers, and then update all of the DAC outputs on the same clock edge by clearing the appropriate DnUDIS bits.

16.3.4 Paired Operating Modes

The DACGCF0 register includes bits that allow alternate sources for the input data applied to DAC1. The DAC1 Data Source field (D1SRC) allows four different data sources for DAC1: the data in DAC1H:DAC1L (the default), the one's complement of the value in DAC1H:DAC1L, the data in DAC0H:DAC0L, and the one's complement of the value in DAC0H:DAC0L. This allows firmware to program DAC0H:DAC0L and see the value reflected on the DAC0 and DAC1 outputs, with DAC1 producing either the same voltage as DAC0 or its complement. The DAC3 Data Source field (D3SRC) operates in a similar fashion with DAC2 and DAC3.

The DAC1 Alternating Mode Enable bit (D1AMEN) provides additional options for generating DAC1 input data. When D1AMEN is cleared, DAC1 always updates from the data source selected by the D1SRC field. When D1AMEN is set, the data source is determined by the logic state of the DAC1 trigger source. When the DAC1 trigger is logic low, then DAC1 uses the data in DAC1H:DAC1L. When the DAC1 trigger is logic high, DAC1 uses the data source specified by D1SRC. When using the Alternating Mode feature, the trigger source must be one of the Configurable Logic Units, and the minimum high and low times for this trigger must be at least two system clock cycles. The DAC3 Alternating Mode Enable bit (D3AMEN) operates in a similar fashion with DAC3.

Because of the highly flexible nature of trigger signals provided by the Configurable Logic module, the Alternating Mode feature may be used to create a wide variety of different waveforms. For example, the DAC0 and DAC1 data registers may be written with adjacent digital values, and an 8-bit PWM signal from PCA0 may be routed through a Configurable Logic unit to serve as the trigger source for DAC1. With this configuration, the DAC1 output voltage will assume an average value representing an interpolation between the two adjacent digital values, thus extending the resolution of the DAC to a theoretical 20 bits. In another example, the DAC0 and DAC1 data registers may be written with digital values representing high and low logic levels, and the serial output of a digital peripheral (e.g. a UART or SMBus/I2C) may be routed through the Configurable Logic module to the DAC1 trigger so that DAC1 reproduces the serial output with arbitrary, programmable high and low levels. Note that in many cases utilizing the Alternating Mode the first DAC in the pair (DAC0 or DAC2) should be left disabled, since the desired output is produced by the second DAC in the pair.

16.4 DAC Control Registers

16.4.1 DACGCF0: DAC Global Configuration 0

Bit	7	6	5	4	3	2	1	0
Name	D23REFSL	D3AMEN	D3SRC		D01REFSL	D1AMEN	D1SRC	
Access	RW	RW	RW		RW	RW	RW	
Reset	1	0	0x0		1	0	0x0	

SFR Page = 0x30; SFR Address: 0x88 (bit-addressable)

Bit	Name	Reset	Access	Description
7	D23REFSL	1	RW	**DAC2 and DAC3 Reference Voltage Select.**

Determines the voltage reference for DAC2 and DAC3.

Value	Name	Description
0	VREF	Select the VREF pin.
1	VDD	Select the VDD supply.

Bit	Name	Reset	Access	Description
6	D3AMEN	0	RW	**DAC3 Alternating Mode Enable.**

Value	Name	Description
0	NORMAL	DAC3 always updates from the data source selected in D3SRC. This mode may be used with any trigger source.
1	ALTERNATE	DAC3 updates occur on the rising or falling edge of the trigger signal. On a falling edge, DAC3 receives the DAC3H/L registers. On a rising edge, DAC3 receives the data source selected in D3SRC. **This mode may only be used with Configurable Logic trigger sources, and the selected trigger source must be high or low for two or more SYSCLK cycles.**

Bit	Name	Reset	Access	Description
5:4	D3SRC	0x0	RW	**DAC3 Data Source.**

Selects the DAC3 data source input.

Value	Name	Description
0x0	DAC3	Input = DAC3H:DAC3L.
0x1	DAC3_INVERT	Input = Inverse of DAC3H:DAC3L (one's complement).
0x2	DAC2	Input = DAC2H:DAC2L.
0x3	DAC2_INVERT	Input = Inverse of DAC2H:DAC2L (one's complement).

Bit	Name	Reset	Access	Description
3	D01REFSL	1	RW	**DAC0 and DAC1 Reference Voltage Select.**

Determines the voltage reference for DAC0 and DAC1.

Value	Name	Description
0	VREF	Select the VREF pin.
1	VDD	Select the VDD supply.

Bit	Name	Reset	Access	Description
2	D1AMEN	0	RW	**DAC1 Alternating Mode Enable.**
	Value	Name		Description
	0	NORMAL		DAC1 always updates from the data source selected in D1SRC. This mode may be used with any trigger source.
	1	ALTERNATE		DAC1 updates occur on the rising or falling edge of the trigger signal. On a falling edge, DAC1 receives the DAC1H/L registers. On a rising edge, DAC1 receives the data source selected in D1SRC. **This mode may only be used with Configurable Logic trigger sources, and the selected trigger source must be high or low for two or more SYSCLK cycles.**
1:0	D1SRC	0x0	RW	**DAC1 Data Source.** Selects the DAC1 data source input.
	Value	Name		Description
	0x0	DAC1		Input = DAC1H:DAC1L.
	0x1	DAC1_INVERT		Input = Inverse of DAC1H:DAC1L (one's complement).
	0x2	DAC0		Input = DAC0H:DAC0L.
	0x3	DAC0_INVERT		Input = Inverse of DAC0H:DAC0L (one's complement).

16.4.2 DACGCF1: DAC Global Configuration 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved				D3UDIS	D2UDIS	D1UDIS	D0UDIS
Access	R				RW	RW	RW	RW
Reset	0x0				0	0	0	0
SFR Page = 0x30; SFR Address: 0x98 (bit-addressable)								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	D3UDIS	0	RW	**DAC3 Update Disable.**
	Value	Name		Description
	0	ENABLE		Allow triggers to update DAC3 output.
	1	DISABLE		Triggers will not update DAC3 output.
2	D2UDIS	0	RW	**DAC2 Update Disable.**
	Value	Name		Description
	0	ENABLE		Allow triggers to update DAC2 output.
	1	DISABLE		Triggers will not update DAC2 output.
1	D1UDIS	0	RW	**DAC1 Update Disable.**
	Value	Name		Description
	0	ENABLE		Allow triggers to update DAC1 output.
	1	DISABLE		Triggers will not update DAC1 output.
0	D0UDIS	0	RW	**DAC0 Update Disable.**
	Value	Name		Description
	0	ENABLE		Allow triggers to update DAC0 output.
	1	DISABLE		Triggers will not update DAC0 output.

16.4.3 DACGCF2: DAC Global Configuration 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved		D23REFGN		Reserved		D01REFGN	
Access	R		RW		R		RW	
Reset	0x0		0x1		0x0		0x1	
SFR Page = 0x30; SFR Address: 0xA2								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5:4	D23REFGN	0x1	RW	**DAC2/3 Reference Buffer Gain.** Selects the gain applied to the reference buffer.

Value	Name	Description
0x0	ATTEN_2P0	Selected reference will be attenuated by a factor of 2.
0x1	ATTEN_2P4	Selected reference will be attenuated by a factor of 2.4 (Gain = 1/2.4).
0x2	ATTEN_3P0	Selected reference will be attenuated by a factor of 3 (Gain = 1/3).

Bit	Name	Reset	Access	Description
3:2	*Reserved*	*Must write reset value.*		
1:0	D01REFGN	0x1	RW	**DAC0/1 Reference Buffer Gain.** Selects the gain applied to the reference buffer.

Value	Name	Description
0x0	ATTEN_2P0	Selected reference will be attenuated by a factor of 2.
0x1	ATTEN_2P4	Selected reference will be attenuated by a factor of 2.4 (Gain = 1/2.4).
0x2	ATTEN_3P0	Selected reference will be attenuated by a factor of 3 (Gain = 1/3).

16.4.4 DAC0CF0: DAC0 Configuration 0

Bit	7	6	5	4	3	2	1	0
Name	EN	RSTMD	LJST	Reserved		UPDATE		
Access	RW	RW	RW	R		RW		
Reset	0	0	0	0x0		0x0		

SFR Page = 0x30; SFR Address: 0x91

Bit	Name	Reset	Access	Description
7	EN	0	RW	**DAC0 Enable.**
	Enables the DAC0 Output.			
	Value	Name		Description
	0	DISABLE		DAC0 is disabled and not driven at the output pin.
	1	ENABLE		DAC0 is enabled and will drive the output pin.
6	RSTMD	0	RW	**DAC0 Reset Mode.**
	Enables "persist through reset" output capability for DAC0.			
	Value	Name		Description
	0	NORMAL		All resets will reset DAC0 and its associated registers.
	1	PERSIST		DAC0 output will persist through all resets except for power-on-resets.
5	LJST	0	RW	**DAC0 Left Justify Enable.**
	Value	Name		Description
	0	RIGHT_JUSTIFY		DAC0 input is treated as right-justified.
	1	LEFT_JUSTIFY		DAC0 input is treated as left-justified.
4:3	*Reserved*	*Must write reset value.*		
2:0	UPDATE	0x0	RW	**DAC0 Update Trigger Source.**
	Selects the trigger source for DAC0 output updates (when updates are not inhibited).			
	Value	Name		Description
	0x0	SYSCLK		DAC0 output updates occur on every clock cycle.
	0x1	TIMER3		DAC0 output updates occur on Timer 3 high byte overflow.
	0x2	TIMER4		DAC0 output updates occur on Timer 4 high byte overflow.
	0x3	TIMER5		DAC0 output updates occur on Timer 5 high byte overflow.
	0x4	CLU0		DAC0 output updates occur on Configurable Logic output 0 rising edge.
	0x5	CLU1		DAC0 output updates occur on Configurable Logic output 1 rising edge.
	0x6	CLU2		DAC0 output updates occur on Configurable Logic output 2 rising edge.
	0x7	CLU3		DAC0 output updates occur on Configurable Logic output 3 rising edge.

16.4.5 DAC0CF1: DAC0 Configuration 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved						DRVGAIN	
Access	R						RW	
Reset	0x00						0x1	
SFR Page = 0x30; SFR Address: 0x92								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	DRVGAIN	0x1	RW	**DAC0 Output Buffer Gain.** Selects the gain to be applied to the DAC output buffer. This field is used in conjunction with the reference gain selection in DACGCF2 to determine the full-scale output voltage of the DAC. The full-scale output of the DAC will be equal to the selected reference voltage, multiplied by the reference gain and the output buffer gain. Typically, both parameters will be set to the same value to produce a full-scale output of 1.0 x VREF.

Value	Name	Description
0x0	GAIN_2P0	DAC output gain is 2.
0x1	GAIN_2P4	DAC output gain is 2.4.
0x2	GAIN_3P0	DAC output gain is 3.

16.4.6 DAC0L: DAC0 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC0L							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x84								

Bit	Name	Reset	Access	Description
7:0	DAC0L	0x00	RW	**Data Word Low Byte.** Low byte of data input. Trigger events are inhibited upon writing to DAC0L and uninhibited when writing to DAC0H to prevent unintentional updates. When updating the DAC input, DAC0L should always be written before DAC0H.

16.4.7 DAC0H: DAC0 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC0H							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x85								

Bit	Name	Reset	Access	Description
7:0	DAC0H	0x00	RW	**Data Word High Byte.** High byte of data input. Trigger events are inhibited upon writing to DAC0L and uninhibited when writing to DAC0H to prevent unintentional updates. When updating the DAC input, DAC0L should always be written before DAC0H.

16.4.8 DAC1CF0: DAC1 Configuration 0

Bit	7	6	5	4	3	2	1	0
Name	EN	RSTMD	LJST	Reserved		UPDATE		
Access	RW	RW	RW	R		RW		
Reset	0	0	0	0x0		0x0		

SFR Page = 0x30; SFR Address: 0x93

Bit	Name	Reset	Access	Description
7	EN	0	RW	**DAC1 Enable.**
	Enables the DAC1 Output.			
	Value	Name		Description
	0	DISABLE		DAC1 is disabled and not driven at the output pin.
	1	ENABLE		DAC1 is enabled and will drive the output pin.
6	RSTMD	0	RW	**DAC1 Reset Mode.**
	Enables "persist through reset" output capability for DAC1.			
	Value	Name		Description
	0	NORMAL		All resets will reset DAC1 and its associated registers.
	1	PERSIST		DAC1 output will persist through all resets except for power-on-resets.
5	LJST	0	RW	**DAC1 Left Justify Enable.**
	Value	Name		Description
	0	RIGHT_JUSTIFY		DAC1 input is treated as right-justified.
	1	LEFT_JUSTIFY		DAC1 input is treated as left-justified.
4:3	*Reserved*	*Must write reset value.*		
2:0	UPDATE	0x0	RW	**DAC1 Update Trigger Source.**
	Selects the trigger source for DAC1 output updates (when updates are not inhibited).			
	Value	Name		Description
	0x0	SYSCLK		DAC1 output updates occur on every clock cycle.
	0x1	TIMER3		DAC1 output updates occur on Timer 3 high byte overflow.
	0x2	TIMER4		DAC1 output updates occur on Timer 4 high byte overflow.
	0x3	TIMER5		DAC1 output updates occur on Timer 5 high byte overflow.
	0x4	CLU0		DAC1 output updates occur on Configurable Logic output 0 rising edge.
	0x5	CLU1		DAC1 output updates occur on Configurable Logic output 1 rising edge.
	0x6	CLU2		DAC1 output updates occur on Configurable Logic output 2 rising edge.
	0x7	CLU3		DAC1 output updates occur on Configurable Logic output 3 rising edge.

16.4.9 DAC1CF1: DAC1 Configuration 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved						DRVGAIN	
Access	R						RW	
Reset	0x00						0x1	
SFR Page = 0x30; SFR Address: 0x94								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	DRVGAIN	0x1	RW	**DAC1 Output Buffer Gain.** Selects the gain to be applied to the DAC output buffer. This field is used in conjunction with the reference gain selection in DACGCF2 to determine the full-scale output voltage of the DAC. The full-scale output of the DAC will be equal to the selected reference voltage, multiplied by the reference gain and the output buffer gain. Typically, both parameters will be set to the same value to produce a full-scale output of 1.0 x VREF.

Value	Name	Description
0x0	GAIN_2P0	DAC output gain is 2.
0x1	GAIN_2P4	DAC output gain is 2.4.
0x2	GAIN_3P0	DAC output gain is 3.

16.4.10 DAC1L: DAC1 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC1L							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x89								

Bit	Name	Reset	Access	Description
7:0	DAC1L	0x00	RW	**Data Word Low Byte.** Low byte of data input. Trigger events are inhibited upon writing to DAC1L and uninhibited when writing to DAC1H to prevent unintentional updates. When updating the DAC input, DAC1L should always be written before DAC1H.

16.4.11 DAC1H: DAC1 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC1H							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x8A								

Bit	Name	Reset	Access	Description
7:0	DAC1H	0x00	RW	**Data Word High Byte.** High byte of data input. Trigger events are inhibited upon writing to DAC1L and uninhibited when writing to DAC1H to prevent unintentional updates. When updating the DAC input, DAC1L should always be written before DAC1H.

16.4.12 DAC2CF0: DAC2 Configuration 0

Bit	7	6	5	4	3	2	1	0
Name	EN	RSTMD	LJST	Reserved		UPDATE		
Access	RW	RW	RW	R		RW		
Reset	0	0	0	0x0		0x0		
SFR Page = 0x30; SFR Address: 0x95								

Bit	Name	Reset	Access	Description
7	EN	0	RW	**DAC2 Enable.**
	Enables the DAC2 Output.			
	Value	Name		Description
	0	DISABLE		DAC2 is disabled and not driven at the output pin.
	1	ENABLE		DAC2 is enabled and will drive the output pin.
6	RSTMD	0	RW	**DAC2 Reset Mode.**
	Enables "persist through reset" output capability for DAC2.			
	Value	Name		Description
	0	NORMAL		All resets will reset DAC2 and its associated registers.
	1	PERSIST		DAC2 output will persist through all resets except for power-on-resets.
5	LJST	0	RW	**DAC2 Left Justify Enable.**
	Value	Name		Description
	0	RIGHT_JUSTIFY		DAC2 input is treated as right-justified.
	1	LEFT_JUSTIFY		DAC2 input is treated as left-justified.
4:3	*Reserved*	*Must write reset value.*		
2:0	UPDATE	0x0	RW	**DAC2 Update Trigger Source.**
	Selects the trigger source for DAC2 output updates (when updates are not inhibited).			
	Value	Name		Description
	0x0	SYSCLK		DAC2 output updates occur on every clock cycle.
	0x1	TIMER3		DAC2 output updates occur on Timer 3 high byte overflow.
	0x2	TIMER4		DAC2 output updates occur on Timer 4 high byte overflow.
	0x3	TIMER5		DAC2 output updates occur on Timer 5 high byte overflow.
	0x4	CLU0		DAC2 output updates occur on Configurable Logic output 0 rising edge.
	0x5	CLU1		DAC2 output updates occur on Configurable Logic output 1 rising edge.
	0x6	CLU2		DAC2 output updates occur on Configurable Logic output 2 rising edge.
	0x7	CLU3		DAC2 output updates occur on Configurable Logic output 3 rising edge.

16.4.13 DAC2CF1: DAC2 Configuration 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved						DRVGAIN	
Access	R						RW	
Reset	0x00						0x1	
SFR Page = 0x30; SFR Address: 0x96								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	DRVGAIN	0x1	RW	**DAC2 Output Buffer Gain.** Selects the gain to be applied to the DAC output buffer. This field is used in conjunction with the reference gain selection in DACGCF2 to determine the full-scale output voltage of the DAC. The full-scale output of the DAC will be equal to the selected reference voltage, multiplied by the reference gain and the output buffer gain. Typically, both parameters will be set to the same value to produce a full-scale output of 1.0 x VREF.

Value	Name	Description
0x0	GAIN_2P0	DAC output gain is 2.
0x1	GAIN_2P4	DAC output gain is 2.4.
0x2	GAIN_3P0	DAC output gain is 3.

16.4.14 DAC2L: DAC2 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC2L							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x8B								

Bit	Name	Reset	Access	Description
7:0	DAC2L	0x00	RW	**Data Word Low Byte.** Low byte of data input. Trigger events are inhibited upon writing to DAC2L and uninhibited when writing to DAC2H to prevent unintentional updates. When updating the DAC input, DAC2L should always be written before DAC2H.

16.4.15 DAC2H: DAC2 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC2H							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x8C								

Bit	Name	Reset	Access	Description
7:0	DAC2H	0x00	RW	**Data Word High Byte.** High byte of data input. Trigger events are inhibited upon writing to DAC2L and uninhibited when writing to DAC2H to prevent unintentional updates. When updating the DAC input, DAC2L should always be written before DAC2H.

16.4.16 DAC3CF0: DAC3 Configuration 0

Bit	7	6	5	4	3	2	1	0
Name	EN	RSTMD	LJST	Reserved		UPDATE		
Access	RW	RW	RW	R		RW		
Reset	0	0	0	0x0		0x0		
SFR Page = 0x30; SFR Address: 0x9A								

Bit	Name	Reset	Access	Description
7	EN	0	RW	**DAC3 Enable.**
	Enables the DAC3 Output.			
	Value	Name		Description
	0	DISABLE		DAC3 is disabled and not driven at the output pin.
	1	ENABLE		DAC3 is enabled and will drive the output pin.
6	RSTMD	0	RW	**DAC3 Reset Mode.**
	Enables "persist through reset" output capability for DAC3.			
	Value	Name		Description
	0	NORMAL		All resets will reset DAC3 and its associated registers.
	1	PERSIST		DAC3 output will persist through all resets except for power-on-resets.
5	LJST	0	RW	**DAC3 Left Justify Enable.**
	Value	Name		Description
	0	RIGHT_JUSTIFY		DAC3 input is treated as right-justified.
	1	LEFT_JUSTIFY		DAC3 input is treated as left-justified.
4:3	*Reserved*	*Must write reset value.*		
2:0	UPDATE	0x0	RW	**DAC3 Update Trigger Source.**
	Selects the trigger source for DAC3 output updates (when updates are not inhibited).			
	Value	Name		Description
	0x0	SYSCLK		DAC3 output updates occur on every clock cycle.
	0x1	TIMER3		DAC3 output updates occur on Timer 3 high byte overflow.
	0x2	TIMER4		DAC3 output updates occur on Timer 4 high byte overflow.
	0x3	TIMER5		DAC3 output updates occur on Timer 5 high byte overflow.
	0x4	CLU0		DAC3 output updates occur on Configurable Logic output 0 rising edge.
	0x5	CLU1		DAC3 output updates occur on Configurable Logic output 1 rising edge.
	0x6	CLU2		DAC3 output updates occur on Configurable Logic output 2 rising edge.
	0x7	CLU3		DAC3 output updates occur on Configurable Logic output 3 rising edge.

16.4.17 DAC3CF1: DAC3 Configuration 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved						DRVGAIN	
Access	R						RW	
Reset	0x00						0x1	
SFR Page = 0x30; SFR Address: 0x9C								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	DRVGAIN	0x1	RW	**DAC3 Output Buffer Gain.** Selects the gain to be applied to the DAC output buffer. This field is used in conjunction with the reference gain selection in DACGCF2 to determine the full-scale output voltage of the DAC. The full-scale output of the DAC will be equal to the selected reference voltage, multiplied by the reference gain and the output buffer gain. Typically, both parameters will be set to the same value to produce a full-scale output of 1.0 x VREF.

Value	Name	Description
0x0	GAIN_2P0	DAC output gain is 2.
0x1	GAIN_2P4	DAC output gain is 2.4.
0x2	GAIN_3P0	DAC output gain is 3.

16.4.18 DAC3L: DAC3 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC3L							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x8D								

Bit	Name	Reset	Access	Description
7:0	DAC3L	0x00	RW	**Data Word Low Byte.** Low byte of data input. Trigger events are inhibited upon writing to DAC3L and uninhibited when writing to DAC3H to prevent unintentional updates. When updating the DAC input, DAC3L should always be written before DAC3H.

16.4.19 DAC3H: DAC3 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	DAC3H							
Access	RW							
Reset	0x00							
SFR Page = 0x30; SFR Address: 0x8E								

Bit	Name	Reset	Access	Description
7:0	DAC3H	0x00	RW	**Data Word High Byte.** High byte of data input. Trigger events are inhibited upon writing to DAC3L and uninhibited when writing to DAC3H to prevent unintentional updates. When updating the DAC input, DAC3L should always be written before DAC3H.

17. I2C Slave (I2CSLAVE0)

17.1 Introduction

The I2C Slave interface is a 2-wire, bidirectional serial bus that is compatible with the I2C Bus Specification 3.0. It is capable of transferring in high-speed mode (HS-mode) at speeds of up to 3.4 Mbps. Firmware can write to the I2C interface, and the I2C interface can autonomously control the serial transfer of data. The interface also supports clock stretching for cases where the core may be temporarily prohibited from transmitting a byte or processing a received byte during an I2C transaction. It can also operate in low power modes without an active system clock and wake the core when a matching slave address is received.

This module operates only as an I2C slave device. The I2C Slave peripheral provides control of the SCL (serial clock) synchronization, SDA (serial data), SCL clock stretching, I2C arbitration logic, and low power mode operation.

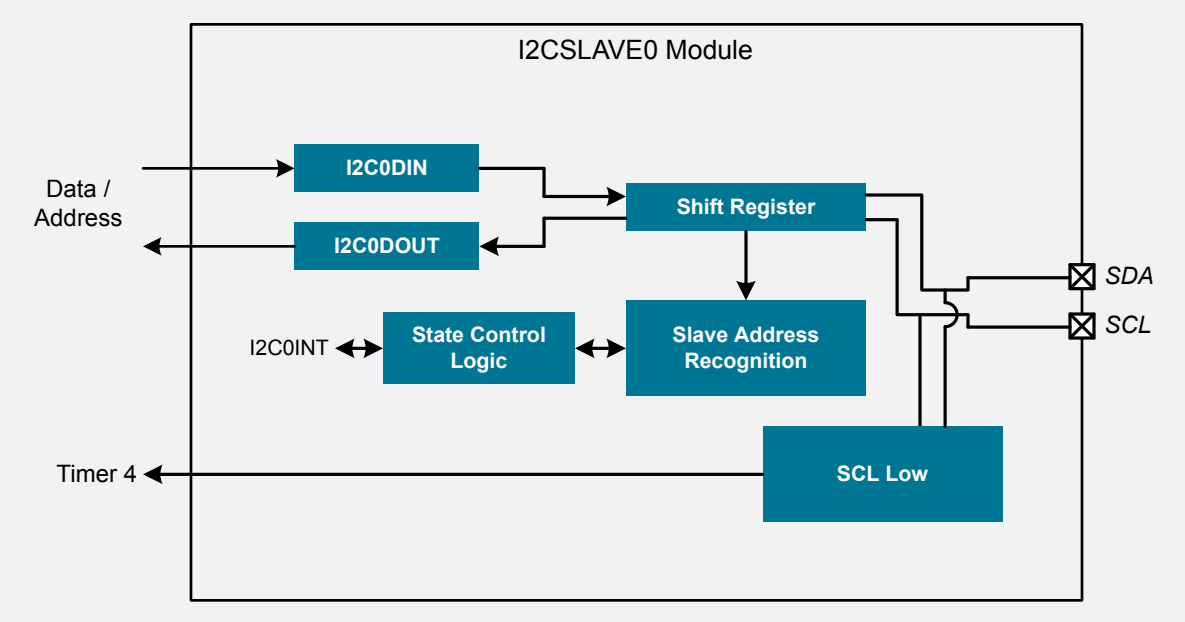


Figure 17.1. I2CSLAVE0 Block Diagram

17.2 Features

The I2C module includes the following features:

- Standard (up to 100 kbps), Fast (400 kbps), Fast Plus (1 Mbps), and High-speed (3.4 Mbps) transfer speeds
- Support for slave mode only
- Clock low extending (clock stretching) to interface with faster masters
- Hardware support for 7-bit slave address recognition
- Transmit and receive FIFOs (two byte) to help increase throughput in faster applications

17.3 Functional Description

17.3.1 Overview

The I2C Slave module operates only in slave mode. The hardware provides timing and shifting control for serial transfers; the higher level protocol is determined by user software. The I2C hardware interface provides the following application-independent features:

- Byte-wise serial data transfers
- SDA data synchronization
- Timeout recognition, as defined by the I2C0CNTL configuration register
- START/STOP detection
- Interrupt generation
- Status information
- High-speed I2C mode detection
- Automatic wakeup from lower power modes when a matching slave address is received
- Hardware recognition of the slave address and automatic acknowledgment of address/data

An I2CSLAVE0 interrupt is generated when the RD, WR or STOP bit is set in the I2C0STAT register. It is also generated when the ACTIVE bit goes low to indicate the end of an I2C bus transfer. Refer to the I2C0STAT register definition for complete details on the conditions for the setting and clearing of these bits.

17.3.2 I2C Protocol

The I2C specification allows any recessive voltage between 3.0 and 5.0 V; different devices on the bus may operate at different voltage levels. However, the maximum voltage on any port pin must conform to the electrical characteristics specifications. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free.

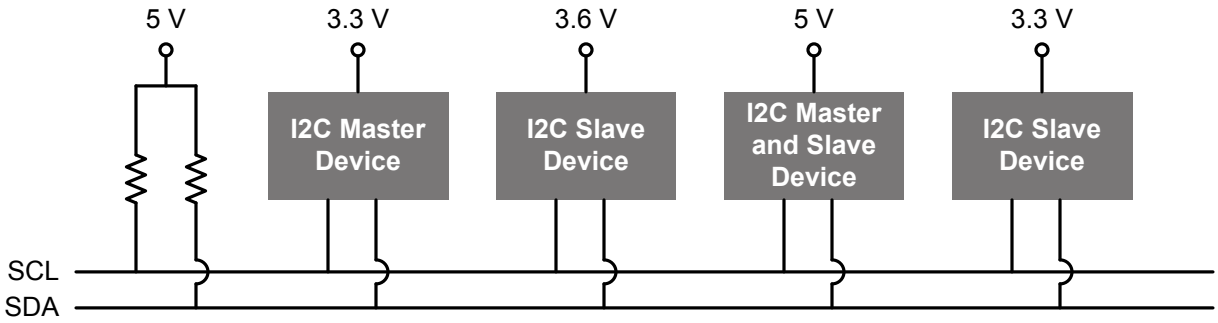


Figure 17.2. Typical I2C System Connection

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE) and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The I2C interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. It is not necessary to specify one device as the Master in a system; any device who transmits a START and a slave address becomes the master for the duration of that transfer.

A typical I2C transaction consists of a START condition followed by an address byte (Bits 7–1: 7-bit slave address; Bit 0: R/W direction bit), one or more bytes of data, and a STOP condition. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 17.3 I2C Transaction on page 237). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address byte. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation.

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 17.3 I2C Transaction on page 237 illustrates a typical I2C transaction.

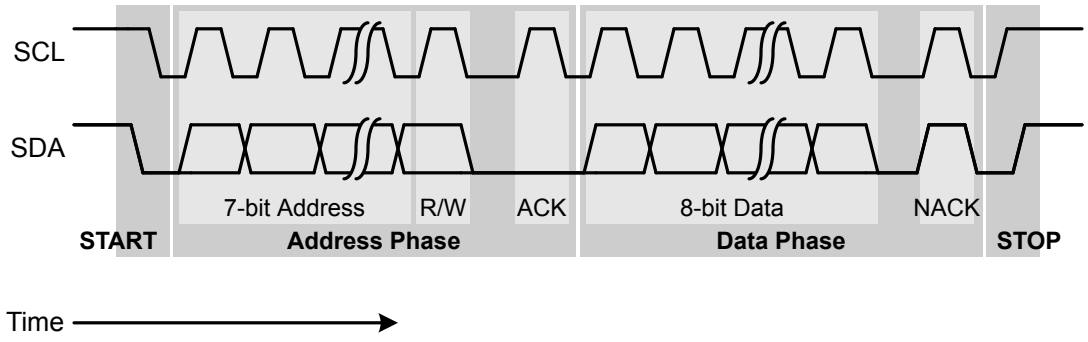


Figure 17.3. I2C Transaction

Transmitter vs. Receiver

On the I2C communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

Clock Low Extension

I2C provides a clock synchronization mechanism which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

In the I2C Slave peripheral, clock stretching is only performed on the SCL falling edge associated with the ACK or NACK bit. Clock stretching is always performed on every byte transaction that is addressed to the peripheral. Clock stretching is completed by the I2CSLAVE0 peripheral when it releases the SCL line from the low state. The I2CSLAVE0 peripheral releases the SCL line when firmware writes a 0 to the I2C0INT bit in the I2C0STAT register.

SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the I2C protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

For the I2C Slave interface, an on-chip timer is used to implement SCL low timeouts. The SCL low timeout feature is enabled by setting the TIMEOUT bit in I2C0CN0. The associated timer is forced to reload when SCL is high, and allowed to count when SCL is low. With the associated timer enabled and configured to overflow after 25 ms (and TIMEOUT set), the timer interrupt service routine can be used to reset (disable and re-enable) the I2C module in the event of an SCL low timeout.

High-Speed Mode

The I2C specification supports High-speed mode (HS-mode) transfers, which allow devices to transfer data at rates of up to 3.4 Mbps and remain fully downward compatible with slower speed devices. This allows HS-mode devices to operate in a mixed-speed bus system. Refer to the I2C Specification for details on the electrical and timing requirements for HS-mode operation. The I2CSLAVE0 peripheral is compatible with the I2C HS-mode operation without any firmware intervention other than requiring that firmware enable the I2CSLAVE0 peripheral.

By default, the I2C bus operates at speeds of up to Fast-mode (F/S mode) only, where the maximum transfer rate is 400 kbps. The I2C bus switches to from F/S mode to HS-mode only after the following sequence of bits appear on the I2C bus:

1. START bit (S)
2. 8-bit master code (0000 1XXX)
3. NACK bit (N)

The HS-mode master codes are reserved 8-bit codes which are not used for slave addressing or other purposes. An HS-mode compatible I2C master device will switch the I2C bus to HS-mode by transmitting the above sequence of bits on the I2C bus at a transfer rate of not more than 400 kbps. After that, the master can switch to HS-mode to transfer data at a rate of up to 3.4 Mbps. The I2C bus switches back to F/S mode when the I2C master transmits a STOP bit.

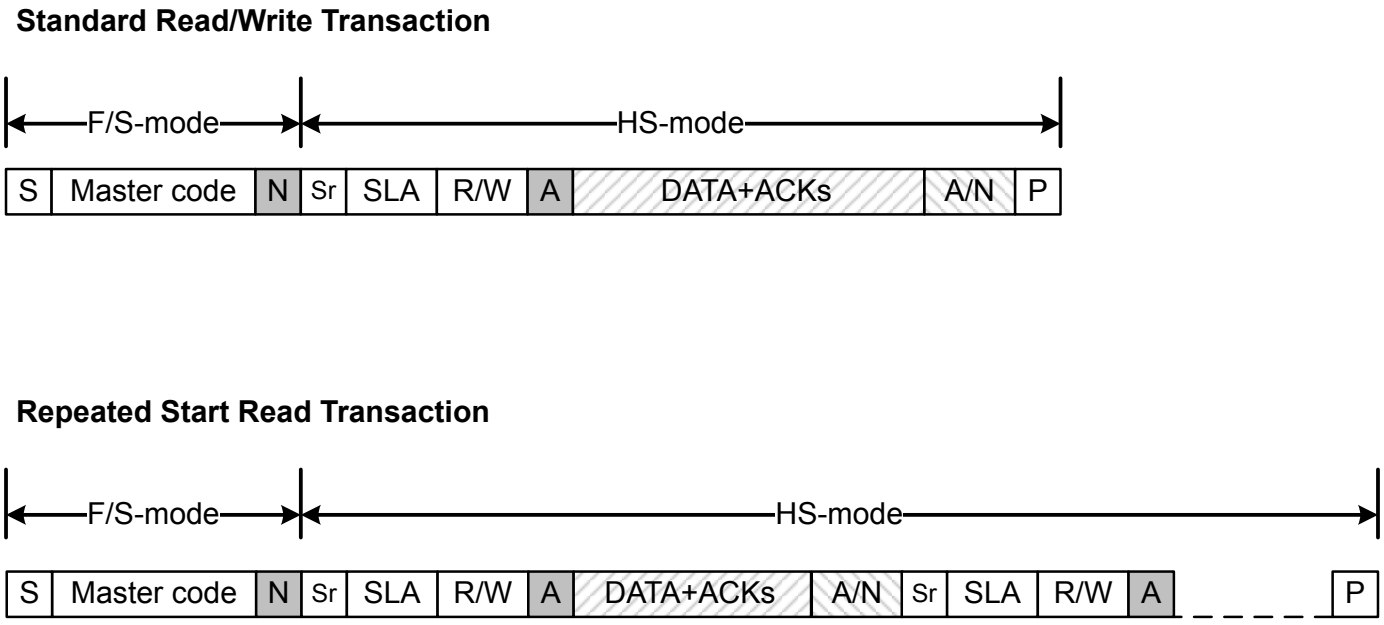


Figure 17.4. Fast-Mode to High-Speed Mode Transition

17.3.3 Automatic Address Recognition

The I2CSLAVE0 peripheral can be configured to recognize a specific slave address and respond with an ACK without any software intervention. This feature is enabled by firmware:

1. Clear BUSY bit in I2C0CN0 to enable automatic ACK response.
2. Write the slave address to I2C0ADM.
3. Set the PINMD bit in I2C0CN0 to 1 to enable the SCL and SDA pins.
4. Set the I2C0EN bit in I2C0CN0 to 1 to enable the I2CSLAVE0 peripheral.

The Slave Address Mask (SLVM in the I2C0ADM register) can be used to define an address mask to enable automatic hardware response to multiple slave addresses. Additionally, if the ADDRCHK bit is set in the I2C0CN0 register, the matching address will be placed in the receive FIFO, allowing firmware to check which address was used to initiate the transaction. In this case, firmware should read the address from the receive FIFO using the I2C0DIN register before proceeding with the data transfer.

17.3.4 Operational Modes

The I2C Slave peripheral supports two types of data transfers: I2C Read data transfers where data is transferred from the I2C Slave peripheral to an I2C master, and I2C Write data transfers where data is transferred from an I2C master to the I2C Slave peripheral. The I2C master initiates both types of data transfers and provides the serial clock pulses that the I2C slave peripheral detects on the SCL pin. This section describes in detail the setting and clearing of various status bits in the I2C0STAT register during different modes of operations. In all modes, the I2CSLAVE0 peripheral performs clock stretching automatically on every SCL falling edge associated with the ACK or NACK bit.

I2C Write Sequence

The I2C Write sequence with the I2C Slave peripheral consists of a series of interrupts and required actions in each interrupt. The write sequence consists of the following steps:

1. An incoming START and Address + W byte causes the peripheral to exit idle mode or wakes the device from a low power state. The peripheral will automatically ACK a matching address if BUSY is cleared to 0.
2. An interrupt occurs after the automatic ACK of the address. The I2C peripheral holds the SCL line low for clock streching until firmware clears I2C0INT. Firmware should take the actions indicated by Figure 17.6 I2C Write Flow Diagram with the I2C Slave Peripheral on page 242.
3. Firmware reads one or more bytes of data from the master on each subsequent data interrupt, acknowledging (ACK) or non-acknowledging (NACK) the data.
4. The master sends a STOP when the entire data transfer completes.

Figure 17.5 Example I2C Write Sequence with the I2C Slave Peripheral on page 241 demonstrates an example sequence, including a repeated start, and Figure 17.6 I2C Write Flow Diagram with the I2C Slave Peripheral on page 242 describes the I2C Write sequence and firmware actions in each interrupt.

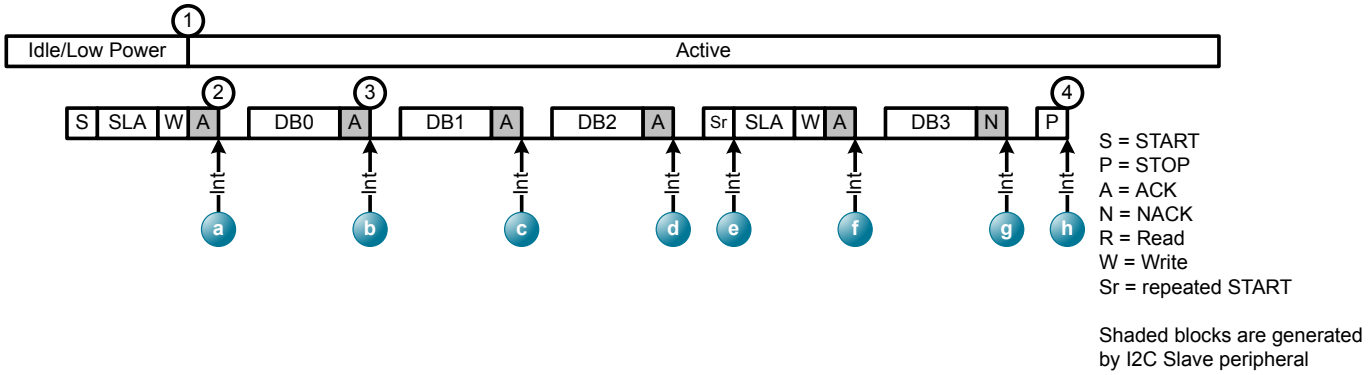


Figure 17.5. Example I2C Write Sequence with the I2C Slave Peripheral

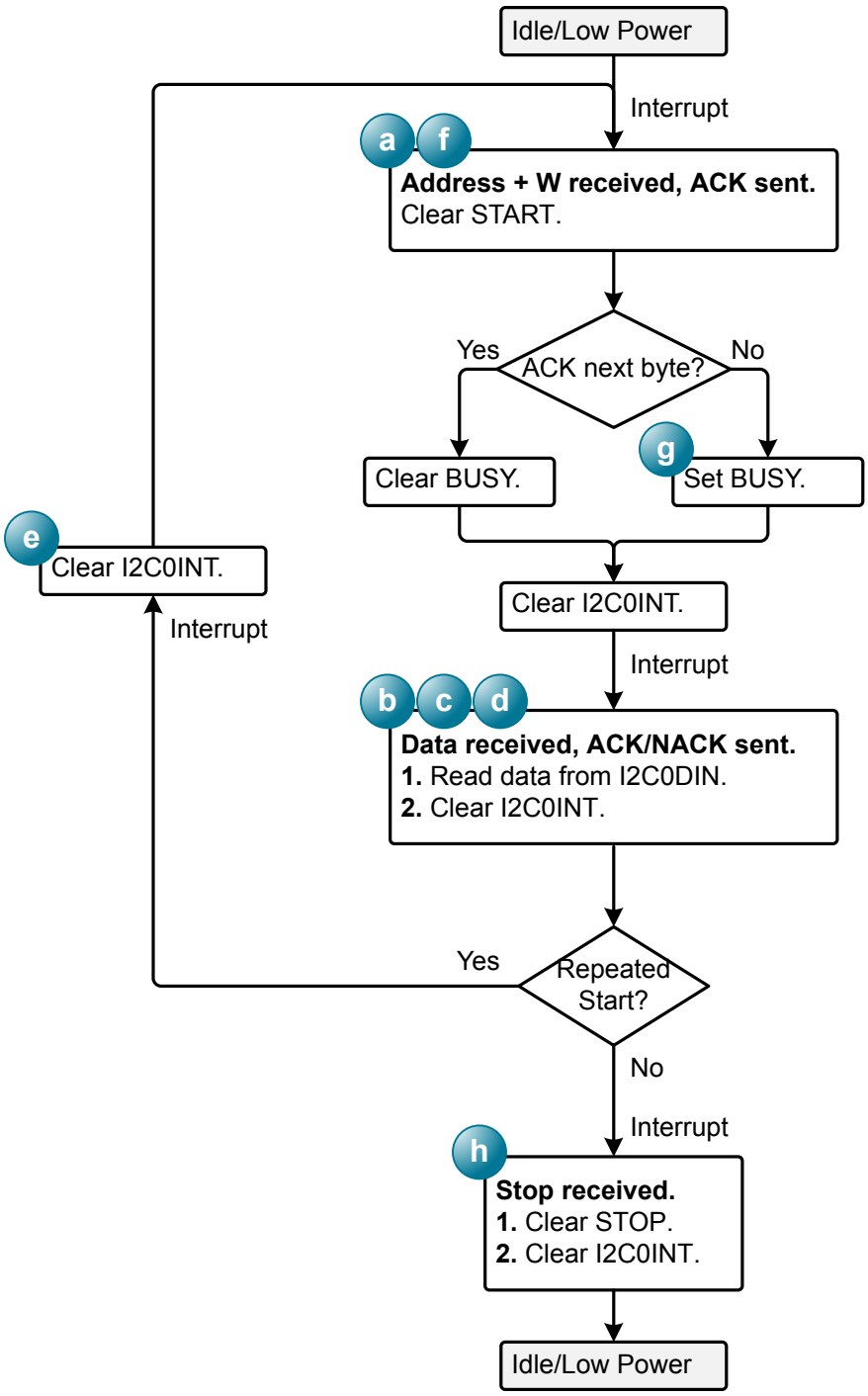


Figure 17.6. I2C Write Flow Diagram with the I2C Slave Peripheral

Note: Firmware can leave the BUSY bit as 0 in step F in the Figure 17.5 Example I2C Write Sequence with the I2C Slave Peripheral on page 241 sequence. In this case, the master will receive an ACK instead at step G could still generate a STOP bit immediately after the ACK.

