EFM8 Universal Bee Family
EFM8UB1 Reference Manual

The EFM8UB1, part of the Universal Bee family of MCUs, is a multi-purpose line of 8-bit microcontrollers with USB feature set in small packages.

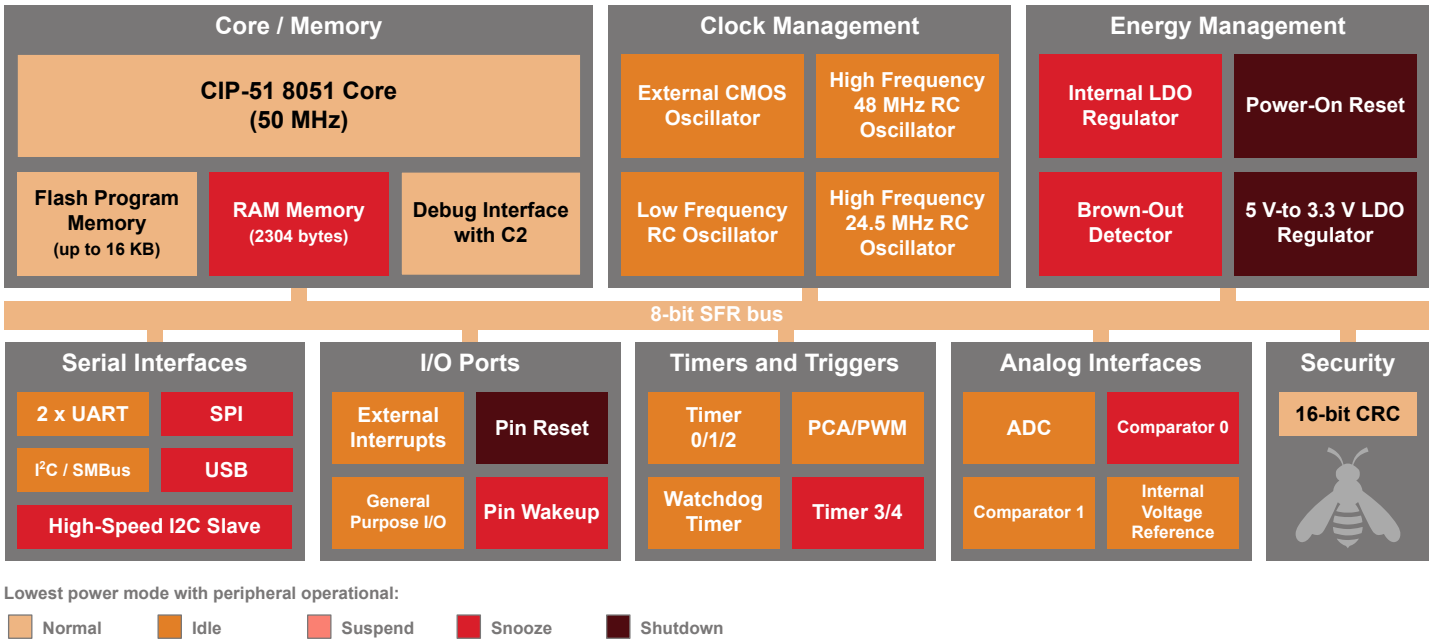
These devices offer high value by integrating an innovative energy-smart USB peripheral interface, charger detect circuit, 8 kV ESD protection, and enhanced high speed communication interfaces into small packages, making them ideal for space-constrained USB applications. With an efficient 8051 core and precision analog, the EFM8UB1 family is also optimal for embedded applications.

EFM8UB1 applications include the following:

- USB I/O controls, dongles
- High-speed communication bridge
- Consumer electronics
- Medical equipment

KEY FEATURES

- Pipelined 8-bit C8051 core with 50 MHz maximum operating frequency
- Up to 22 multifunction, 5 V tolerant I/O pins
- Low Energy USB with full- and low-speed support saves up to 90% of the USB energy
- USB charger detect circuit (USB-BCS 1.2 compliant)
- One 12-bit ADC and two analog comparators with internal voltage DAC as reference input
- Five 16-bit timers
- Two UARTs, SPI, SMBus/I2C master/slave and I2C slave
- Priority crossbar for flexible pin mapping



1. System Overview

1.1 Introduction

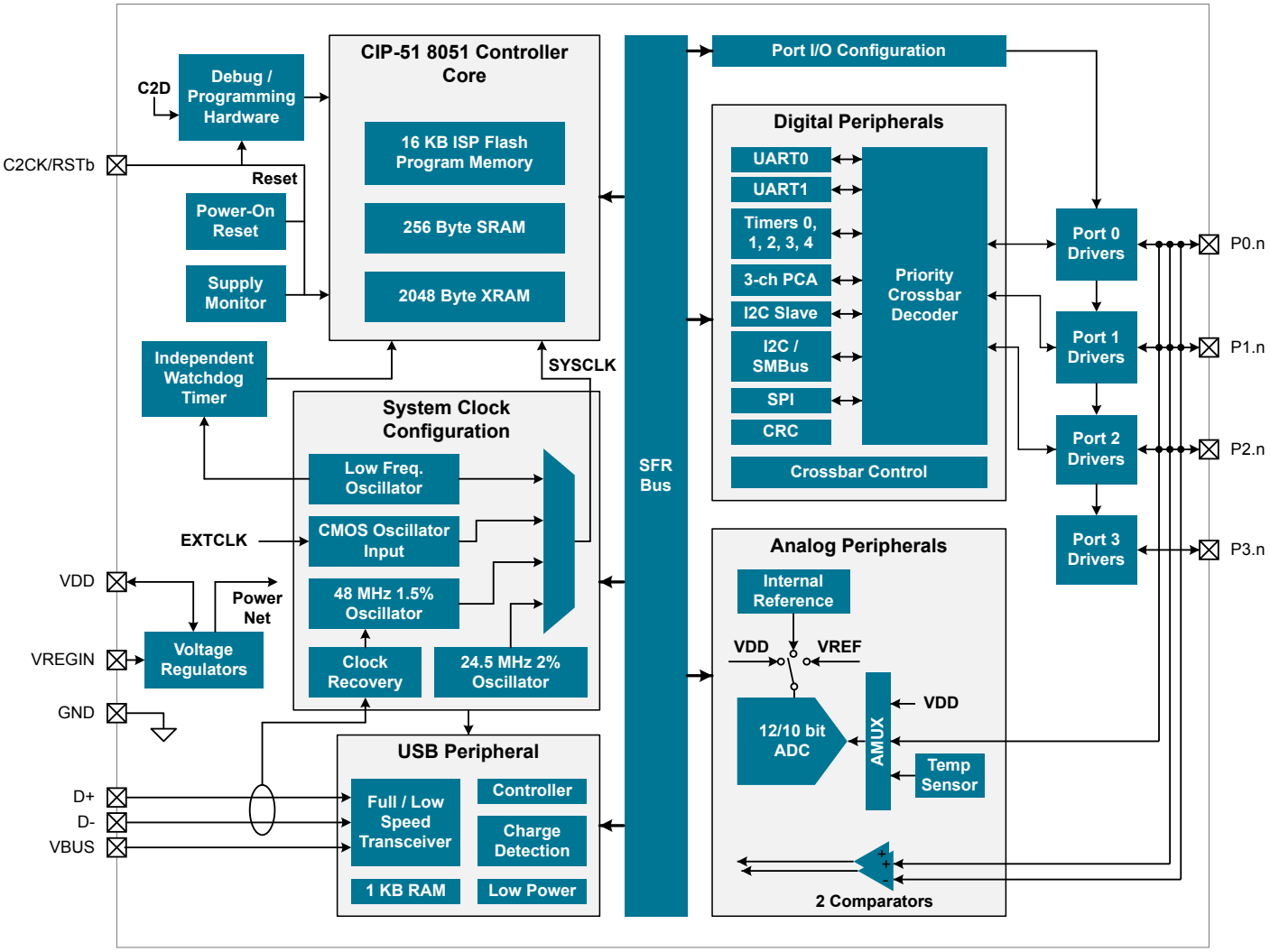


Figure 1.1. Detailed EFM8UB1 Block Diagram

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 • Regulators 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	• USB0 Bus Activity • Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Rising Edge
Snooze	• Core and peripheral clocks halted • HFOSC0 and HFOSC1 oscillators stopped • Regulators 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	• USB0 Bus Activity • Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Rising Edge
Shutdown	• All internal power nets shut down • 5V regulator remains active (if enabled) • 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 P3.0 and P3.1 can be used as GPIO. Additionally, the C2 Interface Data signal (C2D) is shared with P3.0.

The port control block offers the following features:

- Up to 22 multi-functions I/O pins, supporting digital and analog functions.
- Flexible priority crossbar decoder for digital peripheral assignment.
- Two drive strength settings for each port.
- Two direct-pin interrupt sources with dedicated interrupt vectors (INT0 and INT1).
- Up to 20 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.
- 48 MHz internal oscillator (HFOSC1), accurate to ±1.5% over supply and temperature corners.
- 80 kHz low-frequency oscillator (LFOSC0).
- External CMOS clock input (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 three 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

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

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 and Timer 4 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.
- USB Start-of-Frame (SOF) 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 Serial Bus (USB0)

The USB0 peripheral provides a full-speed USB 2.0 compliant device controller and PHY with additional Low Energy USB features. The device supports both full-speed (12MBit/s) and low speed (1.5MBit/s) operation, and includes a dedicated USB oscillator with clock recovery mechanism for crystal-free operation. No external components are required. The USB function controller (USB0) consists of a Serial Interface Engine (SIE), USB transceiver (including matching resistors and configurable pull-up resistors), and 1 KB FIFO block. The Low Energy Mode ensures the current consumption is optimized and enables USB communication on a strict power budget.

The USB0 module includes the following features:

- Full and Low Speed functionality.
- Implements 4 bidirectional endpoints.
- Low Energy Mode to reduce active supply current based on bus bandwidth.
- USB 2.0 compliant USB peripheral support (no host capability).
- Direct module access to 1 KB of RAM for FIFO memory.
- Clock recovery to meet USB clocking requirements with no external components.
- Charger detection circuitry with automatic detection of SDP, CDP, and DCP interfaces.
- D+ and D- can be routed to ADC input to support ACM and proprietary charger architectures.

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

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.
- Four byte FIFO 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).
- Four 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 buffers 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

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

1.7 Analog

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

The ADC is a successive-approximation-register (SAR) ADC with 12-, 10-, and 8-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 200 ksps samples per second in 12-bit mode or 800 ksps samples per second in 10-bit mode.
- Operation in low power modes at lower conversion speeds.
- Asynchronous hardware conversion trigger, selectable between software, external I/O and internal timer sources.
- Output data window comparator allows automatic range checking.
- Support for burst mode, which produces one set of accumulated data per conversion-start trigger with programmable power-on settling and tracking time.
- Conversion complete and window compare interrupts supported.
- Flexible output data formatting.
- Includes an internal fast-settling reference with two levels (1.65 V and 2.4 V) and support for external reference and signal ground.
- Integrated temperature sensor.

Low Current Comparators (CMP0, CMP1)

Analog comparators are 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 12 (CMP1) external positive inputs
- Up to 10 (CMP0) or 12 (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. The Port I/O latches are reset to 1 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. 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:

- 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
- USB reset

1.9 Debugging

The EFM8UB1 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 USB bootloader. This bootloader resides in flash and can be erased if it is not needed.