I2C Read Sequence

The I2C Read sequence with the I2C Slave peripheral consists of a series of interrupts and required actions in each interrupt. The read sequence consists of the following steps:

1. An incoming START and Address + R byte causes the peripheral to exit idle mode or wakes the device from a low power state. The peripheral will automatically ACK a matching address if BUSY is cleared to 0.
2. An interrupt occurs after the automatic ACK of the address. The I2C peripheral holds the SCL line low for clock streching until firmware clears I2C0INT. Firmware should read the data from the master and take the actions indicated by Figure 17.8 I2C Read Flow Diagram with the I2C Slave Peripheral on page 244.
3. Firmware writes one or more bytes of data to the master on each subsequent data interrupt.
4. The master sends a NACKwhen the current data transfer completes and either a repeated START or STOP.
5. The master sends a STOP when the entire data transfer completes.

Figure 17.7 Example I2C Read Sequence with the I2C Slave Peripheral on page 243 demonstrates an example sequence, including a repeated start, and Figure 17.8 I2C Read Flow Diagram with the I2C Slave Peripheral on page 244 describes the I2C Read sequence and firmware actions in each interrupt.

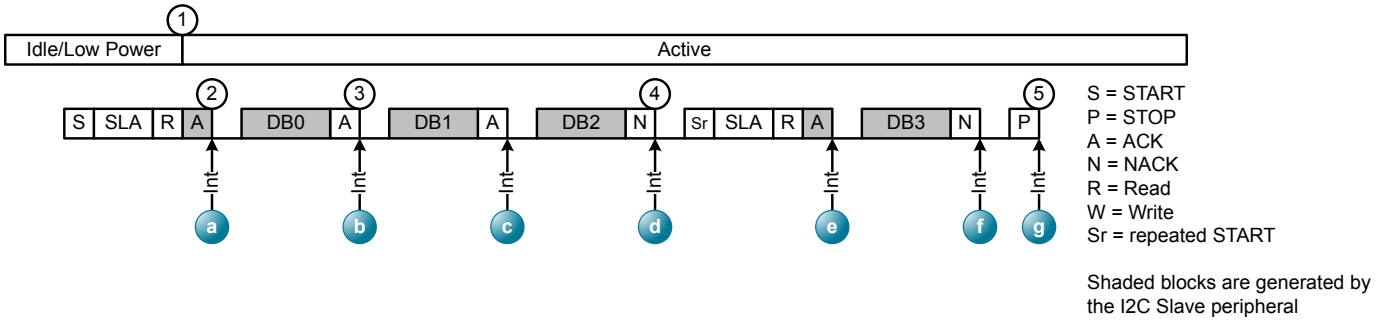


Figure 17.7. Example I2C Read Sequence with the I2C Slave Peripheral

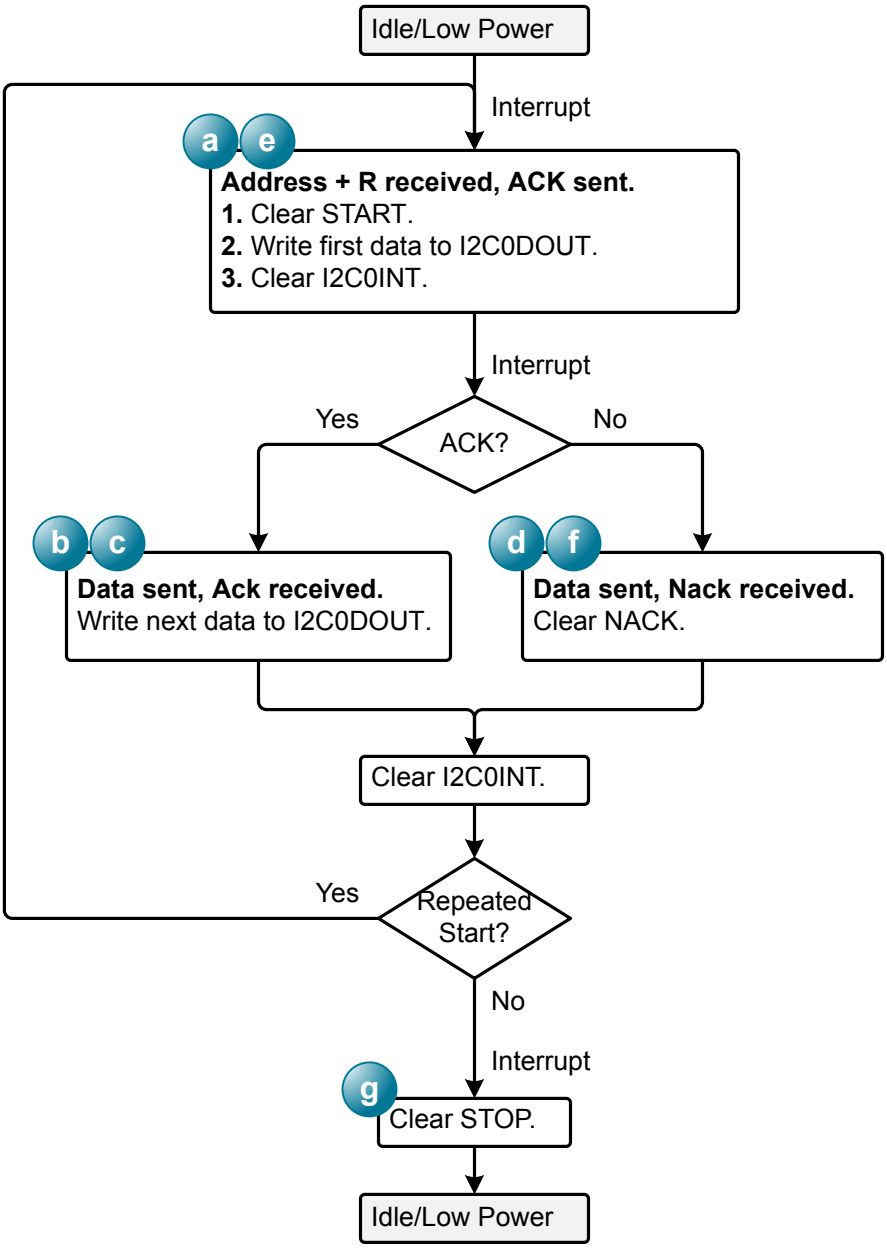


Figure 17.8. I2C Read Flow Diagram with the I2C Slave Peripheral

Note: The I2C master must always generate a NACK before it can generate a repeated START bit or a STOP bit. This NACK causes I2C Slave peripheral to release the SDA line for the I2C master to generate the START or STOP bit.

17.3.5 Status Decoding

The current I2C status can be easily decoded using the I2C0STAT register. Table 17.1 I2C Status Decoding on page 245 describes the typical actions firmware should take in each state. In the tables, STATUS VECTOR refers to the lower five bits of I2C0STAT: NACK, START, STOP, WR, and RD. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the I2C specification.

Note: Interrupts from multiple sources (STOP, START, RD, WR, etc.) can accumulate, so the actual status vector may have additional conditions set and not match the value in the table below. In these cases, the order of operations should be:

1. Service the STOP bit.
2. Service the START bit.
3. Service the START + RD or START + WR bits.
4. Service the RD or WR bits.

Table 17.1. I2C Status Decoding

Mode	Current Status Vector	Current I2C State	Expected Actions	Next Status Vector Expected
Write (Master to Slave)	01010	START + Address + W received, ACK sent	Clear START and I2C0INT.	00010
	00010	Data byte received, ACK sent	Read data from I2C0DIN and clear I2C0INT. Set BUSY to NACK the next byte or keep BUSY clear to ACK the next byte.	00010 or 10010 or 00100
	10010	Data byte received, NACK sent	Read data from I2C0DIN and clear I2C0INT. Clear BUSY to ACK the next byte or keep BUSY set to NACK the next byte.	00010 or 10010 or 00100
	00000	Repeated Start	Clear I2C0INT.	01010
	00100	STOP received	Clear STOP and I2C0INT.	
Read (Slave to Master)	01001	START + Address + R received, ACK sent	Clear START, write data to I2C0DOUT, and clear I2C0INT.	00001
	00001	Data byte sent, master ACK received	Write data to I2C0DOUT and clear I2C0INT.	00100
	10001	Data byte sent, master NACK received	Clear NACK and I2C0INT.	00100
	00100	STOP received	Clear STOP and I2C0INT	

17.4 I2C0 Slave Control Registers

17.4.1 I2C0DIN: I2C0 Received Data

Bit	7	6	5	4	3	2	1	0
Name	I2C0DIN							
Access	R							
Reset	Varies							
SFR Page = 0x20; SFR Address: 0xBC								

Bit	Name	Reset	Access	Description
7:0	I2C0DIN	Varies	R	**I2C0 Received Data.** Reading this register reads any data received from the RX FIFO. I2C0DIN may be read until RXE is set to 1, indicating there is no more data in the RX FIFO. If this register is read when RXE is set to 1, the last byte in the RX FIFO is returned.

17.4.2 I2C0DOUT: I2C0 Transmit Data

Bit	7	6	5	4	3	2	1	0
Name	I2C0DOUT							
Access	RW							
Reset	Varies							
SFR Page = 0x20; SFR Address: 0xBB								

Bit	Name	Reset	Access	Description
7:0	I2C0DOUT	Varies	RW	**I2C0 Transmit Data.** Writing this register writes a byte into the TX FIFO. I2C0DOUT may be written when TXNF is set to 1, which indicates that there is more room available in the TX FIFO. If this register is written when TXNF is cleared to 0, the most recent byte written to the TX FIFO will be overwritten.

17.4.3 I2C0SLAD: I2C0 Slave Address

Bit	7	6	5	4	3	2	1	0
Name	Reserved	I2C0SLAD						
Access	RW	RW						
Reset	0	0x00						
SFR Page = 0x20; SFR Address: 0xBD								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:0	I2C0SLAD	0x00	RW	**I2C Hardware Slave Address.** This field, combined with the SLVM mask in I2C0ADM, defines the I2C0 Slave Address for automatic hardware acknowledgement.

17.4.4 I2C0STAT: I2C0 Status

Bit	7	6	5	4	3	2	1	0
Name	HSMODE	ACTIVE	I2C0INT	NACK	START	STOP	WR	RD
Access	R	R	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x20; SFR Address: 0xB9								

Bit	Name	Reset	Access	Description
7	HSMODE	0	R	**High Speed Mode.** This bit is set to 1 by hardware when a High Speed master code is received and automatically clears when a STOP event occurs.
6	ACTIVE	0	R	**Bus Active.** This bit is set to 1 by hardware when an incoming slave address matches and automatically clears when the transfer completes with either a STOP or a NACK event.
5	I2C0INT	0	RW	**I2C Interrupt.** This bit is set when a read (RD), write (WR), or a STOP event (STOP) occurs. This bit will also set when the ACTIVE bit goes low to indicate the end of a transfer. This bit will generate an interrupt, and hardware will automatically clear this bit after the RD and WR bits clear.
4	NACK	0	RW	**NACK.** This bit is set by hardware when one of the following conditions are met: - A NACK is transmitted by either a Master or a Slave when the ACTIVE bit is high. - An I2C slave transmits a NACK to a matching slave address. Hardware will automatically clear this bit.
3	START	0	RW	**Start.** This bit is set by hardware when a START is received and a matching slave address is received. Software must clear this bit.
2	STOP	0	RW	**Stop.** This bit is set by hardware when a STOP is received and the last slave address received was a match. Software must clear this bit.
1	WR	0	RW	**I2C Write.** This bit is set by hardware on the 9th SCL falling edge when one of the following conditions are met: - The I2C0 Slave responds with an ACK, and the RX FIFO is full. - The I2C0 Slave responds with a NACK, and the RX FIFO is full. - The current byte transaction has a matching I2C0 Slave address, the 8th bit was a WRITE bit (0), and FACS is set to 1. This bit will set the I2C0INT bit and generate an interrupt, if enabled. Software must clear this bit.
0	RD	0	RW	**I2C Read.** This bit is set by hardware on the 9th SCL falling edge when one of the following conditions are met: - The I2C Master responds with an ACK (data or address), and there is no more data in the TX FIFO. - I2C Master responds with a NACK. - The current byte transaction has a matching I2C slave address, the 8th bit was a READ bit (1), and FACS is set to 1. This bit will set the I2C0INT bit and generate an interrupt, if enabled. Software must clear this bit.

17.4.5 I2C0CN0: I2C0 Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved	ADDRCHK	PINDRV	PINMD	TIMEOUT	PRELOAD	I2C0EN	BUSY
Access	R	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	1	0	1
SFR Page = 0x20; SFR Address: 0xBA								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	ADDRCHK	0	RW	**Address Check Enable.**

Setting this bit to 1 writes the matching slave address into the receive FIFO, enabling firmware to determine which address caused the match interrupt.

Value	Name	Description
0	DISABLED	The matching slave address is not copied into the receive FIFO.
1	ENABLED	The matching slave address is copied into the receive FIFO.

5	PINDRV	0	RW	**Pin Drive Strength.**

When this bit is set, the SCL and SDA pins will use high drive strength to drive low. When cleared, the pins will use low drive strength. This overrides the drive strength setting for the I/O port.

Value	Name	Description
0	LOW_DRIVE	SDA and SCL will use low drive strength.
1	HIGH_DRIVE	SDA and SCL will use high drive strength.

4	PINMD	0	RW	**Pin Mode Enable.**

Value	Name	Description
0	GPIO_MODE	Set the I2C0 Slave pins in GPIO mode.
1	I2C_MODE	Set the I2C0 Slave pins in I2C mode.

3	TIMEOUT	0	RW	**SCL Low Timeout Enable.**

When this bit is set and RLFSEL in TMR4CN1 is set correctly, Timer 4 will start counting only when SCL is low. When SCL is high, Timer 4 will auto-reload with the value from the reload registers. If Timer 4 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL is high. The Timer 4 interrupt service routine should reset I2C communication.

Value	Name	Description
0	DISABLED	Disable I2C SCL low timeout detection using Timer 4.
1	ENABLED	Enable I2C SCL low timeout detection using Timer 4 if Timer 4 RLFSEL is set appropriately.

2	PRELOAD	1	RW	**Preload Disable.**

Value	Name	Description
0	ENABLED	Data bytes must be written into the TX FIFO via the I2C0DOUT register before the 8th SCL clock of the matching slave address byte transfer arrives for an I2C read operation.

Bit	Name	Reset	Access	Description
	1	DISABLED		Data bytes need not be preloaded for I2C read operations. The data byte can be written to I2C0DOUT during interrupt servicing.
1	I2C0EN	0	RW	**I2C Enable.**
	This bit enables the I2C0 Slave module. PINMD must be enabled first before this bit is enabled.			
	Value	Name		Description
	0	DISABLED		Disable the I2C0 Slave module.
	1	ENABLED		Enable the I2C0 Slave module.
0	BUSY	1	RW	**Busy.**
	Value	Name		Description
	0	NOT_SET		Device will acknowledge an I2C master.
	1	SET		Device will not respond to an I2C master. All I2C data sent to the device will be NACKed.

17.4.6 I2C0FCN0: I2C0 FIFO Control 0

Bit	7	6	5	4	3	2	1	0
Name	TFRQE	TFLSH	TXTH		RFRQE	RFLSH	RXTH	
Access	RW	RW	RW		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x20; SFR Address: 0xAD								

Bit	Name	Reset	Access	Description
7	TFRQE	0	RW	**Write Request Interrupt Enable.**
	When set to 1, an I2C0 interrupt will be generated any time TFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		I2C0 interrupts will not be generated when TFRQ is set.
	1	ENABLED		I2C0 interrupts will be generated if TFRQ is set.
6	TFLSH	0	RW	**TX FIFO Flush.**
	This bit flushes the TX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will not be sent. Hardware will clear the TFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
5:4	TXTH	0x0	RW	**TX FIFO Threshold.**
	This field configures when hardware will set the transmit FIFO request bit (TFRQ). TFRQ is set whenever the number of bytes in the TX FIFO is equal to or less than the value in TXTH.			
	Value	Name		Description
	0x0	ZERO		TFRQ will be set when the TX FIFO is empty.
	0x1	ONE		TFRQ will be set when the TX FIFO contains one or fewer bytes.
3	RFRQE	0	RW	**Read Request Interrupt Enable.**
	When set to 1, an I2C0 interrupt will be generated any time RFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		I2C0 interrupts will not be generated when RFRQ is set.
	1	ENABLED		I2C0 interrupts will be generated if RFRQ is set.
2	RFLSH	0	RW	**RX FIFO Flush.**
	This bit flushes the RX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will be lost. Hardware will clear the RFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
1:0	RXTH	0x0	RW	**RX FIFO Threshold.**
	This field configures when hardware will set the receive FIFO request bit (RFRQ). RFRQ is set whenever the number of bytes in the RX FIFO exceeds the value in RXTH.			
	Value	Name		Description
	0x0	ZERO		RFRQ will be set anytime new data arrives in the RX FIFO (when the RX FIFO is not empty).
	0x1	ONE		RFRQ will be set if the RX FIFO contains more than one byte.

17.4.7 I2C0FCN1: I2C0 FIFO Control 1

Bit	7	6	5	4	3	2	1	0
Name	TFRQ	TXNF	Reserved		RFRQ	RXE	Reserved	
Access	R	R	R		R	R	R	
Reset	1	1	0x0		0	1	0x0	

SFR Page = 0x20; SFR Address: 0xAB

Bit	Name	Reset	Access	Description
7	TFRQ	1	R	**Transmit FIFO Request.**
	Set to 1 by hardware when the number of bytes in the TX FIFO is less than or equal to the TX FIFO threshold (TXTH).			
	Value	Name		Description
	0	NOT_SET		The number of bytes in the TX FIFO is greater than TXTH.
	1	SET		The number of bytes in the TX FIFO is less than or equal to TXTH.
6	TXNF	1	R	**TX FIFO Not Full.**
	This bit indicates when the TX FIFO is full and can no longer be written to. If a write is performed when TXNF is cleared to 0 it will replace the most recent byte in the FIFO.			
	Value	Name		Description
	0	FULL		The TX FIFO is full.
	1	NOT_FULL		The TX FIFO has room for more data.
5:4	*Reserved*	*Must write reset value.*		
3	RFRQ	0	R	**Receive FIFO Request.**
	Set to 1 by hardware when the number of bytes in the RX FIFO is larger than specified by the RX FIFO threshold (RXTH).			
	Value	Name		Description
	0	NOT_SET		The number of bytes in the RX FIFO is less than or equal to RXTH.
	1	SET		The number of bytes in the RX FIFO is greater than RXTH.
2	RXE	1	R	**RX FIFO Empty.**
	This bit indicates when the RX FIFO is empty. If a read is performed when RXE is set, the last byte will be returned.			
	Value	Name		Description
	0	NOT_EMPTY		The RX FIFO contains data.
	1	EMPTY		The RX FIFO is empty.
1:0	*Reserved*	*Must write reset value.*		

17.4.8 I2C0FCT: I2C0 FIFO Count

Bit	7	6	5	4	3	2	1	0
Name	Reserved	TXCNT			Reserved	RXCNT		
Access	R	R			R	R		
Reset	0	0x0			0	0x0		
SFR Page = 0x20; SFR Address: 0xF5								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	TXCNT	0x0	R	**TX FIFO Count.** This field indicates the number of bytes in the transmit FIFO.
3	*Reserved*	*Must write reset value.*		
2:0	RXCNT	0x0	R	**RX FIFO Count.** This field indicates the number of bytes in the receive FIFO.

17.4.9 I2C0ADM: I2C0 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	FACS	SLVM						
Access	RW	RW						
Reset	1	0x7F						
SFR Page = 0x20; SFR Address: 0xFF								

Bit	Name	Reset	Access	Description
7	FACS	1	RW	**Force Address Clock Stretching.** This bit controls whether clock stretching is enforced (via setting of I2C0INT bit) after the a matching slave address has been acknowledged. When this bit is set, clock stretching always occurs after an ACK of the address byte until firmware clears the I2C0INT bit. When this bit is cleared, the I2C0INT bit won't be set by the address ACK alone, but it may be set due to other conditions as detailed in the descriptions of the RD and WR bits.

Value	Name	Description
0	DONT_FORCE_STRETCH	The I2C0INT bit is not set by acking the slave address alone. Additional conditions are required to set I2C0INT.
1	FORCE_STRETCH	The I2C0INT bit is always set when matching slave address is acknowledged. This will force clock stretching until firmware clears the I2C0INT bit.

6:0	SLVM	0x7F	RW	**I2C Hardware Slave Address.** This field defines which bits of register I2C0SLAD are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM enables comparison with the corresponding bit in I2C0SLAD. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).

18. Programmable Counter Array (PCA0)

18.1 Introduction

The programmable counter array (PCA) provides multiple channels of enhanced timer and PWM functionality while requiring less CPU intervention than standard counter/timers. The PCA consists of a dedicated 16-bit counter/timer and one 16-bit capture/compare module for each channel. The counter/timer is driven by a programmable timebase that has flexible external and internal clocking options. Each capture/compare module may be configured to operate independently in one of five modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, or Pulse-Width Modulated (PWM) Output. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the crossbar to port I/O when enabled.

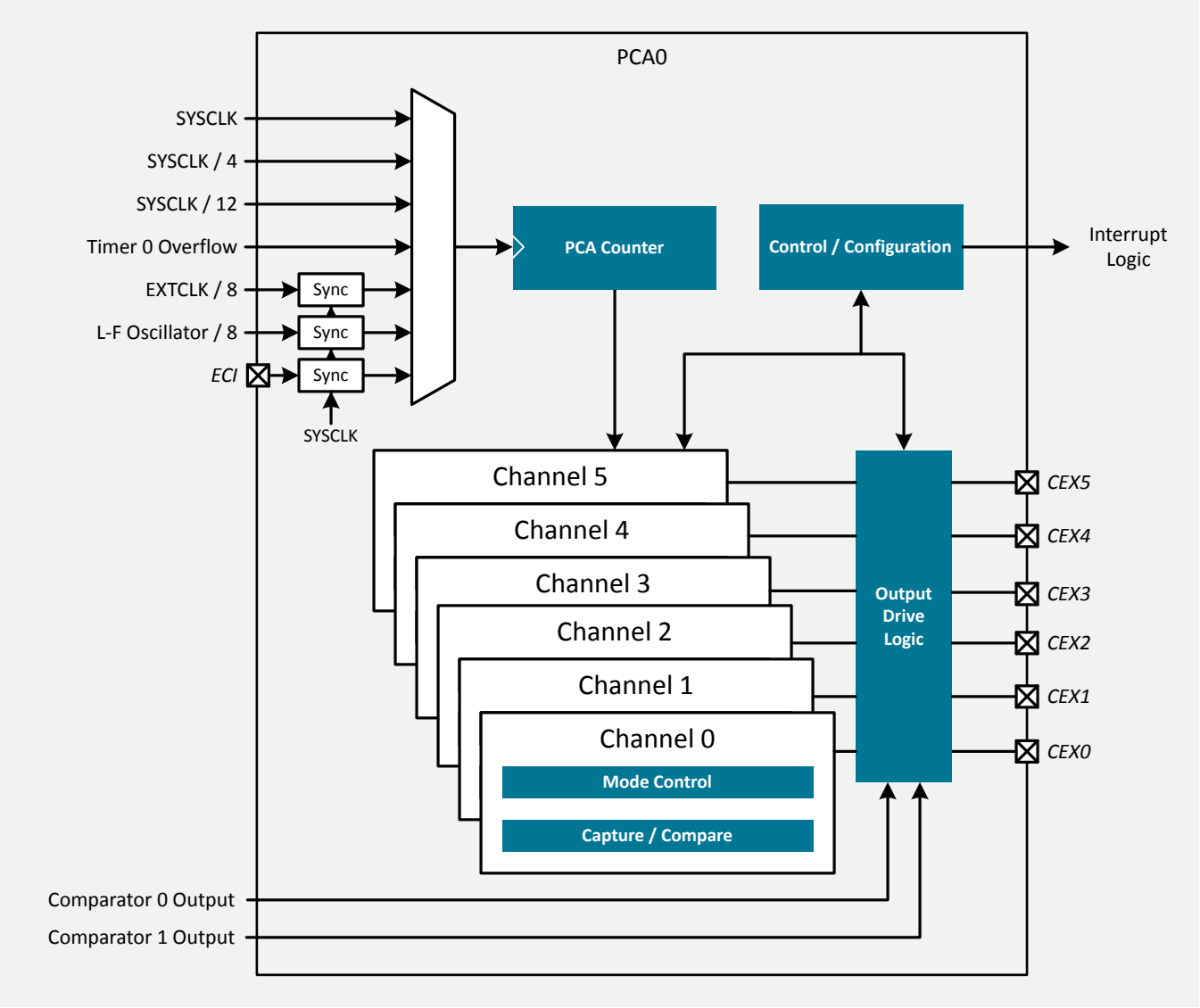


Figure 18.1. PCA Block Diagram

18.2 Features

- 16-bit time base
- Programmable clock divisor and clock source selection
- Up to six independently-configurable channels
- 8, 9, 10, 11 and 16-bit PWM modes (center or edge-aligned operation)
- Output polarity control
- Frequency output mode
- Capture on rising, falling or any edge
- Compare function for arbitrary waveform generation
- Software timer (internal compare) mode
- Can accept hardware “kill” signal from comparator 0 or comparator 1

18.3 Functional Description

18.3.1 Counter / Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte of the 16-bit counter/timer and PCA0L is the low byte. Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register.

Note: Reading the PCA0L Register first guarantees an accurate reading of the entire 16-bit PCA0 counter.

Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer.

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine and must be cleared by software. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

Table 18.1. PCA Timebase Input Options

CPS2:0	Timebase
000	System clock divided by 12
001	System clock divided by 4
010	Timer 0 overflow
011	High-to-low transitions on ECI (max rate = system clock divided by 4) [1]
100	System clock
101	External oscillator source divided by 8 [1]
110	Low frequency oscillator divided by 8 [1]
111	Reserved

Note:
1. Synchronized with the system clock.

18.3.2 Interrupt Sources

The PCA0 module shares one interrupt vector among all of its modules. There are are several event flags that can be used to generate a PCA0 interrupt. They are as follows: the main PCA counter overflow flag (CF), which is set upon a 16-bit overflow of the PCA0 counter; an intermediate overflow flag (COVF), which can be set on an overflow from the 8th–11th bit of the PCA0 counter; and the individual flags for each PCA channel (CCFn), which are set according to the operation mode of that module. These event flags are always set when the trigger condition occurs. Each of these flags can be individually selected to generate a PCA0 interrupt using the corresponding interrupt enable flag (ECF for CF, ECOV for COVF, and ECCFn for each CCFn). PCA0 interrupts must be globally enabled before any individual interrupt sources are recognized by the processor. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1.

18.3.3 Capture/Compare Modules

Each module can be configured to operate independently in one of six operation modes: edge-triggered capture, software timer, high-speed output, frequency output, 8 to 11-bit pulse width modulator, or 16-bit pulse width modulator. Table 18.2 PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules on page 255 summarizes the bit settings in the PCA0CPMn and PCA0PWM registers used to select the PCA capture/compare module's operating mode. All modules set to use 8-, 9-, 10-, or 11-bit PWM mode must use the same cycle length (8–11 bits). Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt.

Table 18.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules

Operational Mode	PCA0CPMn								PCA0PWM				
Bit Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	ARSEL	ECOV	COVF	Reserved	CLSEL
Capture triggered by positive edge on CEXn	X	X	1	0	0	0	0	A	0	X	B	X	X
Capture triggered by negative edge on CEXn	X	X	0	1	0	0	0	A	0	X	B	X	X
Capture triggered by any transition on CEXn	X	X	1	1	0	0	0	A	0	X	B	X	X
Software Timer	X	C	0	0	1	0	0	A	0	X	B	X	X
High Speed Output	X	C	0	0	1	1	0	A	0	X	B	X	X
Frequency Output	X	C	0	0	0	1	1	A	0	X	B	X	X
8-Bit Pulse Width Modulator[7]	0	C	0	0	E	0	1	A	0	X	B	X	0
9-Bit Pulse Width Modulator[7]	0	C	0	0	E	0	1	A	D	X	B	X	1
10-Bit Pulse Width Modulator[7]	0	C	0	0	E	0	1	A	D	X	B	X	2
11-Bit Pulse Width Modulator[7]	0	C	0	0	E	0	1	A	D	X	B	X	3
16-Bit Pulse Width Modulator	1	C	0	0	E	0	1	A	0	X	B	X	X

Notes:
1. X = Don't Care (no functional difference for individual module if 1 or 0).
2. A = Enable interrupts for this module (PCA interrupt triggered on CCFn set to 1).
3. B = Enable 8th–11th bit overflow interrupt (Depends on setting of CLSEL).
4. C = When set to 0, the digital comparator is off. For high speed and frequency output modes, the associated pin will not toggle. In any of the PWM modes, this generates a 0% duty cycle (output = 0).
5. D = Selects whether the Capture/Compare register (0) or the Auto-Reload register (1) for the associated channel is accessed via addresses PCA0CPHn and PCA0CPLn.
6. E = When set, a match event will cause the CCFn flag for the associated channel to be set.
7. All modules set to 8, 9, 10 or 11-bit PWM mode use the same cycle length setting.

18.3.3.1 Output Polarity

The output polarity of each PCA channel is individually selectable using the PCA0POL register. By default, all output channels are configured to drive the PCA output signals (CEXn) with their internal polarity. When the CEXnPOL bit for a specific channel is set to 1, that channel's output signal will be inverted at the pin. All other properties of the channel are unaffected, and the inversion does not apply to PCA input signals. Changes in the PCA0POL register take effect immediately at the associated output pin.

18.3.4 Edge-Triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN0 is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPPn and CAPNn bits are set to logic 1, then the state of the port pin associated with CEXn can be read directly to determine whether a rising-edge or falling-edge caused the capture.

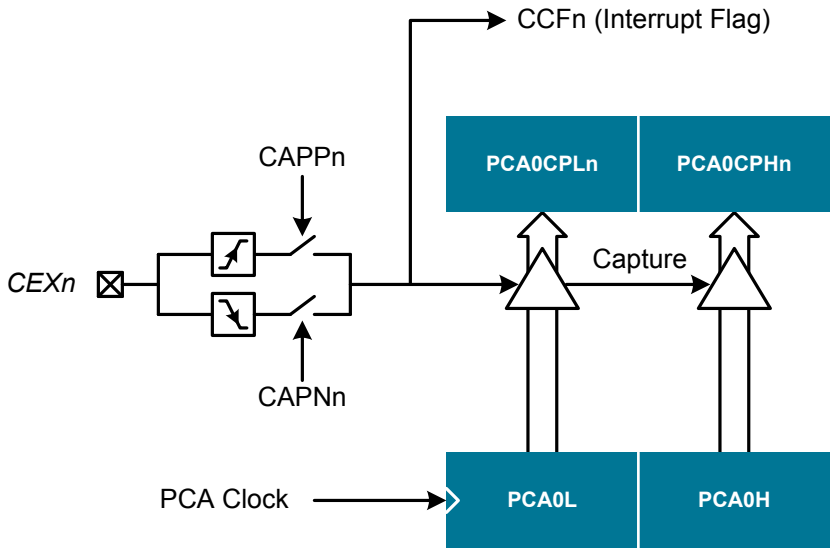


Figure 18.2. PCA Capture Mode Diagram

Note: The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

18.3.5 Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN0 is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and it must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

Note: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

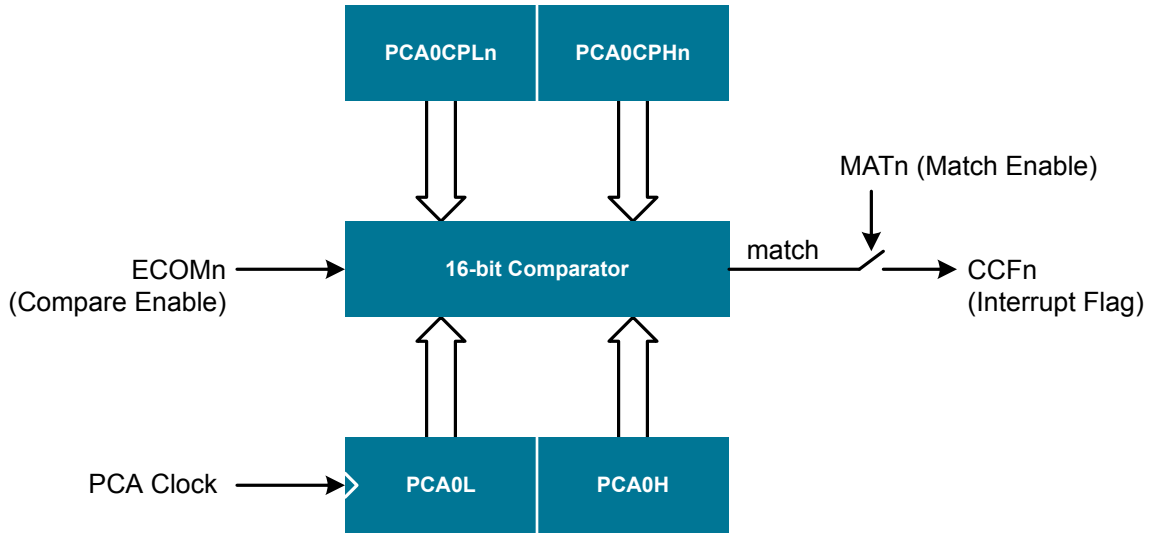


Figure 18.3. PCA Software Timer Mode Diagram

18.3.6 High-Speed Output Mode

In High-Speed Output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the capture/compare flag (CCFn) in PCA0CN0 is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine. It must be cleared by software. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin retains its state and does not toggle on the next match event.

Note: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

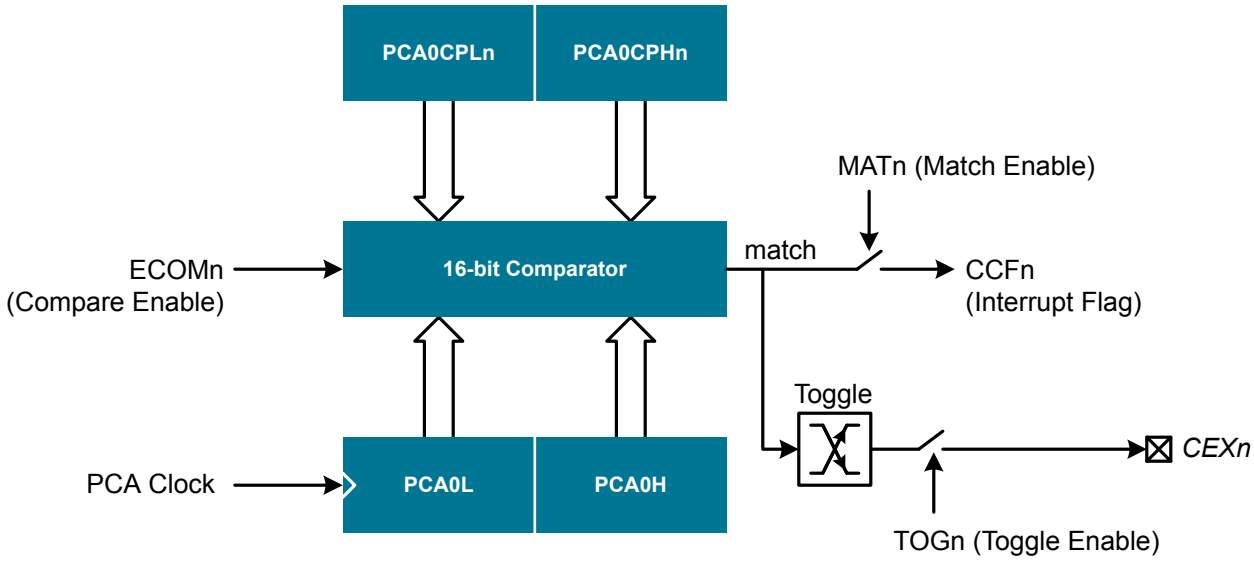


Figure 18.4. PCA High-Speed Output Mode Diagram

18.3.7 Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module's associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined as follows:

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

Note: A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

Where F_{PCA} is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register.

Note: The MATn bit should normally be set to 0 in this mode. If the MATn bit is set to 1, the CCFn flag for the channel will be set when the 16-bit PCA0 counter and the 16-bit capture/compare register for the channel are equal.

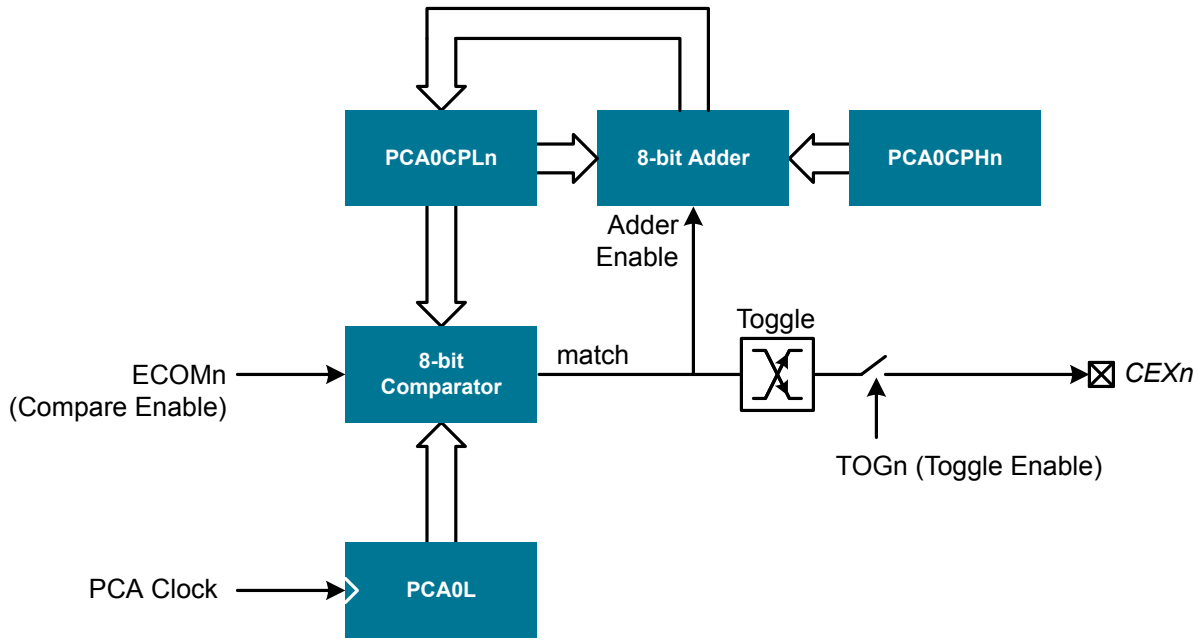


Figure 18.5. PCA Frequency Output Mode

18.3.8 PWM Waveform Generation

The PCA can generate edge- or center-aligned PWM waveforms with resolutions of 8, 9, 10, 11, or 16 bits. PWM resolution depends on the module setup, as specified within the individual module PCA0CPMn registers as well as the PCA0PWM register. Modules can be configured for 8-11 bit mode or for 16-bit mode individually using the PCA0CPMn registers. All modules configured for 8-11 bit mode have the same resolution, specified by the PCA0PWM register. When operating in one of the PWM modes, each module may be individually configured for center or edge-aligned PWM waveforms. Each channel has a single bit in the PCA0CENT register to select between the two options.

Edge Aligned PWM

When configured for edge-aligned mode, a module generates an edge transition at two points for every 2^N PCA clock cycles, where N is the selected PWM resolution in bits. In edge-aligned mode, these two edges are referred to as the “match” and “overflow” edges. The polarity at the output pin is selectable and can be inverted by setting the appropriate channel bit to 1 in the PCA0POL register. Prior to inversion, a match edge sets the channel to logic high, and an overflow edge clears the channel to logic low.

The match edge occurs when the the lowest N bits of the module’s PCA0CPn register match the corresponding bits of the main PCA0 counter register. For example, with 10-bit PWM, the match edge occurs any time bits 9-0 of the PCA0CPn register match bits 9-0 of the PCA0 counter value.

The overflow edge occurs when an overflow of the PCA0 counter happens at the desired resolution. For example, with 10-bit PWM, the overflow edge occurs when bits 0-9 of the PCA0 counter transition from all 1s to all 0s. All modules configured for edge-aligned mode at the same resolution align on the overflow edge of the waveforms.

An example of the PWM timing in edge-aligned mode for two channels is shown here. In this example, the CEX0POL and CEX1POL bits are cleared to 0.

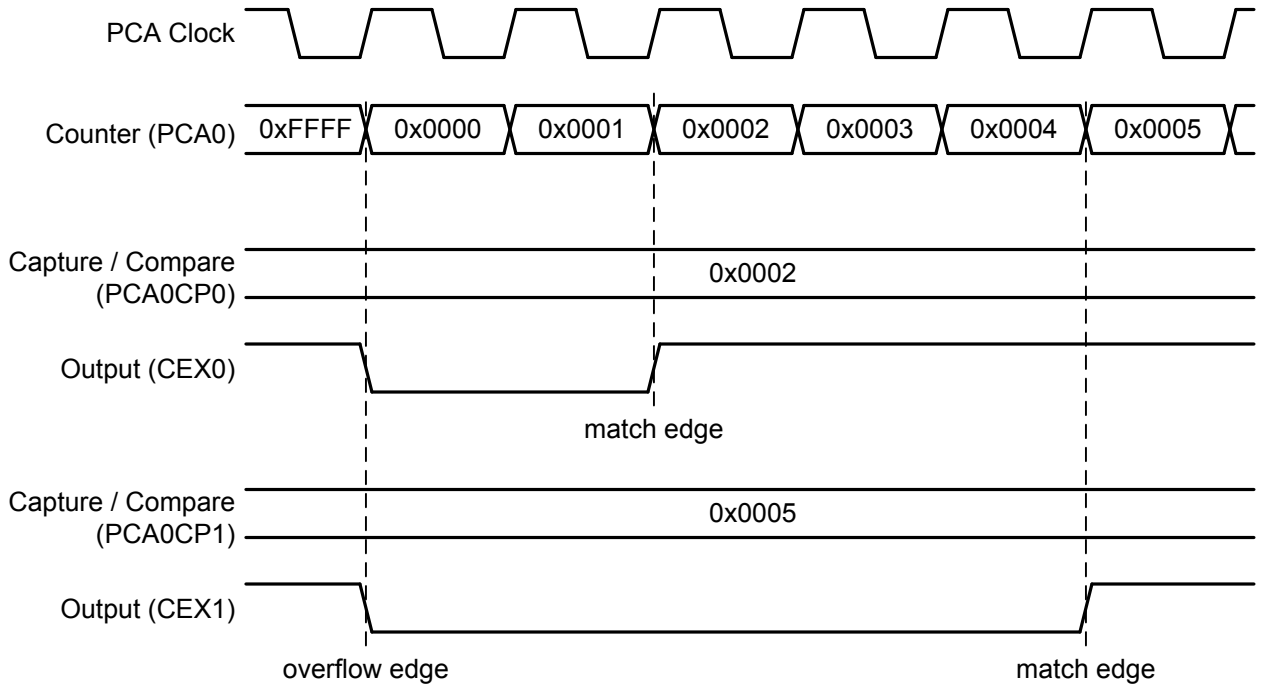


Figure 18.6. Edge-Aligned PWM Timing

For a given PCA resolution, the unused high bits in the PCA0 counter and the PCA0CPn compare registers are ignored, and only the used bits of the PCA0CPn register determine the duty cycle. Figure 18.7 N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution) on page 260 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is cleared to 0. Figure 18.8 N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution) on page 261 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is set to 1. A 0% duty cycle for the channel (with CEXnPOL = 0) is achieved by clearing the module’s ECOM bit to 0. This will disable the comparison, and prevent the match edge from occuring.

Note: Although the PCA0CPn compare register determines the duty cycle, it is not always appropriate for firmware to update this register directly. See the sections on 8 to 11-bit and 16-bit PWM mode for additional details on adjusting duty cycle in the various modes.

$$\text{Duty Cycle} = \frac{2^N - \text{PCA0CPn}}{2^N}$$

Figure 18.7. N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution)

$$\text{Duty Cycle} = \frac{\text{PCA0CPn}}{2^N}$$

Figure 18.8. N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution)

Center Aligned PWM

When configured for center-aligned mode, a module generates an edge transition at two points for every $2^{(N+1)}$ PCA clock cycles, where N is the selected PWM resolution in bits. In center-aligned mode, these two edges are referred to as the “up” and “down” edges. The polarity at the output pin is selectable and can be inverted by setting the appropriate channel bit to 1 in the PCA0POL register.

The generated waveforms are centered about the points where the lower N bits of the PCA0 counter are zero. The $(N+1)^{th}$ bit in the PCA0 counter acts as a selection between up and down edges. In 16-bit mode, a special 17th bit is implemented internally for this purpose. At the center point, the (non-inverted) channel output is low when the $(N+1)^{th}$ bit is 0 and high when the $(N+1)^{th}$ bit is 1, except for cases of 0% and 100% duty cycle. Prior to inversion, an up edge sets the channel to logic high, and a down edge clears the channel to logic low.

Down edges occur when the $(N+1)^{th}$ bit in the PCA0 counter is one and a logical inversion of the value in the module’s PCA0CPn register matches the main PCA0 counter register for the lowest N bits. For example, with 10-bit PWM, the down edge occurs when the one’s complement of bits 9-0 of the PCA0CPn register match bits 9-0 of the PCA0 counter and bit 10 of the PCA0 counter is 1.

Up edges occur when the $(N+1)^{th}$ bit in the PCA0 counter is zero and the lowest N bits of the module’s PCA0CPn register match the value of (PCA0 - 1). For example, with 10-bit PWM, the up edge occurs when bits 9-0 of the PCA0CPn register are one less than bits 9-0 of the PCA0 counter and bit 10 of the PCA0 counter is 0.

An example of the PWM timing in center-aligned mode for two channels is shown here. In this example, the CEX0POL and CEX1POL bits are cleared to 0.

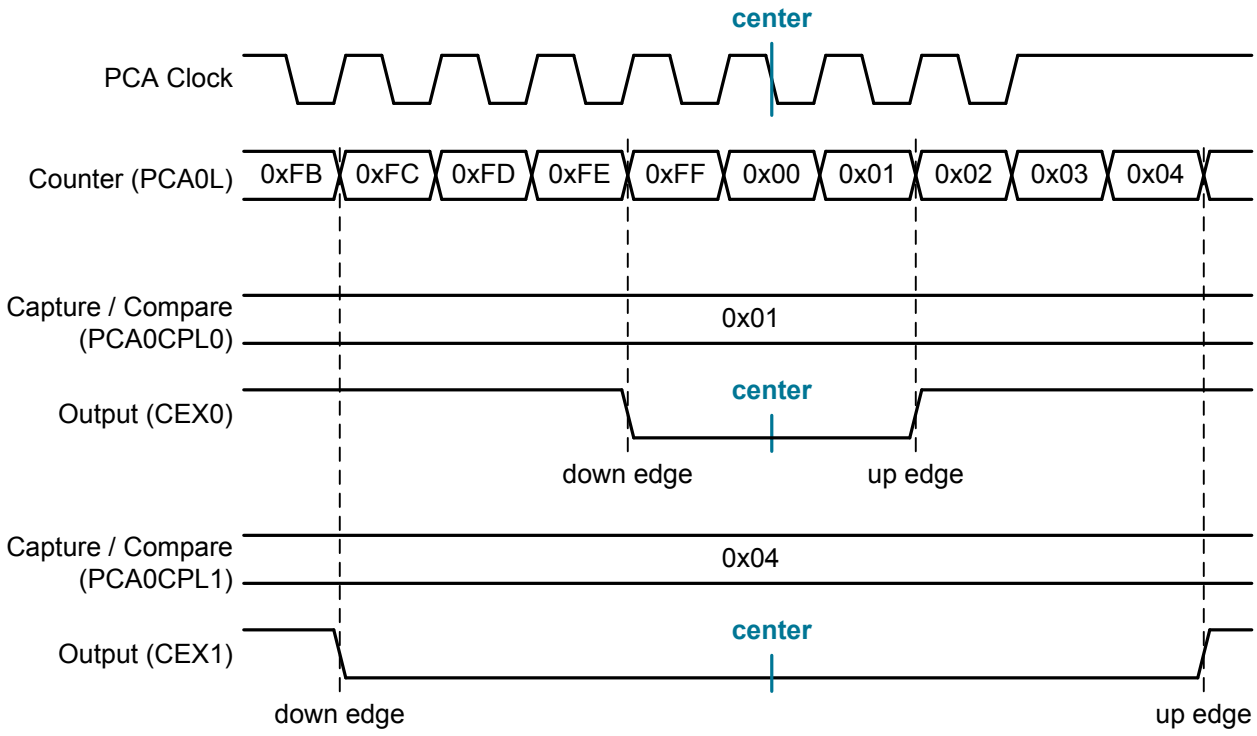


Figure 18.9. Center-Aligned PWM Timing

Figure 18.10 N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution) on page 263 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is cleared to 0. Figure 18.11 N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution) on page 263 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is set to 1. The equations are true only when the lowest N bits of the PCA0CPn register are not all 0s or all 1s. With CEXnPOL equal to zero, 100% duty cycle is produced when the lowest N bits of PCA0CPn are all 0, and 0% duty cycle is produced when the lowest N bits of PCA0CPn are all 1. For a given PCA resolution, the unused high bits in the PCA0 counter and the PCA0CPn compare registers are ignored, and only the used bits of the PCA0CPn register determine the duty cycle.

Note: Although the PCA0CPn compare register determines the duty cycle, it is not always appropriate for firmware to update this register directly. See the sections on 8 to 11-bit and 16-bit PWM mode for additional details on adjusting duty cycle in the various modes.

$$\text{Duty Cycle} = \frac{2^N - \text{PCA0CPn} - \frac{1}{2}}{2^N}$$

Figure 18.10. N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution)

$$\text{Duty Cycle} = \frac{\text{PCA0CPn} + \frac{1}{2}}{2^N}$$

Figure 18.11. N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution)

18.3.8.1 8 to 11-Bit PWM Modes

Each module can be used independently to generate a pulse width modulated (PWM) output on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA counter/timer and the setting of the PWM cycle length (8 through 11-bits). For backwards-compatibility with the 8-bit PWM mode available on other devices, the 8-bit PWM mode operates slightly different than 9 through 11-bit PWM modes.

Important: All channels configured for 8 to 11-bit PWM mode use the same cycle length. It is not possible to configure one channel for 8-bit PWM mode and another for 11-bit mode (for example). However, other PCA channels can be configured to Pin Capture, High-Speed Output, Software Timer, Frequency Output, or 16-bit PWM mode independently. Each channel configured for a PWM mode can be individually selected to operate in edge-aligned or center-aligned mode.

8-bit Pulse Width Modulator Mode

In 8-bit PWM mode, the duty cycle is determined by the value of the low byte of the PCA0CPn register (PCA0CPLn). To adjust the duty cycle, PCA0CPLn should not normally be written directly. Instead, the recommendation is to adjust the duty cycle using the high byte of the PCA0CPn register (register PCA0CPHn). This allows seamless updating of the PWM waveform as PCA0CPLn is reloaded automatically with the value stored in PCA0CPHn during the overflow edge (in edge-aligned mode) or the up edge (in center-aligned mode).

Setting the ECOMn and PWMn bits in the PCA0CPMn register and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit pulse width modulator mode. If the MATn bit is set to 1, the CCFn flag for the module is set each time a match edge or up edge occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which occurs every 256 PCA clock cycles.

9- to 11-bit Pulse Width Modulator Mode

In 9 to 11-bit PWM mode, the duty cycle is determined by the value of the least significant N bits of the PCA0CPn register, where N is the selected PWM resolution.

To adjust the duty cycle, PCA0CPn should not normally be written directly. Instead, the recommendation is to adjust the duty cycle by writing to an “Auto-Reload” register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The data written to define the duty cycle should be right-justified in the registers. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0. This allows seamless updating of the PWM waveform, as the PCA0CPn register is reloaded automatically with the value stored in the auto-reload registers during the overflow edge (in edge-aligned mode) or the up edge (in center-aligned mode).

Setting the ECOMn and PWMn bits in the PCA0CPMn register and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit pulse width modulator mode. If the MATn bit is set to 1, the CCFn flag for the module is set each time a match edge or up edge occurs. The COVF flag in PCA0PWM can be used to detect the overflow or down edge.

The 9 to 11-bit PWM mode is selected by setting the ECOMn and PWMn bits in the PCA0CPMn register and setting the CLSEL bits in register PCA0PWM to the desired cycle length (other than 8-bits). If the MATn bit is set to 1, the CCFn flag for the module is set each time a match edge or up edge occurs. The COVF flag in PCA0PWM can be used to detect the overflow or down edge.

Important: When writing a 16-bit value to the PCA0CPn registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

18.3.8.2 16-Bit PWM Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other PWM modes. The entire PCA0CP register is used to determine the duty cycle in 16-bit PWM mode.

To output a varying duty cycle, new value writes should be synchronized with the PCA CCFn match flag to ensure seamless updates.

16-Bit PWM mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, the match interrupt flag should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module is set each time a match edge or up edge occurs. The CF flag in PCA0CN0 can be used to detect the overflow or down edge.

Important: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

18.3.8.3 Comparator Clear Function

In 8/9/10/11/16-bit PWM modes, the comparator clear function utilizes a Comparator output synchronized to the system clock to clear CEXn to logic low for the current PWM cycle. Comparator 0 or Comparator 1 can be selected for the comparator clear function via the CPCSEL bit in the PCA0CLR SFR. This comparator clear function can be enabled for each PWM channel by setting the CPCEn bits to 1. When the comparator clear function is disabled, CEXn is unaffected.

The asynchronous Comparator output is logic high when the voltage of CPn+ is greater than CPn– and logic low when the voltage of CPn+ is less than CPn–. The polarity of the Comparator output is used to clear CEXn as follows: when CPCPOL = 0, CEXn is cleared on the falling edge of the Comparator output.

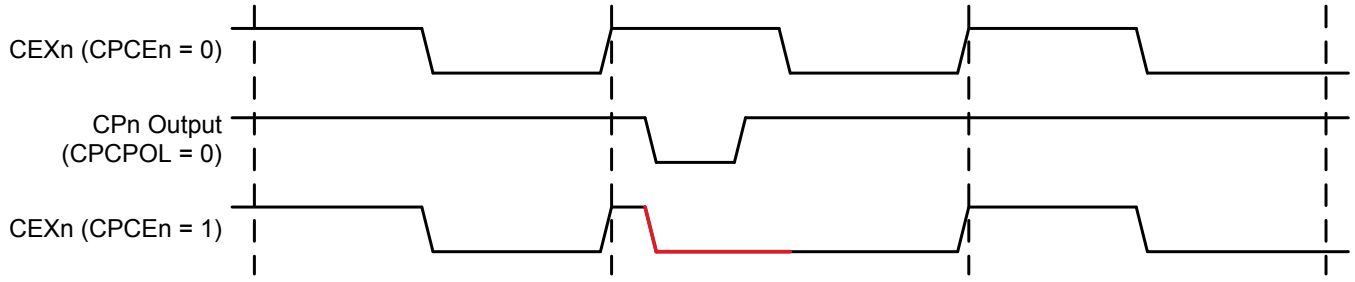


Figure 18.12. CEXn with CPCEn = 1, CPCPOL = 0

When CPCPOL = 1, CEXn is cleared on the rising edge of the Comparator output.

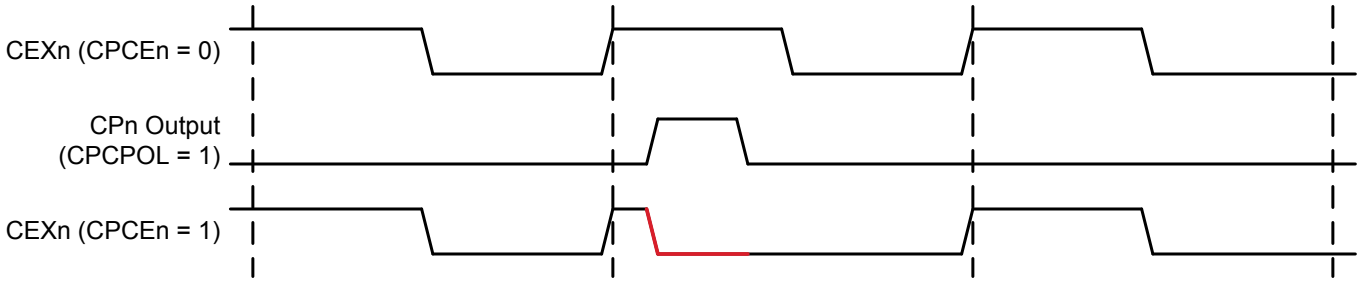


Figure 18.13. CEXn with CPCEn = 1, CPCPOL = 1

In the PWM cycle following the current cycle, should the Comparator output remain logic low when CPCPOL = 0 or logic high when CPCPOL = 1, CEXn will continue to be cleared.

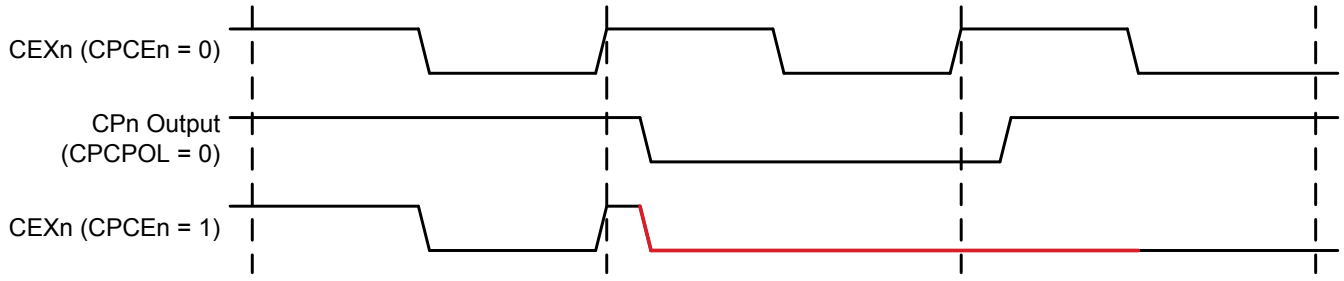


Figure 18.14. CEXn with CPCEn = 1, CPCPOL = 0

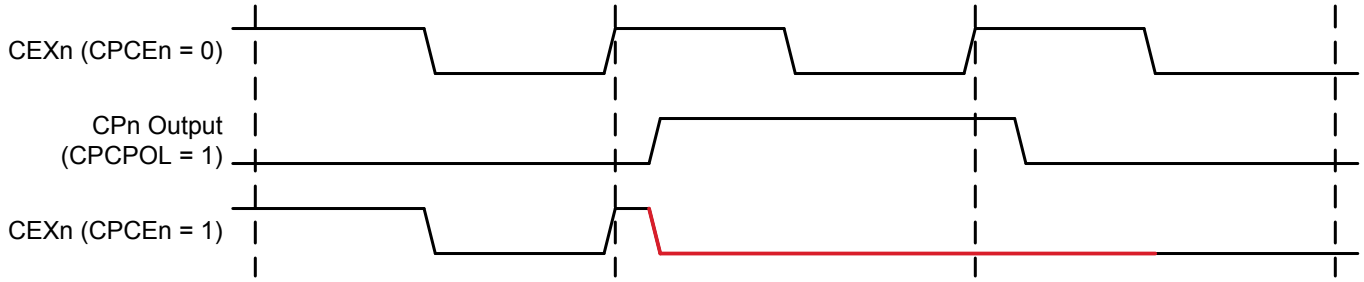


Figure 18.15. CEXn with CPCEn = 1, CPCPOL = 1

18.4 PCA0 Control Registers

18.4.1 PCA0CN0: PCA Control

Bit	7	6	5	4	3	2	1	0
Name	CF	CR	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xD8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	CF	0	RW	**PCA Counter/Timer Overflow Flag.** Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
6	CR	0	RW	**PCA Counter/Timer Run Control.** This bit enables/disables the PCA Counter/Timer. Value: 0 – Name: STOP – Stop the PCA Counter/Timer. Value: 1 – Name: RUN – Start the PCA Counter/Timer running.
5	CCF5	0	RW	**PCA Module 5 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF5 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
4	CCF4	0	RW	**PCA Module 4 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF4 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
3	CCF3	0	RW	**PCA Module 3 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF3 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
2	CCF2	0	RW	**PCA Module 2 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF2 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
1	CCF1	0	RW	**PCA Module 1 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF1 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
0	CCF0	0	RW	**PCA Module 0 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF0 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.

18.4.2 PCA0MD: PCA Mode

Bit	7	6	5	4	3	2	1	0
Name	CIDL	Reserved			CPS			ECF
Access	RW	R			RW			RW
Reset	0	0x0			0x0			0
SFR Page = 0x0, 0x10; SFR Address: 0xD9								

Bit	Name	Reset	Access	Description
7	CIDL	0	RW	**PCA Counter/Timer Idle Control.**
	Specifies PCA behavior when CPU is in Idle Mode.			
	Value	Name		Description
	0	NORMAL		PCA continues to function normally while the system controller is in Idle Mode.
	1	SUSPEND		PCA operation is suspended while the system controller is in Idle Mode.
6:4	*Reserved*	*Must write reset value.*		
3:1	CPS	0x0	RW	**PCA Counter/Timer Pulse Select.**
	These bits select the timebase source for the PCA counter.			
	Value	Name		Description
	0x0	SYSCLK_DIV_12		System clock divided by 12.
	0x1	SYSCLK_DIV_4		System clock divided by 4.
	0x2	T0_OVERFLOW		Timer 0 overflow.
	0x3	ECI		High-to-low transitions on ECI (max rate = system clock divided by 4).
	0x4	SYSCLK		System clock.
	0x5	EXTOSC_DIV_8		External clock divided by 8 (synchronized with the system clock).
	0x6	LFOSC_DIV_8		Low frequency oscillator divided by 8.
0	ECF	0	RW	**PCA Counter/Timer Overflow Interrupt Enable.**
	This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.			
	Value	Name		Description
	0	OVF_INT_DISABLED		Disable the CF interrupt.
	1	OVF_INT_ENABLED		Enable a PCA Counter/Timer Overflow interrupt request when CF is set.

18.4.3 PCA0PWM: PCA PWM Configuration

Bit	7	6	5	4	3	2	1	0
Name	ARSEL	ECOV	COVF	Reserved		CLSEL		
Access	RW	RW	RW	R		RW		
Reset	0	0	0	0x0		0x0		
SFR Page = 0x0, 0x10; SFR Address: 0xF7								

Bit	Name	Reset	Access	Description
7	ARSEL	0	RW	**Auto-Reload Register Select.**

This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9 to 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function.

Value	Name	Description
0	CAPTURE_COMPARE	Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn.
1	AUTORELOAD	Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn.

Bit	Name	Reset	Access	Description
6	ECOV	0	RW	**Cycle Overflow Interrupt Enable.**

This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt.

Value	Name	Description
0	COVF_MASK_DISABLED	COVF will not generate PCA interrupts.
1	COVF_MASK_ENABLED	A PCA interrupt will be generated when COVF is set.

Bit	Name	Reset	Access	Description
5	COVF	0	RW	**Cycle Overflow Flag.**

This bit indicates an overflow of the 8th to 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or firmware, but must be cleared by firmware.

Value	Name	Description
0	NO_OVERFLOW	No overflow has occurred since the last time this bit was cleared.
1	OVERFLOW	An overflow has occurred since the last time this bit was cleared.

Bit	Name	Reset	Access	Description
4:3	*Reserved*	*Must write reset value.*		
2:0	CLSEL	0x0	RW	**Cycle Length Select.**

When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to 16-bit PWM mode.

Value	Name	Description
0x0	8_BITS	8 bits.
0x1	9_BITS	9 bits.
0x2	10_BITS	10 bits.
0x3	11_BITS	11 bits.

18.4.4 PCA0CLR: PCA Comparator Clear Control

Bit	7	6	5	4	3	2	1	0
Name	CPCPOL	CPCSEL	CPCE5	CPCE4	CPCE3	CPCE2	CPCE1	CPCE0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0x9C								

Bit	Name	Reset	Access	Description
7	CPCPOL	0	RW	**Comparator Clear Polarity.**
	Selects the polarity of the comparator result that will clear the PCA channel(s).			
	Value	Name		Description
	0	LOW		PCA channel(s) will be cleared when comparator result goes logic low.
	1	HIGH		PCA channel(s) will be cleared when comparator result goes logic high.
6	CPCSEL	0	RW	**Comparator Clear Select.**
	Selects the comparator to use for the comparator clear function.			
	Value	Name		Description
	0	CMP_0		Comparator 0 will be used for the comparator clear function.
	1	CMP_1		Comparator 1 will be used for the comparator clear function.
5	CPCE5	0	RW	**Comparator Clear Enable for CEX5.**
	Enables the comparator clear function on PCA channel 5.			
4	CPCE4	0	RW	**Comparator Clear Enable for CEX4.**
	Enables the comparator clear function on PCA channel 4.			
3	CPCE3	0	RW	**Comparator Clear Enable for CEX3.**
	Enables the comparator clear function on PCA channel 3.			
2	CPCE2	0	RW	**Comparator Clear Enable for CEX2.**
	Enables the comparator clear function on PCA channel 2.			
1	CPCE1	0	RW	**Comparator Clear Enable for CEX1.**
	Enables the comparator clear function on PCA channel 1.			
0	CPCE0	0	RW	**Comparator Clear Enable for CEX0.**
	Enables the comparator clear function on PCA channel 0.			

18.4.5 PCA0L: PCA Counter/Timer Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0L							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xF9								

Bit	Name	Reset	Access	Description
7:0	PCA0L	0x00	RW	**PCA Counter/Timer Low Byte.** The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.

18.4.6 PCA0H: PCA Counter/Timer High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0H							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xFA								

Bit	Name	Reset	Access	Description
7:0	PCA0H	0x00	RW	**PCA Counter/Timer High Byte.** The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a "snapshot" register, whose contents are updated only when the contents of PCA0L are read.

18.4.7 PCA0POL: PCA Output Polarity

Bit	7	6	5	4	3	2	1	0
Name	Reserved		CEX5POL	CEX4POL	CEX3POL	CEX2POL	CEX1POL	CEX0POL
Access	R		RW	RW	RW	RW	RW	RW
Reset	0x0		0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0x96								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	CEX5POL	0	RW	**CEX5 Output Polarity.**
	Selects the polarity of the CEX5 output channel. When this bit is modified, the change takes effect at the pin immediately.			
	Value	Name		Description
	0	DEFAULT		Use default polarity.
	1	INVERT		Invert polarity.
4	CEX4POL	0	RW	**CEX4 Output Polarity.**
	Selects the polarity of the CEX4 output channel. When this bit is modified, the change takes effect at the pin immediately.			
	Value	Name		Description
	0	DEFAULT		Use default polarity.
	1	INVERT		Invert polarity.
3	CEX3POL	0	RW	**CEX3 Output Polarity.**
	Selects the polarity of the CEX3 output channel. When this bit is modified, the change takes effect at the pin immediately.			
	Value	Name		Description
	0	DEFAULT		Use default polarity.
	1	INVERT		Invert polarity.
2	CEX2POL	0	RW	**CEX2 Output Polarity.**
	Selects the polarity of the CEX2 output channel. When this bit is modified, the change takes effect at the pin immediately.			
	Value	Name		Description
	0	DEFAULT		Use default polarity.
	1	INVERT		Invert polarity.
1	CEX1POL	0	RW	**CEX1 Output Polarity.**
	Selects the polarity of the CEX1 output channel. When this bit is modified, the change takes effect at the pin immediately.			
	Value	Name		Description
	0	DEFAULT		Use default polarity.
	1	INVERT		Invert polarity.

Bit	Name	Reset	Access	Description
0	CEX0POL	0	RW	**CEX0 Output Polarity.**

Selects the polarity of the CEX0 output channel. When this bit is modified, the change takes effect at the pin immediately.

Value	Name	Description
0	DEFAULT	Use default polarity.
1	INVERT	Invert polarity.

18.4.8 PCA0CENT: PCA Center Alignment Enable

Bit	7	6	5	4	3	2	1	0
Name	Reserved		CEX5CEN	CEX4CEN	CEX3CEN	CEX2CEN	CEX1CEN	CEX0CEN
Access	R		RW	RW	RW	RW	RW	RW
Reset	0x0		0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0x9E								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	CEX5CEN	0	RW	**CEX5 Center Alignment Enable.** Selects the alignment properties of the CEX5 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes. Value: 0 – EDGE – Edge-aligned. Value: 1 – CENTER – Center-aligned.
4	CEX4CEN	0	RW	**CEX4 Center Alignment Enable.** Selects the alignment properties of the CEX4 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes. Value: 0 – EDGE – Edge-aligned. Value: 1 – CENTER – Center-aligned.
3	CEX3CEN	0	RW	**CEX3 Center Alignment Enable.** Selects the alignment properties of the CEX3 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes. Value: 0 – EDGE – Edge-aligned. Value: 1 – CENTER – Center-aligned.
2	CEX2CEN	0	RW	**CEX2 Center Alignment Enable.** Selects the alignment properties of the CEX2 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes. Value: 0 – EDGE – Edge-aligned. Value: 1 – CENTER – Center-aligned.
1	CEX1CEN	0	RW	**CEX1 Center Alignment Enable.** Selects the alignment properties of the CEX1 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes. Value: 0 – EDGE – Edge-aligned.

Bit	Name	Reset	Access	Description
	1	CENTER		Center-aligned.
0	CEX0CEN	0	RW	**CEX0 Center Alignment Enable.** Selects the alignment properties of the CEX0 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes.

Value	Name	Description
0	EDGE	Edge-aligned.
1	CENTER	Center-aligned.

18.4.9 PCA0CPM0: PCA Channel 0 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xDA								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 0 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8 to 11-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 0 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 0 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 0 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 0 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF0 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 0 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX0 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 0 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX0 pin. 8 to 11-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 0 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF0) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF0 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF0 is set.

18.4.10 PCA0CPL0: PCA Channel 0 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL0							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xFB								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL0	0x00	RW	**PCA Channel 0 Capture Module Low Byte.** The PCA0CPL0 register holds the low byte (LSB) of the 16-bit capture module. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will clear the module's ECOM bit to a 0.				

18.4.11 PCA0CPH0: PCA Channel 0 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH0							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xFC								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH0	0x00	RW	**PCA Channel 0 Capture Module High Byte.** The PCA0CPH0 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will set the module's ECOM bit to a 1.				

18.4.12 PCA0CPM1: PCA Channel 1 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xDB								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 1 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8 to 11-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 1 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 1 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 1 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 1 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF1 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 1 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX1 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 1 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX1 pin. 8 to 11-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 1 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF1) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF1 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF1 is set.

18.4.13 PCA0CPL1: PCA Channel 1 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL1							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xE9								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL1	0x00	RW	**PCA Channel 1 Capture Module Low Byte.** The PCA0CPL1 register holds the low byte (LSB) of the 16-bit capture module. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will clear the module's ECOM bit to a 0.				

18.4.14 PCA0CPH1: PCA Channel 1 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH1							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xEA								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH1	0x00	RW	**PCA Channel 1 Capture Module High Byte.** The PCA0CPH1 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will set the module's ECOM bit to a 1.				

18.4.15 PCA0CPM2: PCA Channel 2 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xDC								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 2 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8 to 11-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 2 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 2 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 2 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 2 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF2 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 2 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX2 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 2 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX2 pin. 8 to 11-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 2 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF2) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF2 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF2 is set.

18.4.16 PCA0CPL2: PCA Channel 2 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL2							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xEB								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL2	0x00	RW	**PCA Channel 2 Capture Module Low Byte.** The PCA0CPL2 register holds the low byte (LSB) of the 16-bit capture module. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will clear the module's ECOM bit to a 0.				

18.4.17 PCA0CPH2: PCA Channel 2 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH2							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xEC								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH2	0x00	RW	**PCA Channel 2 Capture Module High Byte.** The PCA0CPH2 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will set the module's ECOM bit to a 1.				

18.4.18 PCA0CPM3: PCA Channel 3 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xAE								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 3 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8 to 11-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 3 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 3 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 3 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 3 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF3 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 3 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX3 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 3 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX3 pin. 8 to 11-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 3 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF3) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF3 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF3 is set.

18.4.19 PCA0CPL3: PCA Channel 3 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL3							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xF4								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL3	0x00	RW	**PCA Channel 3 Capture Module Low Byte.** The PCA0CPL3 register holds the low byte (LSB) of the 16-bit capture module. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will clear the module's ECOM bit to a 0.				

18.4.20 PCA0CPH3: PCA Channel 3 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH3							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xF5								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH3	0x00	RW	**PCA Channel 3 Capture Module High Byte.** The PCA0CPH3 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will set the module's ECOM bit to a 1.				

18.4.21 PCA0CPM4: PCA Channel 4 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xAF								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 4 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8 to 11-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 4 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 4 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 4 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 4 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF4 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 4 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX4 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 4 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX4 pin. 8 to 11-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 4 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF4) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF4 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF4 is set.

18.4.22 PCA0CPL4: PCA Channel 4 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL4							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0x84								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL4	0x00	RW	**PCA Channel 4 Capture Module Low Byte.** The PCA0CPL4 register holds the low byte (LSB) of the 16-bit capture module. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will clear the module's ECOM bit to a 0.				

18.4.23 PCA0CPH4: PCA Channel 4 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH4							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0x85								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH4	0x00	RW	**PCA Channel 4 Capture Module High Byte.** The PCA0CPH4 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will set the module's ECOM bit to a 1.				

18.4.24 PCA0CPM5: PCA Channel 5 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0xCC								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 5 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8 to 11-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 5 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 5 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 5 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 5 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF5 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 5 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX5 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 5 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX5 pin. 8 to 11-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 5 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF5) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF5 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF5 is set.

18.4.25 PCA0CPL5: PCA Channel 5 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL5							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xDD								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL5	0x00	RW	**PCA Channel 5 Capture Module Low Byte.** The PCA0CPL5 register holds the low byte (LSB) of the 16-bit capture module. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will clear the module's ECOM bit to a 0.				

18.4.26 PCA0CPH5: PCA Channel 5 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH5							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xDE								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH5	0x00	RW	**PCA Channel 5 Capture Module High Byte.** The PCA0CPH5 register holds the high byte (MSB) of the 16-bit capture module. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
A write to this register will set the module's ECOM bit to a 1.				

19. Serial Peripheral Interface (SPI0)

19.1 Introduction

The serial peripheral interface (SPI) module provides access to a flexible, full-duplex synchronous serial bus. The SPI can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select the SPI in slave mode, or to disable master mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a firmware-controlled chip-select output in master mode, or disabled to reduce the number of pins required. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

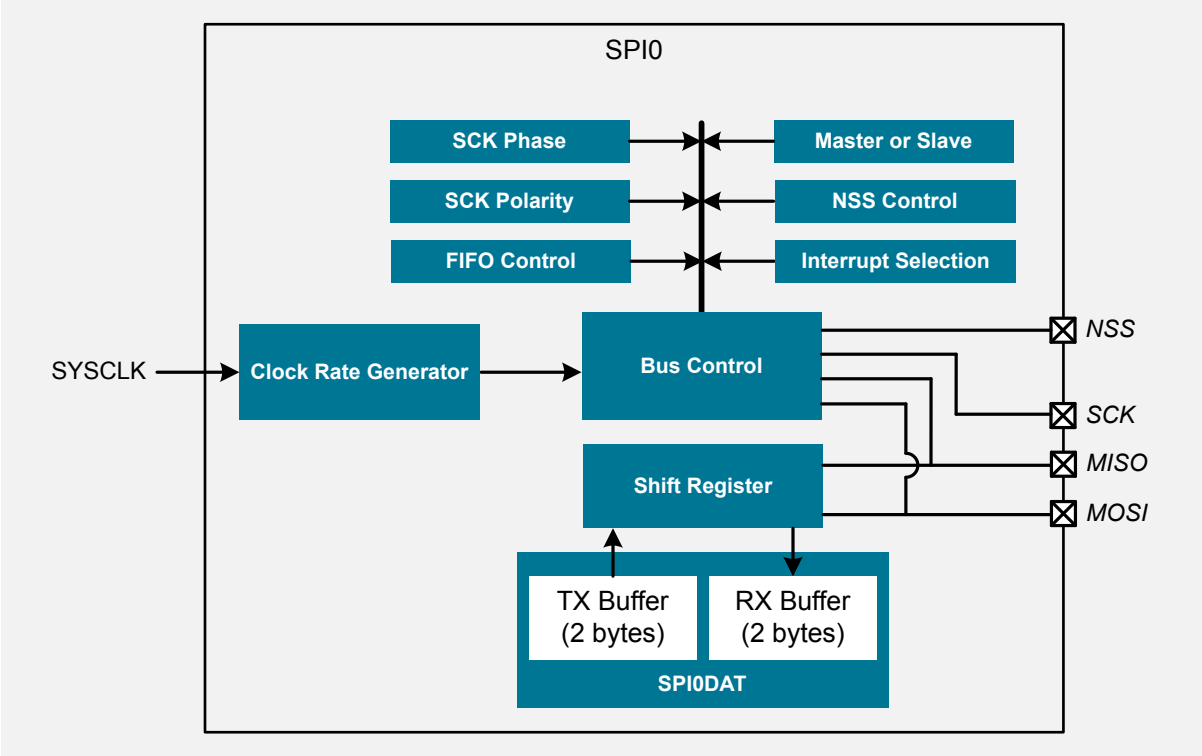


Figure 19.1. SPI Block Diagram

19.2 Features

- Supports 3- or 4-wire master or slave modes
- Supports external clock frequencies up to 12 Mbps in master or slave mode
- Support for all clock phase and polarity modes
- 8-bit programmable clock rate (master)
- Programmable receive timeout (slave)
- Two byte FIFO on transmit and receive
- Can operate in suspend or snooze modes and wake the CPU on reception of a byte
- Support for multiple masters on the same data lines

19.3 Functional Description

19.3.1 Signals

The SPI interface consists of up to four signals: MOSI, MISO, SCK, and NSS.

Master Out, Slave In (MOSI): The MOSI signal is the data output pin when configured as a master device and the data input pin when configured as a slave. It is used to serially transfer data from the master to the slave. Data is transferred on the MOSI pin most-significant bit first. When configured as a master, MOSI is driven from the internal shift register in both 3- and 4-wire mode.

Master In, Slave Out (MISO): The MISO signal is the data input pin when configured as a master device and the data output pin when configured as a slave. It is used to serially transfer data from the slave to the master. Data is transferred on the MISO pin most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled or when the SPI operates in 4-wire mode as a slave that is not selected. When acting as a slave in 3-wire mode, MISO is always driven from the internal shift register.

Serial Clock (SCK): The SCK signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. The SPI module generates this signal when operating as a master and receives it as a slave. The SCK signal is ignored by a SPI slave when the slave is not selected in 4-wire slave mode.

Slave Select (NSS): The function of the slave-select (NSS) signal is dependent on the setting of the NSSMD bitfield. There are three possible modes that can be selected with these bits:

- NSSMD[1:0] = 00: 3-Wire Master or 3-Wire Slave Mode: The SPI operates in 3-wire mode, and NSS is disabled. When operating as a slave device, the SPI is always selected in 3-wire mode. Since no select signal is present, the SPI must be the only slave on the bus in 3-wire mode. This is intended for point-to-point communication between a master and a single slave.
- NSSMD[1:0] = 01: 4-Wire Slave or Multi-Master Mode: The SPI operates in 4-wire mode, and NSS is configured as an input. When operating as a slave, NSS selects the SPI device. When operating as a master, a 1-to- 0 transition of the NSS signal disables the master function of the SPI module so that multiple master devices can be used on the same SPI bus.
- NSSMD[1:0] = 1x: 4-Wire Master Mode: The SPI operates in 4-wire mode, and NSS is enabled as an output. The setting of NSSMD0 determines what logic level the NSS pin will output. This configuration should only be used when operating the SPI as a master device.

The setting of NSSMD bits affects the pinout of the device. When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device.

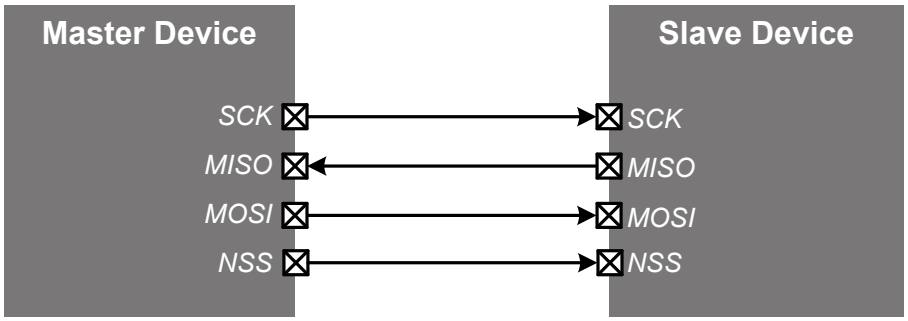


Figure 19.2. 4-Wire Connection Diagram

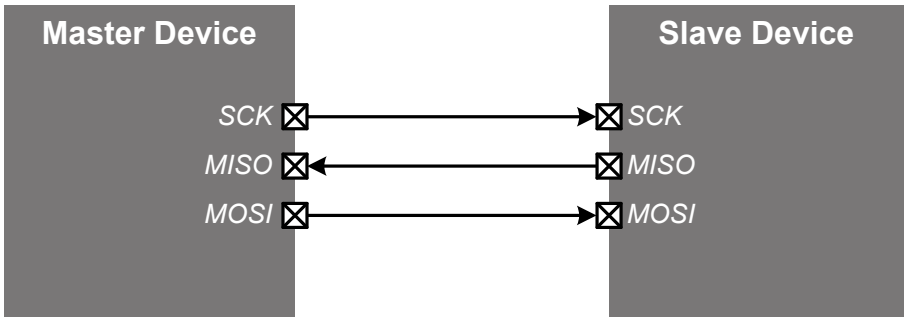


Figure 19.3. 3-Wire Connection Diagram

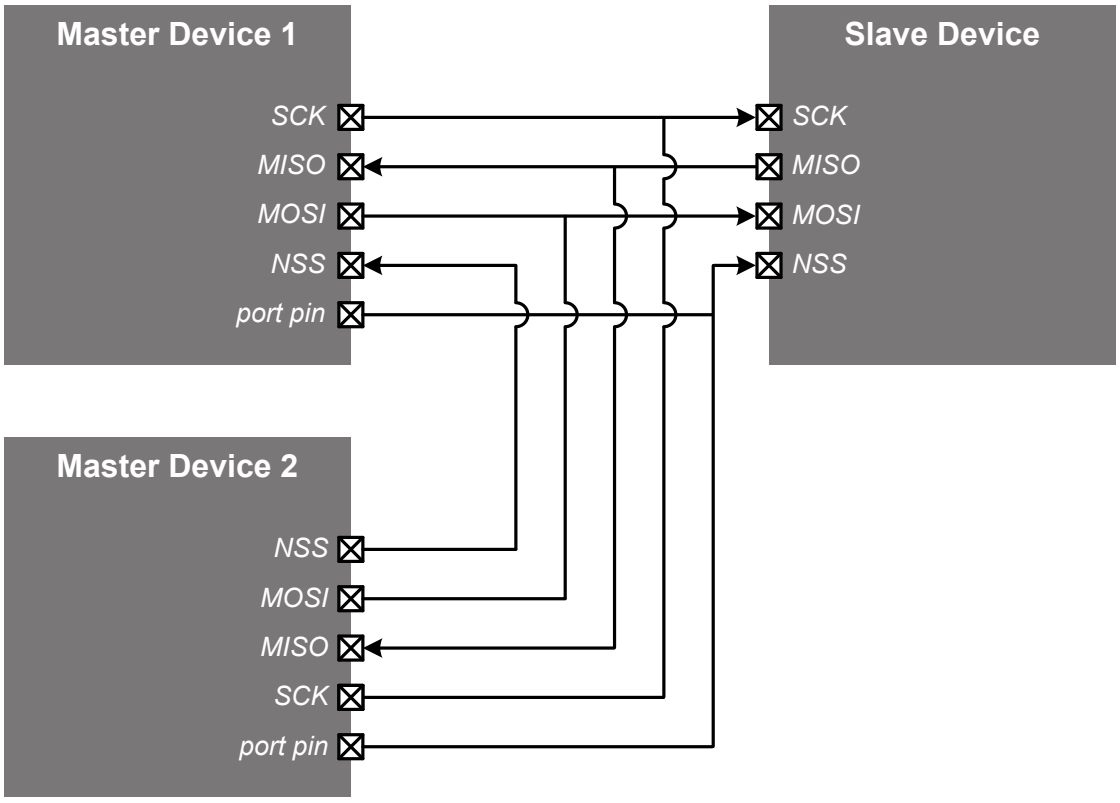


Figure 19.4. Multi-Master Connection Diagram

19.3.1.1 Routing Input Signals Through Configurable Logic

All of the SPI signals are routed through the crossbar by default. It is also possible to route the inputs to the SPI from certain CLU outputs, as controlled by the SPI0PCF register. SCK may route from CLU1, MISO (master mode) may route from CLU2, and MOSI (slave mode) may route from CLU3. Each input selection is controlled individually, allowing the user to apply input logic to one or more of the inputs.

19.3.2 Master Mode Operation

An SPI master device initiates all data transfers on a SPI bus. It drives the SCK line and controls the speed at which data is transferred. To place the SPI in master mode, the MSTEN bit should be set to 1. Writing a byte of data to the SPInDAT register writes to the transmit buffer. If the SPI shift register is empty, a byte is moved from the transmit buffer into the shift register, and a bi-directional data transfer begins. The SPI module provides the serial clock on SCK, while simultaneously shifting data out of the shift register MSB-first on MOSI and into the shift register MSB-first on MISO. Upon completing a transfer, the data received is moved from the shift register into the receive buffer. If the transmit buffer is not empty, the next byte in the transmit buffer will be moved into the shift register and the next data transfer will begin. If no new data is available in the transmit buffer, the SPI will halt and wait for new data to initiate the next transfer. Bytes that have been received and stored in the receive buffer may be read from the buffer via the SPInDAT register.

19.3.3 Slave Mode Operation

When the SPI block is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by an external master device controlling the SCK signal. A bit counter in the SPI logic counts SCK edges. When 8 bits have been shifted through the shift register, a byte is copied into the receive buffer. Data is read from the receive buffer by reading SPInDAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the transmit buffer by writing to SPInDAT and will transfer to the shift register on byte boundaries in the order in which they were written to the buffer.

When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. In the default, 4-wire slave mode, the NSS signal is routed to a port pin and configured as a digital input. The SPI interface is enabled when NSS is logic 0, and disabled when NSS is logic 1. The internal shift register bit counter is reset on a falling edge of NSS. When operated in 3-wire slave mode, NSS is not mapped to an external port pin through the crossbar. Since there is no way of uniquely addressing the device in 3-wire slave mode, the SPI must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling the SPI module with the SPIEN bit.

19.3.4 Clock Phase and Polarity

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPInCFG register. The CKPHA bit selects one of two clock phases (edge used to latch the data). The CKPOL bit selects between an active-high or active-low clock. Both master and slave devices must be configured to use the same clock phase and polarity. The SPI module should be disabled (by clearing the SPIEN bit) when changing the clock phase or polarity. Note that CKPHA should be set to 0 on both the master and slave SPI when communicating between two Silicon Labs devices.