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

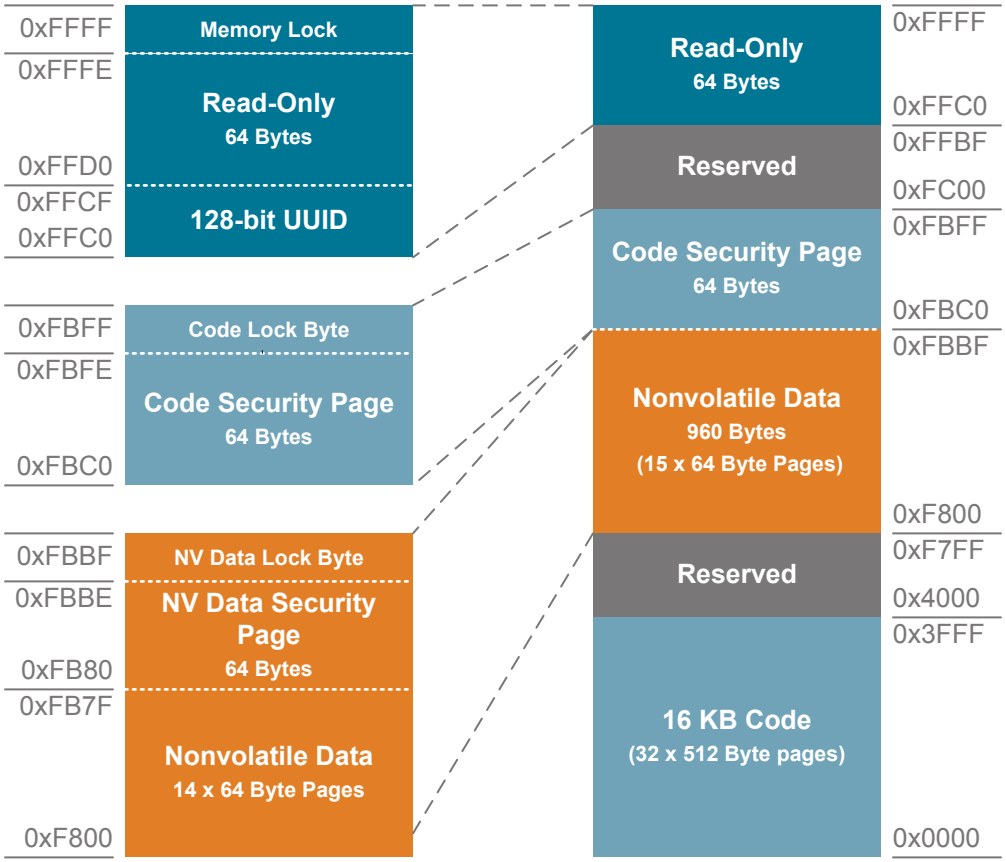


Figure 2.1. Flash Memory Map — 16 KB Devices

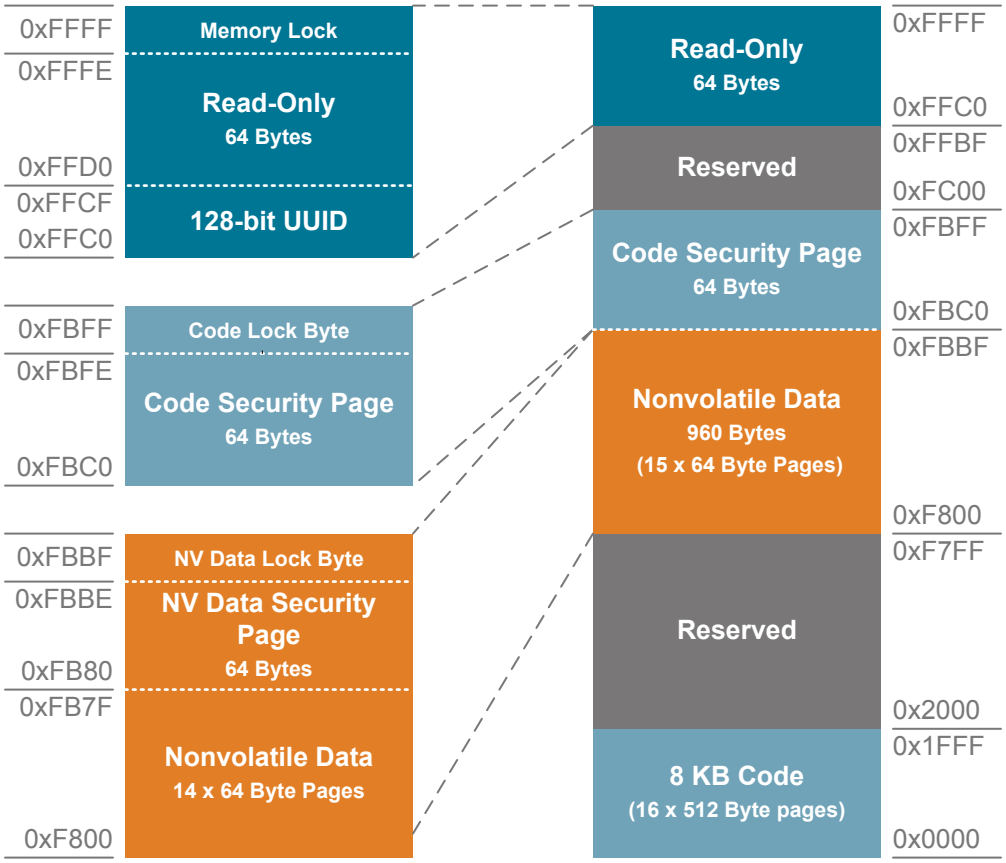


Figure 2.2. Flash Memory Map — 8 KB Devices

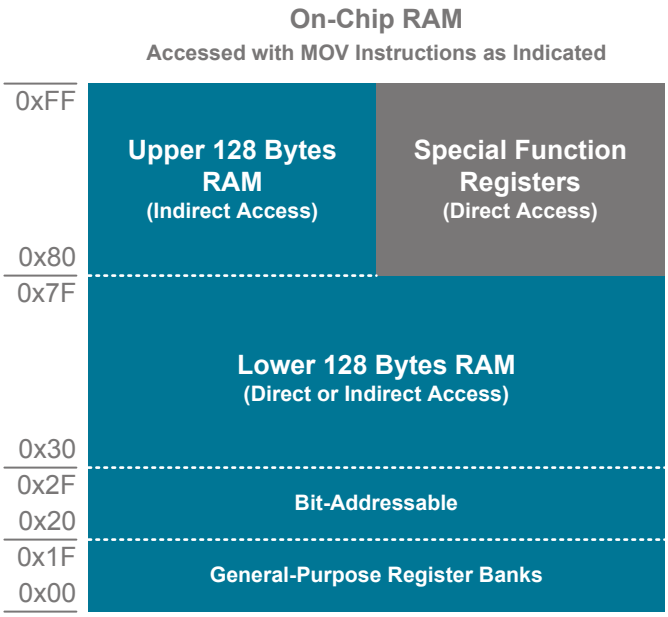


Figure 2.3. Direct / Indirect RAM Memory

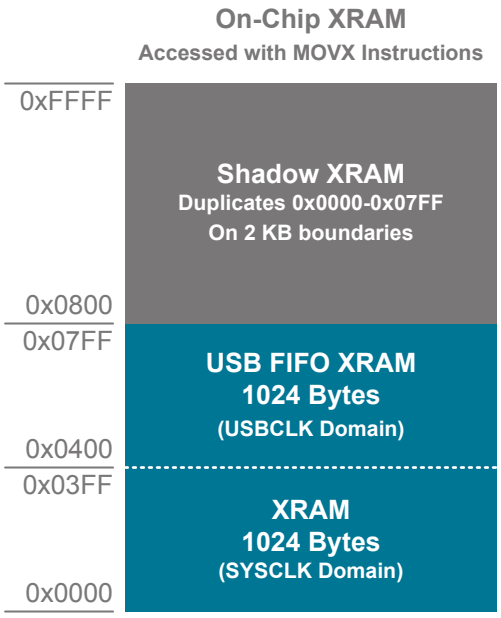


Figure 2.4. XRAM Memory

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	0x00					0x0		
SFR Page = ALL; SFR Address: 0xE7								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
2:0	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 EFM8UB1 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.

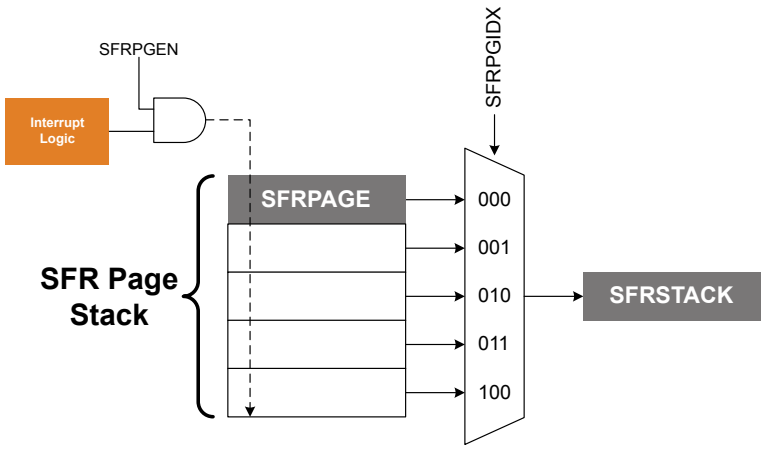
Interrupts and the SFR Page Stack

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 (*bit-address-able)	SFR Page 0x00	0x10	0x20	Address (*bit-address-able)	SFR Page 0x00	0x10	0x20
0x80*	P0			0xC0*	SMB0CN0	-	SMB0CN0
0x81	SP			0xC1	SMB0CF	PFE0CN	SMB0CF
0x82	DPL			0xC2	SMB0DAT	-	SMB0DAT
0x83	DPH			0xC3	ADC0GTL		SMB0FCN0
0x84	-			0xC4	ADC0GTH		SMB0FCN1
0x85	-			0xC5	ADC0LTL		SMB0RXLN
0x86	CRC0CN1	-	CRC0CN1	0xC6	ADC0LTH		REG1CN
0x87	PCON0			0xC7	HFO0CAL		-
0x88*	TCON			0xC8*	TMR2CN0		SCON1
0x89	TMOD			0xC9	REG0CN	-	REG0CN
0x8A	TL0			0xCA	TMR2RLL		-
0x8B	TL1			0xCB	TMR2RLH		-
0x8C	TH0			0xCC	TMR2L		P2SKIP
0x8D	TH1			0xCD	TMR2H		-
0x8E	CKCON0			0xCE	CRC0CN0	EIE2	CRC0CN0
0x8F	PSCTL			0xCF	CRC0FLIP	SFRPGCN	CRC0FLIP
0x90*	P1			0xD0*	PSW		
0x91	TMR3CN0		-	0xD1	REF0CN		-
0x92	TMR3RLL		SBUF1	0xD2	CRC0ST	-	CRC0ST
0x93	TMR3RLH		SMOD1	0xD3	CRC0CNT	-	CRC0CNT
0x94	TMR3L		SBCON1	0xD4	P0SKIP	-	P0SKIP
0x95	TMR3H		SBRLL1	0xD5	P1SKIP	-	P1SKIP
0x96	PCA0POL		SBRLH1	0xD6	SMB0ADM	HFO1CAL	SMB0ADM
0x97	WDTCN			0xD7	SMB0ADR	SFRSTACK	SMB0ADR
0x98*	SCON0	TMR4CN0	SCON0	0xD8*	PCA0CN0		UART1FCN1
0x99	SBUF0	CMP0CN1	SBUF0	0xD9	PCA0MD		-
0x9A	PCON1	-	SPI0FCN0	0xDA	PCA0CPM0		-
0x9B	CMP0CN0		SPI0FCN1	0xDB	PCA0CPM1		-
0x9C	PCA0CLR		P3MDOUT	0xDC	PCA0CPM2		-
0x9D	CMP0MD		UART1FCN0	0xDD	CRC0IN	-	CRC0IN
0x9E	PCA0CENT		UART1LIN	0xDE	CRC0DAT	-	CRC0DAT
0x9F	CMP0MX		-	0xDF	ADC0PWR		-
0xA0*	P2			0xE0*	ACC		

Address (*bit-address-able)	SFR Page 0x00	0x10	0x20	Address (*bit-address-able)	SFR Page 0x00	0x10	0x20
0xA1	SPI0CFG	-	SPI0CFG	0xE1	XBR0	-	XBR0
0xA2	SPI0CKR	TMR4RLL	SPI0CKR	0xE2	XBR1	-	XBR1
0xA3	SPI0DAT	TMR4RLH	SPI0DAT	0xE3	XBR2	-	XBR2
0xA4	P0MDOUT	TMR4L	P0MDOUT	0xE4	IT01CF		-
0xA5	P1MDOUT	TMR4H	P1MDOUT	0xE5	-		
0xA6	P2MDOUT	CKCON1	P2MDOUT	0xE6	EIE1		-
0xA7	SFRPAGE			0xE7	EMI0CN		
0xA8*	IE			0xE8*	ADC0CN0		-
0xA9	CLKSEL			0xE9	PCA0CPL1		-
0xAA	CMP1MX		-	0xEA	PCA0CPH1		-
0xAB	CMP1MD		I2C0FCN1	0xEB	PCA0CPL2		-
0xAC	SMB0TC	CMP1CN1	SMB0TC	0xEC	PCA0CPH2		-
0xAD	DERIVID	-	I2C0FCN0	0xED	P1MAT	-	P1MAT
0xAE	USB0ADR			0xEE	P1MASK	-	P1MASK
0xAF	USB0DAT			0xEF	RSTSRC	HFOCN	SMB0FCT
0xB0*	P3			0xF0*	B		
0xB1	LFO0CN		-	0xF1	P0MDIN	-	P0MDIN
0xB2	ADC0CN1		USB0AEC	0xF2	P1MDIN	IPH	P1MDIN
0xB3	ADC0AC		USB0XCN	0xF3	EIP1		P2MDIN
0xB4	-			0xF4	-	EIP2	P3MDIN
0xB5	DEVICEID	-	USB0CF	0xF5	-	EIP1H	I2C0FCT
0xB6	REVID	-	USB0CDCF	0xF6	PRTDRV	EIP2H	PRTDRV
0xB7	FLKEY			0xF7	PCA0PWM		SPI0FCT
0xB8*	IP			0xF8*	SPI0CN0	-	SPI0CN0
0xB9	ADC0TK		I2C0STAT	0xF9	PCA0L		-
0xBA	-		I2C0CN0	0xFA	PCA0H		UART1FCT
0xBB	ADC0MX		I2C0DOUT	0xFB	PCA0CPL0		P2MAT
0xBC	ADC0CF		I2C0DIN	0xFC	PCA0CPH0		P2MASK
0xBD	ADC0L		I2C0SLAD	0xFD	P0MAT	TMR2CN1	P0MAT
0xBE	ADC0H		USB0CDCN	0xFE	P0MASK	TMR3CN1	P0MASK
0xBF	CMP1CN0		USB0CDSTA	0xFF	VDM0CN	TMR4CN1	-

Table 3.3. Special Function Registers by Name

Register	Address	SFR Pages	Description
ACC	0xE0	ALL	Accumulator

Register	Address	SFR Pages	Description
ADC0AC	0xB3	0x00, 0x10	ADC0 Accumulator Configuration
ADC0CF	0xBC	0x00, 0x10	ADC0 Configuration
ADC0CN0	0xE8	0x00, 0x10	ADC0 Control 0
ADC0CN1	0xB2	0x00, 0x10	ADC0 Control 1
ADC0GTH	0xC4	0x00, 0x10	ADC0 Greater-Than High Byte
ADC0GTL	0xC3	0x00, 0x10	ADC0 Greater-Than Low Byte
ADC0H	0xBE	0x00, 0x10	ADC0 Data Word High Byte
ADC0L	0xBD	0x00, 0x10	ADC0 Data Word Low Byte
ADC0LTH	0xC6	0x00, 0x10	ADC0 Less-Than High Byte
ADC0LTL	0xC5	0x00, 0x10	ADC0 Less-Than Low Byte
ADC0MX	0xBB	0x00, 0x10	ADC0 Multiplexer Selection
ADC0PWR	0xDF	0x00, 0x10	ADC0 Power Control
ADC0TK	0xB9	0x00, 0x10	ADC0 Burst Mode Track Time
B	0xF0	ALL	B Register
CKCON0	0x8E	ALL	Clock Control 0
CKCON1	0xA6	0x10	Clock Control 1
CLKSEL	0xA9	ALL	Clock Select
CMP0CN0	0x9B	0x00, 0x10	Comparator 0 Control 0
CMP0CN1	0x99	0x10	Comparator 0 Control 1
CMP0MD	0x9D	0x00, 0x10	Comparator 0 Mode
CMP0MX	0x9F	0x00, 0x10	Comparator 0 Multiplexer Selection
CMP1CN0	0xBF	0x00, 0x10	Comparator 1 Control 0
CMP1CN1	0xAC	0x10	Comparator 1 Control 1
CMP1MD	0xAB	0x00, 0x10	Comparator 1 Mode
CMP1MX	0xAA	0x00, 0x10	Comparator 1 Multiplexer Selection
CRC0CN0	0xCE	0x00, 0x20	CRC0 Control 0
CRC0CN1	0x86	0x00, 0x20	CRC0 Control 1
CRC0CNT	0xD3	0x00, 0x20	CRC0 Automatic Flash Sector Count
CRC0DAT	0xDE	0x00, 0x20	CRC0 Data Output
CRC0FLIP	0xCF	0x00, 0x20	CRC0 Bit Flip
CRC0IN	0xDD	0x00, 0x20	CRC0 Data Input
CRC0ST	0xD2	0x00, 0x20	CRC0 Automatic Flash Sector Start
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