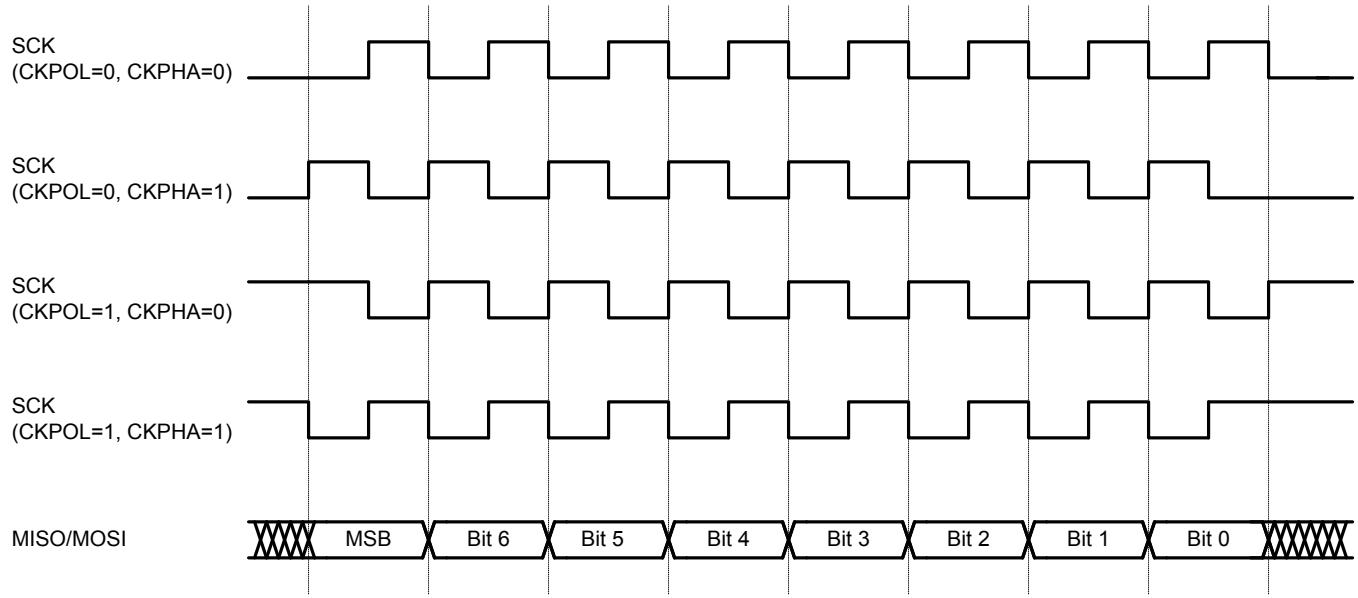


Figure 19.5. Master Mode Data/Clock Timing

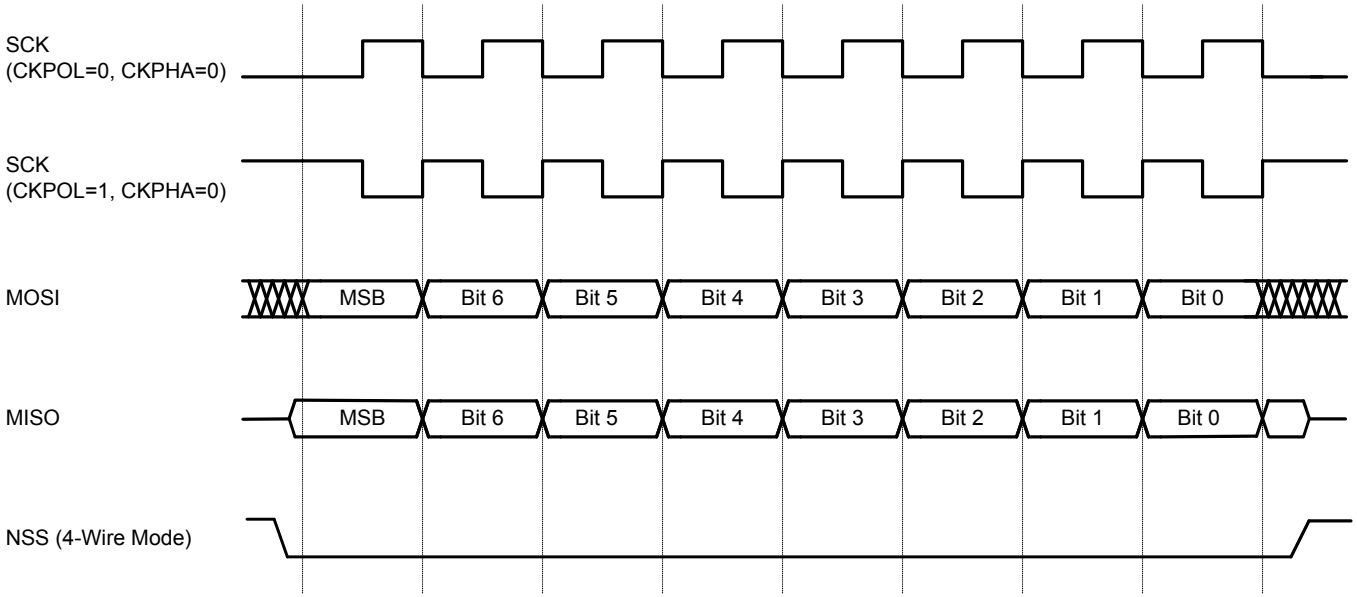


Figure 19.6. Slave Mode Data/Clock Timing (CKPHA = 0)

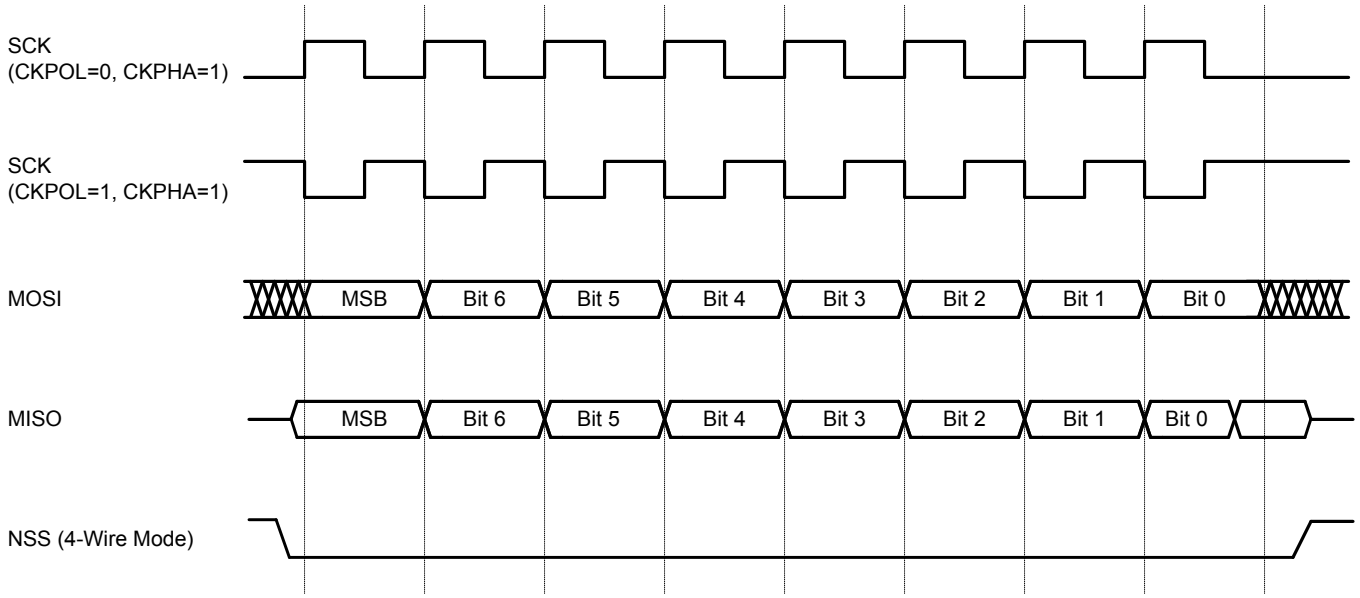


Figure 19.7. Slave Mode Data/Clock Timing (CKPHA = 1)

19.3.5 Basic Data Transfer

The SPI bus is inherently full-duplex. It sends and receives a single byte on every transfer. The SPI peripheral may be operated on a byte-by-byte basis using the SPInDAT register and the SPIF flag. The method firmware uses to send and receive data through the SPI interface is the same in either mode, but the hardware will react differently.

Master Transfers

As an SPI master, all transfers are initiated with a write to SPInDAT, and the SPIF flag will be set by hardware to indicate the end of each transfer. The general method for a single-byte master transfer follows:

1. Write the data to be sent to SPInDAT. The transfer will begin on the bus at this time.
2. Wait for the SPIF flag to generate an interrupt, or poll SPIF until it is set to 1.
3. Read the received data from SPInDAT.
4. Clear the SPIF flag to 0.
5. Repeat the sequence for any additional transfers.

Slave Transfers

As a SPI slave, the transfers are initiated by an external master device driving the bus. Slave firmware may anticipate any output data needs by pre-loading the SPInDAT register before the master begins the transfer.

1. Write any data to be sent to SPInDAT. The transfer will not begin until the external master device initiates it.
2. Wait for the SPIF flag to generate an interrupt, or poll SPIF until it is set to 1.
3. Read the received data from SPInDAT.
4. Clear the SPIF flag to 0.
5. Repeat the sequence for any additional transfers.

19.3.6 Using the SPI FIFOs

The SPI peripheral implements independent four-byte FIFOs for both the transmit and receive paths. The FIFOs are active in both master and slave modes, and a number of configuration features are available to accomodate a variety of SPI implementations.

FIFO Data Interface

Writing and reading the FIFOs is straightforward, and similar to the procedure outlined in 19.3.5 Basic Data Transfer. All FIFO writes and reads are performed through the SPInDAT register. To write data into the transmit buffer, firmware should first check the status of the TXNF bit. If TXNF reads 1, there is room in the buffer and firmware may write to the SPInDAT register. Writing the transmit buffer when TXNF is 0 will cause a write collision error, and the data written will not be accepted into the buffer.

To read data from the receive FIFO, firmware should check the state of the RXE bit. When RXE is 0, it means there is data available in the receive FIFO, and it may be read using the SPInDAT register. When RXE is 1 the receive FIFO is empty. Reading an empty receive FIFO returns the most recently-received byte.

The data in either FIFO may be flushed (i.e. FIFO pointers reset) by setting the corresponding flush bit to 1. TFLSH will reset the transmit FIFO, and RFLSH will reset the receive FIFO.

Half-Duplex Operation

SPI transfers are inherently full-duplex. However, the operation of either FIFO may be disabled to facilitate half-duplex operation.

The TXHOLD bit is used to stall transmission of bytes from the transmit FIFO. TXHOLD is checked by hardware at the beginning of a byte transfer. If TXHOLD is 1 at the beginning of a byte transfer, data will not be pulled from the transmit FIFO. Instead, the SPI interface will hold the output pin at the logic level defined by the TXPOL bit.

The RXFIFOE bit may be used to disable the receive FIFO. If RXFIFOE is 0 at the end of a byte transfer, the received byte will be discarded and the receive FIFO will not be updated.

TXHOLD and RXFIFOE can be changed by firmware at any time during a transfer. Any data currently being shifted out on the SPI interface has already been pulled from the transmit FIFO, and changing TXFLSH will not abort that data transfer.

FIFO Thresholds and Interrupts

The number of bytes present in the FIFOs is stored in the SPInFCT register. The TXCNT field indicates the number of bytes in the transmit FIFO while the RXCNT field indicates the number of bytes in the receive FIFO.

Each FIFO has a threshold field which firmware may use to define when transmit and receive requests will occur. The transmit threshold (TXTH) is continually compared with the TXCNT field. If TXCNT is less than or equal to TXTH, hardware will set the TFRQ flag to 1. The receive threshold (RXTH) is continually compared with RXCNT. If RXCNT is greater than RXTH, hardware will set the RFRQ flag to 1.

The thresholds can be used in interrupt-based systems to specify when the associated interrupt occurs. Both the RFRQ and TFRQ flags may be individually enabled to generate an SPI interrupt using the RFRQE and TFRQE bits, respecitvely. In most applications, when RFRQ or TFRQ are used to generate interrupts the SPIF flag should be disabled as an interrupt source by clearing the SPIFEN control bit to 0.

Applications may choose to use any combination of interrupt sources as needed. In general, the following settings are recommended for different applications:

- **Master mode, transmit only**: Use only the TFRQ flag as an interrupt source. Inside the ISR, check TXNF before writing more data to the FIFO. When all data to be sent has been processed through the ISR, the ISR may clear TFRQE to 0 to prevent further interrupts. Main threads may then set TFRQE back to 1 when additional data is to be sent.
- **Master mode, full-duplex or receive only**: Use only the RFRQ flag as an interrupt source. Transfers may be started by a write to SPInDAT. Inside the ISR, check RXE and read bytes from the FIFO as they are available. For every byte read, a new byte may be written to the transmit FIFO until there are no more bytes to send. If operating half-duplex in receive-only mode, the SPInDAT register must still be written to initiate new transfers.
- **Slave mode, transmit only**: Use the TFRQ flag as an interrupt source. Inside the ISR, check TXNF before writing more data to the FIFO. The receive FIFO may also be disabled if desired.
- **Slave mode, receive only**: Use the RFRQ flag as an interrupt source. If the RXTH field is set to anything other than 0, it is recommended to configure and enable RX timeouts. Inside the ISR, check RXE and read bytes from the FIFO as they are available. The transmit FIFO may be disabled if desired. Note that if the transmit FIFO is not disabled and firmware does not write to SPInDAT, bytes received in the shift register could be sent back out on the SPI MISO pin.
- **Slave mode, full-duplex**: Pre-load the transmit FIFO with the initial bytes to be sent. Use the RFRQ flag as an interrupt source. If the RXTH field is set to anything other than 0, it is recommended to configure and enable RX timeouts. Inside the ISR, check RXE and read bytes from the FIFO as they are available. For every byte read, a new byte may be written to the transmit FIFO.

Slave Receiver Timeout

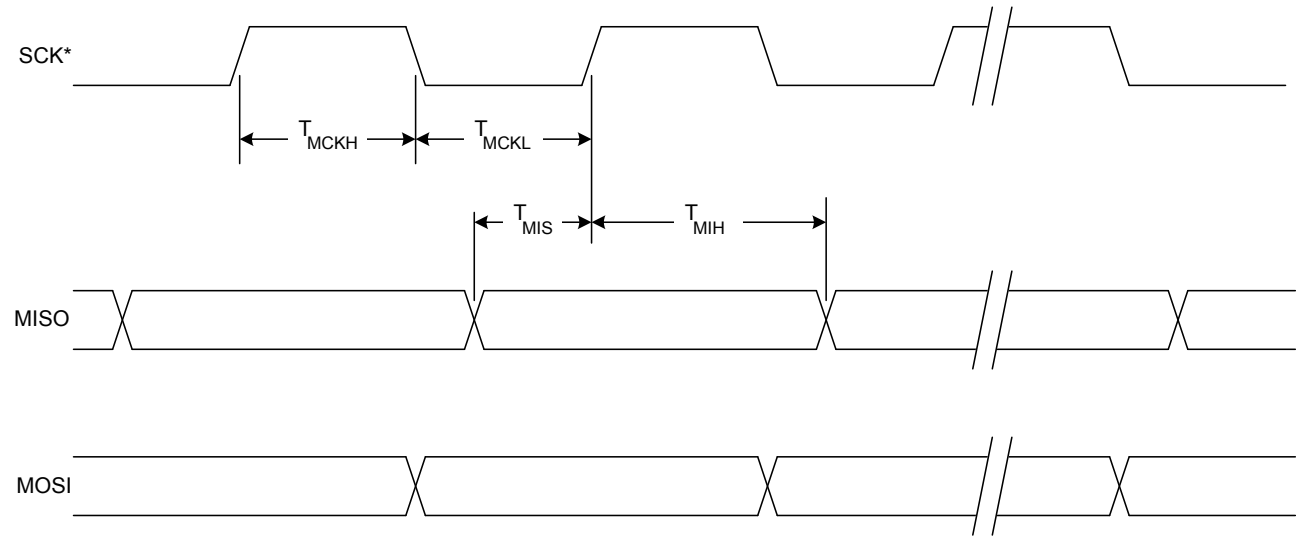
When acting as a SPI slave using RFRQ interrupts and with the RXTH field set to a value greater than 0, it is possible for the external master to write too few bytes to the device to immediately generate an interrupt. To avoid leaving lingering bytes in the receive FIFO, the slave receiver timeout feature may be used. Receive timeouts are enabled by setting the RXTOE bit to 1.

The length of a receive timeout may be specified in the SPInCKR register, and is equivalent to SPInCKR x 32 system clock cycles (SYSCLKs). The internal timeout counter will run when at least one byte has been received in the receive FIFO, but the RFRQ flag is not set (the RXTH threshold has not been crossed). The counter is reloaded from the SPInCKR register under any of the following conditions:

- The receive buffer is read by firmware.
- The RFRQ flag is set.
- A valid SCK occurs on the SPI interface.

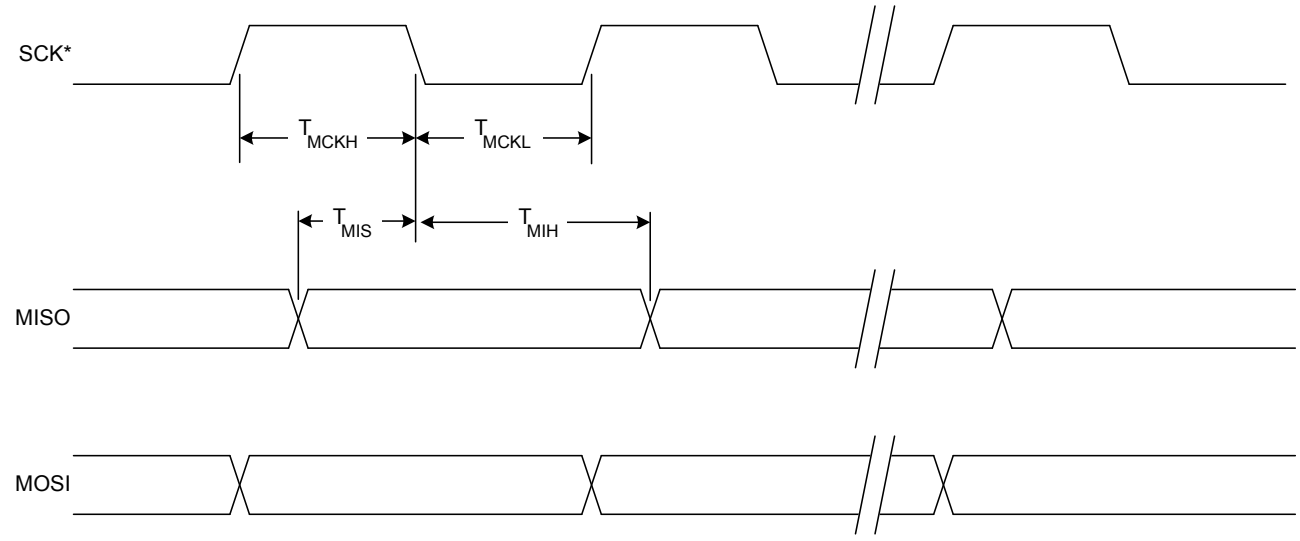
If the internal counter runs out, a SPI interrupt will be generated, allowing firmware to read any bytes remaining in the receive FIFO.

19.3.7 SPI Timing Diagrams



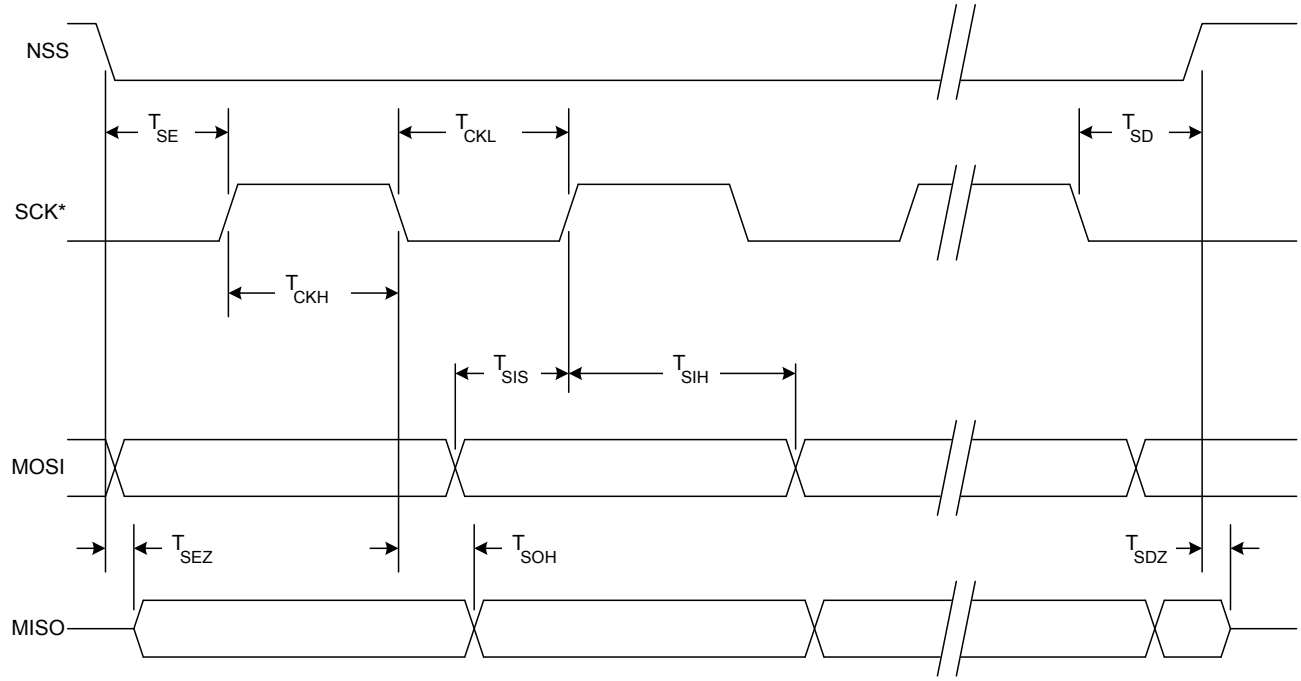
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 19.8. SPI Master Timing (CKPHA = 0)



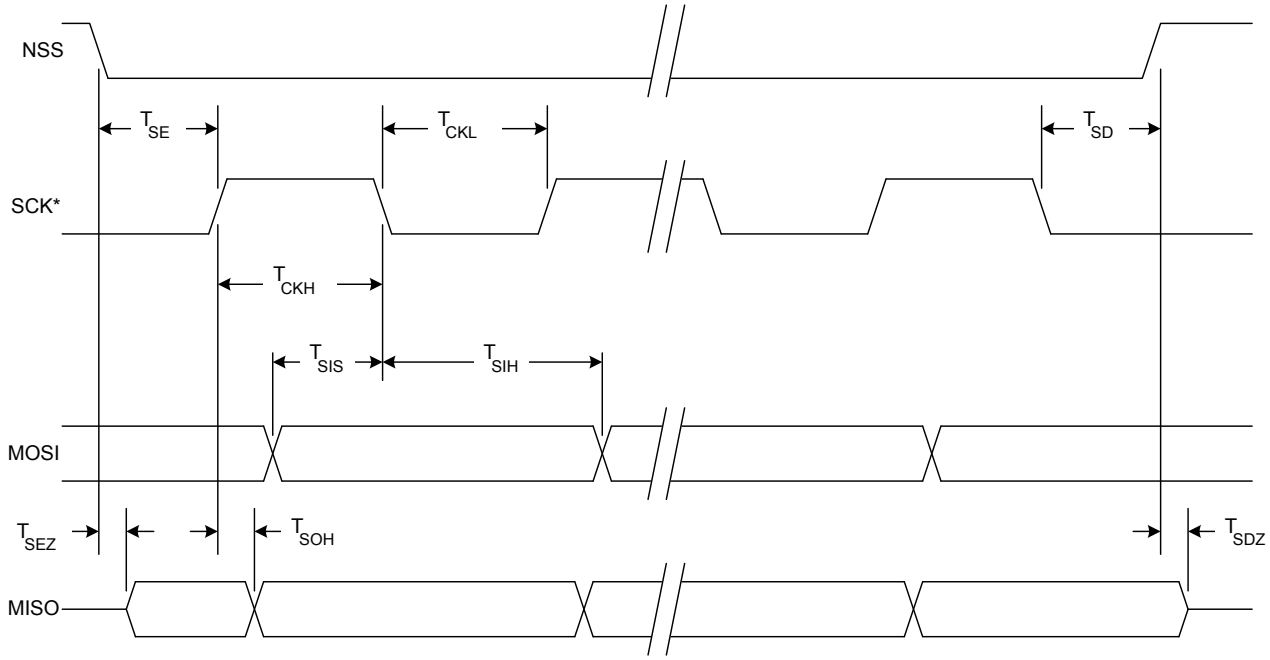
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 19.9. SPI Master Timing (CKPHA = 1)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 19.10. SPI Slave Timing (CKPHA = 0)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 19.11. SPI Slave Timing (CKPHA = 1)

Table 19.1. SPI Timing Parameters

Parameter	Description	Min	Max	Units
Master Mode Timing				
T_{MCKH}	SCK High Time	1 x T_{SYSCLK}	—	ns
T_{MCKL}	SCK Low Time	1 x T_{SYSCLK}	—	ns
T_{MIS}	MISO Valid to SCK Sample Edge	20	—	ns
T_{MIH}	SCK Sample Edge to MISO Change	5	—	ns
Slave Mode Timing				
T_{SE}	NSS Falling to First SCK Edge	5	—	ns
T_{SD}	Last SCK Edge to NSS Rising	5	—	ns
T_{SEZ}	NSS Falling to MISO Valid	—	20	ns
T_{SDZ}	NSS Rising to MISO High-Z	—	20	ns
T_{CKH}	SCK High Time	40	—	ns
T_{CKL}	SCK Low Time	40	—	ns
T_{SIS}	MOSI Valid to SCK Sample Edge	20	—	ns
T_{SIH}	SCK Sample Edge to MOSI Change	5	—	ns
T_{SOH}	SCK Shift Edge to MISO Change	—	20	ns

Note:
1. T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).

19.4 SPI0 Control Registers

19.4.1 SPI0CFG: SPI0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	SPIBSY	MSTEN	CKPHA	CKPOL	SLVSEL	NSSIN	SRMT	RXE
Access	R	RW	RW	RW	R	R	R	R
Reset	0	0	0	0	0	1	1	1
SFR Page = 0x0, 0x20; SFR Address: 0xA1								

Bit	Name	Reset	Access	Description
7	SPIBSY	0	R	**SPI Busy.** This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode).
6	MSTEN	0	RW	**Master Mode Enable.**
	Value	Name		Description
	0	MASTER_DISABLED		Disable master mode. Operate in slave mode.
	1	MASTER_ENABLED		Enable master mode. Operate as a master.
5	CKPHA	0	RW	**SPI0 Clock Phase.**
	Value	Name		Description
	0	DATA_CEN-TERED_FIRST		Data centered on first edge of SCK period.
	1	DATA_CEN-TERED_SECOND		Data centered on second edge of SCK period.
4	CKPOL	0	RW	**SPI0 Clock Polarity.**
	Value	Name		Description
	0	IDLE_LOW		SCK line low in idle state.
	1	IDLE_HIGH		SCK line high in idle state.
3	SLVSEL	0	R	**Slave Selected Flag.** This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input.
2	NSSIN	1	R	**NSS Instantaneous Pin Input.** This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched.
1	SRMT	1	R	**Shift Register Empty.** This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK.
0	RXE	1	R	**RX FIFO Empty.** This bit indicates when the RX FIFO is empty. If a read is performed when RXE is set, the last byte will be returned.

Bit	Name	Reset	Access	Description
	Value	Name		Description
	0	NOT_EMPTY		The RX FIFO contains data.
	1	EMPTY		The RX FIFO is empty.

19.4.2 SPI0CN0: SPI0 Control

Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSMD		TXNF	SPIEN
Access	RW	RW	RW	RW	RW		R	RW
Reset	0	0	0	0	0x1		1	0
SFR Page = 0x0, 0x20; SFR Address: 0xF8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	SPIF	0	RW	**SPI0 Interrupt Flag.** This bit is set to logic 1 by hardware at the end of a data transfer. If SPIF interrupts are enabled with the SPIFEN bit, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by firmware.
6	WCOL	0	RW	**Write Collision Flag.** This bit is set to logic 1 if a write to SPI0DAT is attempted when TXNF is 0. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by firmware.
5	MODF	0	RW	**Mode Fault Flag.** This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD = 01). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by firmware.
4	RXOVRN	0	RW	**Receive Overrun Flag.** This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by firmware.
3:2	NSSMD	0x1	RW	**Slave Select Mode.** Selects between the following NSS operation modes:

Value	Name	Description
0x0	3_WIRE	3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin.
0x1	4_WIRE_SLAVE	4-Wire Slave or Multi-Master Mode. NSS is an input to the device.
0x2	4_WIRE_MASTER_NSS_LOW	4-Wire Single-Master Mode. NSS is an output and logic low.
0x3	4_WIRE_MASTER_NSS_HIGH	4-Wire Single-Master Mode. NSS is an output and logic high.

1	TXNF	1	R	**TX FIFO Not Full.** This bit indicates when the TX FIFO is full and can no longer be written to. If a write is performed when TXF is cleared to 0, a WCOL error will be generated.

Value	Name	Description
0	FULL	The TX FIFO is full.
1	NOT_FULL	The TX FIFO has room for more data.

0	SPIEN	0	RW	**SPI0 Enable.**

Value	Name	Description
0	DISABLED	Disable the SPI module.

Bit	Name	Reset	Access	Description
	1	ENABLED		Enable the SPI module.

19.4.3 SPI0CKR: SPI0 Clock Rate

Bit	7	6	5	4	3	2	1	0
Name	SPI0CKR							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x20; SFR Address: 0xA2								

Bit	Name	Reset	Access	Description
7:0	SPI0CKR	0x00	RW	**SPI0 Clock Rate.** These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where SYSCLK is the system clock frequency and SPI0CKR is the 8-bit value held in the SPI0CKR register. fsck = SYSCLK / (2 * (SPI0CKR + 1)) for 0 <= SPI0CKR <= 255

19.4.4 SPI0DAT: SPI0 Data

Bit	7	6	5	4	3	2	1	0
Name	SPI0DAT							
Access	RW							
Reset	Varies							
SFR Page = 0x0, 0x20; SFR Address: 0xA3								

Bit	Name	Reset	Access	Description
7:0	SPI0DAT	Varies	RW	**SPI0 Transmit and Receive Data.** The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0DAT places the data into the transmit buffer and initiates a transfer when in master mode. A read of SPI0DAT returns the contents of the receive buffer.

19.4.5 SPI0FCN0: SPI0 FIFO Control 0

Bit	7	6	5	4	3	2	1	0
Name	TFRQE	TFLSH	TXTH		RFRQE	RFLSH	RXTH	
Access	RW	RW	RW		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x20; SFR Address: 0x9A								

Bit	Name	Reset	Access	Description
7	TFRQE	0	RW	**Write Request Interrupt Enable.**
	When set to 1, a SPI0 interrupt will be generated any time TFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		SPI0 interrupts will not be generated when TFRQ is set.
	1	ENABLED		SPI0 interrupts will be generated if TFRQ is set.
6	TFLSH	0	RW	**TX FIFO Flush.**
	This bit flushes the TX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will not be sent. Hardware will clear the TFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
5:4	TXTH	0x0	RW	**TX FIFO Threshold.**
	This field configures when hardware will set the transmit FIFO request bit (TFRQ). TFRQ is set whenever the number of bytes in the TX FIFO is equal to or less than the value in TXTH.			
	Value	Name		Description
	0x0	ZERO		TFRQ will be set when the TX FIFO is empty.
	0x1	ONE		TFRQ will be set when the TX FIFO contains one or fewer bytes.
3	RFRQE	0	RW	**Read Request Interrupt Enable.**
	When set to 1, a SPI0 interrupt will be generated any time RFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		SPI0 interrupts will not be generated when RFRQ is set.
	1	ENABLED		SPI0 interrupts will be generated if RFRQ is set.
2	RFLSH	0	RW	**RX FIFO Flush.**
	This bit flushes the RX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will be lost. Hardware will clear the RFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
1:0	RXTH	0x0	RW	**RX FIFO Threshold.**
	This field configures when hardware will set the receive FIFO request bit (RFRQ). RFRQ is set whenever the number of bytes in the RX FIFO exceeds the value in RXTH.			
	Value	Name		Description
	0x0	ZERO		RFRQ will be set anytime new data arrives in the RX FIFO (when the RX FIFO is not empty).
	0x1	ONE		RFRQ will be set if the RX FIFO contains more than one byte.

19.4.6 SPI0FCN1: SPI0 FIFO Control 1

Bit	7	6	5	4	3	2	1	0
Name	TFRQ	THPOL	TXHOLD	SPIFEN	RFRQ	Reserved	RXTOE	RXFIFOE
Access	R	RW	RW	RW	R	R	RW	RW
Reset	1	1	0	1	0	0	0	1

SFR Page = 0x20; SFR Address: 0x9B

Bit	Name	Reset	Access	Description
7	TFRQ	1	R	**Transmit FIFO Request.**

Set to 1 by hardware when the number of bytes in the TX FIFO is less than or equal to the TX FIFO threshold (TXTH).

Value	Name	Description
0	NOT_SET	The number of bytes in the TX FIFO is greater than TXTH.
1	SET	The number of bytes in the TX FIFO is less than or equal to TXTH.

Bit	Name	Reset	Access	Description
6	THPOL	1	RW	**Transmit Hold Polarity.**

Selects the polarity of the data out signal when TXHOLD is active.

Value	Name	Description
0	HOLD_0	Data output will be held at logic low when TXHOLD is set.
1	HOLD_1	Data output will be held at logic high when TXHOLD is set.

Bit	Name	Reset	Access	Description
5	TXHOLD	0	RW	**Transmit Hold.**

This bit allows firmware to stall transmission of bytes from the TX FIFO until cleared. When set, the SPI will complete any byte transmission in progress, but any new transfers will be 0xFF, and not pull data from the TX FIFO. Bytes will continue to be pulled from the TX FIFO when the TXHOLD bit is cleared.

Value	Name	Description
0	CONTINUE	The UART will continue to transmit any available data in the TX FIFO.
1	HOLD	The UART will not transmit any new data from the TX FIFO.

Bit	Name	Reset	Access	Description
4	SPIFEN	1	RW	**SPIF Interrupt Enable.**

When set to 1, a SPI0 interrupt will be generated any time SPIF is set to 1.

Value	Name	Description
0	DISABLED	SPI0 interrupts will not be generated when SPIF is set.
1	ENABLED	SPI0 interrupts will be generated if SPIF is set.

Bit	Name	Reset	Access	Description
3	RFRQ	0	R	**Receive FIFO Request.**

Set to 1 by hardware when the number of bytes in the RX FIFO is larger than specified by the RX FIFO threshold (RXTH).

Value	Name	Description
0	NOT_SET	The number of bytes in the RX FIFO is less than or equal to RXTH.
1	SET	The number of bytes in the RX FIFO is greater than RXTH.

Bit	Name	Description
2	*Reserved*	*Must write reset value.*

Bit	Name	Reset	Access	Description
1	RXTOE	0	RW	**Receive Timeout Enable.**
	This bit enables the RX FIFO timeout function. If the RX FIFO is not empty, the number of bytes in the FIFO is not enough to generate a Receive FIFO request, and the timeout is reached, a SPI0 interrupt will be generated.			
	Value	Name		Description
	0	DISABLED		Lingering bytes in the RX FIFO will not generate an interrupt.
	1	ENABLED		Lingering bytes in the RX FIFO will generate an interrupt after timeout.
0	RXFIFOE	1	RW	**Receive FIFO Enable.**
	This bit enables the SPI receive FIFO. When enabled, any received bytes will be placed into the RX FIFO.			
	Value	Name		Description
	0	DISABLED		Received bytes will be discarded.
	1	ENABLED		Received bytes will be placed in the RX FIFO.

19.4.7 SPI0FCT: SPI0 FIFO Count

Bit	7	6	5	4	3	2	1	0
Name	Reserved	TXCNT			Reserved	RXCNT		
Access	R	R			R	R		
Reset	0	0x0			0	0x0		
SFR Page = 0x20; SFR Address: 0xF7								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	TXCNT	0x0	R	**TX FIFO Count.**
	This field indicates the number of bytes in the transmit FIFO.			
3	*Reserved*	*Must write reset value.*		
2:0	RXCNT	0x0	R	**RX FIFO Count.**
	This field indicates the number of bytes in the receive FIFO.			

19.4.8 SPI0PCF: SPI0 Pin Configuration

Bit	7	6	5	4	3	2	1	0
Name	Reserved					SCKSEL	MISEL	SISEL
Access	R					RW	RW	RW
Reset	0x00					0	0	0
SFR Page = 0x20; SFR Address: 0xDF								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
2	SCKSEL	0	RW	**Slave Clock Input Select.** This bit selects the source of the SCK clock input signal in slave mode.
	Value	Name		Description
	0	CROSSBAR		SCK (slave mode clock input) is connected to the pin assigned by the crossbar.
	1	CLU1		SCK (slave mode clock input) is connected to the CLU1 output.
1	MISEL	0	RW	**Master Data Input Select.** This bit selects the source of the MISO data input signal in master mode.
	Value	Name		Description
	0	CROSSBAR		MISO (master mode data input) is connected to the pin assigned by the crossbar.
	1	CLU2		MISO (master mode data input) is connected to the CLU2 output.
0	SISEL	0	RW	**Slave Data Input Select.** This bit selects the source of the MOSI data input signal in slave mode.
	Value	Name		Description
	0	CROSSBAR		MOSI (slave mode data input) is connected to the pin assigned by the crossbar.
	1	CLU3		MOSI (slave mode data input) is connected to the CLU3 output.

20. System Management Bus / I2C (SMB0)

20.1 Introduction

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I^2C serial bus.

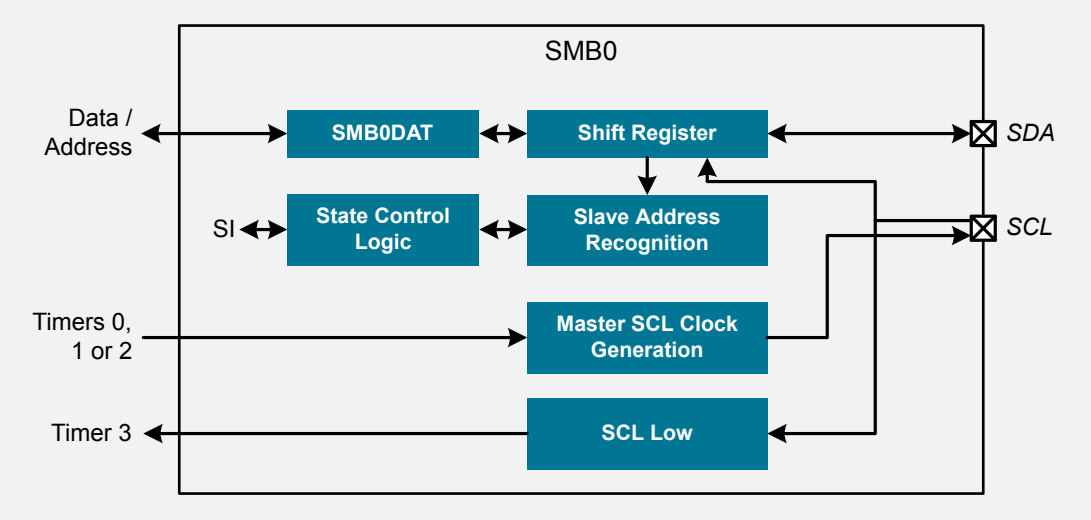


Figure 20.1. SMBus 0 Block Diagram

20.2 Features

The SMBus module includes the following features:

- Standard (up to 100 kbps) and Fast (400 kbps) transfer speeds
- Support for master, slave, and multi-master modes
- Hardware synchronization and arbitration for multi-master mode
- Clock low extending (clock stretching) to interface with faster masters
- Hardware support for 7-bit slave and general call address recognition
- Firmware support for 10-bit slave address decoding
- Ability to inhibit all slave states
- Programmable data setup/hold times
- Transmit and receive FIFOs (one byte) to help increase throughput in faster applications

20.3 Functional Description

20.3.1 Supporting Documents

It is assumed the reader is familiar with or has access to the following supporting documents:

- The I^2C-Bus and How to Use It (including specifications), Philips Semiconductor.
- The I^2C-Bus Specification—Version 2.0, Philips Semiconductor.
- System Management Bus Specification—Version 1.1, SBS Implementers Forum.

20.3.2 SMBus Protocol

The SMBus specification allows any recessive voltage between 3.0 and 5.0 V; different devices on the bus may operate at different voltage levels. However, the maximum voltage on any port pin must conform to the electrical characteristics specifications. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

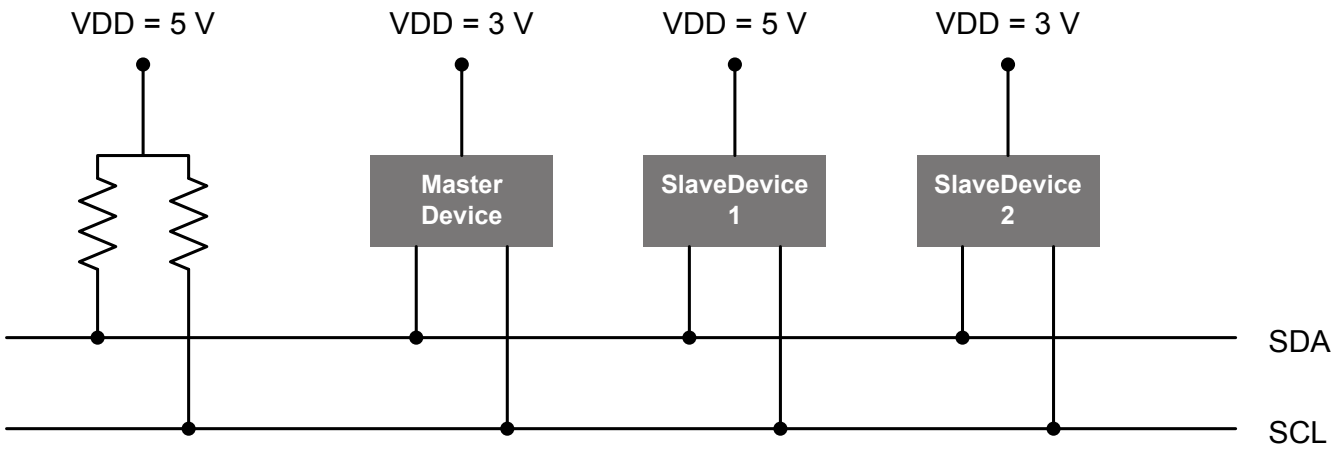


Figure 20.2. Typical SMBus System Connection

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. It is not necessary to specify one device as the Master in a system; any device who transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 20.3 SMBus Transaction on page 309). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address byte. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation.

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 20.3 SMBus Transaction on page 309 illustrates a typical SMBus transaction.