Register	Address	SFR Pages	Description
EIE2	0xCE	0x10	Extended Interrupt Enable 2
EIP1	0xF3	0x00, 0x10	Extended Interrupt Priority 1 Low
EIP1H	0xF5	0x10	Extended Interrupt Priority 1 High
EIP2	0xF4	0x10	Extended Interrupt Priority 2
EIP2H	0xF6	0x10	Extended Interrupt Priority 2 High
EMI0CN	0xE7	ALL	External Memory Interface Control
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
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

Register	Address	SFR Pages	Description
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
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
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
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
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	0x9A	0x00	Power Control 1
PFE0CN	0xC1	0x10	Prefetch Engine Control
PRTDRV	0xF6	0x00, 0x20	Port Drive Strength
PSCTL	0x8F	ALL	Program Store Control
PSW	0xD0	ALL	Program Status Word
REF0CN	0xD1	0x00, 0x10	Voltage Reference Control
REG0CN	0xC9	0x00, 0x20	Voltage Regulator 0 Control
REG1CN	0xC6	0x20	Voltage Regulator 1 Control
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

Register	Address	SFR Pages	Description
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	0xCF	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
TCON	0x88	ALL	Timer 0/1 Control
TH0	0x8C	ALL	Timer 0 High Byte
TH1	0x8D	ALL	Timer 1 High Byte
TL0	0x8A	ALL	Timer 0 Low Byte
TL1	0x8B	ALL	Timer 1 Low Byte
TMOD	0x89	ALL	Timer 0/1 Mode
TMR2CN0	0xC8	0x00, 0x10	Timer 2 Control 0
TMR2CN1	0xFD	0x10	Timer 2 Control 1
TMR2H	0xCD	0x00, 0x10	Timer 2 High Byte
TMR2L	0xCC	0x00, 0x10	Timer 2 Low Byte

Register	Address	SFR Pages	Description
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
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
USB0ADR	0xAE	ALL	USB0 Indirect Address
USB0AEC	0xB2	0x20	USB0 Advanced Energy Control
USB0CDCF	0xB6	0x20	USB0 Charger Detect Configuration
USB0CDCN	0xBE	0x20	USB0 Charger Detect Control
USB0CDSTA	0xBF	0x20	USB0 Charger Detect Status
USB0CF	0xB5	0x20	USB0 Configuration
USB0DAT	0xAF	ALL	USB0 Data
USB0XCN	0xB3	0x20	USB0 Transceiver Control
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

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: 0xCF								

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.

Bit	Name	Reset	Access	Description
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 bulk of the flash memory is organized in 512-byte pages. 1 KB of the flash is organized in 64-byte pages to simplify EEPROM emulation or other non-volatile data storage tasks. Either of the flash areas may be used to store code or non-volatile data. Flash memory may 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.

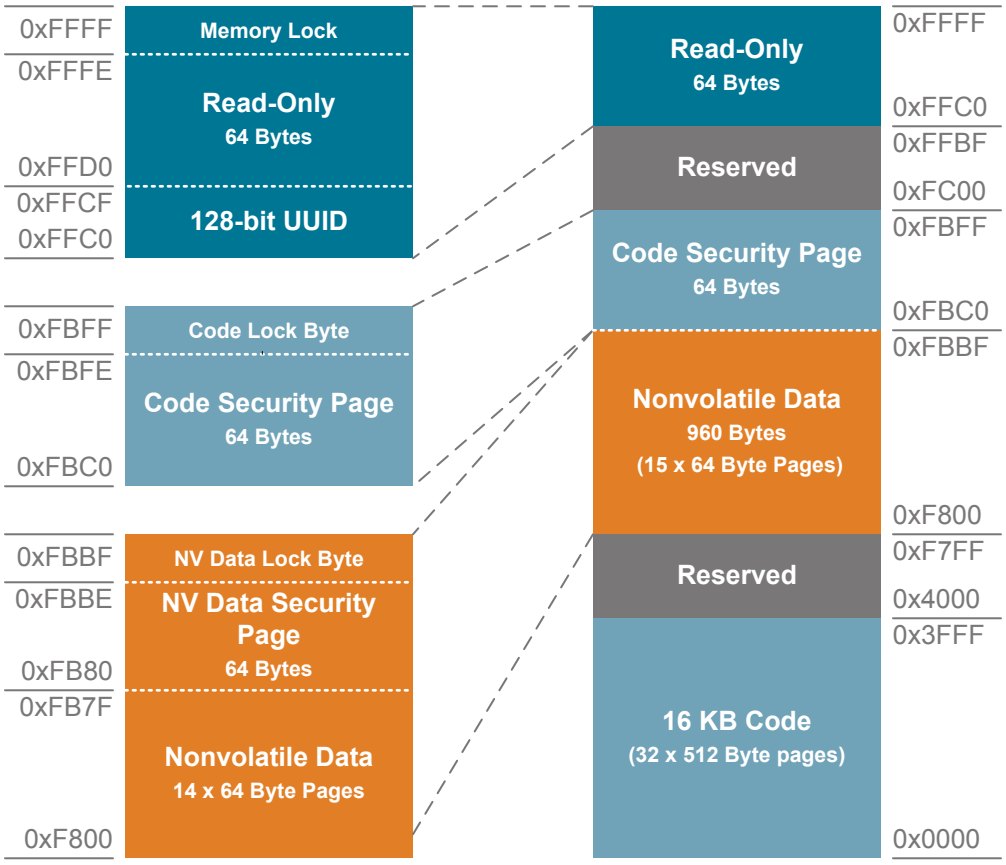


Figure 4.1. Flash Memory Map — 16 KB Devices

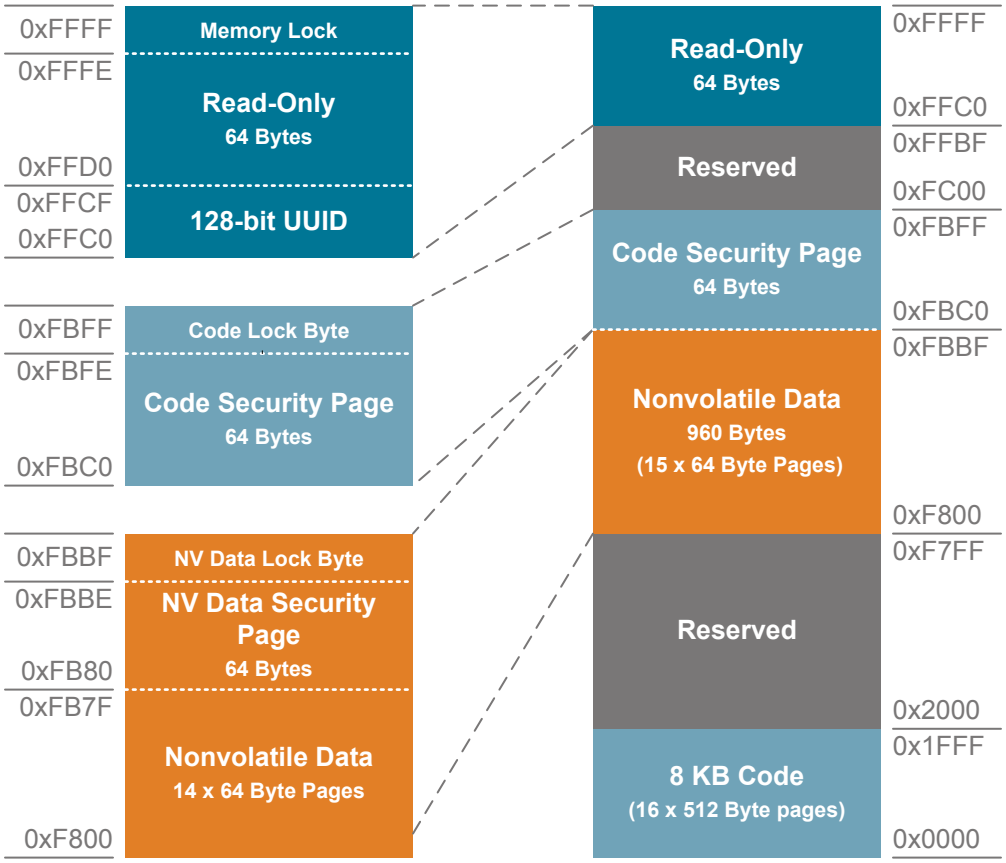


Figure 4.2. Flash Memory Map — 8 KB Devices

4.2 Features

The flash memory has the following features:

- Up to 16 KB in 512-byte sectors, and 1 KB in 64-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.

Security lock bytes located in flash user space offer individual protection for the "data flash" and "user flash" regions of flash memory. Read, write and erase access can be restricted from both unprotected code or the C2 interface. See the specific device memory map for the location of the security bytes, and the regions they protect. The user-lock security byte controls access to the "user flash" region, and allows the user to lock "n" flash pages, starting at page 0, where "n" is the 1s complement number represented by the user-lock security byte. The data-lock security byte controls access to the "data flash" region, and operates as an all-or-nothing lock. If the data-lock security byte is 0xFF, all of data area, including the page containing the lock byte, will be open. If the data-lock security bytes is a non-0xFF value, all of data area, will be locked.
Note: The page containing the user-lock security byte is unlocked when no other flash pages are locked (all bits of the user-lock security byte are 1) and locked when any other flash pages are locked (any bit of the user-lock security byte is 0).

Table 4.1. User-Lock Security Byte Decoding

User-Lock Security Lock Byte	111111101b
1s Complement	00000010b
Flash Pages Locked	3 (First two flash pages in user flash + user-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. Additional restrictions between the two regions of flash are also enforced per the following tables.

Table 4.2. Flash Security Summary - Firmware Permissions

	Permissions according to the area firmware is executing from:			
Target Area for Read / Write / Erase	**Unlocked User Page**	**Locked User Page**	**Unlocked Data Page**	**Locked Data Page**
Any Unlocked User Page	[R] [W] [E]	[R] [W] [E]	[R] [W] [E]	[R] [W] [E]
Locked User Page (except user security page)	reset	[R] [W] [E]	reset	[R] [W] [E]
Locked User Security Page	reset	[R] [W]	reset	[R] [W]
Any Unlocked Data Page	[R] [W] [E]	[R] [W] [E]	[R] [W] [E]	n/a
Locked Data Page (except data security page)	reset	[R] [W] [E]	n/a	[R] [W] [E]
Locked Data Security Page	reset	[R] [W]	n/a	[R] [W]
Read-Only Area	[R]	[R]	[R]	[R]
Reserved Area	reset	reset	reset	reset

	Permissions according to the area firmware is executing from:			
Target Area for Read / Write / Erase	**Unlocked User Page**	**Locked User Page**	**Unlocked Data Page**	**Locked Data Page**
[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 User Page	[R] [W] [E]
Any Locked User Page	Device Erase Only
Any Unlocked Data Page	[R] [W] [E]
Any Locked Data 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 crystal-based source, be advised that crystal 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.
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

A128-bit universally unique identifier (UUID) is pre-programmed into all devices. 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.

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	0x32							
SFR Page = 0x0; SFR Address: 0xB5								

Bit	Name	Reset	Access	Description
7:0	DEVICEID	0x32	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
0x41	EFM8UB10F16G_QFN28	EFM8UB10F16G-{R}-QFN28
0x43	EFM8UB10F16G_QFN20	EFM8UB10F16G-{R}-QFN20
0x45	EFM8UB11F16G_QSOP24	EFM8UB11F16G-{R}-QSOP24
0x49	EFM8UB10F8G_QFN20	EFM8UB10F8G-{R}-QFN20

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
0x02	REV_A	Revision A.
0x03	REV_B	Revision B.

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 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 interrupt-pending flag is ignored by the hardware and program execution continues as normal. The interrupt-pending flag is set to logic 1 regardless of whether the interrupt is enabled.

Each interrupt source can be individually enabled or disabled through the use of an associated interrupt enable bit in the IE and EIEn registers. However, interrupts must first be globally enabled by setting the EA bit to logic 1 before the individual interrupt enables are recognized. 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. However, most 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 return-from-interrupt (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 data sheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the 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 35. 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)
USB0 Events	0x007B	15	EIE2_EUSB0	CMIE_RSTINTE CMIE_RSUINTE CMIE_SOFE CMIE_SUSINTE IN1IE_EP0E IN1IE_IN1E IN1IE_IN2E IN1IE_IN3E OUT1IE_OUT1E OUT1IE_OUT2E OUT1IE_OUT3E	CMINT_RSTINT CMINT_RSUINT CMINT_SOF CMINT_SUSINT IN1INT_EP0 IN1INT_IN1 IN1INT_IN2 IN1INT_IN3 OUT1INT_OUT1 OUT1INT_OUT2 OUT1INT_OUT3
VBUS / USB Charge Detect	0x0083	16	EIE2_EVBUS	USB0CDCF_DCDIE USB0CDCF_PDIE USB0CDCF_SDIE USB0CF_VBUSIE	USB0CDSTA_DCDI USB0CDSTA_ERR USB0CDSTA_PDI USB0CDSTA_SDI USB0CF_VBUSI
UART1	0x008B	17	EIE2_ES1	UART1FCN0_RFRQE UART1FCN0_TFRQE UART1FCN1_RIE UART1FCN1_RXTO UART1FCN1_TIE	SCON1_RI SCON1_TI UART1FCN1_RFRQ UART1FCN1_TFRQ
I2C0 Slave	0x0093	18	EIE2_EI2C0	I2C0FCN0_RFRQE I2C0FCN0_TFRQE	I2C0STAT_I2C0INT I2C0FCN1_RFRQ I2C0FCN1_TFRQ
Timer 4 Overflow / Capture	0x009B	19	EIE2_ET4	TMR4CN0_TF4CEN TMR4CN0_TF4LEN	TMR4CN0_TF4H TMR4CN0_TF4L

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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.
2	EWADC0	0	RW	**ADC0 Window Comparison Interrupt Enable.** This bit sets the masking of ADC0 Window Comparison interrupt.

Bit	Name	Reset	Access	Description
	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 = 0x0, 0x10; SFR Address: 0xF3								

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: 0xF5								

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			ET4	EI2C0	ES1	EVBUS	EUSB0
Access	RW			RW	RW	RW	RW	RW
Reset	0x0			0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xCE								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	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.
3	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.
2	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.
1	EVBUS	0	RW	**VBUS and USB Charger Detect Interrupt.**
	This bit sets the masking of the VBUS and VBUS and USB Charger Detect interrupts.			
	Value	Name		Description
	0	DISABLED		Disable all VBUS and VBUS and USB Charger Detect interrupts.
	1	ENABLED		Enable interrupt requests generated by VBUS and VBUS and USB Charger Detect.
0	EUSB0	0	RW	**USB (USB0) Interrupt Enable.**
	This bit sets the masking of the USB0 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all USB0 interrupts.
	1	ENABLED		Enable interrupt requests generated by USB0.

6.3.8 EIP2: Extended Interrupt Priority 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved			PT4	PI2C0	PS1	PVBUS	PUSB0
Access	RW			RW	RW	RW	RW	RW
Reset	0x0			0	0	0	0	0
SFR Page = 0x10; SFR Address: 0xF4								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	PT4	0	RW	**Timer 4 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 4 interrupt.
3	PI2C0	0	RW	**I2C0 Slave Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the I2C0 Slave interrupt.
2	PS1	0	RW	**UART1 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the UART1 interrupt.
1	PVBUS	0	RW	**VBUS and USB Charger Detect Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the VBUS and USB Charger Detect interrupt.
0	PUSB0	0	RW	**USB (USB0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for USB0 interrupts.

6.3.9 EIP2H: Extended Interrupt Priority 2 High

Bit	7	6	5	4	3	2	1	0
Name	Reserved			PHT4	PHI2C0	PHS1	PHVBUS	PHUSB0
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	PHT4	0	RW	**Timer 4 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 4 interrupt.
3	PHI2C0	0	RW	**I2C0 Slave Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the I2C0 Slave interrupt.
2	PHS1	0	RW	**UART1 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the UART1 interrupt.
1	PHVBUS	0	RW	**VBUS and USB Charger Detect Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the VBUS and USB Charger Detect interrupt.
0	PHUSB0	0	RW	**USB (USB0) Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for USB0 interrupts.

7. Power Management and Internal Regulators

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.

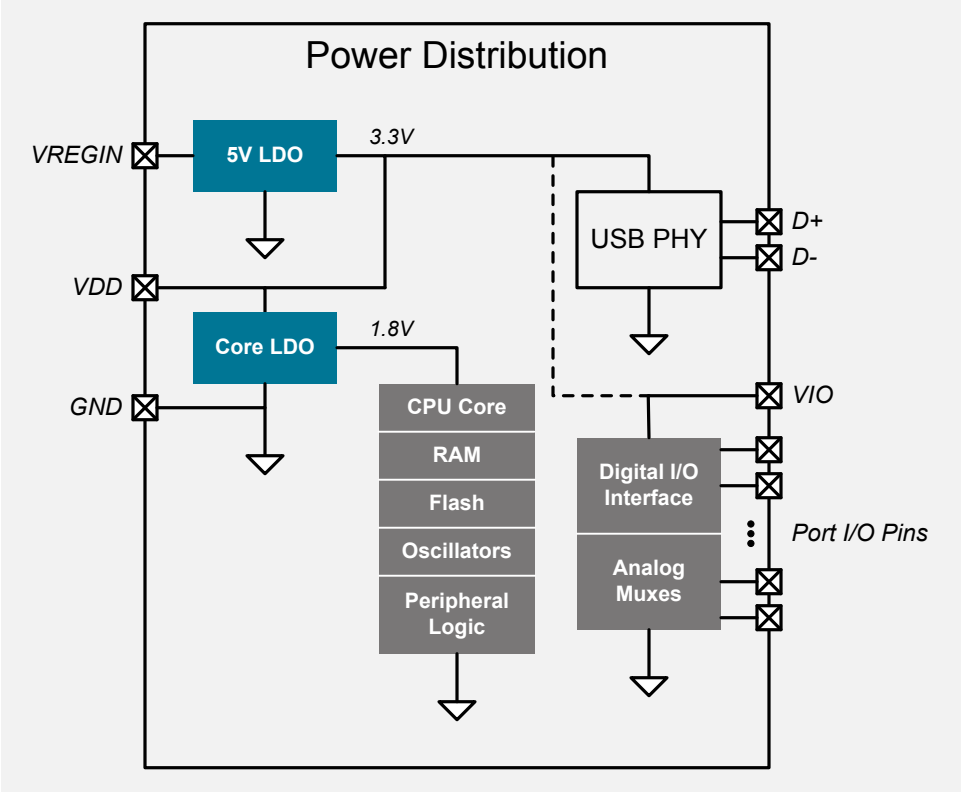


Figure 7.1. Power System Block Diagram

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 • Regulators 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	• USB0 Bus Activity • Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Rising Edge

Power Mode	Details	Mode Entry	Wake-Up Sources
Snooze	• Core and peripheral clocks halted • HFOSC0 and HFOSC1 oscillators stopped • Regulators 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	• USB0 Bus Activity • Timer 4 Event • SPI0 Activity • I2C0 Slave Activity • Port Match Event • Comparator 0 Rising Edge
Shutdown	• All internal power nets shut down • 5V regulator remains active (if enabled) • 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:

- 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.
- 5V-to-3.3V Regulator:
  - Allows direct connection to USB supply net.
  - Provides up to 100 mA for system-level use.
  - Low power consumption in Snooze 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 the 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 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.

7.5 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 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. 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.

7.6 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, 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. Coming out of Shutdown mode, whether by POR or pin reset, will appear as a power-on reset of the device.

7.8 5V-to-3.3V Regulator

The 5-to-3.3 V regulator is powered from the VREGIN pin on the device. When active, it regulates the input voltage to 3.3 V at the VDD pin, providing up to 100 mA for the device and system. In addition to the normal mode of operation, the regulator has two low power modes which may be used to reduce the supply current, and may be disabled when not in use.

Table 7.2. Voltage Regulator Operational Modes

Regulator Condition	SUSEN Bit	BIASENB Bit	REG1ENB Bit	Relative Power Consumption
Normal	0	0	0	highest
Suspend	1	0	0	low
Bias Disabled	x	1	0	extremely low
Disabled	x	1	1	off

The voltage regulator is enabled in normal mode by default. Normal mode offers the fastest response times, for systems with dynamically-changing loads.

For applications which can tolerate a lower regulator bandwidth but still require a tightly regulated output voltage, the regulator may be placed in suspend mode. Suspend mode is activated when firmware sets the SUSEN bit. Suspend mode reduces the regulator bias current at the expense of bandwidth.

For low power applications that can tolerate reduced output voltage accuracy and load regulation, the internal bias current may be disabled completely using the BIASENB bit. If firmware sets the BIASENB bit, the regulator will regulate the voltage using a method that is more susceptible to process and temperature variations. In addition, the actual output voltage may drop substantially under heavy loads. The bias should only be disabled for light loads (5 mA or less) or when the voltage regulator is disabled.

If the regulator is not used in a system, the VREGIN and VDD pins should be connected together. Firmware may disable the regulator by writing both the REG1ENB and BIASENB bits in REG1CN to turn off the regulator and all associated bias currents.

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					
Access	RW	RW	R					
Reset	0	0	0x00					
SFR Page = 0x0; SFR Address: 0x9A								

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.
5:0	*Reserved*	*Must write reset value.*		

7.9.3 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 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. Only the RSTb pin or power cycle can reset the device.

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

7.9.4 REG1CN: Voltage Regulator 1 Control

Bit	7	6	5	4	3	2	1	0
Name	REG1ENB	Reserved				BIASENB	SUSEN	Reserved
Access	RW	R				RW	RW	R
Reset	0	0x0				0	0	0
SFR Page = 0x20; SFR Address: 0xC6								

Bit	Name	Reset	Access	Description
7	REG1ENB	0	RW	**Voltage Regulator 1 Disable.** This bit may be used to disable the 5V regulator if an external regulator is used to power VDD. VREGIN should be tied to VDD in any system that disables this regulator.
6:3	*Reserved*	*Must write reset value.*		
2	BIASENB	0	RW	**Regulator Bias Disable.** The BIASENB bit disables the regulator bias voltage when set to 1.

Value	Name	Description
0	ENABLED	Regulator bias is enabled.
1	DISABLED	Regulator bias is disabled.

Bit	Name	Reset	Access	Description
1	SUSEN	0	RW	**Voltage Regulator 1 Suspend Enable.** When set to 1, this bit places the 5V regulator into suspend mode.

Value	Name	Description
0	NORMAL	The 5V regulator is in normal power mode. Normal mode is the highest performance mode for the regulator.
1	SUSPEND	The 5V regulator is in suspend power mode. Suspend mode reduces the regulator bias current, but increases the response times.

Bit	Name	Reset	Access	Description
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.

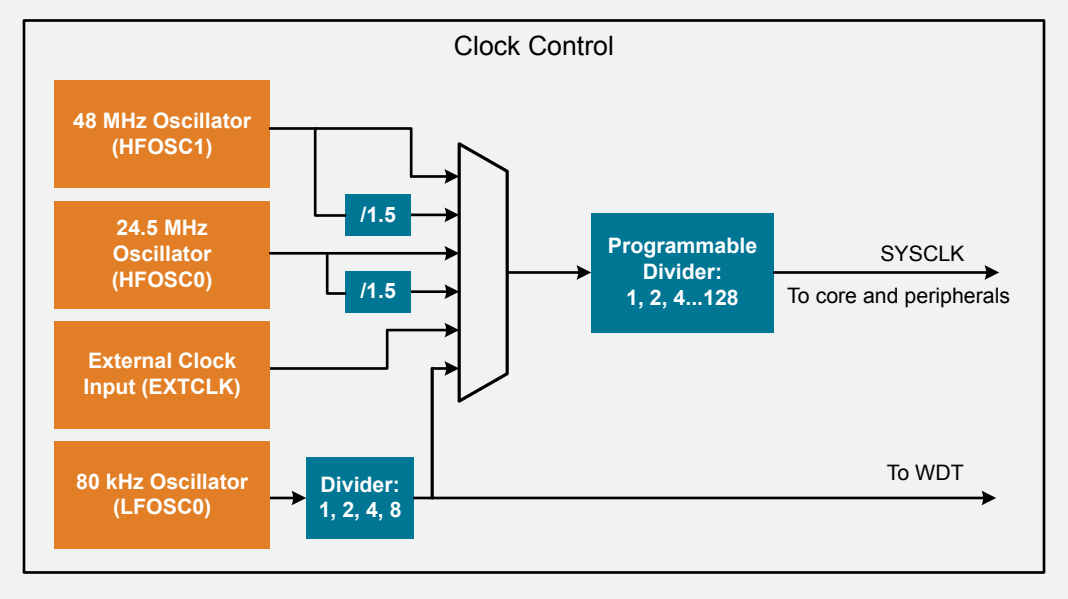


Figure 8.1. Clock Control Block Diagram

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.
- 48 MHz internal oscillator (HFOSC1), accurate to ±1.5% over supply and temperature corners.
- 80 kHz low-frequency oscillator (LFOSC0).
- External CMOS clock input (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.

8.3.3 HFOSC1 48 MHz Internal Oscillator

HFOSC1 is a programmable internal high-frequency oscillator that is factory-calibrated to 48 MHz. The oscillator is automatically enabled when it is requested. The oscillator period can be adjusted via the HFO1CAL register to obtain other frequencies.

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 rising 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 Clock

An external CMOS clock source is also supported as a core clock source. The EXTCLK pin on the device serves as the external clock input when running in this mode. The EXTCLK input may also be used to clock certain digital peripherals (e.g., Timers, PCA, etc.) while SYSCLK runs from one of the internal oscillator sources. 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.

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 proagated 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).
	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).
	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.				

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. The Port I/O latches are reset to 1 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. 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.

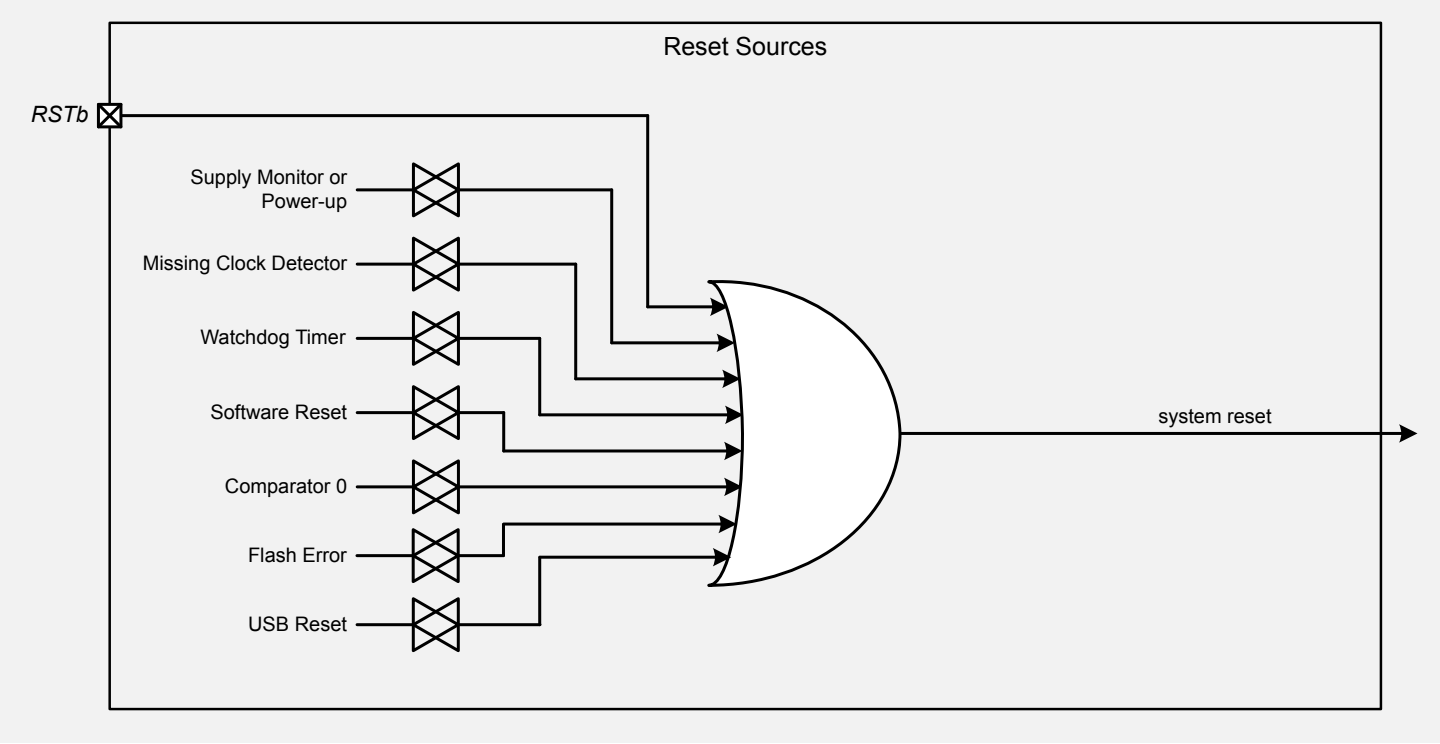


Figure 9.1. Reset Sources Block Diagram

9.2 Features

Reset sources on the device include:

- 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
- USB 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.