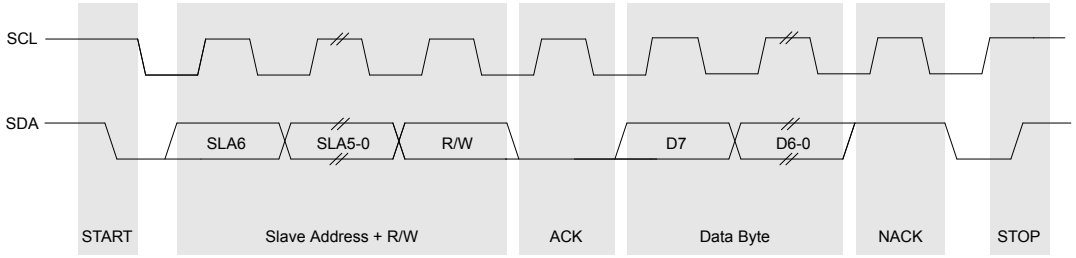


Figure 20.3. SMBus Transaction

Transmitter vs. Receiver

On the SMBus communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see ● SCL High (SMBus Free) Timeout on page 309). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I^2C, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

For the SMBus 0 interface, Timer 3 is used to implement SCL low timeouts. The SCL low timeout feature is enabled by setting the SMB0TOE bit in SMB0CF. The associated timer is forced to reload when SCL is high, and allowed to count when SCL is low. With the associated timer enabled and configured to overflow after 25 ms (and SMB0TOE set), the timer interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMB0FTE bit in SMB0CF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods (as defined by the timer configured for the SMBus clock source). If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. A clock source is required for free timeout detection, even in a slave-only implementation.

20.3.3 Configuring the SMBus Module

The SMBus can operate in both Master and Slave modes. The interface provides timing and shifting control for serial transfers; higher level protocol is determined by user software. The SMBus interface provides the following application-independent features:

- Byte-wise serial data transfers
- Clock signal generation on SCL (Master Mode only) and SDA data synchronization
- Timeout/bus error recognition, as defined by the SMB0CF configuration register
- START/STOP timing, detection, and generation
- Bus arbitration
- Interrupt generation
- Status information
- Optional hardware recognition of slave address and automatic acknowledgement of address/data

SMBus interrupts are generated for each data byte or slave address that is transferred. When hardware acknowledgement is disabled, the point at which the interrupt is generated depends on whether the hardware is acting as a data transmitter or receiver. When a transmitter (i.e., sending address/data, receiving an ACK), this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data (i.e., receiving address/data, sending an ACK), this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. If hardware acknowledgement is enabled, these interrupts are always generated after the ACK cycle. Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN0 register to find the cause of the SMBus interrupt.

SMBus Configuration Register

The SMBus Configuration register (SMB0CF) is used to enable the SMBus master and/or slave modes, select the SMBus clock source, and select the SMBus timing and timeout options. When the ENSMB bit is set, the SMBus is enabled for all master and slave events. Slave events may be disabled by setting the INH bit. With slave events inhibited, the SMBus interface will still monitor the SCL and SDA pins; however, the interface will NACK all received addresses and will not generate any slave interrupts. When the INH bit is set, all slave events will be inhibited following the next START (interrupts will continue for the duration of the current transfer).

The SMBCS bit field selects the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine both the bit rate and the absolute minimum SCL low and high times. The selected clock source may be shared by other peripherals so long as the timer is left running at all times. The selected clock source should typically be configured to overflow at three times the desired bit rate. When the interface is operating as a master (and SCL is not driven or extended by any other devices on the bus), the device will hold the SCL line low for one overflow period, and release it for two overflow periods. T_{HIGH} is typically twice as large as T_{LOW}. The actual SCL output may vary due to other devices on the bus (SCL may be extended low by slower slave devices, driven low by contending master devices, or have long ramp times). The SMBus hardware will ensure that once SCL does return high, it reads a logic high state for a minimum of one overflow period.

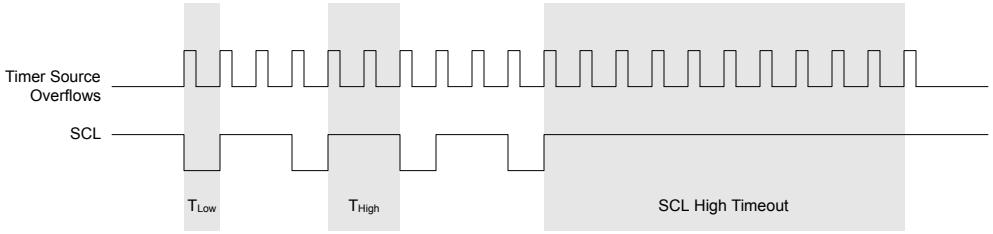


Figure 20.4. Typical SMBus SCL Generation

Setting the EXTHOLD bit extends the minimum setup and hold times for the SDA line. The minimum SDA setup time defines the absolute minimum time that SDA is stable before SCL transitions from low-to-high. The minimum SDA hold time defines the absolute minimum time that the current SDA value remains stable after SCL transitions from high-to-low. EXTHOLD should be set so that the minimum setup and hold times meet the SMBus Specification requirements of 250 ns and 300 ns, respectively. Setup and hold time extensions are typically necessary for SMBus compliance when SYSCLK is above 10 MHz.

Table 20.1. Minimum SDA Setup and Hold Times

EXTHOLD	Minimum SDA Setup Time	Minimum SDA Hold Time
0	T_{low} – 4 system clocks or 1 system clock + s/w delay	3 system clocks
1	11 system clocks	12 system clocks
Note: Setup Time for ACK bit transmissions and the MSB of all data transfers. When using software acknowledgment, the s/w delay occurs between the time SMB0DAT or ACK is written and when SI is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero.		

With the SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts. The SMBus interface will force the associated timer to reload while SCL is high, and allow the timer to count when SCL is low. The timer interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus. SMBus Free Timeout detection can be enabled by setting the SMBFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods.

SMBus Pin Swap

The SMBus peripheral is assigned to pins using the priority crossbar decoder. By default, the SMBus signals are assigned to port pins starting with SDA on the lower-numbered pin, and SCL on the next available pin. The SWAP bit in the SMBus Timing Control register can be set to 1 to reverse the order in which the SMBus signals are assigned.

SMBus Timing Control

The SDD field in the SMBus Timing Control register is used to delay the recognition of the falling edge of the SDA signal. This feature should be applied in cases where a data bit transition occurs close to the SCL falling edge that may cause a false START detection when there is a significant mismatch between the impedance or capacitance on the SDA and SCL lines. This feature should also be applied to improve the recognition of the repeated START bit when the SCL bus capacitance is very high. These kinds of events are not expected in a standard SMBus- or I2C-compliant system.

Note: In most systems this parameter should not be adjusted, and it is recommended that it be left at its default value.

The SDD field can be used to delay the recognition of the SDA falling edge by the SMBus hardware by 2, 4, or 8 SYSCLKs.

SMBus Control Register

SMB0CN0 is used to control the interface and to provide status information. The higher four bits of SMB0CN0 (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

STA and STO indicate that a START and/or STOP has been detected or generated since the last SMBus interrupt. STA and STO are also used to generate START and STOP conditions when operating as a master. Writing a 1 to STA will cause the SMBus interface to enter Master Mode and generate a START when the bus becomes free (STA is not cleared by hardware after the START is generated). Writing a 1 to STO while in Master Mode will cause the interface to generate a STOP and end the current transfer after the next ACK cycle. If STO and STA are both set (while in Master Mode), a STOP followed by a START will be generated.

The ARBLOST bit indicates that the interface has lost an arbitration. This may occur anytime the interface is transmitting (master or slave). A lost arbitration while operating as a slave indicates a bus error condition. ARBLOST is cleared by hardware each time SI is cleared.

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost.

Note: The SMBus interface is stalled while SI is set; if SCL is held low at this time, the bus is stalled until software clears SI.

Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 20.2. Sources for Hardware Changes to SMB0CN0

Bit	Set by Hardware When:	Cleared by Hardware When:
MASTER	A START is generated.	A STOP is generated. Arbitration is lost.
TXMODE	START is generated. SMB0DAT is written before the start of an SMBus frame.	A START is detected. Arbitration is lost[1]. SMB0DAT is not written before the start of an SMBus frame.
STA	A START followed by an address byte is received.	Must be cleared by software.
STO	A STOP is detected while addressed as a slave. Arbitration is lost due to a detected STOP.	A pending STOP is generated.
ACKRQ	A byte has been received and an ACK response value is needed (only when hardware ACK is not enabled).	After each ACK cycle.
ARBLOST	A repeated START is detected as a MASTER when STA is low (unwanted repeated START). SCL is sensed low while attempting to generate a STOP or repeated START condition. SDA is sensed low while transmitting a 1 (excluding ACK bits).	Each time SIn is cleared.
ACK	The incoming ACK value is low (ACKNOWLEDGE).	The incoming ACK value is high (NOT ACKNOWLEDGE).
SI	A START has been generated. Lost arbitration. A byte has been transmitted and an ACK/NACK received. A byte has been received. A START or repeated START followed by a slave address + R/W has been received. A STOP has been received.	Must be cleared by software.

Note:
1. When arbitration is lost, hardware clears TXMODE after software clears SI.

Hardware Slave Address Recognition

The SMBus hardware has the capability to automatically recognize incoming slave addresses and send an ACK without software intervention. Automatic slave address recognition is enabled by setting the EHACK bit in register SMB0ADM to 1. This will enable both automatic slave address recognition and automatic hardware ACK generation for received bytes (as a master or slave).

The registers used to define which address(es) are recognized by the hardware are the SMBus Slave Address register and the SMBus Slave Address Mask register. A single address or range of addresses (including the General Call Address 0x00) can be specified using these two registers. The most-significant seven bits of the two registers are used to define which addresses will be ACKed. A 1 in a bit of the slave address mask SLVM enables a comparison between the received slave address and the hardware's slave address SLV for that bit. A 0 in a bit of the slave address mask means that bit will be treated as a "don't care" for comparison purposes. In this case, either a 1 or a 0 value are acceptable on the incoming slave address. Additionally, if the GC bit in register SMB0ADR is set to 1, hardware will recognize the General Call Address (0x00).

Table 20.3. Hardware Address Recognition Examples (EHACK=1)

Hardware Slave Address SLV	Slave Address Mask SLVM	GC bit	Slave Addresses Recognized by Hardware
0x34	0x7F	0	0x34
0x34	0x7F	1	0x34, 0x00 (General Call)
0x34	0x7E	0	0x34, 0x35
0x34	0x7E	1	0x34, 0x35, 0x00 (General Call)
0x70	0x73	0	0x70, 0x74, 0x78, 0x7C
Note: These addresses must be shifted to the left by one bit when writing to the SMB0ADR register.			

Software ACK Generation

In general, it is recommended for applications to use hardware ACK and address recognition. In some cases it may be desirable to drive ACK generation and address recognition from firmware. When the EHACK bit in register SMB0ADM is cleared to 0, the firmware on the device must detect incoming slave addresses and ACK or NACK the slave address and incoming data bytes. As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received during the last ACK cycle. ACKRQ is set each time a byte is received, indicating that an outgoing ACK value is needed. When ACKRQ is set, software should write the desired outgoing value to the ACK bit before clearing SI. A NACK will be generated if software does not write the ACK bit before clearing SI. SDA will reflect the defined ACK value immediately following a write to the ACK bit; however SCL will remain low until SI is cleared. If a received slave address is not acknowledged, further slave events will be ignored until the next START is detected.

SMBus Data Register

The SMBus Data register SMB0DAT holds a byte of serial data to be transmitted or one that has just been received. Software may safely read or write to the data register when the SI flag is set. Software should not attempt to access the SMB0DAT register when the SMBus is enabled and the SI flag is cleared to logic 0.

Note: Certain device families have a transmit and receive buffer interface which is accessed by reading and writing the SMB0DAT register. To promote software portability between devices with and without this buffer interface it is recommended that SMB0DAT not be used as a temporary storage location. On buffer-enabled devices, writing the register multiple times will push multiple bytes into the transmit FIFO.

20.3.4 Operational Modes

The SMBus interface may be configured to operate as master and/or slave. At any particular time, it will be operating in one of the following four modes: Master Transmitter, Master Receiver, Slave Transmitter, or Slave Receiver. The SMBus interface enters Master Mode any time a START is generated, and remains in Master Mode until it loses an arbitration or generates a STOP. An SMBus interrupt is generated at the end of all SMBus byte frames. The position of the ACK interrupt when operating as a receiver depends on whether hardware ACK generation is enabled. As a receiver, the interrupt for an ACK occurs before the ACK with hardware ACK generation disabled, and after the ACK when hardware ACK generation is enabled. As a transmitter, interrupts occur after the ACK, regardless of whether hardware ACK generation is enabled or not.

Master Write Sequence

During a write sequence, an SMBus master writes data to a slave device. The master in this transfer will be a transmitter during the address byte, and a transmitter during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 0 (WRITE). The master then transmits one or more bytes of serial data. After each byte is transmitted, an acknowledge bit is generated by the slave. The transfer is ended when the STO bit is set and a STOP is generated. The interface will switch to Master Receiver Mode if SMB0DAT is not written following a Master Transmitter interrupt. Figure 20.5 Typical Master Write Sequence on page 316 shows a typical master write sequence as it appears on the bus, and Figure 20.6 Master Write Sequence State Diagram (EHACK = 1) on page 317 shows the corresponding firmware state machine. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur after the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

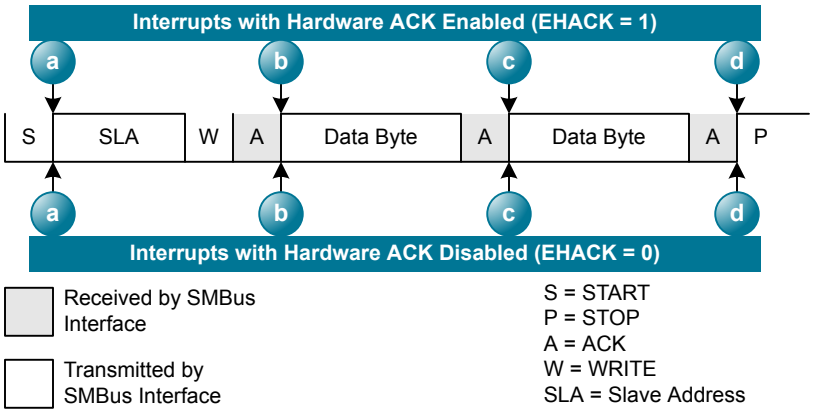


Figure 20.5. Typical Master Write Sequence

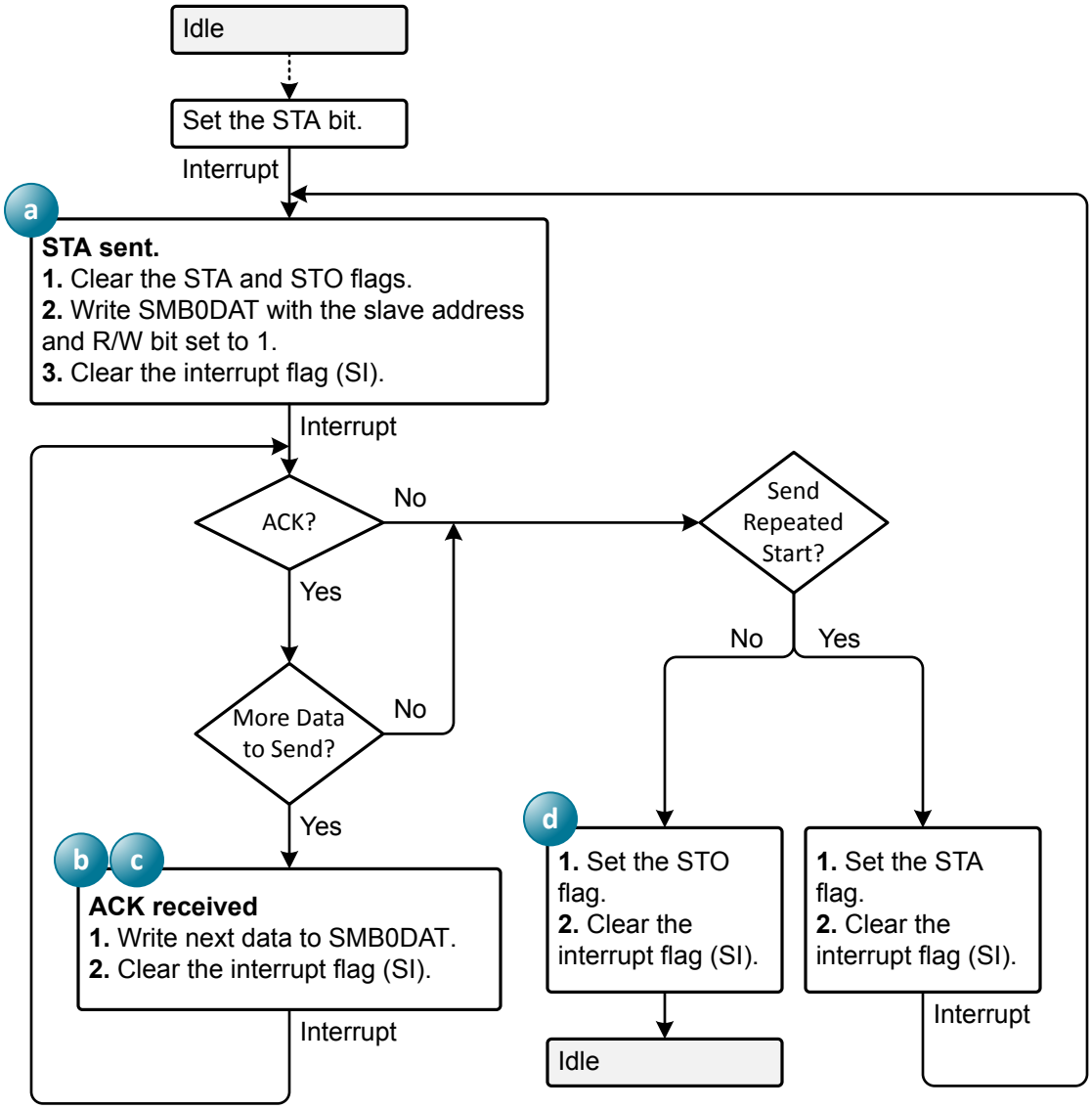


Figure 20.6. Master Write Sequence State Diagram (EHACK = 1)

Master Read Sequence

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.

Writing a 1 to the ACK bit generates an ACK; writing a 0 generates a NACK. Software should write a 0 to the ACK bit for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 20.7 Typical Master Read Sequence on page 318 shows a typical master read sequence as it appears on the bus, and Figure 20.8 Master Read Sequence State Diagram (EHACK = 1) on page 319 shows the corresponding firmware state machine. Two received data bytes are shown, though any number of bytes may be received. Notice that the "data byte transferred" interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs before the ACK with hardware ACK generation disabled, and after the ACK when hardware ACK generation is enabled.

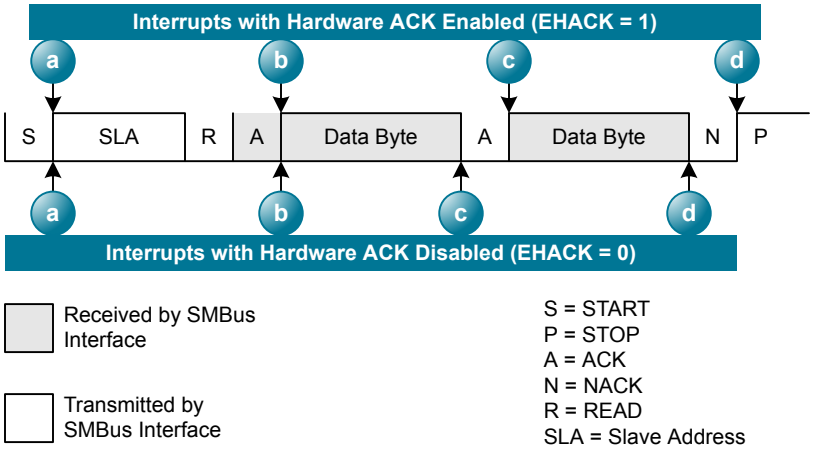


Figure 20.7. Typical Master Read Sequence

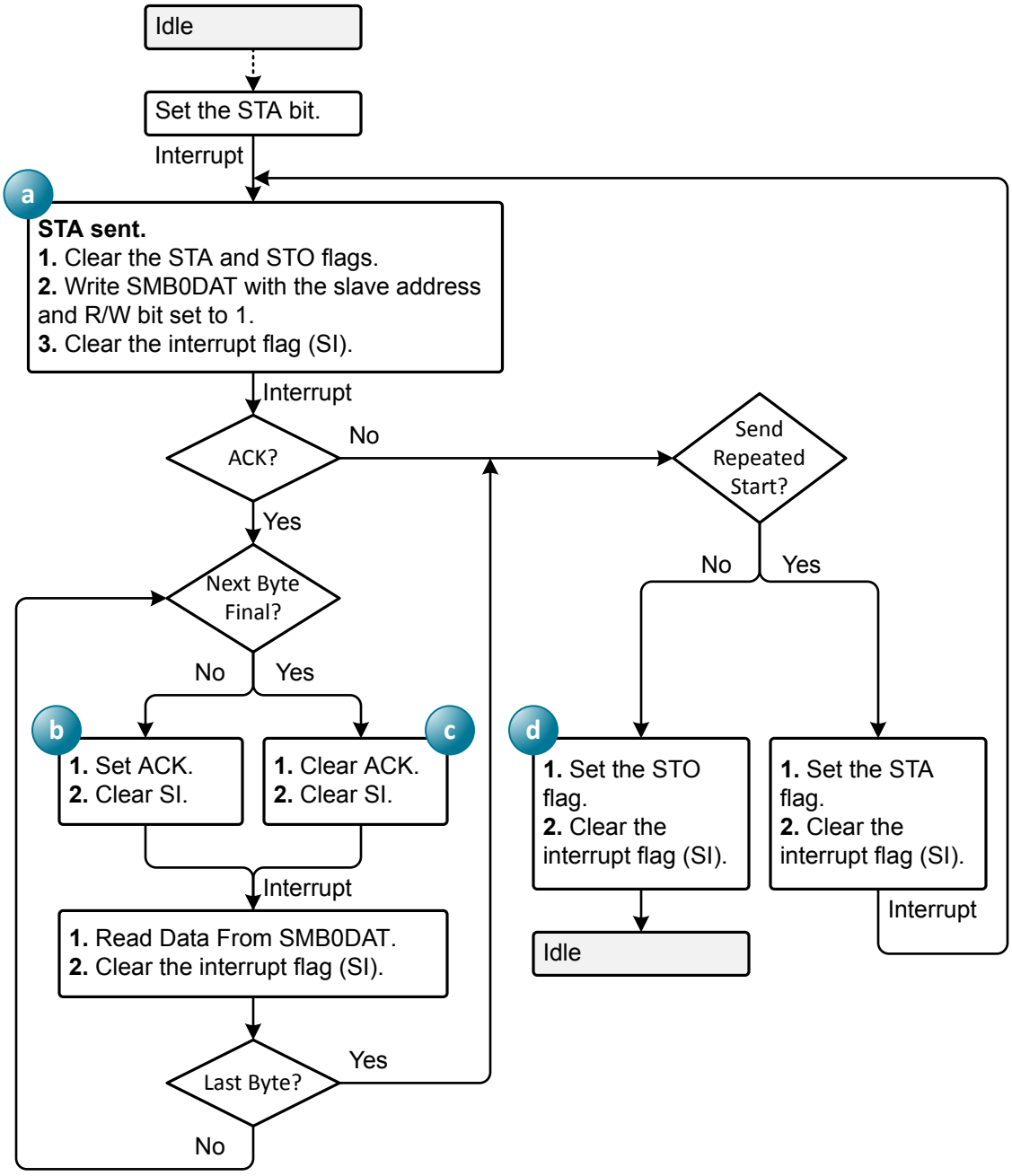


Figure 20.8. Master Read Sequence State Diagram (EHACK = 1)

Slave Write Sequence

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.

The interface exits Slave Receiver Mode after receiving a STOP. The interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 20.9 Typical Slave Write Sequence on page 320 shows a typical slave write sequence as it appears on the bus. The corresponding firmware state diagram (combined with the slave read sequence) is shown in Figure 20.10 Slave State Diagram (EHACK = 1) on page 321. Two received data bytes are shown, though any number of bytes may be received. Notice that the "data byte transferred" interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs before the ACK with hardware ACK generation disabled, and after the ACK when hardware ACK generation is enabled.

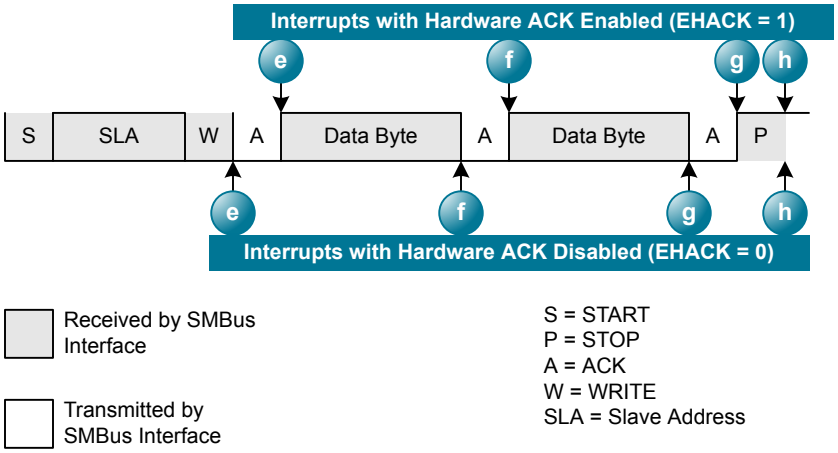


Figure 20.9. Typical Slave Write Sequence

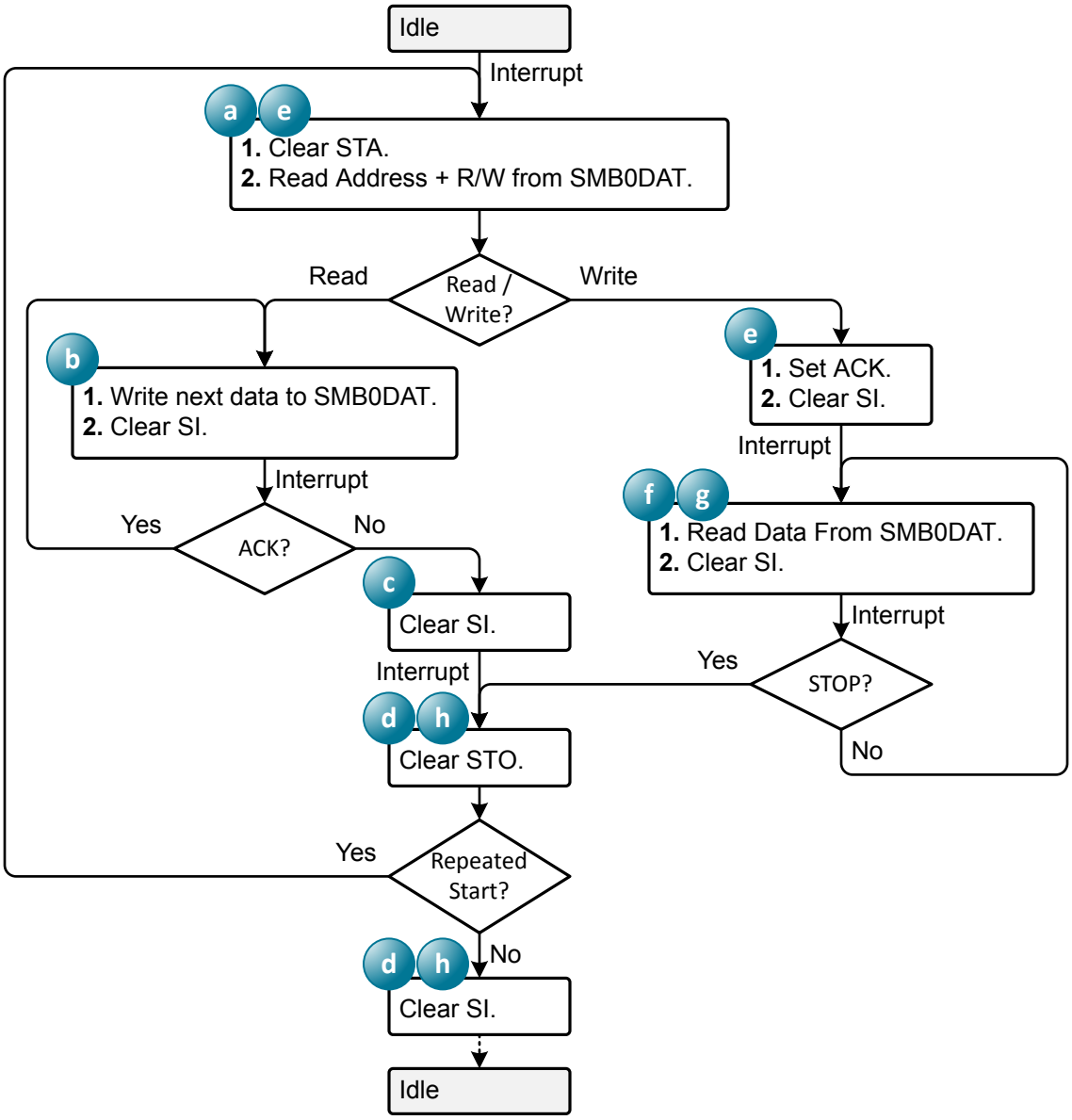


Figure 20.10. Slave State Diagram (EHACK = 1)

Slave Read Sequence

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters slave transmitter mode, and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (an error condition may be generated if SMB0DAT is written following a received NACK while in slave transmitter mode). The interface exits slave transmitter mode after receiving a STOP. The interface will switch to slave receiver mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 20.11 Typical Slave Read Sequence on page 322 shows a typical slave read sequence as it appears on the bus. The corresponding firmware state diagram (combined with the slave read sequence) is shown in Figure 20.10 Slave State Diagram (EHACK = 1) on page 321. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur after the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

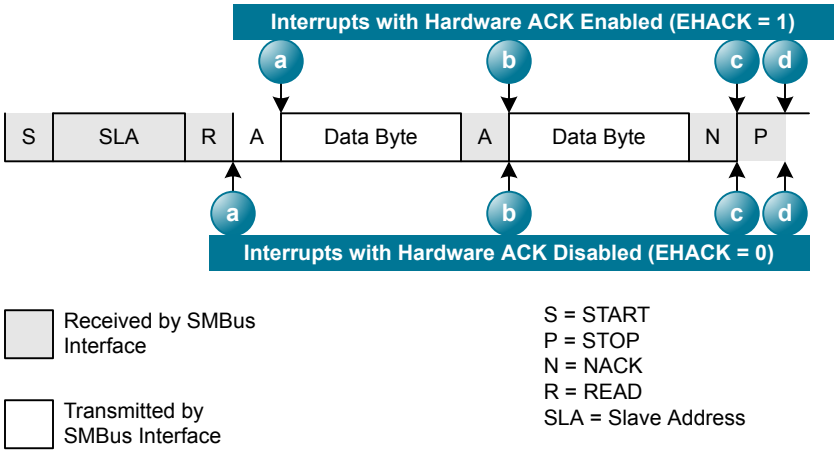


Figure 20.11. Typical Slave Read Sequence

20.4 SMB0 Control Registers

20.4.1 SMB0CF: SMBus 0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	ENSMB	INH	BUSY	EXTHOLD	SMBTOE	SMBFTE	SMBCS	
Access	RW	RW	R	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0x0	
SFR Page = 0x0, 0x20; SFR Address: 0xC1								

Bit	Name	Reset	Access	Description
7	ENSMB	0	RW	**SMBus Enable.** This bit enables the SMBus interface when set to 1. When enabled, the interface constantly monitors the SDA and SCL pins.
6	INH	0	RW	**SMBus Slave Inhibit.** When this bit is set to logic 1, the SMBus does not generate an interrupt when slave events occur. This effectively removes the SMBus slave from the bus. Master Mode interrupts are not affected.
5	BUSY	0	R	**SMBus Busy Indicator.** This bit is set to logic 1 by hardware when a transfer is in progress. It is cleared to logic 0 when a STOP or free-timeout is sensed.
4	EXTHOLD	0	RW	**SMBus Setup and Hold Time Extension Enable.** This bit enables SDA setup and hold time extensions for SMBus operation. When set, the DLYEXT bit controls the length of the SDA setup and hold times.

Value	Name	Description
0	DISABLED	Disable SDA extended setup and hold times.
1	ENABLED	Enable SDA extended setup and hold times.

Bit	Name	Reset	Access	Description
3	SMBTOE	0	RW	**SMBus SCL Timeout Detection Enable.** This bit enables SCL low timeout detection. If set to logic 1 and RLFSEL in TMR3CN1 is set correctly, the SMBus forces Timer 3 to reload while SCL is high and allows Timer 3 to count when SCL goes low. If Timer 3 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL is high. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus communication.
2	SMBFTE	0	RW	**SMBus Free Timeout Detection Enable.** When this bit is set to logic 1, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods.
1:0	SMBCS	0x0	RW	**SMBus Clock Source Selection.** This field selects the SMBus clock source, which is used to generate the SMBus bit rate. See the SMBus clock timing section for additional details.

Value	Name	Description
0x0	TIMER0	Timer 0 Overflow.
0x1	TIMER1	Timer 1 Overflow.
0x2	TIMER2_HIGH	Timer 2 High Byte Overflow.
0x3	TIMER2_LOW	Timer 2 Low Byte Overflow.

20.4.2 SMB0TC: SMBus 0 Timing and Pin Control

Bit	7	6	5	4	3	2	1	0
Name	SWAP	Reserved		DLYEXT	Reserved		SDD	
Access	RW	R		RW	R		RW	
Reset	0	0x0		0	0x0		0x0	
SFR Page = 0x0, 0x20; SFR Address: 0xAC								

Bit	Name	Reset	Access	Description
7	SWAP	0	RW	**SMBus Swap Pins.**

This bit swaps the order of the SMBus pins on the crossbar.

Value	Name	Description
0	SDA_LOW_PIN	SDA is mapped to the lower-numbered port pin, and SCL is mapped to the higher-numbered port pin.
1	SDA_HIGH_PIN	SCL is mapped to the lower-numbered port pin, and SDA is mapped to the higher-numbered port pin.

6:5	*Reserved*	*Must write reset value.*		
4	DLYEXT	0	RW	**Setup and Hold Delay Extension.**

This bit selects how long the setup and hold times will be extended when EXTHOLD is set to 1.

Value	Name	Description
0	STANDARD	SDA setup time is 11 SYSCLKs and SDA hold time is 12 SYSCLKs.
1	EXTENDED	SDA setup time is 31 SYSCLKs and SDA hold time is 31 SYSCLKs.

3:2	*Reserved*	*Must write reset value.*		
1:0	SDD	0x0	RW	**SMBus Start Detection Window.**

This field is used to delay the recognition of the falling edge of the SDA signal. This feature should be applied in cases where a data bit transition occurs close to the SCL falling edge that may cause a false START detection when there is a significant mismatch between the impedance or capacitance on the SDA and SCL lines. This field should be left in the default setting in most cases.

Value	Name	Description
0x0	NONE	No additional SDA falling edge recognition delay (0-1 SYSCLK).
0x1	ADD_2_SYSCLKS	Increase SDA falling edge recognition time window to 2-3 SYSCLKs after the SCL falling edge.
0x2	ADD_4_SYSCLKS	Increase SDA falling edge recognition window to 4-5 SYSCLKs after the SCL falling edge.
0x3	ADD_8_SYSCLKS	Increase SDA falling edge recognition window to 8-9 SYSCLKs after the SCL falling edge.

20.4.3 SMB0CN0: SMBus 0 Control

Bit	7	6	5	4	3	2	1	0
Name	MASTER	TXMODE	STA	STO	ACKRQ	ARBLOST	ACK	SI
Access	R	R	RW	RW	R	R	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xC0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	MASTER	0	R	**SMBus Master/Slave Indicator.**
	This read-only bit indicates when the SMBus is operating as a master.			
	Value	Name		Description
	0	SLAVE		SMBus operating in slave mode.
	1	MASTER		SMBus operating in master mode.
6	TXMODE	0	R	**SMBus Transmit Mode Indicator.**
	This read-only bit indicates when the SMBus is operating as a transmitter.			
	Value	Name		Description
	0	RECEIVER		SMBus in Receiver Mode.
	1	TRANSMITTER		SMBus in Transmitter Mode.
5	STA	0	RW	**SMBus Start Flag.**
	When reading STA, a '1' indicates that a start or repeated start condition was detected on the bus.			
	Writing a '1' to the STA bit initiates a start or repeated start on the bus.			
4	STO	0	RW	**SMBus Stop Flag.**
	When reading STO, a '1' indicates that a stop condition was detected on the bus (in slave mode) or is pending (in master mode).			
	When acting as a master, writing a '1' to the STO bit initiates a stop condition on the bus. This bit is cleared by hardware.			
3	ACKRQ	0	R	**SMBus Acknowledge Request.**
	Value	Name		Description
	0	NOT_SET		No ACK requested.
	1	REQUESTED		ACK requested.
2	ARBLOST	0	R	**SMBus Arbitration Lost Indicator.**
	Value	Name		Description
	0	NOT_SET		No arbitration error.
	1	ERROR		Arbitration error occurred.

Bit	Name	Reset	Access	Description
1	ACK	0	RW	**SMBus Acknowledge.** When read as a master, the ACK bit indicates whether an ACK (1) or NACK (0) is received during the most recent byte transfer. As a slave, this bit should be written to send an ACK (1) or NACK (0) to a master request. Note that the logic level of the ACK bit on the SMBus interface is inverted from the logic of the register ACK bit.
0	SI	0	RW	**SMBus Interrupt Flag.** This bit is set by hardware to indicate that the current SMBus state machine operation (such as writing a data or address byte) is complete, and the hardware needs additional control from the firmware to proceed. While SI is set, SCL is held low and SMBus is stalled. SI must be cleared by firmware. Clearing SI initiates the next SMBus state machine operation.

20.4.4 SMB0ADR: SMBus 0 Slave Address

Bit	7	6	5	4	3	2	1	0
Name	SLV							GC
Access	RW							RW
Reset	0x00							0
SFR Page = 0x0, 0x20; SFR Address: 0xD7								

Bit	Name	Reset	Access	Description
7:1	SLV	0x00	RW	**SMBus Hardware Slave Address.** Defines the SMBus Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM are checked against the incoming address. This allows multiple addresses to be recognized.
0	GC	0	RW	**General Call Address Enable.** When hardware address recognition is enabled (EHACK = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware.

Value	Name	Description
0	IGNORED	General Call Address is ignored.
1	RECOGNIZED	General Call Address is recognized.

20.4.5 SMB0ADM: SMBus 0 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM							EHACK
Access	RW							RW
Reset	0x7F							0
SFR Page = 0x0, 0x20; SFR Address: 0xD6								

Bit	Name	Reset	Access	Description
7:1	SLVM	0x7F	RW	**SMBus Slave Address Mask.** Defines which bits of register SMB0ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM enables comparisons with the corresponding bit in SLV. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
0	EHACK	0	RW	**Hardware Acknowledge Enable.** Enables hardware acknowledgement of slave address and received data bytes.

Value	Name	Description
0	ADR_ACK_MANUAL	Firmware must manually acknowledge all incoming address and data bytes.
1	ADR_ACK_AUTOMAT-IC	Automatic slave address recognition and hardware acknowledge is enabled.

20.4.6 SMB0DAT: SMBus 0 Data

Bit	7	6	5	4	3	2	1	0
Name	SMB0DAT							
Access	RW							
Reset	Varies							
SFR Page = 0x0, 0x20; SFR Address: 0xC2								

Bit	Name	Reset	Access	Description
7:0	SMB0DAT	Varies	RW	**SMBus 0 Data.** The SMB0DAT register is used to access the TX and RX FIFOs. When written, data will go into the TX FIFO. Reading SMB0DAT reads data from the RX FIFO. If SMB0DAT is written when TXNF is 0, the data will over-write the last data byte present in the TX FIFO. If SMB0DAT is read when RXE is set, the last byte in the RX FIFO will be returned.

20.4.7 SMB0FCN0: SMBus 0 FIFO Control 0

Bit	7	6	5	4	3	2	1	0
Name	TFRQE	TFLSH	TXTH		RFRQE	RFLSH	RXTH	
Access	RW	RW	RW		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x20; SFR Address: 0xC3								

Bit	Name	Reset	Access	Description
7	TFRQE	0	RW	**Write Request Interrupt Enable.**
	When set to 1, an SMBus 0 interrupt will be generated any time TFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		SMBus 0 interrupts will not be generated when TFRQ is set.
	1	ENABLED		SMBus 0 interrupts will be generated if TFRQ is set.
6	TFLSH	0	RW	**TX FIFO Flush.**
	This bit flushes the TX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will not be sent. Hardware will clear the TFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
5:4	TXTH	0x0	RW	**TX FIFO Threshold.**
	This field configures when hardware will set the transmit FIFO request bit (TFRQ). TFRQ is set whenever the number of bytes in the TX FIFO is equal to or less than the value in TXTH.			
	Value	Name		Description
	0x0	ZERO		TFRQ will be set when the TX FIFO is empty.
3	RFRQE	0	RW	**Read Request Interrupt Enable.**
	When set to 1, an SMBus 0 interrupt will be generated any time RFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		SMBus 0 interrupts will not be generated when RFRQ is set.
	1	ENABLED		SMBus 0 interrupts will be generated if RFRQ is set.
2	RFLSH	0	RW	**RX FIFO Flush.**
	This bit flushes the RX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will be lost. Hardware will clear the RFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
1:0	RXTH	0x0	RW	**RX FIFO Threshold.**
	This field configures when hardware will set the receive FIFO request bit (RFRQ). RFRQ is set whenever the number of bytes in the RX FIFO exceeds the value in RXTH.			
	Value	Name		Description
	0x0	ZERO		RFRQ will be set anytime new data arrives in the RX FIFO (when the RX FIFO is not empty).

20.4.8 SMB0FCN1: SMBus 0 FIFO Control 1

Bit	7	6	5	4	3	2	1	0
Name	TFRQ	TXNF	Reserved		RFRQ	RXE	Reserved	
Access	R	R	R		R	R	R	
Reset	1	1	0x0		0	1	0x0	

SFR Page = 0x20; SFR Address: 0xC4

Bit	Name	Reset	Access	Description
7	TFRQ	1	R	**Transmit FIFO Request.**
	Set to 1 by hardware when the number of bytes in the TX FIFO is less than or equal to the TX FIFO threshold (TXTH).			
	Value	Name		Description
	0	NOT_SET		The number of bytes in the TX FIFO is greater than TXTH.
	1	SET		The number of bytes in the TX FIFO is less than or equal to TXTH.
6	TXNF	1	R	**TX FIFO Not Full.**
	This bit indicates when the TX FIFO is full and can no longer be written to. If a write is performed when TXNF is cleared to 0 it will replace the most recent byte in the FIFO.			
	Value	Name		Description
	0	FULL		The TX FIFO is full.
	1	NOT_FULL		The TX FIFO has room for more data.
5:4	*Reserved*	*Must write reset value.*		
3	RFRQ	0	R	**Receive FIFO Request.**
	Set to 1 by hardware when the number of bytes in the RX FIFO is larger than specified by the RX FIFO threshold (RXTH).			
	Value	Name		Description
	0	NOT_SET		The number of bytes in the RX FIFO is less than or equal to RXTH.
	1	SET		The number of bytes in the RX FIFO is greater than RXTH.
2	RXE	1	R	**RX FIFO Empty.**
	This bit indicates when the RX FIFO is empty. If a read is performed when RXE is set, the last byte will be returned.			
	Value	Name		Description
	0	NOT_EMPTY		The RX FIFO contains data.
	1	EMPTY		The RX FIFO is empty.
1:0	*Reserved*	*Must write reset value.*		

20.4.9 SMB0RXLN: SMBus 0 Receive Length Counter

Bit	7	6	5	4	3	2	1	0
Name	RXLN							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0xC5								

Bit	Name	Reset	Access	Description
7:0	RXLN	0x00	RW	**SMBus Receive Length Counter.** Master Receiver: This field allows firmware to set the number of bytes to receive as a master receiver (with EHACK set to 1), before stalling the bus. As long as the RX FIFO is serviced and RXLN is greater than zero, hardware will continue to read new bytes from the slave device and send ACKs. Each received byte decrements RXLN until RXLN reaches 0. If RXLN is 0 and a new byte is received, hardware will set the SI bit and stall the bus. The last byte recieved will be ACKed if the ACK bit is set to 1, or NAKed if the ACK bit is cleared to 0. Slave Receiver: When RXLN is cleared to 0, the bus will stall and generate an interrupt after every received byte, regardless of the FIFO status. Any other value programmed here will allow the FIFO to operate. RXLN is not decremented as new bytes arrive in slave receiver mode.
This register should not be modified by firmware in the middle of a transfer, except when SI = 1 and the bus is stalled.				