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_{RST}. 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 increases (supply ramp time is defined as how fast the supply pin ramps from 0 V to V_{RST}). For ramp times less than 1 ms, the power-on reset time (T_{POR}) is typically less than 0.3 ms. Additionally, the power supply must reach V_{RST} 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.

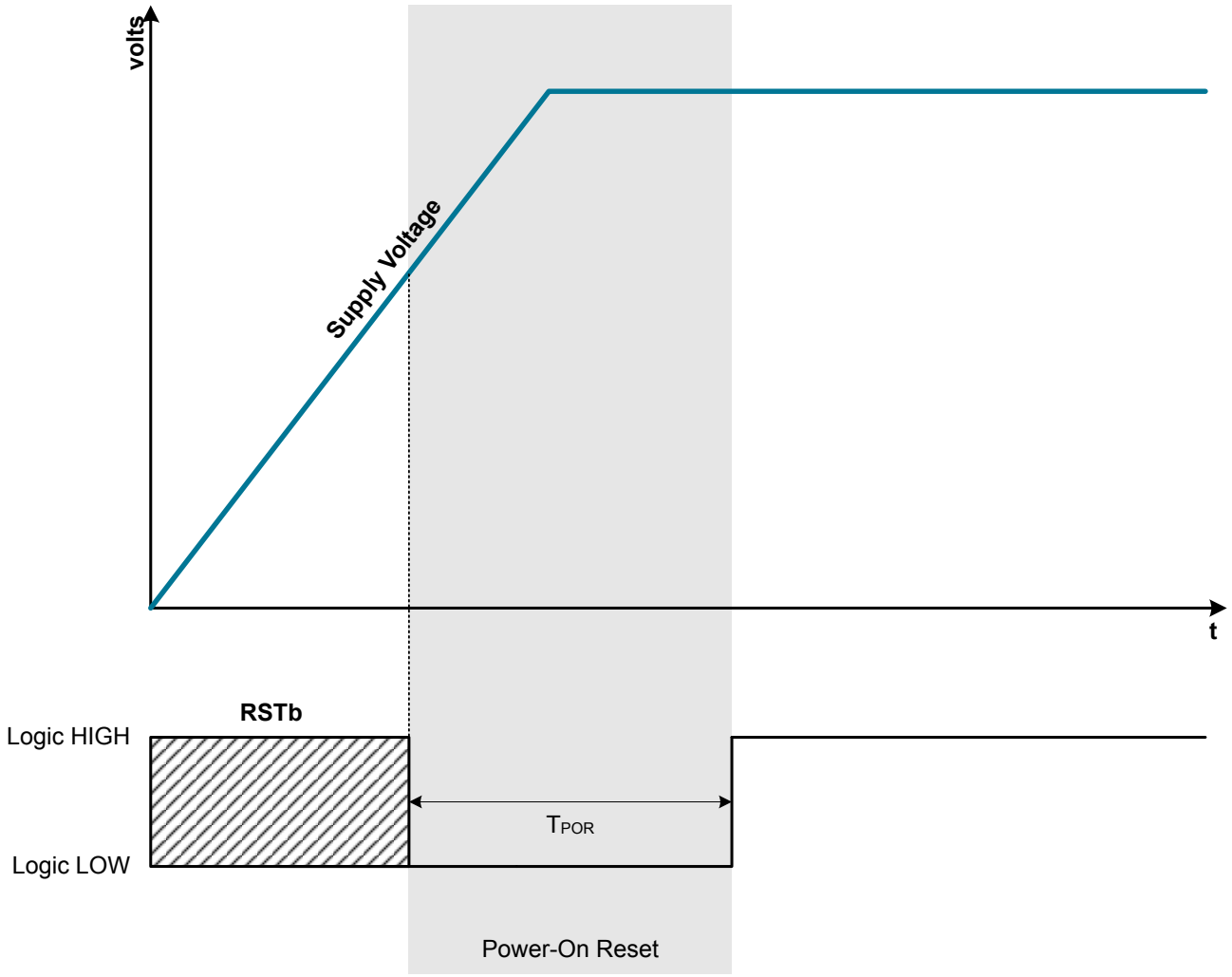


Figure 9.2. Power-On Reset Timing

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.

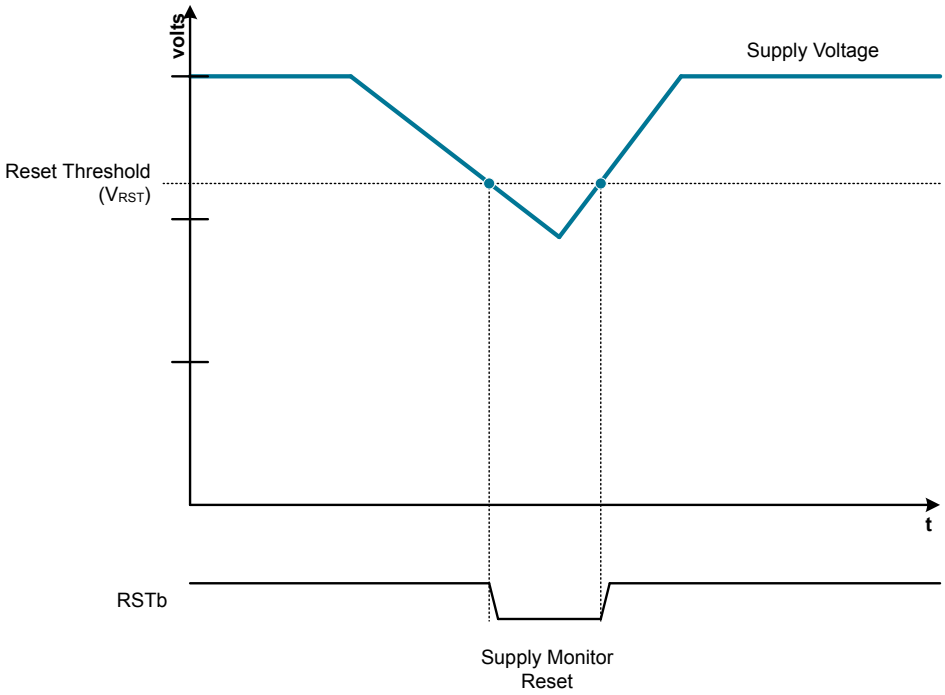


Figure 9.3. Reset Sources

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.

9.3.10 USB Reset

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

- RESET signaling is detected on the USB network. The USB Function Controller (USB0) must be enabled for RESET signaling to be detected.
- A falling or rising voltage on the VBUS pin.

The USBRSF bit will read 1 following a USB 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	USBRSF	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	USBRSF	Varies	RW	**USB Reset Enable and Flag.** Read: This bit reads 1 if USB caused the last reset. Write: Writing a 1 to this bit enables the USB0 module as a reset source.
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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

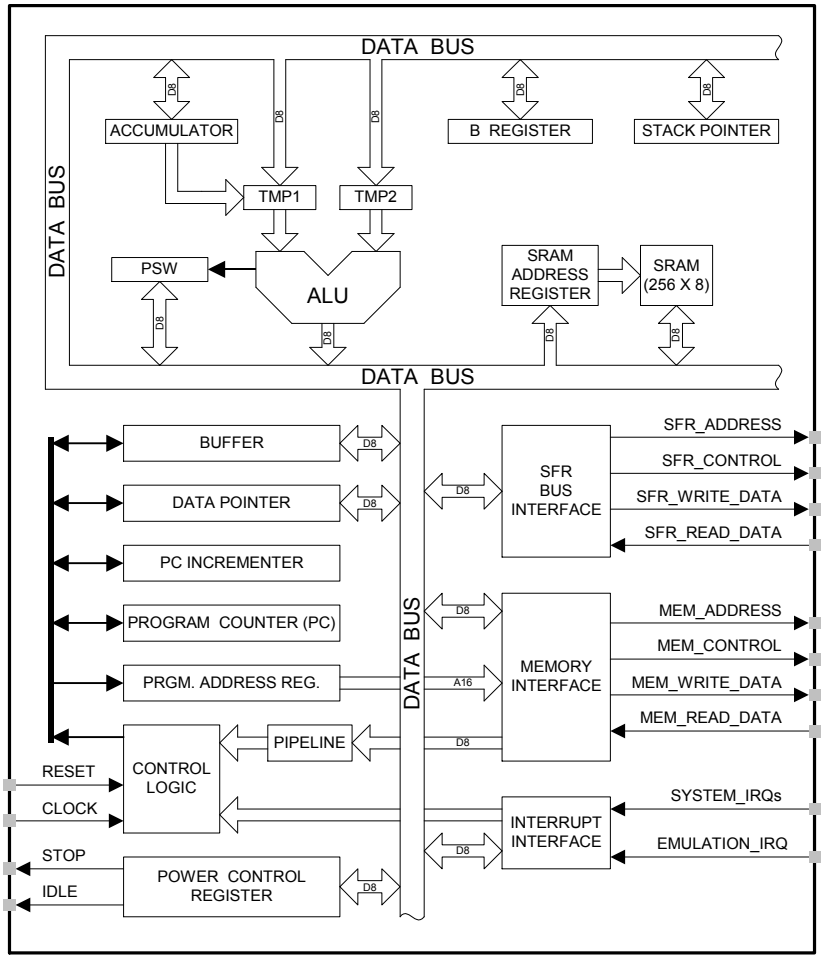


Figure 10.1. CIP-51 Block Diagram

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 2-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 20 ns, the prefetch engine is necessary for full-speed code execution. Instructions are read from flash memory two bytes at a time 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 two clock cycles while the next set of code bytes is retrieved from flash memory.

The PFE0CN register controls the behavior of the prefetch engine. When operating at speeds greater than 25 MHz, the prefetch engine must be enabled. To enable the prefetch engine, both the FLRT and PFEN bit should be set to 1.

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	
			prefetch off	prefetch on
Arithmetic Operations				
ADD A, Rn	Add register to A	1	1	1
ADD A, direct	Add direct byte to A	2	2	2
ADD A, @Ri	Add indirect RAM to A	1	2	2
ADD A, #data	Add immediate to A	2	2	2
ADDC A, Rn	Add register to A with carry	1	1	1
ADDC A, direct	Add direct byte to A with carry	2	2	2
ADDC A, @Ri	Add indirect RAM to A with carry	1	2	2
ADDC A, #data	Add immediate to A with carry	2	2	2
SUBB A, Rn	Subtract register from A with borrow	1	1	1
SUBB A, direct	Subtract direct byte from A with borrow	2	2	2
SUBB A, @Ri	Subtract indirect RAM from A with borrow	1	2	2
SUBB A, #data	Subtract immediate from A with borrow	2	2	2
INC A	Increment A	1	1	1
INC Rn	Increment register	1	1	1
INC direct	Increment direct byte	2	2	2
INC @Ri	Increment indirect RAM	1	2	2
DEC A	Decrement A	1	1	1
DEC Rn	Decrement register	1	1	1
DEC direct	Decrement direct byte	2	2	2
DEC @Ri	Decrement indirect RAM	1	2	2
INC DPTR	Increment Data Pointer	1	1	1
MUL AB	Multiply A and B	1	4	4
DIV AB	Divide A by B	1	8	8
DA A	Decimal adjust A	1	1	1
Logical Operations				
ANL A, Rn	AND Register to A	1	1	1

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

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch off	prefetch on
MOV direct, #data	Move immediate to direct byte	3	3	3
MOV @Ri, A	Move A to indirect RAM	1	2	2
MOV @Ri, direct	Move direct byte to indirect RAM	2	2	2
MOV @Ri, #data	Move immediate to indirect RAM	2	2	2
MOV DPTR, #data16	Load DPTR with 16-bit constant	3	3	3
MOVC A, @A+DPTR	Move code byte relative DPTR to A	1	3	6
MOVC A, @A+PC	Move code byte relative PC to A	1	3	3
MOVX A, @Ri	Move external data (8-bit address) to A	1	3	3
MOVX @Ri, A	Move A to external data (8-bit address)	1	3	3
MOVX A, @DPTR	Move external data (16-bit address) to A	1	3	3
MOVX @DPTR, A	Move A to external data (16-bit address)	1	3	3
PUSH direct	Push direct byte onto stack	2	2	2
POP direct	Pop direct byte from stack	2	2	2
XCH A, Rn	Exchange Register with A	1	1	1
XCH A, direct	Exchange direct byte with A	2	2	2
XCH A, @Ri	Exchange indirect RAM with A	1	2	2
XCHD A, @Ri	Exchange low nibble of indirect RAM with A	1	2	2
Boolean Manipulation				
CLR C	Clear Carry	1	1	1
CLR bit	Clear direct bit	2	2	2
SETB C	Set Carry	1	1	2
SETB bit	Set direct bit	2	2	2
CPL C	Complement Carry	1	1	1
CPL bit	Complement direct bit	2	2	2
ANL C, bit	AND direct bit to Carry	2	2	2
ANL C, /bit	AND complement of direct bit to Carry	2	2	2
ORL C, bit	OR direct bit to carry	2	2	2
ORL C, /bit	OR complement of direct bit to Carry	2	2	2
MOV C, bit	Move direct bit to Carry	2	2	2
MOV bit, C	Move Carry to direct bit	2	2	2
JC rel	Jump if Carry is set	2	2 or 3	2 or 6
JNC rel	Jump if Carry is not set	2	2 or 3	2 or 5
JB bit, rel	Jump if direct bit is set	3	3 or 4	3 or 7
JNB bit, rel	Jump if direct bit is not set	3	3 or 4	3 or 6
JBC bit, rel	Jump if direct bit is set and clear bit	3	3 or 4	3 or 7
Program Branching				

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch off	prefetch on
ACALL addr11	Absolute subroutine call	2	3	6
LCALL addr16	Long subroutine call	3	4	7
RET	Return from subroutine	1	5	8
RETI	Return from interrupt	1	5	7
AJMP addr11	Absolute jump	2	3	6
LJMP addr16	Long jump	3	4	6
SJMP rel	Short jump (relative address)	2	3	6
JMP @A+DPTR	Jump indirect relative to DPTR	1	3	5
JZ rel	Jump if A equals zero	2	2 or 3	2 or 5
JNZ rel	Jump if A does not equal zero	2	2 or 3	2 or 5
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	4 or 5	4 or 7
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3 or 4	3 or 6
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3 or 4	3 or 6
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4 or 5	4 or 7
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2 or 3	2 or 5
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3 or 4	3 or 7
NOP	No operation	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		PFEN	FLRT	Reserved			
Access	R		RW	RW	R			
Reset	0x0		0	0	0x0			
SFR Page = 0x10; SFR Address: 0xC1								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	PFEN	0	RW	**Prefetch Enable.** The prefetch engine should be disabled when the device is in suspend mode to save power.