20.4.10 SMB0FCT: SMBus 0 FIFO Count

Bit	7	6	5	4	3	2	1	0
Name	Reserved			TXCNT	Reserved			RXCNT
Access	R			R	R			R
Reset	0x0			0	0x0			0
SFR Page = 0x20; SFR Address: 0xEF								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	TXCNT	0	R	**TX FIFO Count.** This field indicates the number of bytes in the transmit FIFO.
3:1	*Reserved*	*Must write reset value.*		
0	RXCNT	0	R	**RX FIFO Count.** This field indicates the number of bytes in the receive FIFO.

21. Timers (Timer0, Timer1, Timer2, Timer3, Timer4, and Timer5)

21.1 Introduction

Six counter/timers are included in the device: two are 16-bit counter/timers compatible with those found in the standard 8051, and four are 16-bit auto-reload timers for timing peripherals or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2, Timer 3, Timer 4 and Timer 5 are also similar, and offer both 16-bit and split 8-bit timer functionality with auto-reload capabilities. Timer 2, 3, 4, and 5 offer capture functions that may be selected from several on-chip sources or an external pin, and may also be forced to reload on CLU output signals.

Timers 0 and 1 may be clocked by one of five sources, determined by the Timer Mode Select bits (T1M–T0M) and the Clock Scale bits (SCA1–SCA0). The Clock Scale bits define a pre-scaled clock from which Timer 0 and/or Timer 1 may be clocked.

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timers 2, 3, 4, and 5 may be clocked by the system clock, the system clock divided by 12, or the external clock divided by 8. Additionally, Timer 3 and Timer 4 may be clocked from the LFOSC0 divided by 8, and operate in Suspend or Snooze modes. Timer 4 is a wake source for the device, and may be chained together with Timer 3 to produce long sleep intervals.

Timer 0 and Timer 1 may also be operated as counters. When functioning as a counter, a counter/timer register is incremented on each high-to-low transition at the selected input pin (T0 or T1). Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it must be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

Table 21.1. Timer Modes

Timer 0 and Timer 1 Modes	Timer 2 and 5 Modes	Timer 3 and 4 Modes
13-bit counter/timer	16-bit timer with auto-reload	16-bit timer with auto-reload
16-bit counter/timer	Two 8-bit timers with auto-reload	Two 8-bit timers with auto-reload
8-bit counter/timer with auto-reload	Input capture	Input capture
Two 8-bit counter/timers (Timer 0 only)		Suspend / Snooze wake timer

21.2 Features

Timer 0 and Timer 1 include the following features:

- Standard 8051 timers, supporting backwards-compatibility with firmware and hardware.
- Clock sources include SYSCLK, SYSCLK divided by 12, 4, or 48, the External Clock divided by 8, or an external pin.
- 8-bit auto-reload counter/timer mode
- 13-bit counter/timer mode
- 16-bit counter/timer mode
- Dual 8-bit counter/timer mode (Timer 0)

Timer 2, Timer 3, Timer 4, and Timer 5 are 16-bit timers including the following features:

- Clock sources for all timers include SYSCLK, SYSCLK divided by 12, or the External Clock divided by 8
- LFOSC0 divided by 8 may be used to clock Timer 3 and Timer 4 in active or suspend/snooze power modes
- Timer 4 is a low-power wake source, and can be chained together with Timer 3
- 16-bit auto-reload timer mode
- Dual 8-bit auto-reload timer mode
- External pin capture
- LFOSC0 capture
- Comparator 0 capture
- Configurable Logic output capture

21.3 Functional Description

21.3.1 System Connections

All five timers are capable of clocking other peripherals and triggering events in the system. The individual peripherals select which timer to use for their respective functions. Note that the Timer 2, 3, and 4 high overflows apply to the full timer when operating in 16-bit mode or the high-byte timer when operating in 8-bit split mode.

Table 21.2. Timer Peripheral Clocking / Event Triggering

Function	T0 Overflow	T1 Overflow	T2 High Overflow	T2 Low Overflow	T2 Input Capture	T3 High Overflow	T3 Low Overflow	T3 Input Capture	T4 High Overflow	T4 Low Overflow	T4 Input Capture	T5 High Overflow	T5 Low Overflow	T5 Input Capture
UART0 Baud Rate		Yes												
SMBus 0 Clock Rate (Master)	Yes	Yes	Yes	Yes										
SMBus 0 SCL Low Timeout						Yes								
I2C0 Slave SCL Low Timeout									Yes					
PCA0 Clock	Yes													
ADC0 Conversion Start	Yes		Yes[1]	Yes[1]		Yes[1]	Yes[1]		Yes[1]	Yes[1]		Yes[1]	Yes[1]	
T2 Input Capture Pin					Yes			Yes			Yes			Yes
LFOSC0 Capture					Yes			Yes			Yes			Yes
Comparator 0 Output Capture					Yes			Yes			Yes			Yes
DAC Output Trigger						DAC0/1/2/3			DAC0/1/2/3			DAC0/1/2/3		
CLU Input / CLU Clock			CLU0A			CLU1A			CLU2A			CLU3A		
CLU Output Capture					CLU0/1/2/3			CLU0/1/2/3			CLU0/1/2/3			CLU0/1/2/3
CLU Output Reload			CLU0/2			CLU1/3			CLU0/2			CLU1/3		

Notes:
1. The high-side overflow is used when the timer is in 16-bit mode. The low-side overflow is used in 8-bit mode.

21.3.2 Timer 0 and Timer 1

Timer 0 and Timer 1 are each implemented as a 16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register. Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register. Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently for the supported operating modes.

21.3.2.1 Operational Modes

Mode 0: 13-bit Counter/Timer

Timer 0 and Timer 1 operate as 13-bit counter/timers in Mode 0. The following describes the configuration and operation of Timer 0. However, both timers operate identically, and Timer 1 is configured in the same manner as described for Timer 0.

The TH0 register holds the eight MSBs of the 13-bit counter/timer. TL0 holds the five LSBs in bit positions TL0.4–TL0.0. The three upper bits of TL0 (TL0.7–TL0.5) are indeterminate and should be masked out or ignored when reading. As the 13-bit timer register increments and overflows from 0x1FFF (all ones) to 0x0000, the timer overflow flag TF0 in TCON is set and an interrupt occurs if Timer 0 interrupts are enabled. The overflow rate for Timer 0 in 13-bit mode is:

$$F_{\text{TIMER0}} = \frac{F_{\text{Input Clock}}}{2^{13} - \text{TH0:TL0}} = \frac{F_{\text{Input Clock}}}{8192 - \text{TH0:TL0}}$$

The CT0 bit in the TMOD register selects the counter/timer's clock source. When CT0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register. Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it must be held at a given level for at least two full system clock cycles to ensure the level is properly sampled. Clearing CT selects the clock defined by the T0M bit in register CKCON0. When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON0.

Setting the TR0 bit enables the timer when either GATE0 in the TMOD register is logic 0 or based on the input signal INT0. The IN0PL bit setting in IT01CF changes which state of INT0 input starts the timer counting. Setting GATE0 to 1 allows the timer to be controlled by the external input signal INT0, facilitating pulse width measurements.

Table 21.3. Timer 0 Run Control Options

TR0	GATE0	INT0	IN0PL	Counter/Timer
0	X	X	X	Disabled
1	0	X	X	Enabled
1	1	0	0	Disabled
1	1	0	1	Enabled
1	1	1	0	Enabled
1	1	1	1	Disabled

Note:
1. X = Don't Care

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal INT1 is used with Timer 1, and IN1PL in register IT01CF determines the INT1 state that starts Timer 1 counting.

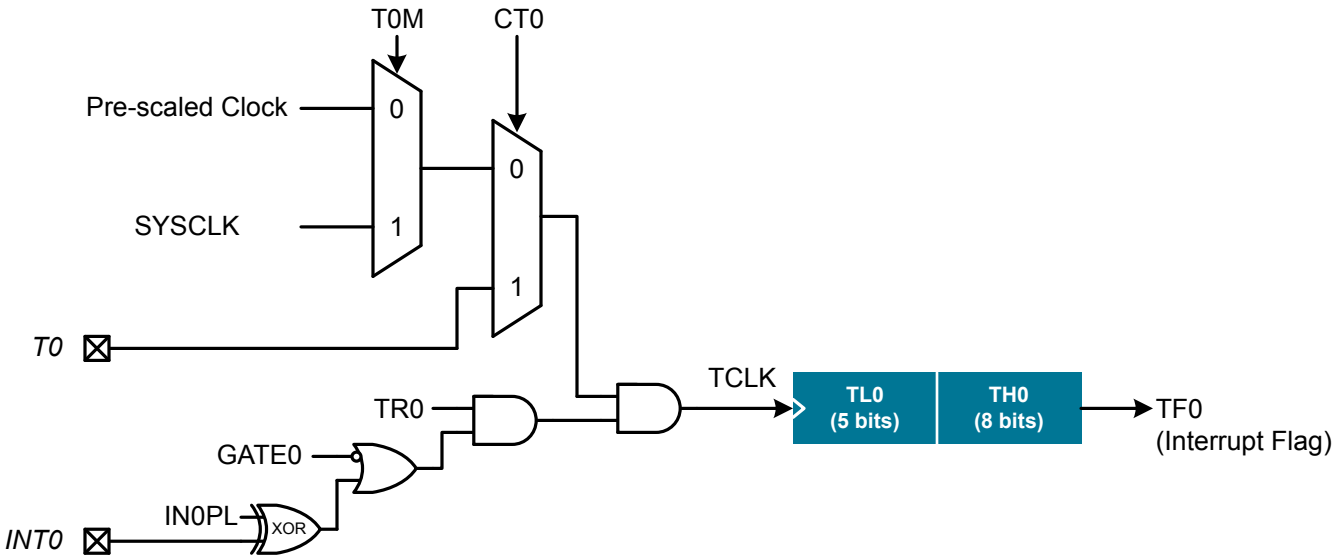


Figure 21.1. T0 Mode 0 Block Diagram

Mode 1: 16-bit Counter/Timer

Mode 1 operation is the same as Mode 0, except that the counter/timer registers use all 16 bits. The counter/timers are enabled and configured in Mode 1 in the same manner as for Mode 0. The overflow rate for Timer 0 in 16-bit mode is:

$$F_{TIMER0} = \frac{F_{Input\ Clock}}{2^{16} - TH0:TL0} = \frac{F_{Input\ Clock}}{65536 - TH0:TL0}$$

Mode 2: 8-bit Counter/Timer with Auto-Reload

Mode 2 configures Timer 0 and Timer 1 to operate as 8-bit counter/timers with automatic reload of the start value. TL0 holds the count and TH0 holds the reload value. When the counter in TL0 overflows from all ones to 0x00, the timer overflow flag TF0 in the TCON register is set and the counter in TL0 is reloaded from TH0. If Timer 0 interrupts are enabled, an interrupt will occur when the TF0 flag is set. The reload value in TH0 is not changed. TL0 must be initialized to the desired value before enabling the timer for the first count to be correct. When in Mode 2, Timer 1 operates identically to Timer 0.

The overflow rate for Timer 0 in 8-bit auto-reload mode is:

$$F_{TIMER0} = \frac{F_{Input\ Clock}}{2^8 - TH0} = \frac{F_{Input\ Clock}}{256 - TH0}$$

Both counter/timers are enabled and configured in Mode 2 in the same manner as Mode 0. Setting the TR0 bit enables the timer when either GATE0 in the TMOD register is logic 0 or when the input signal INT0 is active as defined by bit IN0PL in register IT01CF.

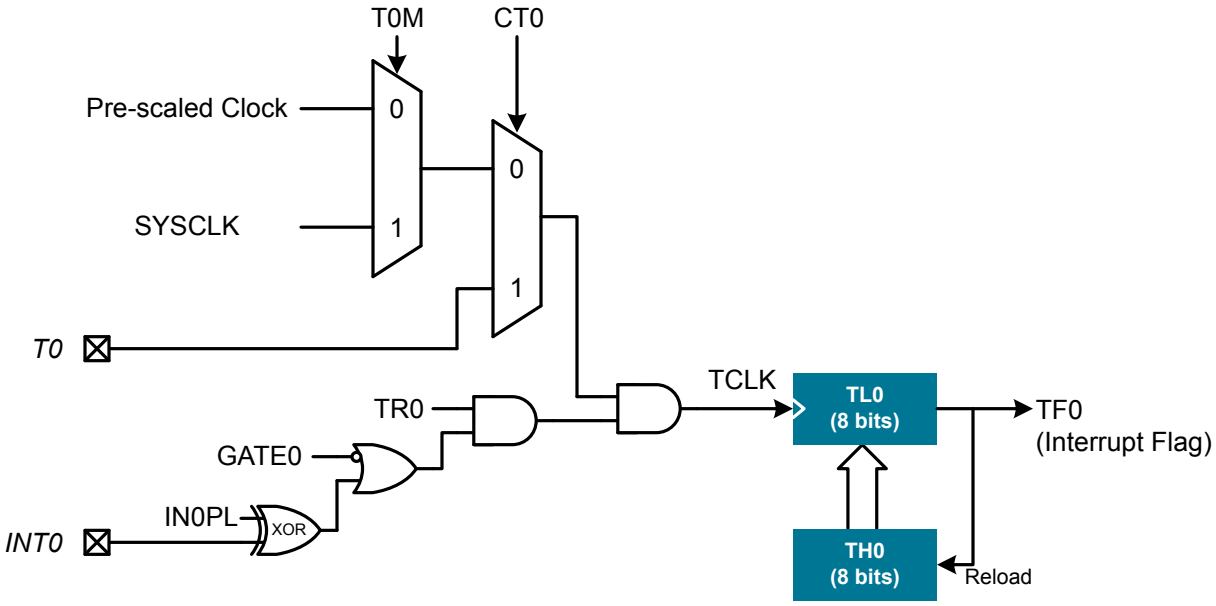


Figure 21.2. T0 Mode 2 Block Diagram

Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, CT0, GATE0, and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

The overflow rate for Timer 0 Low in 8-bit mode is:

$$F_{TIMER0} = \frac{F_{Input\ Clock}}{2^8 - TL0} = \frac{F_{Input\ Clock}}{256 - TL0}$$

The overflow rate for Timer 0 High in 8-bit mode is:

$$F_{TIMER0} = \frac{F_{Input\ Clock}}{2^8 - TH0} = \frac{F_{Input\ Clock}}{256 - TH0}$$

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

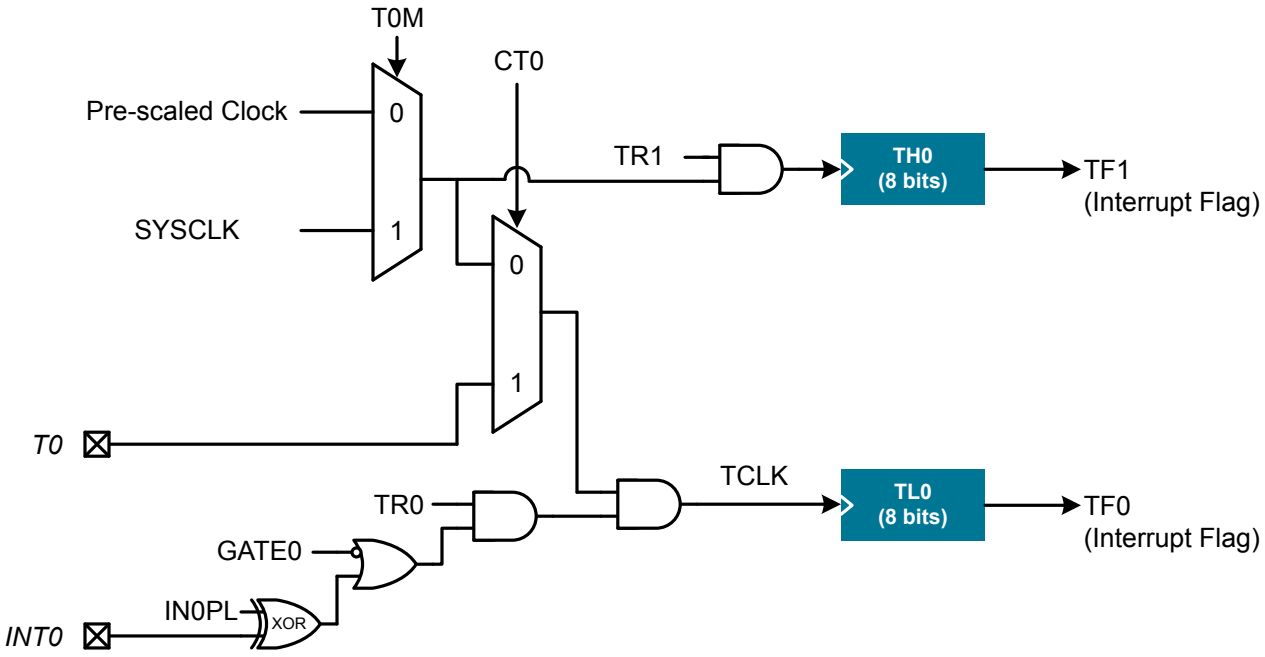


Figure 21.3. T0 Mode 3 Block Diagram

21.3.3 Timer 2, Timer 3, Timer 4, and Timer 5

Timer 2, Timer 3, Timer 4, and Timer 5 are functionally equivalent, with the only differences being the top-level connections to other parts of the system.

The timers are 16 bits wide, formed by two 8-bit SFRs: TMRnL (low byte) and TMRnH (high byte). Each timer may operate in 16-bit auto-reload mode, dual 8-bit auto-reload (split) mode, or capture mode.

Clock Selection

Clocking for each timer is configured using the TnXCLK bit field and the TnML and TnMH bits. Timer 2 may be clocked by the system clock, the system clock divided by 12, or the external clock source divided by 8 (synchronized with SYSCLK). The maximum frequency for the external clock is:

$$F_{SYSCLK} > F_{EXTCLK} \times \frac{6}{7}$$

Timers 3 and 4 may additionally be clocked from the LFOSC0 output divided by 8, and are capable of operating in both the Suspend and Snooze power modes. Timer 4 includes Timer 3 overflows as a clock source, allowing the two to be chained together for longer sleep intervals. When operating in one of the 16-bit modes, the low-side timer clock is used to clock the entire 16-bit timer.

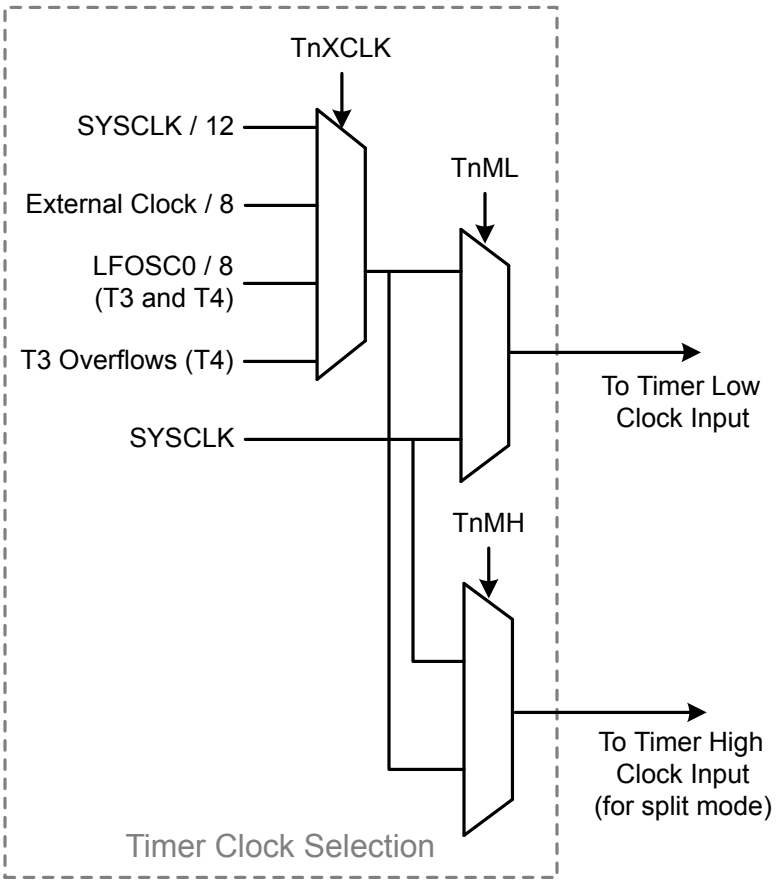


Figure 21.4. Timer 2, 3, 4, and 5 Clock Source Selection

Capture Source Selection

Capture mode allows an external input, the low-frequency oscillator clock, or comparator 0 events to be measured against the selected clock source.

Each timer may individually select one of four capture sources in capture mode: an external input (T2, routed through the crossbar), the low-frequency oscillator clock, comparator 0 events, or CLUn outputs. The capture input signal for the timer is selected using the TnCSEL field in the TMRnCN1 register.

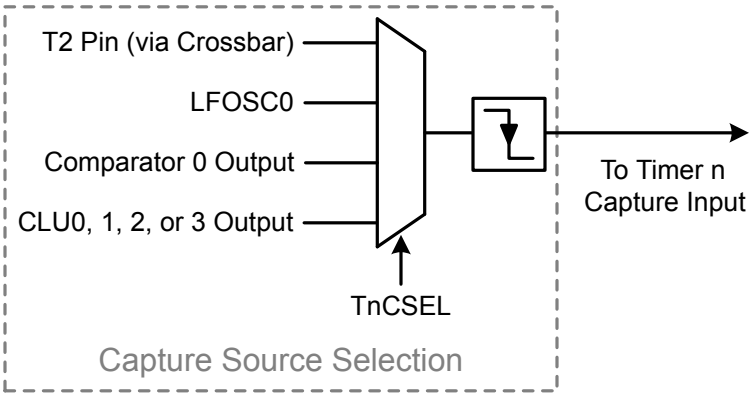


Figure 21.5. Timer 2, 3, 4, and 5 Capture Source Selection

21.3.3.1 16-bit Timer with Auto-Reload

When TnSPLIT is zero, the timer operates as a 16-bit timer with auto-reload. In this mode, the selected clock source increments the timer on every clock. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the timer reload registers (TMRnRLH and TMRnRLL) is loaded into the main timer count register, and the High Byte Overflow Flag (TFnH) is set. If the timer interrupts are enabled, an interrupt is generated on each timer overflow. Additionally, if the timer interrupts are enabled and the TFnLEN bit is set, an interrupt is generated each time the lower 8 bits (TMRnL) overflow from 0xFF to 0x00.

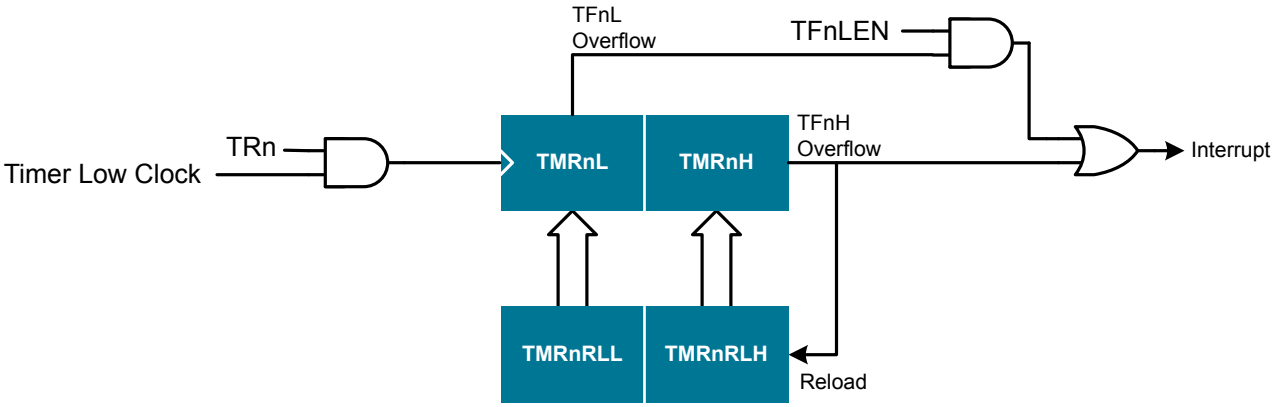


Figure 21.6. 16-Bit Mode Block Diagram

It is also possible to connect the timer up with a CLU output to force a timer reload. The RLFSEL field in the TMRnCN1 register selects this option. When RLFSEL is set to a CLU output option, the timer will reload as normal on overflows, but will also be reloaded whenever the CLU synchronous output is logic high.

21.3.3.2 8-bit Timers with Auto-Reload (Split Mode)

When TnSPLIT is set, the timer operates as two 8-bit timers (TMRnH and TMRnL). Both 8-bit timers operate in auto-reload mode. TMRnRLL holds the reload value for TMRnL; TMRnRLH holds the reload value for TMRnH. The TRn bit in TMRnCN handles the run control for TMRnH. TMRnL is always running when configured for 8-bit auto-reload mode. As shown in the clock source selection tree, the two halves of the timer may be clocked from SYSCLK or by the source selected by the TnXCLK bits.

The overflow rate of the low timer in split 8-bit auto-reload mode is:

$$F_{\text{TIMERn Low}} = \frac{F_{\text{Input Clock}}}{2^8 - \text{TMRnRLL}} = \frac{F_{\text{Input Clock}}}{256 - \text{TMRnRLL}}$$

The overflow rate of the high timer in split 8-bit auto-reload mode is:

$$F_{\text{TIMERn High}} = \frac{F_{\text{Input Clock}}}{2^8 - \text{TMRnRLH}} = \frac{F_{\text{Input Clock}}}{256 - \text{TMRnRLH}}$$

The TFnH bit is set when TMRnH overflows from 0xFF to 0x00; the TFnL bit is set when TMRnL overflows from 0xFF to 0x00. When timer interrupts are enabled, an interrupt is generated each time TMRnH overflows. If timer interrupts are enabled and TFnLEN is set, an interrupt is generated each time either TMRnL or TMRnH overflows. When TFnLEN is enabled, software must check the TFnH and TFnL flags to determine the source of the timer interrupt. The TFnH and TFnL interrupt flags are not cleared by hardware and must be manually cleared by software.

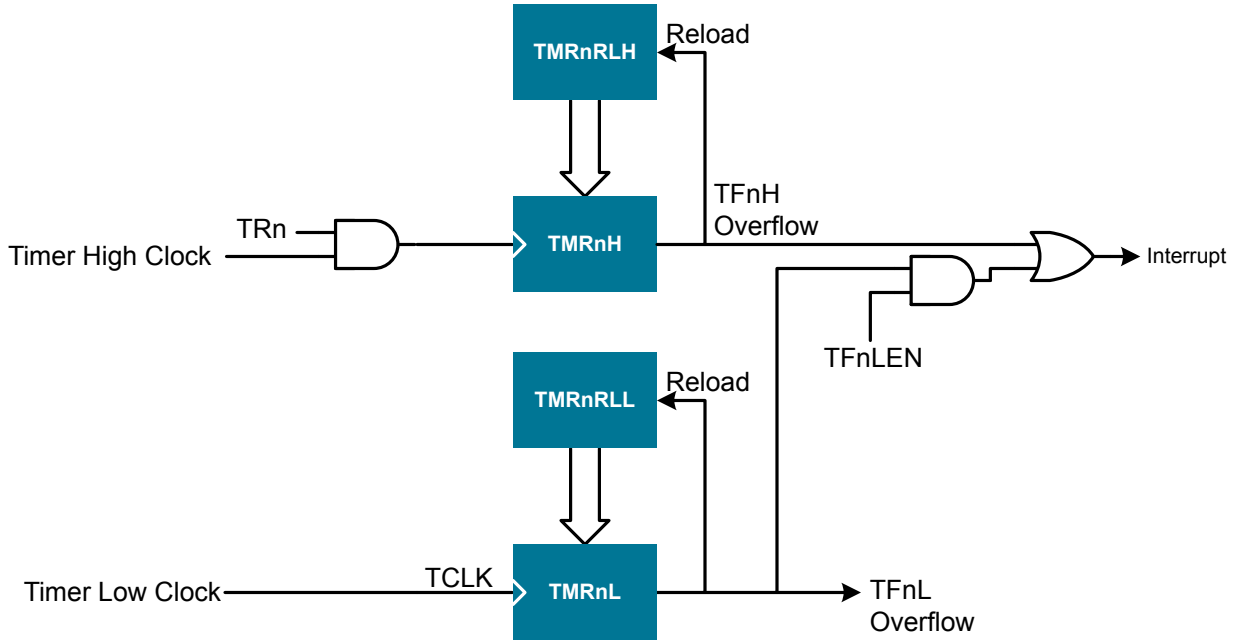


Figure 21.7. 8-Bit Split Mode Block Diagram

21.3.3.3 Capture Mode

Capture mode allows a system event to be measured against the selected clock source. When used in capture mode, the timer clocks normally from the selected clock source through the entire range of 16-bit values from 0x0000 to 0xFFFF.

Setting TFnCEN to 1 enables capture mode. In this mode, TnSPLIT should be set to 0, as the full 16-bit timer is used. Upon a falling edge of the input capture signal, the contents of the timer register (TMRnH:TMRnL) are loaded into the reload registers (TMRnRLH:TMRnRLL) and the TFnH flag is set. By recording the difference between two successive timer capture values, the period of the captured signal can be determined with respect to the selected timer clock.

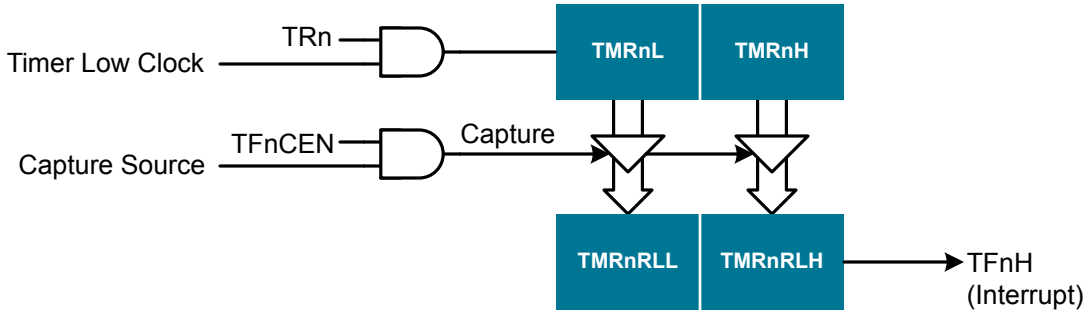


Figure 21.8. Capture Mode Block Diagram

21.3.3.4 Timer 3 and Timer 4 Chaining and Wake Source

Timer 3 and Timer 4 may be chained together to provide a longer counter option. This is accomplished by configuring Timer 4's T4XCLK field to clock from Timer 3 overflows. The primary use of this mode is to wake the device from long-term Suspend or Snooze operations, but it may also be used effectively as a 32-bit capture source.

It is important to note the relationship between the two timers when they are chained together in this manner. The timer 3 overflow rate becomes the Timer 4 clock, and essentially acts as a prescaler to the 16-bit Timer 4 function. For example, if Timer 3 is configured to overflow every 3 SYSCLKs, and Timer 4 is configured to overflow every 5 clocks (coming from Timer 3 overflows), the Timer 4 overflow will occur every 15 SYSCLKs.

Timer 4 is capable of waking the device from the low-power Suspend and Snooze modes. To operate in either mode, the timer must be running from either the LFOSC / 8 option, or Timer 3 overflows (with Timer 3 configured to run from LFOSC / 8). If running in one of these modes, the overflow event from Timer 4 will trigger a wake for the device.

21.4 Timer 0, 1, 2, 3, and 4 Control Registers

21.4.1 CKCON0: Clock Control 0

Bit	7	6	5	4	3	2	1	0
Name	T3MH	T3ML	T2MH	T2ML	T1M	T0M	SCA	
Access	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0x0	
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x8E								

Bit	Name	Reset	Access	Description
7	T3MH	0	RW	**Timer 3 High Byte Clock Select.**

Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only).

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 3 high byte uses the clock defined by T3XCLK in TMR3CN0.
1	SYSCLK	Timer 3 high byte uses the system clock.

Bit	Name	Reset	Access	Description
6	T3ML	0	RW	**Timer 3 Low Byte Clock Select.**

Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode.

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 3 low byte uses the clock defined by T3XCLK in TMR3CN0.
1	SYSCLK	Timer 3 low byte uses the system clock.

Bit	Name	Reset	Access	Description
5	T2MH	0	RW	**Timer 2 High Byte Clock Select.**

Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only).

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 2 high byte uses the clock defined by T2XCLK in TMR2CN0.
1	SYSCLK	Timer 2 high byte uses the system clock.

Bit	Name	Reset	Access	Description
4	T2ML	0	RW	**Timer 2 Low Byte Clock Select.**

Selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 2 low byte uses the clock defined by T2XCLK in TMR2CN0.
1	SYSCLK	Timer 2 low byte uses the system clock.

Bit	Name	Reset	Access	Description
3	T1M	0	RW	**Timer 1 Clock Select.**

Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1.

Value	Name	Description
0	PRESCALE	Timer 1 uses the clock defined by the prescale field, SCA.
1	SYSCLK	Timer 1 uses the system clock.

Bit	Name	Reset	Access	Description
2	T0M	0	RW	**Timer 0 Clock Select.**
	Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1.			

Value	Name	Description
0	PRESCALE	Counter/Timer 0 uses the clock defined by the prescale field, SCA.
1	SYSCLK	Counter/Timer 0 uses the system clock.

Bit	Name	Reset	Access	Description
1:0	SCA	0x0	RW	**Timer 0/1 Prescale.**
	These bits control the Timer 0/1 Clock Prescaler:			

Value	Name	Description
0x0	SYSCLK_DIV_12	System clock divided by 12.
0x1	SYSCLK_DIV_4	System clock divided by 4.
0x2	SYSCLK_DIV_48	System clock divided by 48.
0x3	EXTOSC_DIV_8	External oscillator divided by 8 (synchronized with the system clock).

21.4.2 CKCON1: Clock Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved				T5MH	T5ML	T4MH	T4ML
Access	R				RW	RW	RW	RW
Reset	0x0				0	0	0	0
SFR Page = 0x10; SFR Address: 0xA6								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	T5MH	0	RW	**Timer 5 High Byte Clock Select.**

Selects the clock supplied to the Timer 5 high byte (split 8-bit timer mode only).

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 5 high byte uses the clock defined by T5XCLK in TMR5CN0.
1	SYSCLK	Timer 5 high byte uses the system clock.

Bit	Name	Reset	Access	Description
2	T5ML	0	RW	**Timer 5 Low Byte Clock Select.**

Selects the clock supplied to Timer 5. If Timer 5 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 5 low byte uses the clock defined by T5XCLK in TMR5CN0.
1	SYSCLK	Timer 5 low byte uses the system clock.

Bit	Name	Reset	Access	Description
1	T4MH	0	RW	**Timer 4 High Byte Clock Select.**

Selects the clock supplied to the Timer 4 high byte (split 8-bit timer mode only).

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 4 high byte uses the clock defined by T4XCLK in TMR4CN0.
1	SYSCLK	Timer 4 high byte uses the system clock.

Bit	Name	Reset	Access	Description
0	T4ML	0	RW	**Timer 4 Low Byte Clock Select.**

Selects the clock supplied to Timer 4. If Timer 4 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 4 low byte uses the clock defined by T4XCLK in TMR4CN0.
1	SYSCLK	Timer 4 low byte uses the system clock.

21.4.3 TCON: Timer 0/1 Control

Bit	7	6	5	4	3	2	1	0
Name	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x88 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	TF1	0	RW	**Timer 1 Overflow Flag.** Set to 1 by hardware when Timer 1 overflows. This flag can be cleared by firmware but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine.
6	TR1	0	RW	**Timer 1 Run Control.** Timer 1 is enabled by setting this bit to 1.
5	TF0	0	RW	**Timer 0 Overflow Flag.** Set to 1 by hardware when Timer 0 overflows. This flag can be cleared by firmware but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine.
4	TR0	0	RW	**Timer 0 Run Control.** Timer 0 is enabled by setting this bit to 1.
3	IE1	0	RW	**External Interrupt 1.** This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by firmware but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine in edge-triggered mode.
2	IT1	0	RW	**Interrupt 1 Type Select.** This bit selects whether the configured INT1 interrupt will be edge or level sensitive. INT1 is configured active low or high by the IN1PL bit in register IT01CF.

Value	Name	Description
0	LEVEL	INT1 is level triggered.
1	EDGE	INT1 is edge triggered.

Bit	Name	Reset	Access	Description
1	IE0	0	RW	**External Interrupt 0.** This flag is set by hardware when an edge/level of type defined by IT0 is detected. It can be cleared by firmware but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode.
0	IT0	0	RW	**Interrupt 0 Type Select.** This bit selects whether the configured INT0 interrupt will be edge or level sensitive. INT0 is configured active low or high by the IN0PL bit in register IT01CF.

Value	Name	Description
0	LEVEL	INT0 is level triggered.
1	EDGE	INT0 is edge triggered.

21.4.4 TMOD: Timer 0/1 Mode

Bit	7	6	5	4	3	2	1	0
Name	GATE1	CT1	T1M		GATE0	CT0	T0M	
Access	RW	RW	RW		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x89								

Bit	Name	Reset	Access	Description
7	GATE1	0	RW	**Timer 1 Gate Control.**
	Value	Name		Description
	0	DISABLED		Timer 1 enabled when TR1 = 1 irrespective of INT1 logic level.
	1	ENABLED		Timer 1 enabled only when TR1 = 1 and INT1 is active as defined by bit IN1PL in register IT01CF.
6	CT1	0	RW	**Counter/Timer 1 Select.**
	Value	Name		Description
	0	TIMER		Timer Mode. Timer 1 increments on the clock defined by T1M in the CKCON0 register.
	1	COUNTER		Counter Mode. Timer 1 increments on high-to-low transitions of an external pin (T1).
5:4	T1M	0x0	RW	**Timer 1 Mode Select.**
	These bits select the Timer 1 operation mode.			
	Value	Name		Description
	0x0	MODE0		Mode 0, 13-bit Counter/Timer
	0x1	MODE1		Mode 1, 16-bit Counter/Timer
	0x2	MODE2		Mode 2, 8-bit Counter/Timer with Auto-Reload
	0x3	MODE3		Mode 3, Timer 1 Inactive
3	GATE0	0	RW	**Timer 0 Gate Control.**
	Value	Name		Description
	0	DISABLED		Timer 0 enabled when TR0 = 1 irrespective of INT0 logic level.
	1	ENABLED		Timer 0 enabled only when TR0 = 1 and INT0 is active as defined by bit IN0PL in register IT01CF.
2	CT0	0	RW	**Counter/Timer 0 Select.**
	Value	Name		Description
	0	TIMER		Timer Mode. Timer 0 increments on the clock defined by T0M in the CKCON0 register.
	1	COUNTER		Counter Mode. Timer 0 increments on high-to-low transitions of an external pin (T0).

Bit	Name	Reset	Access	Description
1:0	T0M	0x0	RW	**Timer 0 Mode Select.**
	These bits select the Timer 0 operation mode.			

Value	Name	Description
0x0	MODE0	Mode 0, 13-bit Counter/Timer
0x1	MODE1	Mode 1, 16-bit Counter/Timer
0x2	MODE2	Mode 2, 8-bit Counter/Timer with Auto-Reload
0x3	MODE3	Mode 3, Two 8-bit Counter/Timers

21.4.5 TL0: Timer 0 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL0							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x8A								

Bit	Name	Reset	Access	Description
7:0	TL0	0x00	RW	**Timer 0 Low Byte.**
	The TL0 register is the low byte of the 16-bit Timer 0.			

21.4.6 TL1: Timer 1 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL1							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x8B								

Bit	Name	Reset	Access	Description
7:0	TL1	0x00	RW	**Timer 1 Low Byte.**
	The TL1 register is the low byte of the 16-bit Timer 1.			

21.4.7 TH0: Timer 0 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH0							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x8C								

Bit	Name	Reset	Access	Description
7:0	TH0	0x00	RW	**Timer 0 High Byte.** The TH0 register is the high byte of the 16-bit Timer 0.

21.4.8 TH1: Timer 1 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH1							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10, 0x20; SFR Address: 0x8D								

Bit	Name	Reset	Access	Description
7:0	TH1	0x00	RW	**Timer 1 High Byte.** The TH1 register is the high byte of the 16-bit Timer 1.

21.4.9 TMR2RLL: Timer 2 Reload Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLL							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xCA								

Bit	Name	Reset	Access	Description
7:0	TMR2RLL	0x00	RW	**Timer 2 Reload Low Byte.** When operating in one of the auto-reload modes, TMR2RLL holds the reload value for the low byte of Timer 2 (TMR2L). When operating in capture mode, TMR2RLL is the captured value of TMR2L.

21.4.10 TMR2RLH: Timer 2 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLH							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xCB								

Bit	Name	Reset	Access	Description
7:0	TMR2RLH	0x00	RW	**Timer 2 Reload High Byte.** When operating in one of the auto-reload modes, TMR2RLH holds the reload value for the high byte of Timer 2 (TMR2H). When operating in capture mode, TMR2RLH is the captured value of TMR2H.

21.4.11 TMR2L: Timer 2 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2L							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xCE								

Bit	Name	Reset	Access	Description
7:0	TMR2L	0x00	RW	**Timer 2 Low Byte.** In 16-bit mode, the TMR2L register contains the low byte of the 16-bit Timer 2. In 8-bit mode, TMR2L contains the 8-bit low byte timer value.

21.4.12 TMR2H: Timer 2 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2H							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0xCF								

Bit	Name	Reset	Access	Description
7:0	TMR2H	0x00	RW	**Timer 2 High Byte.** In 16-bit mode, the TMR2H register contains the high byte of the 16-bit Timer 2. In 8-bit mode, TMR2H contains the 8-bit high byte timer value.

21.4.13 TMR2CN0: Timer 2 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF2H	TF2L	TF2LEN	TF2CEN	T2SPLIT	TR2	T2XCLK	
Access	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0x0	
SFR Page = 0x0, 0x10; SFR Address: 0xC8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	TF2H	0	RW	**Timer 2 High Byte Overflow Flag.** Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16-bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit must be cleared by firmware.
6	TF2L	0	RW	**Timer 2 Low Byte Overflow Flag.** Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. TF2L will be set when the low byte overflows regardless of the Timer 2 mode. This bit must be cleared by firmware.
5	TF2LEN	0	RW	**Timer 2 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 2 Low Byte interrupts. If Timer 2 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 2 overflows.
4	TF2CEN	0	RW	**Timer 2 Capture Enable.** When set to 1, this bit enables Timer 2 Capture Mode. If TF2CEN is set and Timer 2 interrupts are enabled, an interrupt will be generated according to the capture source selected by the T2CSEL bits, and the current 16-bit timer value in TMR2H:TMR2L will be copied to TMR2RLH:TMR2RLL.
3	T2SPLIT	0	RW	**Timer 2 Split Mode Enable.** When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload.

Value	Name	Description
0	16_BIT_RELOAD	Timer 2 operates in 16-bit auto-reload mode.
1	8_BIT_RELOAD	Timer 2 operates as two 8-bit auto-reload timers.

Bit	Name	Reset	Access	Description
2	TR2	0	RW	**Timer 2 Run Control.** Timer 2 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in split mode.
1:0	T2XCLK	0x0	RW	**Timer 2 External Clock Select.** This bit selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 2 Clock Select bits (T2MH and T2ML) may still be used to select between the external clock and the system clock for either timer.