Value	Name	Description
0	DISABLED	Disable the prefetch engine (SYSCLK < 25 MHz).
1	ENABLED	Enable the prefetch engine (SYSCLK > 25 MHz).

Bit	Name	Reset	Access	Description
4	FLRT	0	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
0	SYSCLK_BELOW_25_MHZ	SYSCLK < 25 MHz.
1	SYSCLK_BELOW_50_MHZ	SYSCLK < 50 MHz.

Bit	Name	Reset	Access	Description
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 P3.0 and P3.1 can be used as GPIO. Additionally, the C2 Interface Data signal (C2D) is shared with P3.0.

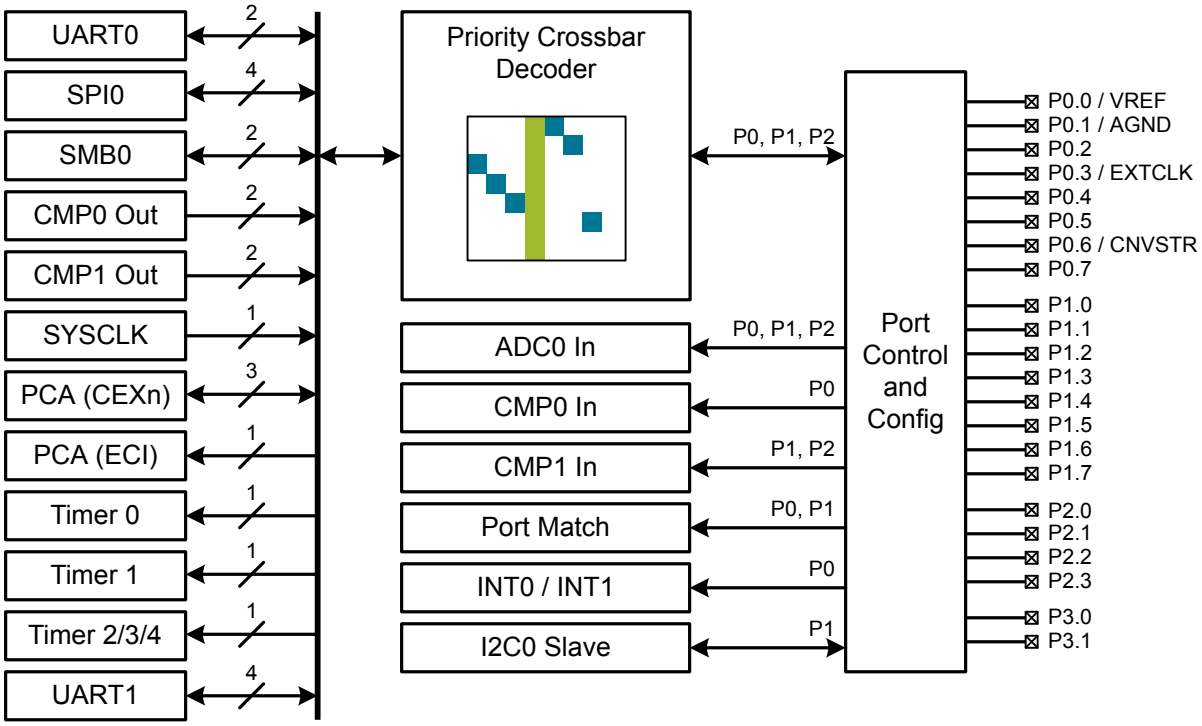


Figure 11.1. Port I/O Block Diagram

11.2 Features

The port control block offers the following features:

- Up to 22 multi-functions I/O pins, supporting digital and analog functions.
- Flexible priority crossbar decoder for digital peripheral assignment.
- Two drive strength settings for each port.
- Two direct-pin interrupt sources with dedicated interrupt vectors (INT0 and INT1).
- Up to 20 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.

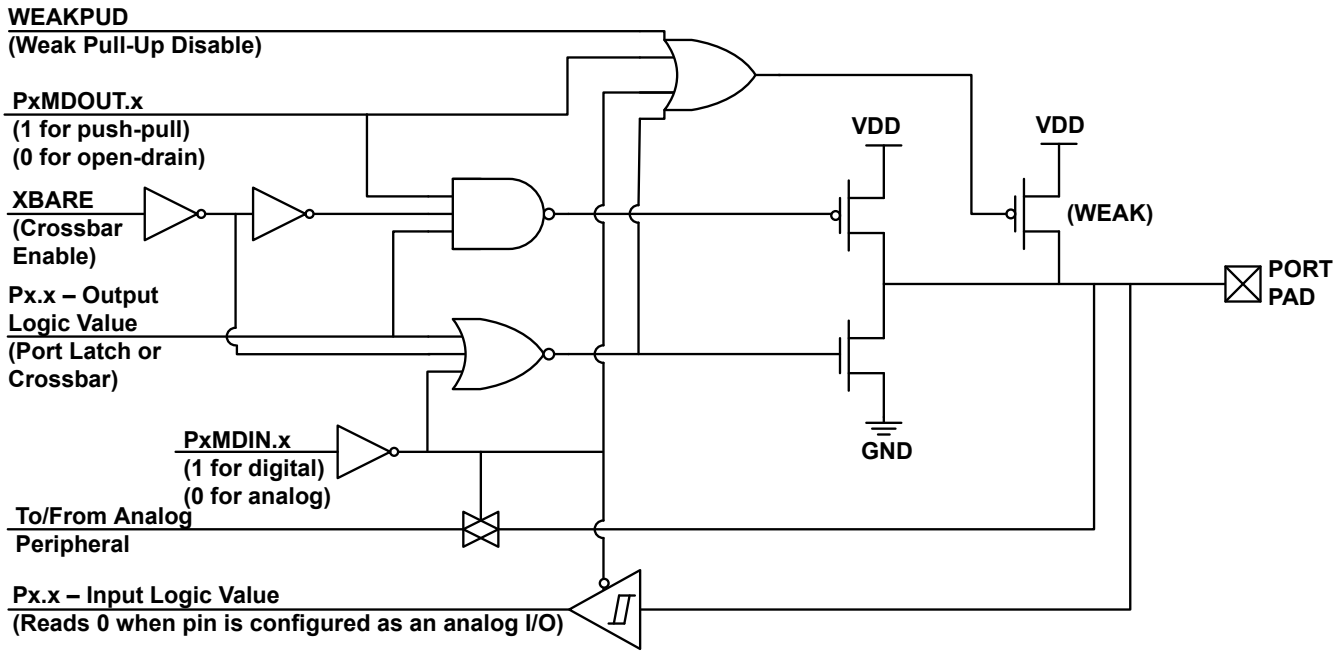


Figure 11.2. Port I/O Cell Block Diagram

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. lear 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	SFR(s) Used For Assignment
ADC Input	P0.0 – P2.3 D+, D- (USB)	ADC0MX, PnSKIP, PnMDIN
Comparator 0 Input	P0.0 – P1.2 (QFN28) P0.0 – P0.7 (QSOP24, QFN20)	CMP0MX, PnSKIP, PnMDIN
Comparator 1 Input	P1.0 – P2.3 (QFN28) P0.6 – P2.3 (QSOP24, QFN20)	CMP1MX, PnSKIP, PnMDIN
Voltage Reference (VREF)	P0.0	REF0CN, PnSKIP, PnMDIN
Reference Ground (AGND)	P0.1	REF0CN, PnSKIP, PnMDIN

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 (CEX0-2 and ECI), T0, T1, T2/3/4	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	P1.1 – P1.2 (QFN20, QSOP24) P1.5 – P1.6 (QFN28)	I2C0CN0
External Interrupt 0, External Interrupt 1	P0.0 – P0.7	IT01CF
Conversion Start (CNVSTR)	P0.6	ADC0CN0
External Clock Input (EXTCLK)	P0.3	CLKSEL
Port Match	P0.0 – P2.3	P0MASK, P0MAT, P1MASK, P1MAT, P2MASK, P2MAT
VBUS	P3.1	USB0CF
Any pin used for GPIO	P0.0 – P3.1	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 83 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.

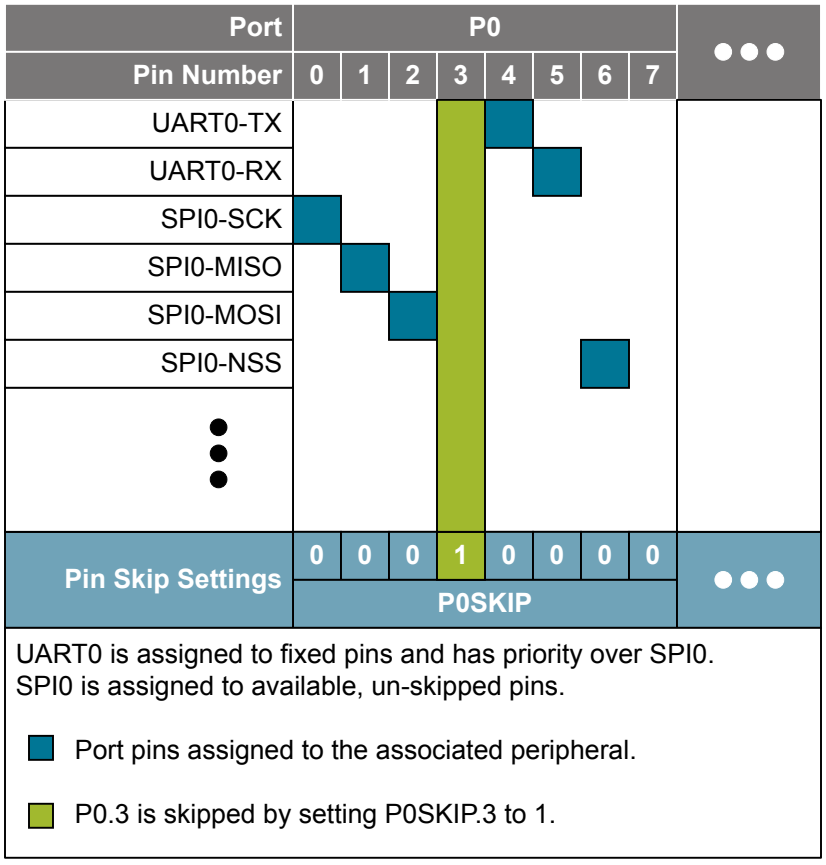
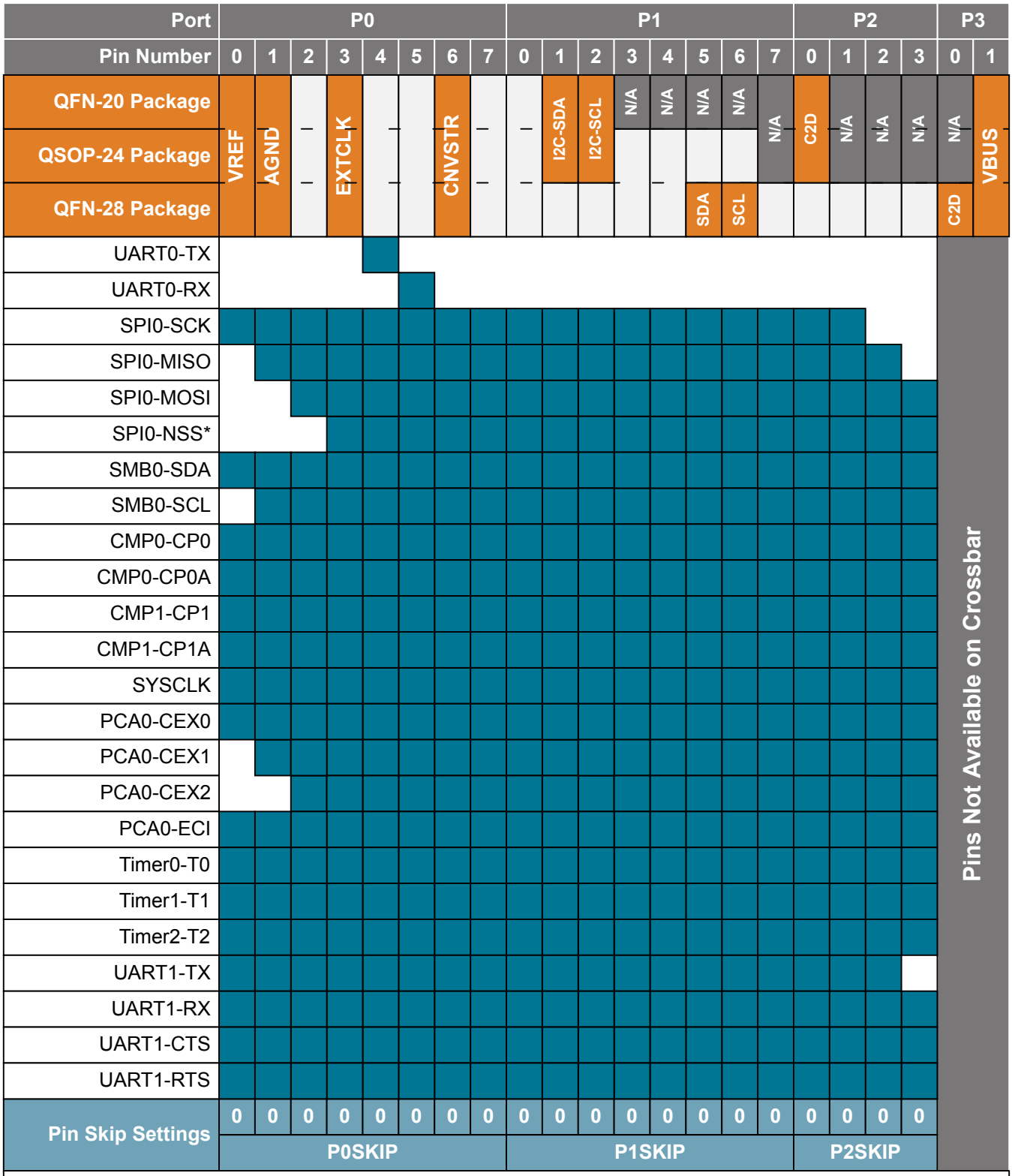


Figure 11.3. Crossbar Priority Decoder Example Assignments

11.3.3.1 Crossbar Functional Map

Figure 11.4 Full Crossbar Map on page 85 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.

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.

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	0x0		0	0	0	0	0x0	
SFR Page = 0x0, 0x20; SFR Address: 0xE2								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	T2E	0	RW	**T2 Enable.**
	Value	Name		Description
	0	DISABLED		T2 unavailable at Port pin.
	1	ENABLED		T2 routed to Port pin.
4	T1E	0	RW	**T1 Enable.**
	Value	Name		Description
	0	DISABLED		T1 unavailable at Port pin.
	1	ENABLED		T1 routed to Port pin.
3	T0E	0	RW	**T0 Enable.**
	Value	Name		Description
	0	DISABLED		T0 unavailable at Port pin.
	1	ENABLED		T0 routed to Port pin.
2	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.
1:0	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_CEX1		CEX0, CEX1 routed to Port pins.
	0x3	CEX0_CEX1_CEX2		CEX0, CEX1, CEX2 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	1	RW	**Port 0 Bit 6 Match Value.**
	See bit 7 description			
5	B5	1	RW	**Port 0 Bit 5 Match Value.**
	See bit 7 description			
4	B4	1	RW	**Port 0 Bit 4 Match Value.**
	See bit 7 description			
3	B3	1	RW	**Port 0 Bit 3 Match Value.**
	See bit 7 description			
2	B2	1	RW	**Port 0 Bit 2 Match Value.**
	See bit 7 description			
1	B1	1	RW	**Port 0 Bit 1 Match Value.**
	See bit 7 description			
0	B0	1	RW	**Port 0 Bit 0 Match Value.**
	See bit 7 description			

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	1	RW	**Port 0 Bit 6 Latch.**
	See bit 7 description			
5	B5	1	RW	**Port 0 Bit 5 Latch.**
	See bit 7 description			
4	B4	1	RW	**Port 0 Bit 4 Latch.**
	See bit 7 description			
3	B3	1	RW	**Port 0 Bit 3 Latch.**
	See bit 7 description			
2	B2	1	RW	**Port 0 Bit 2 Latch.**
	See bit 7 description			
1	B1	1	RW	**Port 0 Bit 1 Latch.**
	See bit 7 description			
0	B0	1	RW	**Port 0 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.

11.4.8 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	1	RW	**Port 0 Bit 6 Input Mode.**
	See bit 7 description			
5	B5	1	RW	**Port 0 Bit 5 Input Mode.**
	See bit 7 description			
4	B4	1	RW	**Port 0 Bit 4 Input Mode.**
	See bit 7 description			
3	B3	1	RW	**Port 0 Bit 3 Input Mode.**
	See bit 7 description			
2	B2	1	RW	**Port 0 Bit 2 Input Mode.**
	See bit 7 description			
1	B1	1	RW	**Port 0 Bit 1 Input Mode.**
	See bit 7 description			
0	B0	1	RW	**Port 0 Bit 0 Input Mode.**
	See bit 7 description			
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.9 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	1	RW	**Port 1 Bit 6 Match Value.**
	See bit 7 description			
5	B5	1	RW	**Port 1 Bit 5 Match Value.**
	See bit 7 description			
4	B4	1	RW	**Port 1 Bit 4 Match Value.**
	See bit 7 description			
3	B3	1	RW	**Port 1 Bit 3 Match Value.**
	See bit 7 description			
2	B2	1	RW	**Port 1 Bit 2 Match Value.**
	See bit 7 description			
1	B1	1	RW	**Port 1 Bit 1 Match Value.**
	See bit 7 description			
0	B0	1	RW	**Port 1 Bit 0 Match Value.**
	See bit 7 description			

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	1	RW	**Port 1 Bit 6 Latch.**
	See bit 7 description			
5	B5	1	RW	**Port 1 Bit 5 Latch.**
	See bit 7 description			
4	B4	1	RW	**Port 1 Bit 4 Latch.**
	See bit 7 description			
3	B3	1	RW	**Port 1 Bit 3 Latch.**
	See bit 7 description			
2	B2	1	RW	**Port 1 Bit 2 Latch.**
	See bit 7 description			
1	B1	1	RW	**Port 1 Bit 1 Latch.**
	See bit 7 description			
0	B0	1	RW	**Port 1 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.

11.4.14 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	1	RW	**Port 1 Bit 6 Input Mode.**
	See bit 7 description			
5	B5	1	RW	**Port 1 Bit 5 Input Mode.**
	See bit 7 description			
4	B4	1	RW	**Port 1 Bit 4 Input Mode.**
	See bit 7 description			
3	B3	1	RW	**Port 1 Bit 3 Input Mode.**
	See bit 7 description			
2	B2	1	RW	**Port 1 Bit 2 Input Mode.**
	See bit 7 description			
1	B1	1	RW	**Port 1 Bit 1 Input Mode.**
	See bit 7 description			
0	B0	1	RW	**Port 1 Bit 0 Input Mode.**
	See bit 7 description			
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.15 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	0	RW	**Port 1 Bit 6 Skip.**
	See bit 7 description			
5	B5	0	RW	**Port 1 Bit 5 Skip.**
	See bit 7 description			
4	B4	0	RW	**Port 1 Bit 4 Skip.**
	See bit 7 description			
3	B3	0	RW	**Port 1 Bit 3 Skip.**
	See bit 7 description			
2	B2	0	RW	**Port 1 Bit 2 Skip.**
	See bit 7 description			
1	B1	0	RW	**Port 1 Bit 1 Skip.**
	See bit 7 description			
0	B0	0	RW	**Port 1 Bit 0 Skip.**
	See bit 7 description			

11.4.17 P2MASK: Port 2 Mask

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: 0xFC								

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

11.4.18 P2MAT: Port 2 Match

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

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

11.4.19 P2: Port 2 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	Reserved				B3	B2	B1	B0
Access	R				RW	RW	RW	RW
Reset	0x0				1	1	1	1
SFR Page = ALL; SFR Address: 0xA0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	B3	1	RW	**Port 2 Bit 3 Latch.**
	Value	Name		Description
	0	LOW		P2.3 is low. Set P2.3 to drive low.
	1	HIGH		P2.3 is high. Set P2.3 to drive or float high.
2	B2	1	RW	**Port 2 Bit 2 Latch.**
	See bit 3 description			
1	B1	1	RW	**Port 2 Bit 1 Latch.**
	See bit 3 description			
0	B0	1	RW	**Port 2 Bit 0 Latch.**
	See bit 3 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				B3	B2	B1	B0
Access	R				RW	RW	RW	RW
Reset	0x0				1	1	1	1
SFR Page = 0x20; SFR Address: 0xF3								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	B3	1	RW	**Port 2 Bit 3 Input Mode.**
	Value	Name		Description
	0	ANALOG		P2.3 pin is configured for analog mode.
	1	DIGITAL		P2.3 pin is configured for digital mode.
2	B2	1	RW	**Port 2 Bit 2 Input Mode.**
	See bit 3 description			
1	B1	1	RW	**Port 2 Bit 1 Input Mode.**
	See bit 3 description			
0	B0	1	RW	**Port 2 Bit 0 Input Mode.**
	See bit 3 description			
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				B3	B2	B1	B0
Access	R				RW	RW	RW	RW
Reset	0x0				0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xA6								

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

11.4.23 P3: Port 3 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	Reserved						B1	B0
Access	R						RW	RW
Reset	0x00						1	1
SFR Page = ALL; SFR Address: 0xB0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1	B1	1	RW	**Port 3 Bit 1 Latch.**
	Value	Name		Description
	0	LOW		P3.1 is low. Set P3.1 to drive low.
	1	HIGH		P3.1 is high. Set P3.1 to drive or float high.
0	B0	1	RW	**Port 3 Bit 0 Latch.**
	See bit 1 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	Reserved						B1	B0
Access	R						RW	RW
Reset	0x00						1	1
SFR Page = 0x20; SFR Address: 0xF4								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1	B1	1	RW	**Port 3 Bit 1 Input Mode.**
	Value	Name		Description
	0	ANALOG		P3.1 pin is configured for analog mode.
	1	DIGITAL		P3.1 pin is configured for digital mode.
0	B0	1	RW	**Port 3 Bit 0 Input Mode.**
	See bit 1 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	Reserved						B1	B0
Access	R						RW	RW
Reset	0x00						0	0
SFR Page = 0x20; SFR Address: 0x9C								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1	B1	0	RW	**Port 3 Bit 1 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P3.1 output is open-drain.
	1	PUSH_PULL		P3.1 output is push-pull.
0	B0	0	RW	**Port 3 Bit 0 Output Mode.**
	See bit 1 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-, 10-, and 8-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.

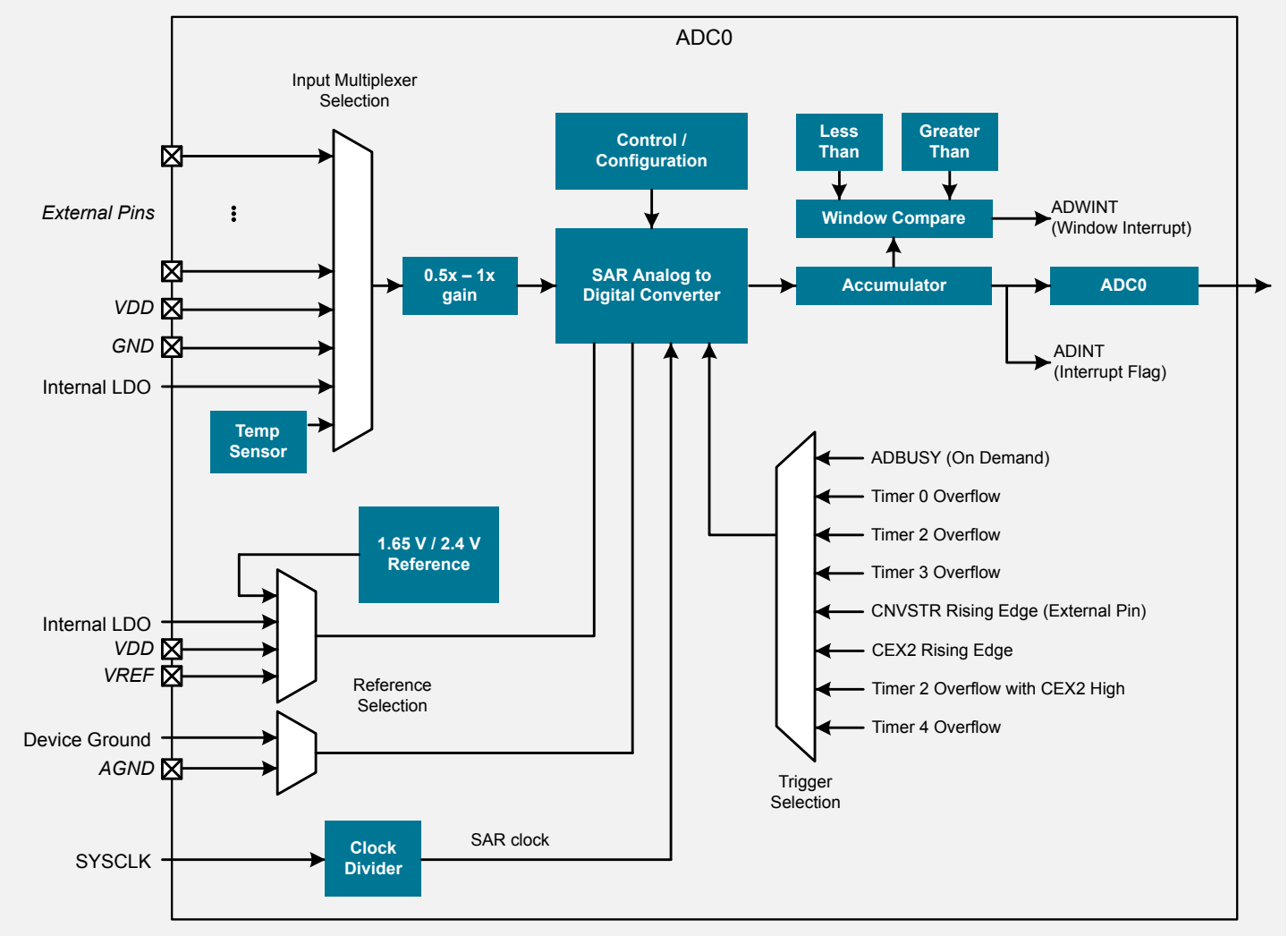


Figure 12.1. ADC Block Diagram

12.2 Features

- Up to 20 external inputs.
- Single-ended 12-bit and 10-bit modes.
- Supports an output update rate of 200 ksps samples per second in 12-bit mode or 800 ksps samples per second in 10-bit mode.
- Operation in low power modes at lower conversion speeds.
- Asynchronous hardware conversion trigger, selectable between software, external I/O and internal timer sources.
- Output data window comparator allows automatic range checking.
- Support for burst mode, which produces one set of accumulated data per conversion-start trigger with programmable power-on settling and tracking time.
- Conversion complete and window compare interrupts supported.
- Flexible output data formatting.
- Includes an internal fast-settling reference with two levels (1.65 V and 2.4 V) and support for external reference and signal ground.
- Integrated temperature sensor.

12.3 Functional Description

12.3.1 Clocking

The ADC is clocked by an adjustable conversion clock (SARCLK). SARCLK is a divided version of the selected system clock when burst mode is disabled (ADBMEN = 0), or a divided version of the HFOSC0 oscillator when burst mode is enabled (ADBMEN = 1). The clock divide value is determined by the AD0SC field. In most applications, SARCLK should be adjusted to operate as fast as possible, without exceeding the maximum electrical specifications. The SARCLK does not directly determine sampling times or sampling rates.

12.3.2 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.

12.3.2.1 Internal Voltage Reference

The high-speed internal reference offers two programmable voltage levels, and 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 can be set to one of two voltage values: 1.65 V or 2.4 V, depending on the value of the IREFLVL bit. The electrical specifications tables detail SAR clock and throughput limitations for each reference source.

12.3.2.2 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 1.8 V 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.

12.3.2.3 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.

12.3.2.4 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.

12.3.3 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.