Value	Name	Description
0x0	SYSCLK_DIV_12	Timer 2 clock is the system clock divided by 12.
0x1	EXTOSC_DIV_8	Timer 2 clock is the external oscillator divided by 8 (synchronized with SYSCLK when not in suspend or snooze mode).

21.4.14 TMR2CN1: Timer 2 Control 1

Bit	7	6	5	4	3	2	1	0
Name	RLFSEL			Reserved		T2CSEL		
Access	RW			R		RW		
Reset	0x0			0x0		0x1		
SFR Page = 0x10; SFR Address: 0xFD								

Bit	Name	Reset	Access	Description
7:5	RLFSEL	0x0	RW	**Force Reload Select.**

Selects the signal that can force the Timer to reload the timer from the Timer Reload SFRs regardless of whether an overflow has occured. A logic high on the selected signal will force the Timer to reload.

Value	Name	Description
0x0	NONE	Timer will only reload on overflow events.
0x1	CLU0_OUT	Timer will reload on overflow events and CLU0 synchronous output high.
0x2	CLU2_OUT	Timer will reload on overflow events and CLU2 synchronous output high.

Bit	Name	Reset	Access	Description
4:3	*Reserved*	*Must write reset value.*		
2:0	T2CSEL	0x1	RW	**Timer 2 Capture Select.**

When used in capture mode, the T2CSEL field selects the input capture signal.

Value	Name	Description
0x0	PIN	Capture high-to-low transitions on the T2 input pin.
0x1	LFOSC	Capture high-to-low transitions of the LFO oscillator.
0x2	COMPARATOR0	Capture high-to-low transitions of the Comparator 0 output.
0x4	CLU0_OUT	Capture high-to-low transitions on the configurable logic unit 0 synchronous output.
0x5	CLU1_OUT	Capture high-to-low transitions on the configurable logic unit 1 synchronous output.
0x6	CLU2_OUT	Capture high-to-low transitions on the configurable logic unit 2 synchronous output.
0x7	CLU3_OUT	Capture high-to-low transitions on the configurable logic unit 3 synchronous output.

21.4.15 TMR3RLL: Timer 3 Reload Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLL							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0x92								

Bit	Name	Reset	Access	Description
7:0	TMR3RLL	0x00	RW	**Timer 3 Reload Low Byte.** When operating in one of the auto-reload modes, TMR3RLL holds the reload value for the low byte of Timer 3 (TMR3L). When operating in capture mode, TMR3RLL is the captured value of TMR3L.

21.4.16 TMR3RLH: Timer 3 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLH							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0x93								

Bit	Name	Reset	Access	Description
7:0	TMR3RLH	0x00	RW	**Timer 3 Reload High Byte.** When operating in one of the auto-reload modes, TMR3RLH holds the reload value for the high byte of Timer 3 (TMR3H). When operating in capture mode, TMR3RLH is the captured value of TMR3H.

21.4.17 TMR3L: Timer 3 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3L							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0x94								

Bit	Name	Reset	Access	Description
7:0	TMR3L	0x00	RW	**Timer 3 Low Byte.** In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value.

21.4.18 TMR3H: Timer 3 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3H							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; SFR Address: 0x95								

Bit	Name	Reset	Access	Description
7:0	TMR3H	0x00	RW	**Timer 3 High Byte.** In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8-bit mode, TMR3H contains the 8-bit high byte timer value.

21.4.19 TMR3CN0: Timer 3 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF3H	TF3L	TF3LEN	TF3CEN	T3SPLIT	TR3	T3XCLK	
Access	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0x0	
SFR Page = 0x0, 0x10; SFR Address: 0x91								

Bit	Name	Reset	Access	Description
7	TF3H	0	RW	**Timer 3 High Byte Overflow Flag.** Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16-bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit must be cleared by firmware.
6	TF3L	0	RW	**Timer 3 Low Byte Overflow Flag.** Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. TF3L will be set when the low byte overflows regardless of the Timer 3 mode. This bit must be cleared by firmware.
5	TF3LEN	0	RW	**Timer 3 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows.
4	TF3CEN	0	RW	**Timer 3 Capture Enable.** When set to 1, this bit enables Timer 3 Capture Mode. If TF3CEN is set and Timer 3 interrupts are enabled, an interrupt will be generated according to the capture source selected by the T3CSEL bits, and the current 16-bit timer value in TMR3H:TMR3L will be copied to TMR3RLH:TMR3RLL.
3	T3SPLIT	0	RW	**Timer 3 Split Mode Enable.** When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload.

Value	Name	Description
0	16_BIT_RELOAD	Timer 3 operates in 16-bit auto-reload mode.
1	8_BIT_RELOAD	Timer 3 operates as two 8-bit auto-reload timers.

Bit	Name	Reset	Access	Description
2	TR3	0	RW	**Timer 3 Run Control.** Timer 3 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in split mode.
1:0	T3XCLK	0x0	RW	**Timer 3 External Clock Select.** This bit selects the external clock source for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 3 Clock Select bits (T3MH and T3ML) may still be used to select between the external clock and the system clock for either timer.

Value	Name	Description
0x0	SYSCLK_DIV_12	Timer 3 clock is the system clock divided by 12.
0x1	EXTOSC_DIV_8	Timer 3 clock is the external oscillator divided by 8 (synchronized with SYSCLK when not in suspend or snooze mode).
0x3	LFOSC_DIV_8	Timer 3 clock is the low-frequency oscillator divided by 8 (synchronized with SYSCLK when not in suspend or snooze mode).

21.4.20 TMR3CN1: Timer 3 Control 1

Bit	7	6	5	4	3	2	1	0
Name	RLFSEL			STSYNC	Reserved	T3CSEL		
Access	RW			RW	R	RW		
Reset	0x0			0	0	0x0		
SFR Page = 0x10; SFR Address: 0xFE								

Bit	Name	Reset	Access	Description
7:5	RLFSEL	0x0	RW	**Force Reload Select.**

Selects the signal that can force the Timer to reload the timer from the Timer Reload SFRs regardless of whether an overflow has occured. A logic high on the selected signal will force the Timer to reload.

Value	Name	Description
0x0	SMB0_SCL	If the SMBTOE bit in the SMB0CF register is 0, then the timer will only reload on overflow events. If SMBTOE is 1, the timer will reload on overflow events and when the SMB0 SCL signal is high.
0x1	CLU1_OUT	Timer will reload on overflow events and CLU1 synchronous output high.
0x2	CLU3_OUT	Timer will reload on overflow events and CLU3 synchronous output high.
0x3	NONE	Timer will only reload on overflow events.

4	STSYNC	0	RW	**Suspend Timer Synchronization Status.**

This bit is used to indicate when it is safe to read and write the registers associated with the suspend wake-up timer. If a suspend wake-up source other than the timer has brought the oscillator out of suspend mode, it may take up to three timer clocks before the timer can be read or written. When STSYNC reads '1', reads and writes of the timer register should not be performed. When STSYNC reads '0', it is safe to read and write the timer registers.

3	*Reserved*	*Must write reset value.*		
2:0	T3CSEL	0x0	RW	**Timer 3 Capture Select.**

When used in capture mode, the T3CSEL field selects the input capture signal.

Value	Name	Description
0x0	PIN	Capture high-to-low transitions on the T2 input pin.
0x1	LFOSC	Capture high-to-low transitions of the LFO oscillator.
0x2	COMPARATOR0	Capture high-to-low transitions of the Comparator 0 output.
0x4	CLU0_OUT	Capture high-to-low transitions on the configurable logic unit 0 synchronous output.
0x5	CLU1_OUT	Capture high-to-low transitions on the configurable logic unit 1 synchronous output.
0x6	CLU2_OUT	Capture high-to-low transitions on the configurable logic unit 2 synchronous output.
0x7	CLU3_OUT	Capture high-to-low transitions on the configurable logic unit 3 synchronous output.

21.4.21 TMR4RLL: Timer 4 Reload Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLL							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xA2								

Bit	Name	Reset	Access	Description
7:0	TMR4RLL	0x00	RW	**Timer 4 Reload Low Byte.** When operating in one of the auto-reload modes, TMR4RLL holds the reload value for the low byte of Timer 4 (TMR4L). When operating in capture mode, TMR4RLL is the captured value of TMR4L.

21.4.22 TMR4RLH: Timer 4 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLH							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xA3								

Bit	Name	Reset	Access	Description
7:0	TMR4RLH	0x00	RW	**Timer 4 Reload High Byte.** When operating in one of the auto-reload modes, TMR4RLH holds the reload value for the high byte of Timer 4 (TMR4H). When operating in capture mode, TMR4RLH is the captured value of TMR4H.

21.4.23 TMR4L: Timer 4 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4L							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xA4								

Bit	Name	Reset	Access	Description
7:0	TMR4L	0x00	RW	**Timer 4 Low Byte.** In 16-bit mode, the TMR4L register contains the low byte of the 16-bit Timer 4. In 8-bit mode, TMR4L contains the 8-bit low byte timer value.

21.4.24 TMR4H: Timer 4 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4H							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xA5								

Bit	Name	Reset	Access	Description
7:0	TMR4H	0x00	RW	**Timer 4 High Byte.** In 16-bit mode, the TMR4H register contains the high byte of the 16-bit Timer 4. In 8-bit mode, TMR4H contains the 8-bit high byte timer value.

21.4.25 TMR4CN0: Timer 4 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF4H	TF4L	TF4LEN	TF4CEN	T4SPLIT	TR4	T4XCLK	
Access	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0x0	
SFR Page = 0x10; SFR Address: 0x98 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	TF4H	0	RW	**Timer 4 High Byte Overflow Flag.** Set by hardware when the Timer 4 high byte overflows from 0xFF to 0x00. In 16-bit mode, this will occur when Timer 4 overflows from 0xFFFF to 0x0000. When the Timer 4 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 4 interrupt service routine. This bit must be cleared by firmware.
6	TF4L	0	RW	**Timer 4 Low Byte Overflow Flag.** Set by hardware when the Timer 4 low byte overflows from 0xFF to 0x00. TF4L will be set when the low byte overflows regardless of the Timer 4 mode. This bit must be cleared by firmware.
5	TF4LEN	0	RW	**Timer 4 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 4 Low Byte interrupts. If Timer 4 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 4 overflows.
4	TF4CEN	0	RW	**Timer 4 Capture Enable.** When set to 1, this bit enables Timer 4 Capture Mode. If TF4CEN is set and Timer 4 interrupts are enabled, an interrupt will be generated according to the capture source selected by the T4CSEL bits, and the current 16-bit timer value in TMR4H:TMR4L will be copied to TMR4RLH:TMR4RLL.
3	T4SPLIT	0	RW	**Timer 4 Split Mode Enable.** When this bit is set, Timer 4 operates as two 8-bit timers with auto-reload.

Value	Name	Description
0	16_BIT_RELOAD	Timer 4 operates in 16-bit auto-reload mode.
1	8_BIT_RELOAD	Timer 4 operates as two 8-bit auto-reload timers.

Bit	Name	Reset	Access	Description
2	TR4	0	RW	**Timer 4 Run Control.** Timer 4 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR4H only; TMR4L is always enabled in split mode.
1:0	T4XCLK	0x0	RW	**Timer 4 External Clock Select.** This bit selects the external clock source for Timer 4. If Timer 4 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 4 Clock Select bits (T4MH and T4ML) may still be used to select between the external clock and the system clock for either timer.

Value	Name	Description
0x0	SYSCLK_DIV_12	Timer 4 clock is the system clock divided by 12.
0x1	EXTOSC_DIV_8	Timer 4 clock is the external oscillator divided by 8 (synchronized with SYSCLK when not in suspend or snooze mode).
0x2	TIMER3	Timer 4 is clocked by Timer 3 overflows.
0x3	LFOSC_DIV_8	Timer 4 clock is the low-frequency oscillator divided by 8 (synchronized with SYSCLK when not in suspend or snooze mode).

21.4.26 TMR4CN1: Timer 4 Control 1

Bit	7	6	5	4	3	2	1	0
Name	RLFSEL			STSYNC	Reserved	T4CSEL		
Access	RW			RW	R	RW		
Reset	0x0			0	0	0x0		
SFR Page = 0x10; SFR Address: 0xFF								

Bit	Name	Reset	Access	Description
7:5	RLFSEL	0x0	RW	**Force Reload Select.**

Selects the signal that can force the Timer to reload the timer from the Timer Reload SFRs regardless of whether an overflow has occured. A logic high on the selected signal will force the Timer to reload.

Value	Name	Description
0x0	I2CSLAVE0_SCL	If the TIMEOUT bit in I2C0CN0 is 0, then the timer will only reload on overflow events. If TIMEOUT is 1, the timer will reload on overflow events and when the I2C0 SCL signal is high.
0x1	CLU0_OUT	Timer will reload on overflow events and CLU0 synchronous output high.
0x2	CLU2_OUT	Timer will reload on overflow events and CLU2 synchronous output high.
0x3	NONE	Timer will only reload on overflow events.

4	STSYNC	0	RW	**Suspend Timer Synchronization Status.**

This bit is used to indicate when it is safe to read and write the registers associated with the suspend wake-up timer. If a suspend wake-up source other than the timer has brought the oscillator out of suspend mode, it may take up to three timer clocks before the timer can be read or written. When STSYNC reads '1', reads and writes of the timer register should not be performed. When STSYNC reads '0', it is safe to read and write the timer registers.

3	*Reserved*	*Must write reset value.*		
2:0	T4CSEL	0x0	RW	**Timer 4 Capture Select.**

When used in capture mode, the T4CSEL field selects the input capture signal.

Value	Name	Description
0x0	PIN	Capture high-to-low transitions on the T2 input pin.
0x1	LFOSC	Capture high-to-low transitions of the LFO oscillator.
0x2	COMPARATOR0	Capture high-to-low transitions of the Comparator 0 output.
0x4	CLU0_OUT	Capture high-to-low transitions on the configurable logic unit 0 synchronous output.
0x5	CLU1_OUT	Capture high-to-low transitions on the configurable logic unit 1 synchronous output.
0x6	CLU2_OUT	Capture high-to-low transitions on the configurable logic unit 2 synchronous output.
0x7	CLU3_OUT	Capture high-to-low transitions on the configurable logic unit 3 synchronous output.

21.4.27 TMR5RLL: Timer 5 Reload Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLL							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xD2								

Bit	Name	Reset	Access	Description
7:0	TMR5RLL	0x00	RW	**Timer 5 Reload Low Byte.** When operating in one of the auto-reload modes, TMR5RLL holds the reload value for the low byte of Timer 5 (TMR5L). When operating in capture mode, TMR5RLL is the captured value of TMR5L.

21.4.28 TMR5RLH: Timer 5 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLH							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xD3								

Bit	Name	Reset	Access	Description
7:0	TMR5RLH	0x00	RW	**Timer 5 Reload High Byte.** When operating in one of the auto-reload modes, TMR5RLH holds the reload value for the high byte of Timer 5 (TMR5H). When operating in capture mode, TMR5RLH is the captured value of TMR5H.

21.4.29 TMR5L: Timer 5 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5L							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xD4								

Bit	Name	Reset	Access	Description
7:0	TMR5L	0x00	RW	**Timer 5 Low Byte.** In 16-bit mode, the TMR5L register contains the low byte of the 16-bit Timer 5. In 8-bit mode, TMR5L contains the 8-bit low byte timer value.

21.4.30 TMR5H: Timer 5 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5H							
Access	RW							
Reset	0x00							
SFR Page = 0x10; SFR Address: 0xD5								

Bit	Name	Reset	Access	Description
7:0	TMR5H	0x00	RW	**Timer 5 High Byte.** In 16-bit mode, the TMR5H register contains the high byte of the 16-bit Timer 5. In 8-bit mode, TMR5H contains the 8-bit high byte timer value.

21.4.31 TMR5CN0: Timer 5 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF5H	TF5L	TF5LEN	TF5CEN	T5SPLIT	TR5	T5XCLK	
Access	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0x0	
SFR Page = 0x10; SFR Address: 0xC0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	TF5H	0	RW	**Timer 5 High Byte Overflow Flag.** Set by hardware when the Timer 5 high byte overflows from 0xFF to 0x00. In 16-bit mode, this will occur when Timer 5 overflows from 0xFFFF to 0x0000. When the Timer 5 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 5 interrupt service routine. This bit must be cleared by firmware.
6	TF5L	0	RW	**Timer 5 Low Byte Overflow Flag.** Set by hardware when the Timer 5 low byte overflows from 0xFF to 0x00. TF5L will be set when the low byte overflows regardless of the Timer 5 mode. This bit must be cleared by firmware.
5	TF5LEN	0	RW	**Timer 5 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 5 Low Byte interrupts. If Timer 5 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 5 overflows.
4	TF5CEN	0	RW	**Timer 5 Capture Enable.** When set to 1, this bit enables Timer 5 Capture Mode. If TF5CEN is set and Timer 5 interrupts are enabled, an interrupt will be generated according to the capture source selected by the T5CSEL bits, and the current 16-bit timer value in TMR5H:TMR5L will be copied to TMR5RLH:TMR5RLL.
3	T5SPLIT	0	RW	**Timer 5 Split Mode Enable.** When this bit is set, Timer 5 operates as two 8-bit timers with auto-reload.

Value	Name	Description
0	16_BIT_RELOAD	Timer 5 operates in 16-bit auto-reload mode.
1	8_BIT_RELOAD	Timer 5 operates as two 8-bit auto-reload timers.

Bit	Name	Reset	Access	Description
2	TR5	0	RW	**Timer 5 Run Control.** Timer 5 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR5H only; TMR5L is always enabled in split mode.
1:0	T5XCLK	0x0	RW	**Timer 5 External Clock Select.** This bit selects the external clock source for Timer 5. If Timer 5 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 5 Clock Select bits (T5MH and T5ML) may still be used to select between the external clock and the system clock for either timer.

Value	Name	Description
0x0	SYSCLK_DIV_12	Timer 5 clock is the system clock divided by 12.
0x1	EXTOSC_DIV_8	Timer 5 clock is the external oscillator divided by 8 (synchronized with SYSCLK when not in suspend or snooze mode).

21.4.32 TMR5CN1: Timer 5 Control 1

Bit	7	6	5	4	3	2	1	0
Name	RLFSEL			Reserved		T5CSEL		
Access	RW			R		RW		
Reset	0x0			0x0		0x1		
SFR Page = 0x10; SFR Address: 0xF1								

Bit	Name	Reset	Access	Description
7:5	RLFSEL	0x0	RW	**Force Reload Select.**
	Selects the signal that can force the Timer to reload the timer from the Timer Reload SFRs regardless of whether an overflow has occured. A logic high on the selected signal will force the Timer to reload.			
	Value	Name		Description
	0x0	NONE		Timer will only reload on overflow events.
	0x1	CLU1_OUT		Timer will reload on overflow events and CLU1 synchronous output high.
	0x2	CLU3_OUT		Timer will reload on overflow events and CLU3 synchronous output high.
4:3	*Reserved*	*Must write reset value.*		
2:0	T5CSEL	0x1	RW	**Timer 5 Capture Select.**
	When used in capture mode, the T5CSEL field selects the input capture signal.			
	Value	Name		Description
	0x0	PIN		Capture high-to-low transitions on the T2 input pin.
	0x1	LFOSC		Capture high-to-low transitions of the LFO oscillator.
	0x2	COMPARATOR0		Capture high-to-low transitions of the Comparator 0 output.
	0x4	CLU0_OUT		Capture high-to-low transitions on the configurable logic unit 0 synchronous output.
	0x5	CLU1_OUT		Capture high-to-low transitions on the configurable logic unit 1 synchronous output.
	0x6	CLU2_OUT		Capture high-to-low transitions on the configurable logic unit 2 synchronous output.
	0x7	CLU3_OUT		Capture high-to-low transitions on the configurable logic unit 3 synchronous output.

22. Universal Asynchronous Receiver/Transmitter 0 (UART0)

22.1 Introduction

UART0 is an asynchronous, full duplex serial port offering modes 1 and 3 of the standard 8051 UART. Enhanced baud rate support allows a wide range of clock sources to generate standard baud rates. Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

UART0 has two associated SFRs: Serial Control Register 0 (SCON0) and Serial Data Buffer 0 (SBUF0). The single SBUF0 location provides access to both transmit and receive registers.

Note: Writes to SBUF0 always access the transmit register. Reads of SBUF0 always access the buffered receive register; it is not possible to read data from the transmit register.

With UART0 interrupts enabled, an interrupt is generated each time a transmit is completed (TI is set in SCON0), or a data byte has been received (RI is set in SCON0). The UART0 interrupt flags are not cleared by hardware when the CPU vectors to the interrupt service routine. They must be cleared manually by software, allowing software to determine the cause of the UART0 interrupt (transmit complete or receive complete).

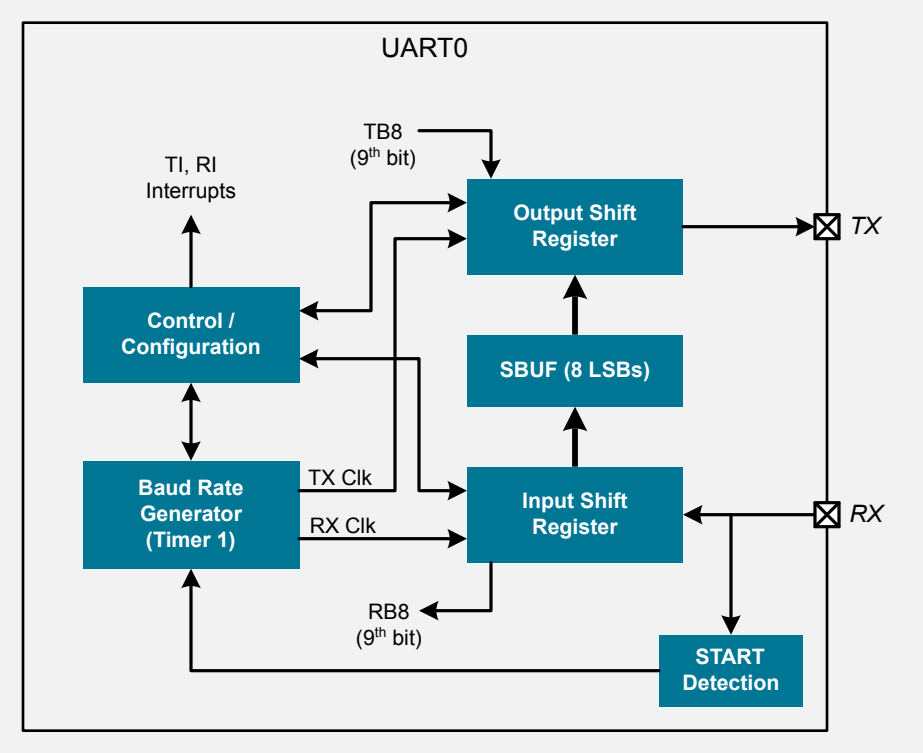


Figure 22.1. UART0 Block Diagram

22.2 Features

The UART uses two signals (TX and RX) and a predetermined fixed baud rate to provide asynchronous communications with other devices.

The UART module provides the following features:

- Asynchronous transmissions and receptions.
- Baud rates up to SYSCLK/2 (transmit) or SYSCLK/8 (receive).
- 8- or 9-bit data.
- Automatic start and stop generation.
- Single-byte FIFO on transmit and receive.

22.3 Functional Description

22.3.1 Baud Rate Generation

The UART0 baud rate is generated by Timer 1 in 8-bit auto-reload mode. The TX clock is generated by TL1; the RX clock is generated by a copy of TL1, which is not user-accessible. Both TX and RX timer overflows are divided by two to generate the TX and RX baud rates. The RX timer runs when Timer 1 is enabled and uses the same reload value (TH1). However, an RX timer reload is forced when a START condition is detected on the RX pin. This allows a receive to begin any time a START is detected, independent of the TX timer state.

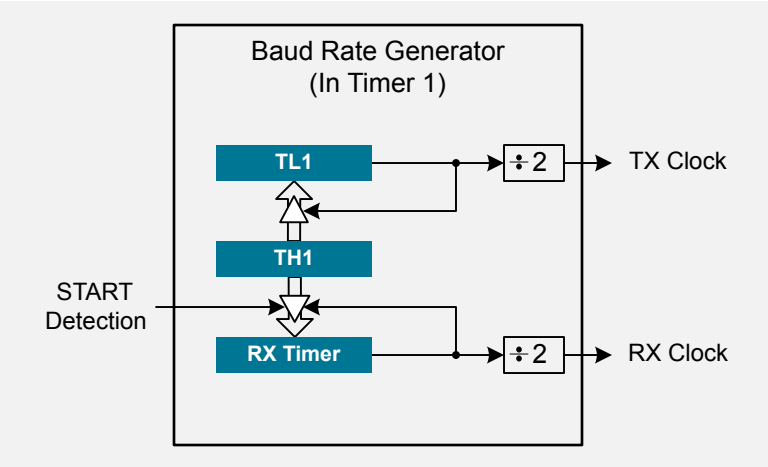


Figure 22.2. UART0 Baud Rate Logic Block Diagram

Timer 1 should be configured for 8-bit auto-reload mode (mode 2). The Timer 1 reload value and prescaler should be set so that overflows occur at twice the desired UART0 baud rate. The UART0 baud rate is half of the Timer 1 overflow rate. Configuring the Timer 1 overflow rate is discussed in the timer sections.

22.3.2 Data Format

UART0 has two options for data formatting. All data transfers begin with a start bit (logic low), followed by the data (sent LSB-first), and end with a stop bit (logic high). The data length of the UART0 module is normally 8 bits. An extra 9th bit may be added to the MSB of data field for use in multi-processor communications or for implementing parity checks on the data. The S0MODE bit in the SCON register selects between 8 or 9-bit data transfers.

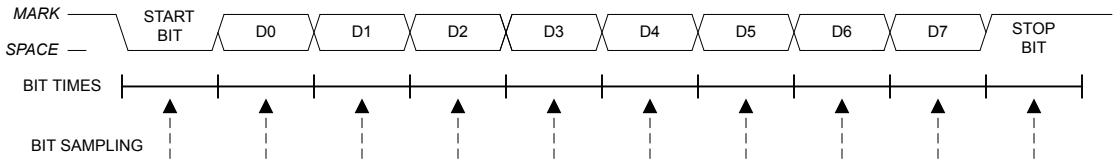


Figure 22.3. 8-Bit Data Transfer

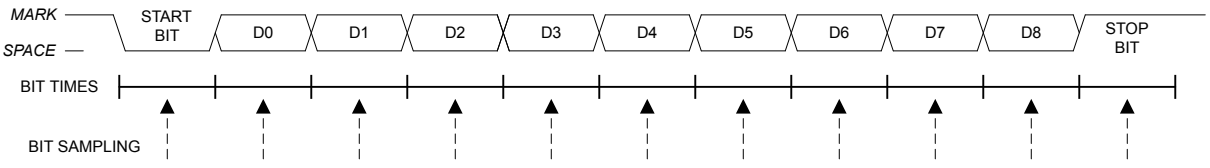


Figure 22.4. 9-Bit Data Transfer

22.3.3 Data Transfer

UART0 provides standard asynchronous, full duplex communication. All data sent or received goes through the SBUF0 register and (in 9-bit mode) the RB8 bit in the SCON0 register.

Transmitting Data

Data transmission is initiated when software writes a data byte to the SBUF0 register. If 9-bit mode is used, software should set up the desired 9th bit in TB8 prior to writing SBUF0. Data is transmitted LSB first from the TX pin. The TI flag in SCON0 is set at the end of the transmission (at the beginning of the stop-bit time). If TI interrupts are enabled, TI will trigger an interrupt.

Receiving Data

To enable data reception, firmware should write the REN bit to 1. Data reception begins when a start condition is recognized on the RX pin. Data will be received at the selected baud rate through the end of the data phase. Data will be transferred into the receive buffer under the following conditions:

- There is room in the receive buffer for the data.
- MCE is set to 1 and the stop bit is also 1 (8-bit mode).
- MCE is set to 1 and the 9th bit is also 1 (9-bit mode).
- MCE is 0 (stop or 9th bit will be ignored).

In the event that there is not room in the receive buffer for the data, the most recently received data will be lost. The RI flag will be set any time that valid data has been pushed into the receive buffer. If RI interrupts are enabled, RI will trigger an interrupt. Firmware may read the 8 LSBs of received data by reading the SBUF0 register. The RB8 bit in SCON0 will represent the 9th received bit (in 9-bit mode) or the stop bit (in 8-bit mode), and should be read prior to reading SBUF0.

22.3.4 Multiprocessor Communications

9-Bit UART mode supports multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its ninth bit is logic 1; in a data byte, the ninth bit is always set to logic 0.

Setting the MCE bit of a slave processor configures its UART such that when a stop bit is received, the UART will generate an interrupt only if the ninth bit is logic 1 (RB8 = 1) signifying an address byte has been received. In the UART interrupt handler, software will compare the received address with the slave's own assigned 8-bit address. If the addresses match, the slave will clear its MCE bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its MCE bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

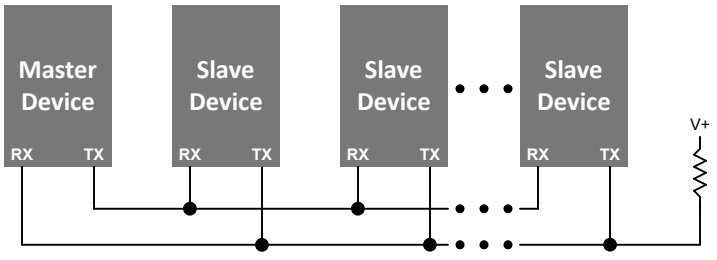


Figure 22.5. Multi-Processor Mode Interconnect Diagram

22.3.5 Routing RX Through Configurable Logic

The RX0 input of the UART is routed through the crossbar by default. It is also possible to route the RX input to the output of CLU0, CLU1 or CLU2. This function is selected by the RXSEL field in register UART0PCF.

22.4 UART0 Control Registers

22.4.1 SCON0: UART0 Serial Port Control

Bit	7	6	5	4	3	2	1	0
Name	SMODE	Reserved	MCE	REN	TB8	RB8	TI	RI
Access	RW	R	RW	RW	RW	R	RW	R
Reset	0	1	0	0	0	Varies	0	0
SFR Page = 0x0, 0x20; SFR Address: 0x98 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	SMODE	0	RW	**Serial Port 0 Operation Mode.**
	Selects the UART0 Operation Mode.			
	Value	Name		Description
	0	8_BIT		8-bit UART with Variable Baud Rate (Mode 0).
	1	9_BIT		9-bit UART with Variable Baud Rate (Mode 1).
6	*Reserved*	*Must write reset value.*		
5	MCE	0	RW	**Multiprocessor Communication Enable.**
	This bit enables checking of the stop bit or the 9th bit in multi-drop communication buses. The function of this bit is dependent on the UART0 operation mode selected by the SMODE bit. In Mode 0 (8-bits), the peripheral will check that the stop bit is logic 1. In Mode 1 (9-bits) the peripheral will check for a logic 1 on the 9th bit.			
	Value	Name		Description
	0	MULTI_DISABLED		Ignore level of 9th bit / Stop bit.
	1	MULTI_ENABLED		RI is set and an interrupt is generated only when the stop bit is logic 1 (Mode 0) or when the 9th bit is logic 1 (Mode 1).
4	REN	0	RW	**Receive Enable.**
	This bit enables/disables the UART receiver. When disabled, bytes can still be read from the receive FIFO, but the receiver will not place new data into the FIFO.			
	Value	Name		Description
	0	RECEIVE_DISABLED		UART0 reception disabled.
	1	RECEIVE_ENABLED		UART0 reception enabled.
3	TB8	0	RW	**Ninth Transmission Bit.**
	The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0).			
2	RB8	Varies	R	**Ninth Receive Bit.**
	RB8 is assigned the value of the STOP bit in Mode 0; it is assigned the value of the 9th data bit in Mode 1.			
1	TI	0	RW	**Transmit Interrupt Flag.**
	Set to a 1 by hardware after data has been transmitted at the beginning of the STOP bit. When the UART0 TI interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared by firmware.			

Bit	Name	Reset	Access	Description
0	RI	0	R	**Receive Interrupt Flag.** Set to 1 by hardware when a byte of data has been received by UART0 (set at the STOP bit sampling time). RI remains set while the receive FIFO contains any data. Hardware will clear this bit when the receive FIFO is empty. If a read of SBUF0 is performed when RI is cleared, the most recently received byte will be returned.

22.4.2 SBUF0: UART0 Serial Port Data Buffer

Bit	7	6	5	4	3	2	1	0
Name	SBUF0							
Access	RW							
Reset	Varies							
SFR Page = 0x0, 0x20; SFR Address: 0x99								

Bit	Name	Reset	Access	Description
7:0	SBUF0	Varies	RW	**Serial Data Buffer.** This SFR accesses the transmit and receive FIFOs. When data is written to SBUF0 and TXNF is 1, the data is placed into the transmit FIFO and is held for serial transmission. Any data in the TX FIFO will initiate a transmission. Writing to SBUF0 while TXNF is 0 will over-write the most recent byte in the TX FIFO. A read of SBUF0 returns the oldest byte in the RX FIFO. Reading SBUF0 when RI is 0 will continue to return the last available data byte in the RX FIFO.

22.4.3 UART0PCF: UART0 Pin Configuration

Bit	7	6	5	4	3	2	1	0
Name	Reserved						RXSEL	
Access	R						R	
Reset	0x00						0x0	
SFR Page = 0x20; SFR Address: 0xD9								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	RXSEL	0x0	R	**RX Input Select.** This field selects the source of the UART0 RX signal.

Value	Name	Description
0x0	CROSSBAR	RX is connected to the pin assigned by the crossbar.
0x1	CLU0	RX is connected to the CLU0 output signal.
0x2	CLU1	RX is connected to the CLU1 output signal.
0x3	CLU2	RX is connected to the CLU2 output signal.

23. Universal Asynchronous Receiver/Transmitter 1 (UART1)

23.1 Introduction

UART1 is an asynchronous, full duplex serial port offering a variety of data formatting options. A dedicated baud rate generator with a 16-bit timer and selectable prescaler is included, which can generate a wide range of baud rates. A received data FIFO allows UART1 to receive multiple bytes before data is lost and an overflow occurs.

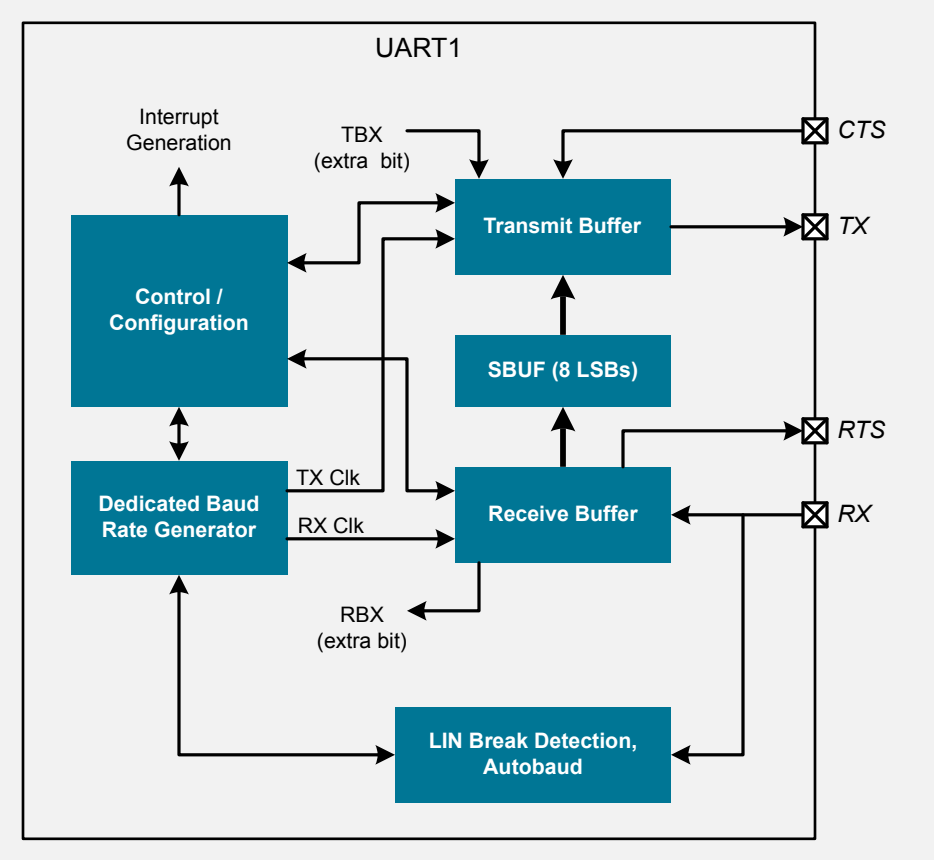


Figure 23.1. UART 1 Block Diagram

23.2 Features

UART1 provides the following features:

- Asynchronous transmissions and receptions
- Dedicated baud rate generator supports baud rates up to SYSCLK/2 (transmit) or SYSCLK/8 (receive)
- 5, 6, 7, 8, or 9 bit data
- Automatic start and stop generation
- Automatic parity generation and checking
- Single-byte buffer on transmit and receive
- Auto-baud detection
- LIN break and sync field detection
- CTS / RTS hardware flow control

23.3 Functional Description

23.3.1 Baud Rate Generation

The UART1 baud rate is generated by a dedicated 16-bit timer which runs from the controller's core clock (SYSCLK), and has prescaler options of 1, 4, 12, or 48. The timer and prescaler options combined allow for a wide selection of baud rates over many SYSCLK frequencies.

The baud rate generator is configured using three registers: SBCON1, SBRLH1, and SBRLL1. The SBCON1 register enables or disables the baud rate generator, and selects the prescaler value for the timer. The baud rate generator must be enabled for UART1 to function. Registers SBRLH1 and SBRLL1 constitute a 16-bit reload value (SBRL1) for the dedicated 16-bit timer. The internal timer counts up from the reload value on every clock tick. On timer overflows (0xFFFF to 0x0000), the timer is reloaded. For reliable UART receive operation, it is typically recommended that the UART baud rate does not exceed SYSCLK/16.

$$\text{Baud Rate} = \frac{\text{SYSCLK}}{(65536 - (\text{SBRL1})) \times 2 \times \text{Prescaler}}$$

23.3.2 Data Format

UART1 has a number of available options for data formatting. Data transfers begin with a start bit (logic low), followed by the data bits (sent LSB-first), a parity or extra bit (if selected), and end with one or two stop bits (logic high). The data length is variable between 5 and 8 bits. A parity bit can be appended to the data, and automatically generated and detected by hardware for even, odd, mark, or space parity. The stop bit length is selectable between short (1 bit time) and long (1.5 or 2 bit times), and a multi-processor communication mode is available for implementing networked UART buses.

All of the data formatting options can be configured using the SMOD1 register. Note that the extra bit feature is not available when parity is enabled, and the second stop bit is only an option for data lengths of 6, 7, or 8 bits.

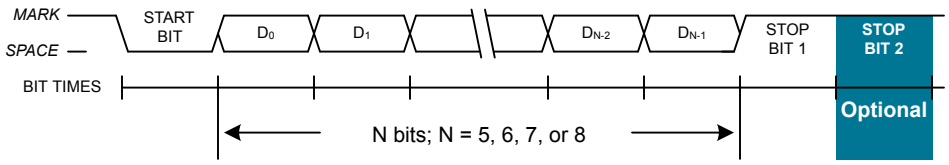


Figure 23.2. UART1 Timing Without Parity or Extra Bit

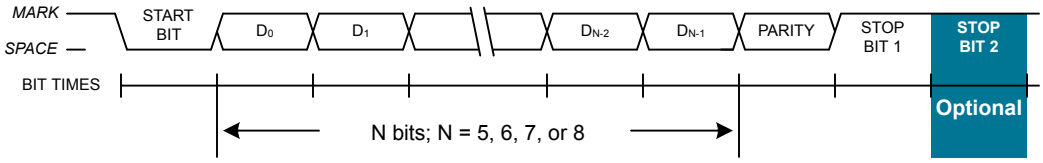


Figure 23.3. UART1 Timing With Parity

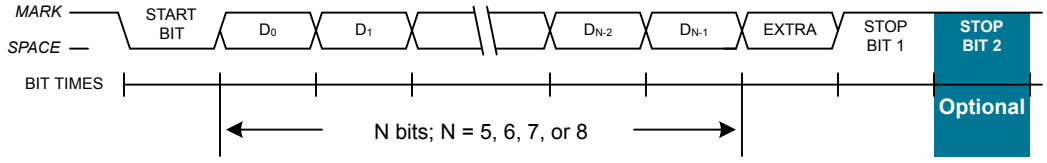


Figure 23.4. UART1 Timing With Extra Bit

23.3.3 Flow Control

The UART provides hardware flow control via the CTS and RTS pins. CTS and RTS may be individually enabled using the crossbar, may be operated independently of one another, and are active only when enabled through the crossbar.

The CTS pin is an input to the device. When CTS is held high, the UART will finish any byte transfer that is currently in progress, and then will halt before sending any more data. CTS must be returned low before data transfer will continue.

The RTS pin is an output from the device. When the receive buffer is full, RTS will toggle high. When data has been read from the buffer and there is additional room available, RTS will be cleared low.

23.3.4 Basic Data Transfer

UART1 provides standard asynchronous, full duplex communication. All data sent or received goes through the SBUF1 register, and (when an extra bit is enabled) the RBX bit in the SCON1 register.

Transmitting Data

Data transmission is initiated when software writes a data byte to the SBUF1 register. If XBE is set (extra bit enable), software should set up the desired extra bit in TBX prior to writing SBUF1. Data is transmitted LSB first from the TX pin. The TI flag in SCON1 is set at the end of the transmission (at the beginning of the stop-bit time). If TI interrupts are enabled, TI will trigger an interrupt.

Receiving Data

To enable data reception, firmware should write the REN bit to 1. Data reception begins when a start condition is recognized on the RX pin. Data will be received at the selected baud rate through the end of the data phase. Data will be transferred into the receive buffer under the following conditions:

- There is room in the receive buffer for the data.
- MCE is set to 1 and the stop bit is also 1 (XBE = 0).
- MCE is set to 1 and the extra bit is also 1 (XBE = 1).
- MCE is 0 (stop or extra bit will be ignored).

In the event that there is not room in the receive buffer for the data, the most recently received data will be lost. The RI flag will be set any time that valid data has been pushed into the receive buffer. If RI interrupts are enabled, RI will trigger an interrupt. Firmware may read the 8 LSBs of received data by reading the SBUF1 register. The RBX bit in SCON1 will represent the extra received bit or the stop bit, depending on whether XBE is enabled. If the extra bit is enabled, it should be read prior to reading SBUF1.

23.3.5 Data Transfer With FIFO

UART1 includes receive and transmit buffers to reduce the amount of overhead required for system interrupts. In applications requiring higher baud rates, the FIFOs may also be used to allow for additional latency when servicing interrupts. The transmit FIFO may be pre-loaded with additional bytes to maximize the outgoing throughput, while the receive FIFO allows the UART to continue receiving additional bytes of data between firmware reads. Configurable thresholds may be set by firmware to dictate when interrupts will be generated, and a receive timeout feature keeps received data from being orphaned in the receive buffer.

Both the receive and transmit FIFOs are configured using the UART1FCN0 and UART1FCN1 registers, and the number of bytes in the FIFOs may be determined at any time by reading UART1FCT.

Using the Transmit FIFO

Prior to using the transmit FIFO, the appropriate configuration settings for the application should be established:

- The TXTH field should be adjusted to the desired level. TXTH determines when the hardware will generate write requests and set the TXRQ flag. TXTH acts as a low watermark for the FIFO data, and the TXRQ flag will be set any time the number of bytes in the FIFO is less than or equal to the value of TXTH. For example, if the TXTH field is configured to 1, TXRQ will be set any time there are zero or one bytes left to send in the transmit FIFO.
- Disable TI interrupts by clearing the TIE bit to 0. TI will still be set at the completion of every byte sent from the UART, but the TI flag is typically not used in conjunction with the FIFO.
- Enable TFRQ interrupts by setting the TFRQE bit to 1.

As with basic data transfer, data transmission is initiated when software writes a data byte to the SBUF1 register. However, software may continue to write bytes to the buffer until the transmit FIFO is full. Software may determine when the FIFO is full either by reading the TXCNT directly from UART1FCT, or by monitoring the TXNF flag. TXNF is normally set to 1 when the transmit FIFO is not full, indicating that more data may be written. Any data written to SBUF1 when the transmit FIFO is full will over-write the most recent data written to the buffer, and a data byte will be lost.

In the course of normal operations, the transmit FIFO may be maintained with an interrupt-based system, filling the FIFO as space allows and servicing any write request interrupts that occur. If no more data is to be sent for some period of time, the TFRQ interrupt should be disabled by firmware until additional data will be sent.

In some situations, it may be necessary to halt transmission when there is still data in the FIFO. To do this, firmware should set the TXHOLD bit to 1. If a data byte is currently in progress, the UART will finish sending that byte and then halt before the nxet data byte. Trasnmission will not continue until TXHOLD is cleared to 0.

If it is necessary to flush the contents of the transmit FIFO entirely, firmware may do so by writing the TFLSH bit to 1. A flush will reset the internal FIFO counters and the UART will cease sending data.

Note: Hardware will clear the TFLSH bit back to 0 when the flush operation is complete. This takes only one SYSCLK cycle, so firmware will always read a 0 on this bit.

Using the Receive FIFO

The receive FIFO also has configuration settings which should be established prior to enabling UART reception:

- The RXTH field should be adjusted to the desired level. RXTH determines when the hardware will generate read requests and set the RXRQ flag. RXTH acts as a high watermark for the FIFO data, and the RXRQ flag will be set any time the number of bytes in the FIFO is greater than the value of RXTH. For example, if the RXTH field is configured to 0, RXRQ will be set any time there is at least one byte in the receive FIFO.
- (Optional) Disable RI interrupt by clearing the RIE bit to 0. The RI bit is still used in conjunction with receive FIFO operation - any time RI is set to 1, it indicates that the receive FIFO has more data. In most applications, it is more efficient to use the RXTH field to allow multiple bytes to be received between interrupts.
- (Optional) Enable RFRQ interrupts by setting the RFRQE bit to 1, and configure the RXTO field to enable receive timeouts. Receive timeouts may be adjusted using the RXTO field, to occur after 2, 4, or 16 idle periods without any activity on the RX pin. An "idle period" is defined as the full length of one transfer at the current baud rate, including start, stop, data, and any additional bits.

Once the receive buffer parameters and interrupts are configured, firmware should write the REN bit to 1 to enable data reception. Data reception begins when a start condition is recognized on the RX pin. Data will be received at the selected baud rate through the end of the data phase. Data will be transferred into the receive buffer under the following conditions:

- There is room in the receive buffer for the data.
- MCE is set to 1 and the stop bit is also 1 (XBE = 0).
- MCE is set to 1 and the extra bit is also 1 (XBE = 1).
- MCE is 0 (stop or extra bit will be ignored).

In the event that there is not room in the receive buffer for the data, the most recently received data will be lost.

The RI flag will be set any time an unread data byte is in the buffer (RXCNT is not equal to 0). Firmware may read the 8 LSBs of received data by reading the SBUF1 register. The RBX bit in SCON1 will represent the extra received bit or the stop bit, depending on whether XBE is enabled. If the extra bit is enabled, it should be read prior to reading SBUF1. Firmware may continue to read the receive buffer until it is empty (RI will be cleared to 0). If firmware reads the buffer while it is empty, the most recent data byte will be returned again.

If it is necessary to flush the contents of the receive FIFO entirely, firmware may do so by writing the RFLSH bit to 1. A flush will reset the internal FIFO counters and any data in the buffer will be lost.

Note: Hardware will clear the RFLSH bit back to 0 when the flush operation is complete. This takes only one SYSCLK cycle, so firmware will always read a 0 on this bit.

23.3.6 Multiprocessor Communications

UART1 supports multiprocessor communication between a master processor and one or more slave processors by special use of the extra data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its extra bit is logic 1; in a data byte, the extra bit is always set to logic 0.

Setting the MCE bit and the XBE bit in the SMOD1 register configures the UART for multi-processor communications. When a stop bit is received, the UART will generate an interrupt only if the extra bit is logic 1 (RBX = 1) signifying an address byte has been received. In the UART interrupt handler, software will compare the received address with the slave's own assigned address. If the addresses match, the slave will clear its MCE bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its MCE bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

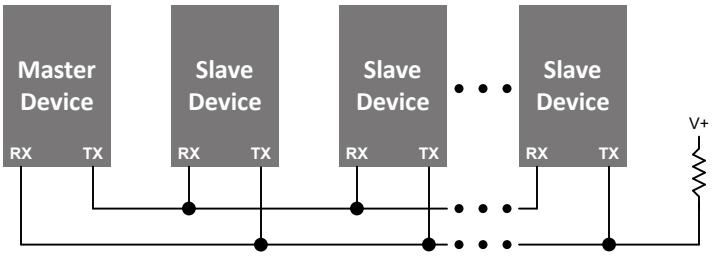


Figure 23.5. Multi-Processor Mode Interconnect Diagram

23.3.7 LIN Break and Sync Detect

UART1 contains dedicated hardware to assist firmware in LIN slave applications. It includes automatic detection of LIN break and sync fields, and can optionally perform automatic baud rate adjustment based on the LIN 0x55 sync word.

The LIN features are enabled by setting the LINMDE bit in UART1LIN to enable LIN mode. When enabled, both break and sync detection will be enabled for all incoming data. The circuitry can detect a break-sync sequence in the middle of an incoming data stream and react accordingly.

The UART will indicate that a break has been detected by setting the BREAKDN flag to 1. Likewise, hardware will set the SYNCD bit if a valid sync is detected, and the SYNCTO bit will indicate if a sync timeout has occured. The break done and sync flags may be individually enabled to generate UART1 interrupts by setting the BREAKDNIE, SYNCDIE, and SYNCTOIE bits to 1.

23.3.8 Autobaud Detection

Automatic baud rate detection and adjustment is supported by the UART. Autobaud may be enabled by setting the AUTOBDE bit in the UART1LIN register to 1. Although the autobaud feature is primarily targeted at LIN applications, it may be used stand-alone as well.

For use in LIN applications, the LINMDE bit should be set to 1. This requires that the UART see a valid LIN break, followed by a delimiter, and then a valid LIN sync word (0x55) before adjusting the baud rate. When used in LIN mode, the autobaud detection circuit may be left on during normal communications.

If LIN mode is not enabled (LINMDE = 0), the autobaud detection circuit will expect to see an 0x55 word on the received data path. The autobaud detection circuit operates by measuring the amount of time it takes to receive a sync word (0x55), and then adjusting the SBRL register value according to the measured time, given the current prescale settings.

Important: Because there is no break involved, when autobaud is used in non-LIN applications, it is important that the autobaud circuit only be enabled when the receiver is expecting an 0x55 sync byte. The SYNCD flag will be set upon detection of the sync byte, and firmware should disable auto-baud once the sync detection flag has been set.

The autobaud feature counts the number of prescaled clocks starting from the first rising edge of the sync field and ending on the last rising edge of the sync field. For 1% accuracy, the prescaler, system clock, and baud rate must be selected such that there are at least 100 clocks per bit. Because the baud rate generator overflows twice per bit, the resulting counts in the SBRLH1:SBRLL1 registers must be at least 50 (i.e. the maximum value of SBRLH1:SBRLL1 must be 65536 – 50, or 65486 and 0xFFCE.

23.3.9 Routing RX Through Configurable Logic

The RX1 input of the UART is routed through the crossbar by default. It is also possible to route the RX input to the output of CLU0, CLU1 or CLU2. This function is selected by the RXSEL field in register UART1PCF.

23.4 UART1 Control Registers

23.4.1 SCON1: UART1 Serial Port Control

Bit	7	6	5	4	3	2	1	0
Name	OVR	PERR	Reserved	REN	TBX	RBX	TI	RI
Access	RW	RW	R	RW	RW	R	RW	R
Reset	0	0	0	0	0	Varies	0	0
SFR Page = 0x20; SFR Address: 0xC8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	OVR	0	RW	**Receive FIFO Overrun Flag.**

This bit indicates a receive FIFO overrun condition, where an incoming character is discarded due to a full FIFO. This bit must be cleared by firmware.

Value	Name	Description
0	NOT_SET	Receive FIFO overrun has not occurred.
1	SET	Receive FIFO overrun has occurred.

Bit	Name	Reset	Access	Description
6	PERR	0	RW	**Parity Error Flag.**

When parity is enabled, this bit indicates that a parity error has occurred. It is set to 1 when the parity of the oldest byte in the FIFO (available when reading SBUF1) does not match the selected parity type. This bit must be cleared by firmware.

Value	Name	Description
0	NOT_SET	Parity error has not occurred.
1	SET	Parity error has occurred.

Bit	Name	Reset	Access	Description
5	*Reserved*	*Must write reset value.*		
4	REN	0	RW	**Receive Enable.**

This bit enables/disables the UART receiver. When disabled, bytes can still be read from the receive FIFO, but the receiver will not place new data into the FIFO.

Value	Name	Description
0	RECEIVE_DISABLED	UART1 reception disabled.
1	RECEIVE_ENABLED	UART1 reception enabled.

Bit	Name	Reset	Access	Description
3	TBX	0	RW	**Extra Transmission Bit.**

The logic level of this bit will be assigned to the extra transmission bit when XBE = 1 in the SMOD1 register. This bit is not used when parity is enabled.

Bit	Name	Reset	Access	Description
2	RBX	Varies	R	**Extra Receive Bit.**

RBX is assigned the value of the extra bit when XBE = 1 in the SMOD1 register. This bit is not valid when parity is enabled or when XBE is cleared to 0.

Bit	Name	Reset	Access	Description
1	TI	0	RW	**Transmit Interrupt Flag.**

Set to a 1 by hardware after data has been transmitted at the beginning of the STOP bit. When the UART1 TI interrupt is enabled, setting this bit causes the CPU to vector to the UART1 interrupt service routine. This bit must be cleared by firmware.

Bit	Name	Reset	Access	Description
0	RI	0	R	**Receive Interrupt Flag.** Set to 1 by hardware when a byte of data has been received by UART1 (set at the STOP bit sampling time). RI remains set while the receive FIFO contains any data. Hardware will clear this bit when the receive FIFO is empty. If a read of SBUF1 is performed when RI is cleared, the most recently received byte will be returned.

23.4.2 SMOD1: UART1 Mode

Bit	7	6	5	4	3	2	1	0
Name	MCE	SPT		PE	SDL		XBE	SBL
Access	RW	RW		RW	RW		RW	RW
Reset	0	0x0		0	0x3		0	0
SFR Page = 0x20; SFR Address: 0x93								

Bit	Name	Reset	Access	Description
7	MCE	0	RW	**Multiprocessor Communication Enable.**
	This function is not available when hardware parity is enabled.			
	Value	Name		Description
	0	MULTI_DISABLED		RI will be activated after the data is received.
	1	MULTI_ENABLED		RI will be activated if the stop bits (and extra bit if enabled) are 1. The extra bit must be enabled using XBE.
6:5	SPT	0x0	RW	**Parity Type.**
	Value	Name		Description
	0x0	ODD_PARITY		Odd.
	0x1	EVEN_PARITY		Even.
	0x2	MARK_PARITY		Mark.
	0x3	SPACE_PARITY		Space.
4	PE	0	RW	**Parity Enable.**
	This bit activates hardware parity generation and checking. The parity type is selected by the SPT field when parity is enabled.			
	Value	Name		Description
	0	PARITY_DISABLED		Disable hardware parity.
	1	PARITY_ENABLED		Enable hardware parity.
3:2	SDL	0x3	RW	**Data Length.**
	Value	Name		Description
	0x0	5_BITS		5 bits.
	0x1	6_BITS		6 bits.
	0x2	7_BITS		7 bits.
	0x3	8_BITS		8 bits.
1	XBE	0	RW	**Extra Bit Enable.**
	When enabled, the value of TBX in the SCON1 register will be appended to the data field.			
	Value	Name		Description
	0	DISABLED		Disable the extra bit.

Bit	Name	Reset	Access	Description
	1	ENABLED		Enable the extra bit.
0	SBL	0	RW	**Stop Bit Length.**
	Value	Name		Description
	0	SHORT		Short: Stop bit is active for one bit time.
	1	LONG		Long: Stop bit is active for two bit times (data length = 6, 7, or 8 bits) or 1.5 bit times (data length = 5 bits).

23.4.3 SBUF1: UART1 Serial Port Data Buffer

Bit	7	6	5	4	3	2	1	0
Name	SBUF1							
Access	RW							
Reset	Varies							
SFR Page = 0x20; SFR Address: 0x92								

Bit	Name	Reset	Access	Description
7:0	SBUF1	Varies	RW	**Serial Port Data Buffer.** This SFR accesses the transmit and receive FIFOs. When data is written to SBUF1 and TXNF is 1, the data is placed into the transmit FIFO and is held for serial transmission. Any data in the TX FIFO will initiate a transmission. Writing to SBUF1 while TXNF is 0 will over-write the most recent byte in the TX FIFO. A read of SBUF1 returns the oldest byte in the RX FIFO. Reading SBUF1 when RI is 0 will continue to return the last available data byte in the RX FIFO.

23.4.4 SBCON1: UART1 Baud Rate Generator Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved	BREN	Reserved			BPS		
Access	RW	RW	RW			RW		
Reset	0	0	0x0			0x0		
SFR Page = 0x20; SFR Address: 0x94								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	BREN	0	RW	**Baud Rate Generator Enable.**
	Value	Name		Description
	0	DISABLED		Disable the baud rate generator. UART1 will not function.
	1	ENABLED		Enable the baud rate generator.
5:3	*Reserved*	*Must write reset value.*		
2:0	BPS	0x0	RW	**Baud Rate Prescaler Select.**
	Value	Name		Description
	0x0	DIV_BY_12		Prescaler = 12.
	0x1	DIV_BY_4		Prescaler = 4.
	0x2	DIV_BY_48		Prescaler = 48.
	0x3	DIV_BY_1		Prescaler = 1.
	0x4	DIV_BY_8		Prescaler = 8.
	0x5	DIV_BY_16		Prescaler = 16.
	0x6	DIV_BY_24		Prescaler = 24.
	0x7	DIV_BY_32		Prescaler = 32.

23.4.5 SBRLH1: UART1 Baud Rate Generator High Byte

Bit	7	6	5	4	3	2	1	0
Name	BRH							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0x96								

Bit	Name	Reset	Access	Description
7:0	BRH	0x00	RW	**UART1 Baud Rate Reload High.**
	This field is the high byte of the 16-bit UART1 baud rate generator. The high byte of the baud rate generator should be written first, then the low byte. The baud rate is determined by the following equation: Baud Rate = (SYSCLK / (65536 - BRH1:BRL1)) * ((1 / 2) * (1 / Prescaler))			

23.4.6 SBRLL1: UART1 Baud Rate Generator Low Byte

Bit	7	6	5	4	3	2	1	0
Name	BRL							
Access	RW							
Reset	0x00							
SFR Page = 0x20; SFR Address: 0x95								

Bit	Name	Reset	Access	Description
7:0	BRL	0x00	RW	**UART1 Baud Rate Reload Low.** This field is the low byte of the 16-bit UART1 baud rate generator. The high byte of the baud rate generator should be written first, then the low byte. The baud rate is determined by the following equation: Baud Rate = (SYSCLK / (65536 - BRH1:BRL1)) * ((1 / 2) * (1 / Prescaler))

23.4.7 UART1FCN0: UART1 FIFO Control 0

Bit	7	6	5	4	3	2	1	0
Name	TFRQE	TFLSH	TXTH		RFRQE	RFLSH	RXTH	
Access	RW	RW	RW		RW	RW	RW	
Reset	0	0	0x0		0	0	0x0	
SFR Page = 0x20; SFR Address: 0x9D								

Bit	Name	Reset	Access	Description
7	TFRQE	0	RW	**Write Request Interrupt Enable.**
	When set to 1, a UART1 interrupt will be generated any time TFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		UART1 interrupts will not be generated when TFRQ is set.
	1	ENABLED		UART1 interrupts will be generated if TFRQ is set.
6	TFLSH	0	RW	**TX FIFO Flush.**
	This bit flushes the TX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will not be sent. Hardware will clear the TFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
5:4	TXTH	0x0	RW	**TX FIFO Threshold.**
	This field configures when hardware will set the transmit FIFO request bit (TFRQ). TFRQ is set whenever the number of bytes in the TX FIFO is equal to or less than the value in TXTH.			
	Value	Name		Description
	0x0	ZERO		TFRQ will be set when the TX FIFO is empty.
3	RFRQE	0	RW	**Read Request Interrupt Enable.**
	When set to 1, a UART1 interrupt will be generated any time RFRQ is logic 1.			
	Value	Name		Description
	0	DISABLED		UART1 interrupts will not be generated when RFRQ is set.
	1	ENABLED		UART1 interrupts will be generated if RFRQ is set.
2	RFLSH	0	RW	**RX FIFO Flush.**
	This bit flushes the RX FIFO. When firmware sets this bit to 1, the internal FIFO counters will be reset, and any remaining data will be lost. Hardware will clear the RFLSH bit back to 0 when the operation is complete (1 SYSCLK cycle).			
1:0	RXTH	0x0	RW	**RX FIFO Threshold.**
	This field configures when hardware will set the receive FIFO request bit (RFRQ). RFRQ is set whenever the number of bytes in the RX FIFO exceeds the value in RXTH.			
	Value	Name		Description
	0x0	ZERO		RFRQ will be set anytime new data arrives in the RX FIFO (when the RX FIFO is not empty).

23.4.8 UART1FCN1: UART1 FIFO Control 1

Bit	7	6	5	4	3	2	1	0
Name	TFRQ	TXNF	TXHOLD	TIE	RFRQ	RXTO		RIE
Access	R	R	RW	RW	R	RW		RW
Reset	1	1	0	1	0	0x0		1

SFR Page = 0x20; SFR Address: 0xD8 (bit-addressable)

Bit	Name	Reset	Access	Description
7	TFRQ	1	R	**Transmit FIFO Request.**

Set to 1 by hardware when the number of bytes in the TX FIFO is less than or equal to the TX FIFO threshold (TXTH).

Value	Name	Description
0	NOT_SET	The number of bytes in the TX FIFO is greater than TXTH.
1	SET	The number of bytes in the TX FIFO is less than or equal to TXTH.

Bit	Name	Reset	Access	Description
6	TXNF	1	R	**TX FIFO Not Full.**

This bit indicates when the TX FIFO is full and can no longer be written to. If a write is performed when TXNF is cleared to 0 it will replace the most recent byte in the FIFO.

Value	Name	Description
0	FULL	The TX FIFO is full.
1	NOT_FULL	The TX FIFO has room for more data.

Bit	Name	Reset	Access	Description
5	TXHOLD	0	RW	**Transmit Hold.**

This bit allows firmware to stall transmission until cleared. When set, the UART will complete any byte transmission in progress, but no further data will be sent. Transmission will continue when the TXHOLD bit is cleared. If CTS is used for hardware flow control, either TXHOLD or CTS assertion will cause transmission to stall.

Value	Name	Description
0	CONTINUE	The UART will continue to transmit any available data in the TX FIFO.
1	HOLD	The UART will not transmit any new data from the TX FIFO.

Bit	Name	Reset	Access	Description
4	TIE	1	RW	**Transmit Interrupt Enable.**

This bit enables the TI flag to generate UART1 interrupts after each byte is sent, regardless of the THTH settings.

Value	Name	Description
0	DISABLED	The TI flag will not generate UART1 interrupts.
1	ENABLED	The TI flag will generate UART1 interrupts when it is set.

Bit	Name	Reset	Access	Description
3	RFRQ	0	R	**Receive FIFO Request.**

Set to 1 by hardware when the number of bytes in the RX FIFO is larger than specified by the RX FIFO threshold (RXTH).

Value	Name	Description
0	NOT_SET	The number of bytes in the RX FIFO is less than or equal to RXTH.
1	SET	The number of bytes in the RX FIFO is greater than RXTH.

Bit	Name	Reset	Access	Description
2:1	RXTO	0x0	RW	**Receive Timeout.**
	This field defines the length of the timeout on the RX FIFO. If the RX FIFO is not empty but the number of bytes in the FIFO is not enough to generate a Receive FIFO request, an RFRQ interrupt will be generated after the specified number of idle frames. An "idle frame is defined as the length of a single transfer on the bus. For example, with a typical 8-N-1 configuration there are 8 data bits, 1 start bit, and 1 stop bit per transfer. An "idle frame" with this configuration is 10 bit times at the selected baud rate.			
	Value	Name		Description
	0x0	DISABLED		The receive timeout feature is disabled.
	0x1	TIMEOUT_2		A receive timeout will occur after 2 idle periods on the UART RX line.
	0x2	TIMEOUT_4		A receive timeout will occur after 4 idle periods on the UART RX line.
	0x3	TIMEOUT_16		A receive timeout will occur after 16 idle periods on the UART RX line.
0	RIE	1	RW	**Receive Interrupt Enable.**
	This bit enables the RI flag to generate UART1 interrupts when there is information available in the receive FIFO, regardless of the RXTH settings.			
	Value	Name		Description
	0	DISABLED		The RI flag will not generate UART1 interrupts.
	1	ENABLED		The RI flag will generate UART1 interrupts when it is set.

23.4.9 UART1FCT: UART1 FIFO Count

Bit	7	6	5	4	3	2	1	0
Name	Reserved	TXCNT			Reserved	RXCNT		
Access	R	R			R	R		
Reset	0	0x0			0	0x0		
SFR Page = 0x20; SFR Address: 0xFA								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	TXCNT	0x0	R	**TX FIFO Count.**
	This field indicates the number of bytes in the transmit FIFO.			
3	*Reserved*	*Must write reset value.*		
2:0	RXCNT	0x0	R	**RX FIFO Count.**
	This field indicates the number of bytes in the receive FIFO.			

23.4.10 UART1LIN: UART1 LIN Configuration

Bit	7	6	5	4	3	2	1	0
Name	AUTOBDE	BREAKDN	SYNCTO	SYNCD	LINMDE	BREAKDNIE	SYNCTOIE	SYNCDIE
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x20; SFR Address: 0x9E								

Bit	Name	Reset	Access	Description
7	AUTOBDE	0	RW	**Auto Baud Detection Enable.**

This bit enables auto-baud detection. Auto-baud measures the time it takes to receive the sync field (an 0x55 byte), and updates the baud rate reload registers accordingly.

Value	Name	Description
0	DISABLED	Autobaud is not enabled.
1	ENABLED	Autobaud is enabled.

Bit	Name	Reset	Access	Description
6	BREAKDN	0	RW	**LIN Break Done Flag.**

This bit is set by hardware after detection of a valid LIN break. This flag must be cleared by software.

Value	Name	Description
0	NOT_SET	A LIN break has not been detected.
1	BREAK	A LIN break was detected since the flag was last cleared.

Bit	Name	Reset	Access	Description
5	SYNCTO	0	RW	**LIN Sync Timeout Flag.**

This bit is set by hardware if a sync measurement in process overflows the baud rate generator. This is usually an indication that the prescaler must be increased. When a sync timeout occurs, the baud rate generator is not updated. Firmware must clear this bit to 0.

Value	Name	Description
0	NOT_SET	A sync timeout has not occured.
1	TIMEOUT	A sync timeout occured.

Bit	Name	Reset	Access	Description
4	SYNCD	0	RW	**LIN Sync Detect Flag.**

This bit is set by hardware after detection of a valid sync word. If LINMDE is set, the sync word must be part of a valid break-sync sequence. This flag must be cleared by software.

Value	Name	Description
0	NOT_SET	A sync has not been detected or is not yet complete.
1	SYNC_DONE	A valid sync word was detected.

Bit	Name	Reset	Access	Description
3	LINMDE	0	RW	**LIN Mode Enable.**

Enables a full LIN check on incoming data.

Value	Name	Description
0	DISABLED	If AUTOBDE is set to 1, sync detection and autobaud will begin on the first falling edge of RX.

Bit	Name	Reset	Access	Description
	1	ENABLED		A valid LIN break field and delimiter must be detected prior to the hardware state machine recognizing a sync word and performing autobaud.
2	BREAKDNIE	0	RW	**LIN Break Done Interrupt Enable.**
	Enables the break done interrupt source.			
	Value	Name		Description
	0	DISABLED		The BREAKDN flag will not generate UART1 interrupts.
	1	ENABLED		The BREAKDN flag will generate UART1 interrupts when it is set.
1	SYNCTOIE	0	RW	**LIN Sync Detect Timeout Interrupt Enable.**
	Enables the synctimeout interrupt source.			
	Value	Name		Description
	0	DISABLED		The SYNCTO flag will not generate UART1 interrupts.
	1	ENABLED		The SYNCTO flag will generate UART1 interrupts when it is set.
0	SYNCDIE	0	RW	**LIN Sync Detect Interrupt Enable.**
	Enables the sync detection interrupt source.			
	Value	Name		Description
	0	DISABLED		The SYNCD flag will not generate UART1 interrupts.
	1	ENABLED		The SYNCD flag will generate UART1 interrupts when it is set.

23.4.11 UART1PCF: UART1 Pin Configuration

Bit	7	6	5	4	3	2	1	0
Name	Reserved						RXSEL	
Access	R						R	
Reset	0x00						0x0	
SFR Page = 0x20; SFR Address: 0xDA								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	RXSEL	0x0	R	**RX Input Select.**
	This field selects the source of the UART1 RX signal.			
	Value	Name		Description
	0x0	CROSSBAR		RX is connected to the pin assigned by the crossbar.
	0x1	CLU0		RX is connected to the CLU0 output signal.
	0x2	CLU1		RX is connected to the CLU1 output signal.
	0x3	CLU2		RX is connected to the CLU2 output signal.

24. Precision Reference (VREF0)

24.1 Introduction

A precision voltage reference is included on-chip. The precision reference may be used to provide the voltage reference for the ADC, DACs, or used by other circuitry connected to the VREF pin.

24.2 Features

The precision voltage reference includes the following features:

- Stable and production-trimmed.
- Routes to VREF pin to source off-chip analog circuits.
- Two selectable levels: 1.2 V and 2.4 V.

24.3 Using the Precision Reference

The precision reference source has one control field, VREFSL located in the REF0CN register. VREFSL selects the output voltage for the reference. When set to NONE, the precision reference is off, and will not be connected to the VREF pin. The VREF_1P2 and VREF_2P4 settings both enable the reference and connect it to the VREF pin. VREF should be configured for analog mode when the reference is used, and an external bypass capacitor of at least 0.1 µF to ground is required.

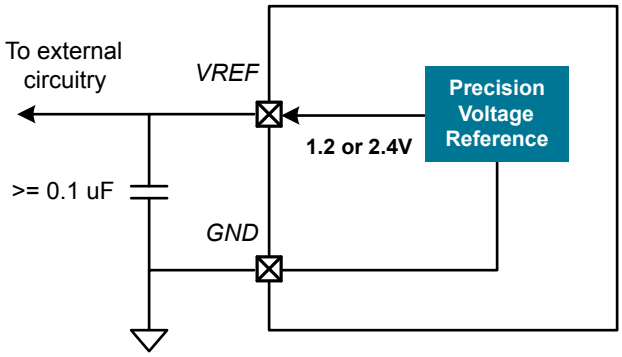


Figure 24.1. Precision Voltage Reference Connections

24.4 VREF Control Registers

24.4.1 REF0CN: Voltage Reference Control

Bit	7	6	5	4	3	2	1	0
Name	VREFSL		Reserved					
Access	RW		R					
Reset	0x0		0x00					
SFR Page = 0x0, 0x30; SFR Address: 0xD1								

Bit	Name	Reset	Access	Description
7:6	VREFSL	0x0	RW	**Voltage Reference Output Select.** Selects an on-chip reference to drive to the VREF pin.

Value	Name	Description
0x0	NONE	The VREF pin is not driven by an on-chip reference. It may be driven with an external reference.
0x1	VREF_1P2	1.2 V reference output to VREF pin.
0x2	VREF_2P4	2.4 V reference output to VREF pin.

5:0	*Reserved*	*Must write reset value.*		

25. Watchdog Timer (WDT0)

25.1 Introduction

The device includes a programmable watchdog timer (WDT) running off the low-frequency oscillator. A WDT overflow forces the MCU into the reset state. To prevent the reset, the WDT must be restarted by application software before overflow. If the system experiences a software or hardware malfunction preventing the software from restarting the WDT, the WDT overflows and causes a reset.

Following a reset, the WDT is automatically enabled and running with the default maximum time interval. If needed, the WDT can be disabled by system software or locked on to prevent accidental disabling. Once locked, the WDT cannot be disabled until the next system reset. The state of the RSTb pin is unaffected by this reset.

The WDT consists of an internal timer running from the low-frequency oscillator. The timer measures the period between specific writes to its control register. If this period exceeds the programmed limit, a WDT reset is generated. The WDT can be enabled and disabled as needed in software, or can be permanently enabled if desired. When the WDT is active, the low-frequency oscillator is forced on. All watchdog features are controlled via the Watchdog Timer Control Register (WDTCN).

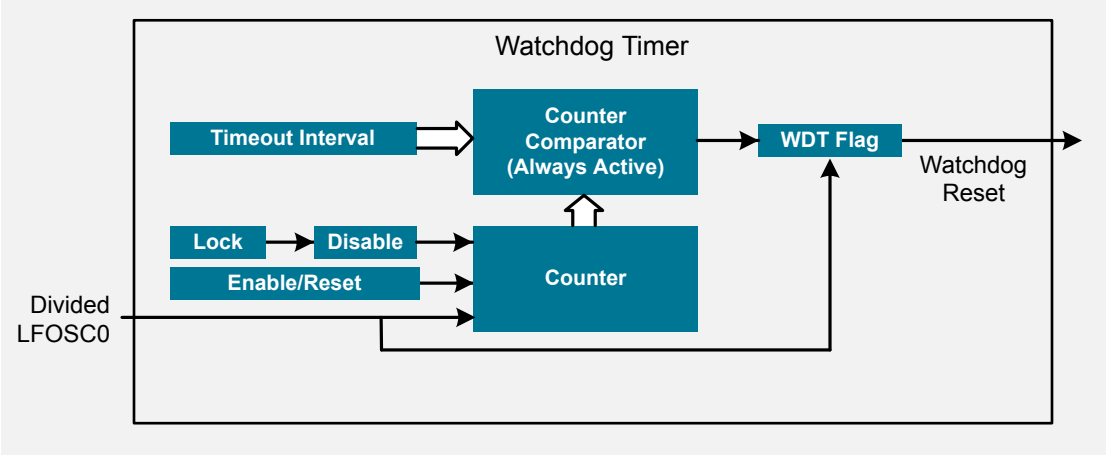


Figure 25.1. Watchdog Timer Block Diagram

25.2 Features

The watchdog timer includes a 16-bit timer with a programmable reset period. The registers are protected from inadvertent access by an independent lock and key interface.

The Watchdog Timer has the following features:

- Programmable timeout interval
- Runs from the low-frequency oscillator
- Lock-out feature to prevent any modification until a system reset

25.3 Using the Watchdog Timer

Enabling/Resetting the WDT

The watchdog timer is both enabled and reset when writing 0xA5 to the WDTCN register. The user's application software should include periodic writes of 0xA5 to WDTCN as needed to prevent a watchdog timer overflow. The counter is incremented on every divided LFOSC0 when the WDT is enabled. The WDT is enabled and reset as a result of any system reset.

Disabling the WDT

Writing 0xDE followed by 0xAD to the WDTCN register disables the WDT. The following code segment illustrates disabling the WDT:

```
CLR EA             ; disable all interrupts
MOV WDTCN,#0DEh    ; disable software watchdog timer
MOV WDTCN,#0ADh
; insert wait for 3 divided LFOSC0 clock periods
SETB EA            ; re-enable interrupts
```

Note: Code that implements the wait must be inserted. Code that implements the wait is not explicity implemented in the above sequence because it depends on the divided LFOSC0 clock and the SYSCLK clock selected.

The writes of 0xDE and 0xAD must occur within 4 clock cycles of each other, or the disable operation is ignored. Interrupts should be disabled during this procedure to avoid delay between the two writes.

The counter retains its value when the WDT is disabled. The counter comparator is always active and can generate a watchdog timer reset even if the watchdog timer is disabled. For example, a watchdog timer reset can be generated when changing from a higher to a lower interval as the counter is not cleared when the WDT is disabled. To avoid this, always clear the counter by resetting the WDT before disabling and changing the timeout interval to a lower interval i.e., follow the code sequence in ● Setting the WDT Interval on page 390.

Disabling the WDT Lockout

Writing 0xFF to WDTCN locks out the disable feature. Once locked out, the disable operation is ignored until the next system reset. Writing 0xFF does not enable or reset the watchdog timer. Applications always intending to use the watchdog should write 0xFF to WDTCN in the initialization code.

Setting the WDT Interval

WDTCN.[2:0] controls the watchdog timeout interval. The interval is given by the following equation, where T_{LFOSC} is the low-frequency oscillator clock period:

$$T_{LFOSC} \times 4^{(WDTCN[2:0]+3)}$$

This provides a nominal interval range of 0.8 ms to 13.1 s when LFOSC0 is configured to run at 80 kHz. WDTCN.7 must be logic 0 when setting this interval. Reading WDTCN returns the programmed interval. WDTCN.[2:0] reads 111b after a system reset.

The following code segment illustrates changing the WDT interval to a lower interval:

```
MOV WDTCN,#0A5h         ; reset watchdog timer
; insert wait for 2 divided LFOSC0 clock periods
CLR EA                  ; disable all interrupts
MOV WDTCN,#0DEh         ; disable software watchdog timer
MOV WDTCN,#0ADh
; insert wait for 3 divided LFOSC0 clock periods
SETB EA                 ; re-enable interrupts
MOV WDTCN,WDT_interval  ; change the current WDT interval to a lower interval with MSB cleared to 0
; insert wait for 1 SYSCLK period
```

Note: Code that implements the wait must be inserted. Code that implements the wait is not explicity implemented in the above sequence because it depends on the divided LFOSC0 clock and the SYSCLK clock selected.

Synchronization

The watchdog timer is controlled via the WDTCN control register using commands. Commands require synchronization between the system clock and WDT clock source, the divided LFOSC0 clock. The table below lists each WDT command and the number of clock periods from the specified clock source for the command to take effect.

Table 25.1. Synchronization Delay

Command	Clock Delay Required to Apply
Enabling/Resetting the WDT	2 divided LFOSC0 clock periods
Setting the WDT Interval	1 SYSCLK clock period
Disabling the WDT	3 divided LFOSC0 clock periods

Due to the WDT command synchronization delay, observe the following guidelines while operating the WDT:

- Only issue one command to WDTCN register within one divided LFOSC0 clock period.
- Change the LFOSC0 divider or disable the LFOSC0 only while the WDT is disabled.
- Change the WDT interval only while the WDT is disabled.

25.4 WDT0 Control Registers

25.4.1 WDTCN: Watchdog Timer Control

Bit	7	6	5	4	3	2	1	0
Name	WDTCN							
Access	RW							
Reset	0x17							
SFR Page = ALL; SFR Address: 0x97								

Bit	Name	Reset	Access	Description
7:0	WDTCN	0x17	RW	**WDT Control.** The WDT control field has different behavior for reads and writes. Read: When reading the WDTCN register, the lower three bits (WDTCN[2:0]) indicate the current timeout interval. Bit WDTCN.4 indicates whether the WDT is active (logic 1) or inactive (logic 0). Write: Writing the WDTCN register can set the timeout interval, enable the WDT, disable the WDT, reset the WDT, or lock the WDT to prevent disabling. Writing to WDTCN with the MSB (WDTCN.7) cleared to 0 will set the timeout interval to the value in bits WDTCN[2:0]. Writing 0xA5 both enables and reloads the WDT. Writing 0xDE followed within 4 system clocks by 0xAD disables the WDT. Writing 0xFF locks out the disable feature until the next device reset.

26. C2 Debug and Programming Interface

26.1 Introduction

The device includes an on-chip Silicon Labs 2-Wire (C2) debug interface that allows flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. Details on the C2 protocol can be found in the C2 Interface Specification.

26.2 Features

The C2 interface provides the following features:

- In-system device programming and debugging.
- Non-intrusive - no firmware or hardware peripheral resources required.
- Allows inspection and modification of all memory spaces and registers.
- Provides hardware breakpoints and single-step capabilites.
- Can be locked via flash security mechanism to prevent unwanted access.

26.3 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and flash programming may be performed. C2CK is shared with the RSTb pin, while the C2D signal is shared with a port I/O pin. This is possible because C2 communication is typically performed when the device is in the halt state, where all on-chip peripherals and user software are stalled. In this halted state, the C2 interface can safely "borrow" the C2CK and C2D pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application.

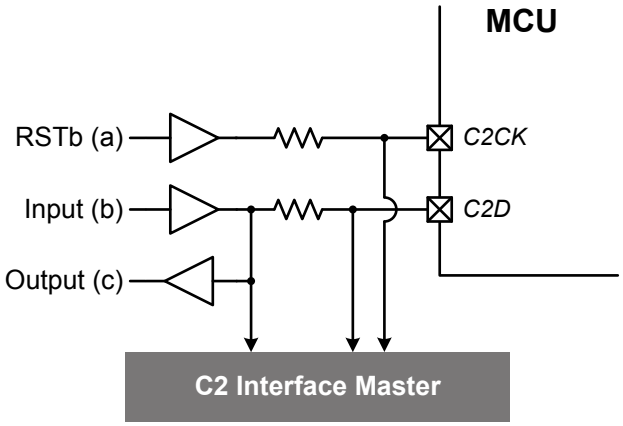


Figure 26.1. Typical C2 Pin Sharing

The configuration above assumes the following:

- The user input (b) cannot change state while the target device is halted.
- The RSTb pin on the target device is used as an input only.

Additional resistors may be necessary depending on the specific application.

26.4 C2 Interface Registers

26.4.1 C2ADD: C2 Address

Bit	7	6	5	4	3	2	1	0
Name	C2ADD							
Access	RW							
Reset	0x00							
This register is part of the C2 protocol.								

Bit	Name	Reset	Access	Description
7:0	C2ADD	0x00	RW	**C2 Address.** The C2ADD register is accessed via the C2 interface. The value written to C2ADD selects the target data register for C2 Data Read and Data Write commands. 0x00: C2DEVID 0x01: C2REVID 0x02: C2FPCTL 0xB4: C2FPDAT

26.4.2 C2DEVID: C2 Device ID

Bit	7	6	5	4	3	2	1	0
Name	C2DEVID							
Access	R							
Reset	0x34							
C2 Address: 0x00								

Bit	Name	Reset	Access	Description
7:0	C2DEVID	0x34	R	**Device ID.** This read-only register returns the 8-bit device ID.

26.4.3 C2REVID: C2 Revision ID

Bit	7	6	5	4	3	2	1	0
Name	C2REVID							
Access	R							
Reset	Varies							
C2 Address: 0x01								

Bit	Name	Reset	Access	Description
7:0	C2REVID	Varies	R	**Revision ID.** This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A.

26.4.4 C2FPCTL: C2 Flash Programming Control

Bit	7	6	5	4	3	2	1	0
Name	C2FPCTL							
Access	RW							
Reset	0x00							
C2 Address: 0x02								

Bit	Name	Reset	Access	Description
7:0	C2FPCTL	0x00	RW	**Flash Programming Control Register.** This register is used to enable flash programming via the C2 interface. To enable C2 flash programming, the following codes must be written in order: 0x02, 0x01. Note that once C2 flash programming is enabled, a system reset must be issued to resume normal operation.

26.4.5 C2FPDAT: C2 Flash Programming Data

Bit	7	6	5	4	3	2	1	0
Name	C2FPDAT							
Access	RW							
Reset	0x00							
C2 Address: 0xB4								

Bit	Name	Reset	Access	Description
7:0	C2FPDAT	0x00	RW	**C2 Flash Programming Data Register.** This register is used to pass flash commands, addresses, and data during C2 flash accesses. Valid commands are listed below. 0x03: Device Erase 0x06: Flash Block Read 0x07: Flash Block Write 0x08: Flash Page Erase

27. Revision History

Revision 0.7

October, 2020

- Added crystal oscillator modes for revision B and D devices.
- In 11.3.2.1 Port I/O Analog Assignments, renamed ADC high-performance inputs to high quality inputs for consistency.
- In 18.3.8 PWM Waveform Generation, updated formula for center aligned PWM edge transitions.
- In Table 20.2 Sources for Hardware Changes to SMB0CN0 on page 313, added footnote.
- Removed ADLPM bit description from 12.4.5 ADC0CF1: ADC0 Configuration as LP mode is fixed enabled and not configurable for this device.
- Updated 23.4.1 SCON1: UART1 Serial Port Control and 23.4.2 SMOD1: UART1 Mode RBX and MCE bit descriptions.
- Revised 15.4.1 CRC0CN0: CRC0 Control 0 CRCINIT bit description.
- Updated 18.3.8.3 Comparator Clear Function to include Comparator 1 clear selection.
- Changed comparator 0 wake up trigger to level triggered instead of edge triggered in 7.1 Introduction.
- Removed timer overflow as an option for the CLU DFF clock.
- Removed comparator VDD min/max value

Revision 0.6

December, 2018

- Updated Silicon Labs recommendation for code security when using the Bootloader in the 1.10 Bootloader section.
- Updated Silicon Labs security recommendation in the 4.3.1 Security Options section.
- Changed V_{RST} to V_{POR} in the 9.3.2 Power-On Reset section.
- Changed CMP0 to CMPn in the 13.3.4 Output Routing section.
- Corrected the code size and the flash pages in Figure 2.1, Figure 2.4 and Figure 4.1 in the 2.4 Memory Map section and 4.1 Introduction sections.
- Corrected the SPI0 FIFO length in Figure 19.1 in the 19.1 Introduction section.
- Updated Figure 1.1 in the 1.1 Introduction section to removed external crystal oscillator as clock source.
- Corrected the definition of I2C0INT in the 17.4.4 I2C0STAT: I2C0 Status register.
- Corrected the CMXN setting for CMPN0.11 channel in the CMP1 Negative Input Multiplexer Channels table.
- Removed all references to XTAL and renamed it to EXTOSC.

Revision 0.5

October, 2018

- Updated the WDT block diagram in 25.1 Introduction.
- Updated the WDT behaviour for all the commands in 25.3 Using the Watchdog Timer section.
- Added Synchronization section in 25.3 Using the Watchdog Timer.

Revision 0.4

September, 2017

- Updated the coloring and formatting.
- Added Revision History.
- Removed preliminary language and updated the applications listed on the front page.
- Added a paragraph to 1.1 Introduction to describe the document organization with the Data Sheet and Reference Manual.
- Added pinout information to 1.10 Bootloader.
- Added all mentions of buffers to FIFOs and added sizing information.
- Updated the 5.3.2 DERIVID: Derivative Identification register to include all I-grade and A-grade part numbers and corrected the DERIVID value for EFM8BB31F32I-{R}-QFN24 part numbers.
- Updated 8.3.4 LFOSC0 80 kHz Internal Oscillator to correct the timer capture source to the falling edge of the LFOSC.
- Corrected the register names in 17.3.3 Automatic Address Recognition.
- Adjusted the language of the SMBus Timing Control section in 20.3.3 Configuring the SMBus Module.
- Fixed the baud rate equation in 23.3.1 Baud Rate Generation.

Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com