12.3.3.1 Multiplexer Channel Selection

Table 12.1. ADC0 Input Multiplexer Channels

ADC0MX setting	Signal Name	Enumeration Name	QFN28 Pin Name	QSOP24 Pin Name	QFN20 Pin Name
00000	ADC0.0	ADC0P0	P0.0	P0.0	P0.0
00001	ADC0.1	ADC0P1	P0.1	P0.1	P0.1
00010	ADC0.2	ADC0P2	P0.2	P0.2	P0.2
00011	ADC0.3	ADC0P3	P0.3	P0.3	P0.3
00100	ADC0.4	ADC0P4	P0.4	P0.4	P0.4
00101	ADC0.5	ADC0P5	P0.5	P0.5	P0.5
00110	ADC0.6	ADC0P6	P0.6	P0.6	P0.6
00111	ADC0.7	ADC0P7	P0.7	P0.7	P0.7
01000	ADC0.8	ADC0P8	P1.0	P1.0	P1.0
01001	ADC0.9	ADC0P9	P1.1	P1.1	P1.1
01010	ADC0.10	ADC0P10	P1.2	P1.2	P1.2
01011	ADC0.11	ADC0P11	P1.3	P1.3	Reserved
01100	ADC0.12	ADC0P12	P1.4	P1.4	Reserved
01101	ADC0.13	ADC0P13	P1.5	P1.5	Reserved
01110	ADC0.14	ADC0P14	P1.6	P1.6	Reserved
01111	ADC0.15	ADC0P15	P1.7	Reserved	Reserved
10000	ADC0.16	TEMP	Internal Temperature Sensor		
10001	ADC0.17	LDO_OUT	Internal 1.8 V LDO Output		
10010	ADC0.18	VDD	VDD Supply Pin		
10011	ADC0.19	GND	GND Supply Pin		
10100	ADC0.20	ADC0P20	P2.0	Reserved	Reserved

ADC0MX setting	Signal Name	Enumeration Name	QFN28 Pin Name	QSOP24 Pin Name	QFN20 Pin Name
10101	ADC0.21	ADC0P21	P2.1	Reserved	Reserved
10110	ADC0.22	ADC0P22	P2.2	Reserved	Reserved
10111	ADC0.23	ADC0P23	P2.3	Reserved	Reserved
11000 - 11011	ADC0.24 - ADC0.27		Reserved	Reserved	Reserved
11100	ADC0.28	USB_DP	USB D+ pin		
11101	ADC0.29	USB_DM	USB D- pin		
11110	ADC0.30	VREGIN_DIV_4	VREGIN / 4		
11111	ADC0.31	NONE	No connection		

12.3.4 Gain Setting

The ADC has gain settings of 1x and 0.5x. In 1x mode, the full scale reading of the ADC is determined directly by VREF. In 0.5x mode, the full-scale reading of the ADC occurs when the input voltage is VREF x 2. The 0.5x gain setting 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 bit in register ADC0CF. Note that even with a gain setting of 0.5, voltages above the supply rail cannot be measured directly by the ADC.

12.3.5 Initiating Conversions

A conversion can be initiated in many ways, depending on the programmed state of the ADCM bitfield. Conversions may be initiated by one of the following:

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

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. When the conversion is complete, the ADC posts the result to its output register and sets the ADC interrupt flag (ADINT). ADINT may be used to trigger a system interrupts, if enabled, or polled by firmware.

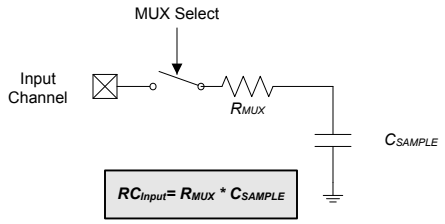
During a conversion, the ADBUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. However, the ADBUSY bit should not be used to poll for ADC conversion completion. The ADC0 interrupt flag (ADINT) should be used instead of the ADBUSY bit. Converted data is available in the ADC0 data registers, ADC0H:ADC0L, when the conversion is complete.
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.

12.3.6 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. 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 12.2. ADC Eqivalent 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

When burst mode is disabled, the ADTM bit controls the ADC track-and-hold mode. In its default state the ADC input is continuously tracked, except when a conversion is in progress. A conversion will begin immediately when the start-of-conversion trigger occurs. When the ADTM bit is logic 1, each conversion is preceded by a tracking period of 4 SAR clocks (after the start-of-conversion signal) for any internal conversion trigger source. When the CNVSTR signal is used to initiate conversions with ADTM set to 1, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR. Setting ADTM to 1 is primarily useful when AMUX settings are frequently changed and conversions are started using the ADBUSY bit.

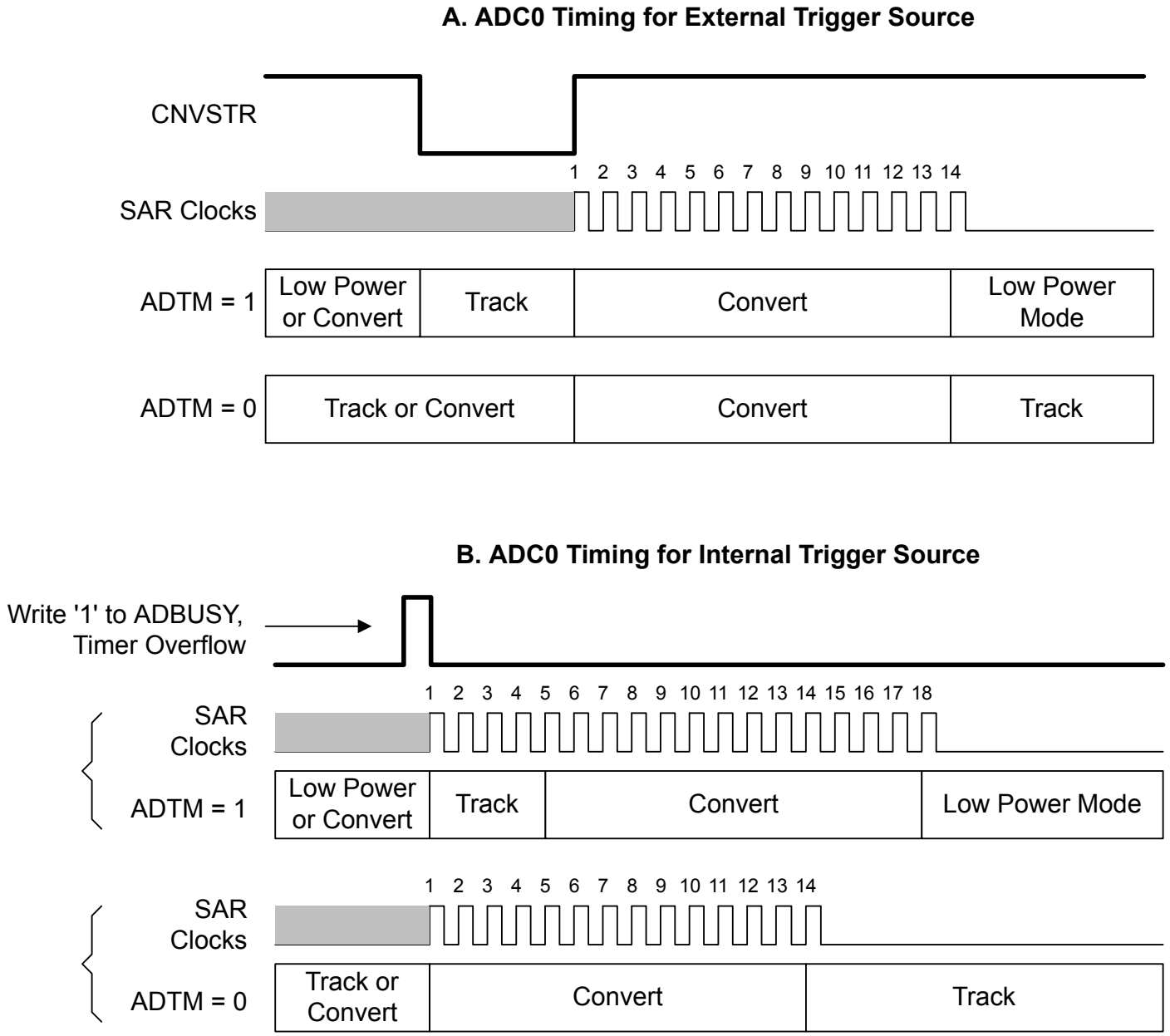
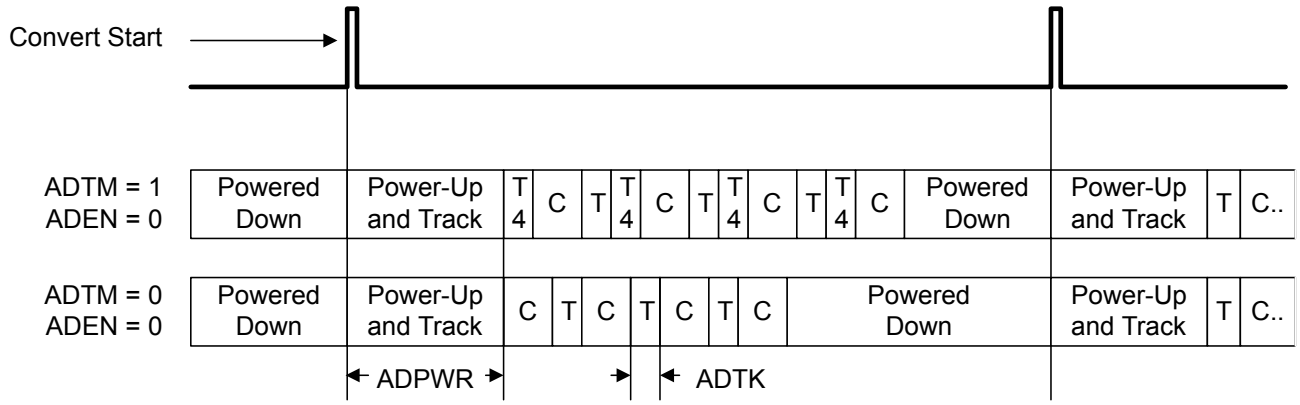


Figure 12.3. Track and Conversion Example Timing (Normal, Non-Burst Operation)

When burst mode is enabled, additional tracking times may need to be specified. Because burst mode may power the ADC on from an unpowered state and take multiple conversions for each start-of-conversion source, two additional timing fields are provided. If the ADC is powered down when the burst sequence begins, it will automatically power up and wait for the time specified in the ADPWR bit field. If the ADC is already powered on, tracking depends solely on ADTM for the first conversion. The ADTK field determines the amount of tracking time given to any subsequent samples in burst mode—essentially, ADTK specifies how long the ADC will wait between burt-mode conversions. If ADTM is set, an additional 4 SAR clocks will be added to the tracking phase of all conversions in burst mode.

Figure 12.4. Burst Mode Timing



T = Tracking set by ADTK
T4 = Tracking set by ADTM (4 SAR clocks)
C = Converting

12.3.7 Burst Mode

Burst mode is a power saving feature that allows the ADC to remain in a low power state between conversions. When burst mode is enabled, the ADC wakes from a low power state, accumulates 1, 4, 8, 16, 32, or 64 samples using the internal low-power high-frequency oscillator, then re-enters a low power state. Since the burst mode clock is independent of the system clock, the ADC can perform multiple conversions then enter a low power state within a single system clock cycle, even if the system clock is running from a slow oscillator.
Note: When using burst mode, care must be taken to issue a convert start signal no faster than once every four SYSCLK periods. This includes external convert start signals. The ADC will ignore convert start signals which arrive before a burst is finished.

Burst mode is enabled by setting ADBMEN to logic 1. When in burst mode, ADEN controls the ADC idle power state (i.e., the state the ADC enters when not tracking or performing conversions). If ADEN is set to logic 0, the ADC is powered down after each burst. If ADEN is set to logic 1, the ADC remains enabled after each burst. On each convert start signal, the ADC is awakened from its idle power state. If the ADC is powered down, it will automatically power up and wait for the amount of time programmed to the ADPWR bits before performing a conversion. Otherwise, the ADC will start tracking and converting immediately.

When burst mode is enabled, a single convert start will initiate a number of conversions equal to the repeat count. When burst mode is disabled, a convert start is required to initiate each conversion. In both modes, the ADC end of conversion interrupt flag (ADINT) will be set after “repeat count” conversions have been accumulated. Similarly, the window comparator will not compare the result to the greater-than and less-than registers until “repeat count” conversions have been accumulated.

12.3.8 8-Bit Mode

Setting the AD8BE bit to 1 will put the ADC in 8-bit mode. In 8-bit mode, only the 8 MSBs of data are converted, allowing the conversion to be completed in fewer SAR clock cycles than a 10-bit conversion. The two LSBs of a conversion are always 00 in this mode, and the ADC0L register will always read back 0x00.

12.3.9 12-Bit Mode

When configured for 12-bit conversions, the ADC performs four 10-bit conversions using four different reference voltages and combines the results into a single 12-bit value. Unlike simple averaging techniques, this method provides true 12-bit resolution of ac or dc input signals without depending on noise to provide dithering. The converter also employs a hardware dynamic element matching algorithm that reconfigures the largest elements of the internal DAC for each of the four 10-bit conversions. This reconfiguration cancels any matching errors and enables the converter to achieve 12-bit linearity performance to go along with its 12-bit resolution.

The 12-bit mode is enabled by setting the AD12BE bit in register ADC0AC to logic 1 and configuring the ADC in burst mode (ADBMEN = 1) for four or more conversions. The conversion can be initiated using any of the conversion start sources, and the 12-bit result will appear in the ADC0H and ADC0L registers. Since the 12-bit result is formed from a combination of four 10-bit results, the maximum output value is 4 x (1023) = 4092, rather than the max value of (2^12 – 1) = 4095 that is produced by a traditional 12-bit converter. To further increase resolution, the burst mode repeat value may be configured to any multiple of four conversions. For example, if a repeat value of 16 is selected, the ADC0 output will be a 14-bit number (sum of four 12-bit numbers) with 13 effective bits of resolution.

The AD12SM bit in register ADC0TK controls when the ADC will track and sample the input signal. When AD12SM is set to 1, the selected input signal will be tracked before the first conversion of a set and held internally during all four conversions. When AD12SM is cleared to 0, the ADC will track and sample the selected input before each of the four conversions in a set. When maximum throughput (180-200 ksps) is needed, it is recommended that AD12SM be set to 1 and ADTK to 0x3F, and that the ADC be placed in always-on mode (ADEN = 1). For sample rates under 180 ksps, or when accumulating multiple samples, AD12SM should normally be cleared to 0, and ADTK should be configured to provide the appropriate settling time for the subsequent conversions.

12.3.10 Output Formatting

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 can be right-justified or left-justified, depending on the setting of the ADSJST field. When the repeat count is set to 1 in 10-bit mode, conversion codes are represented as 10-bit unsigned integers. Inputs are measured from 0 to VREF x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADC0H and ADC0L registers are set to 0.

Table 12.2. 10-Bit Output Code Example

Input Voltage	Right-Justified (ADSJST = 000) ADC0H:L	Left-Justified (ADSJST = 100) ADC0H:L
VREF x 1023/1024	0x03FF	0xFFC0
VREF x 512/1024	0x0200	0x8000
VREF x 256/1024	0x0100	0x4000
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, 32, or 64 consecutive samples can be accumulated and represented in unsigned integer format. The repeat count can be selected using the ADRPT bit field. When a repeat count is higher than 1, the ADC output must be right-justified (ADSJST = 0xx); 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. Notice that accumulating 2n 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

Input Voltage	Repeat Count = 4	Repeat Count = 16	Repeat Count = 64
VREF x 1023/1024	0x0FFC	0x3FF0	0xFFC0
VREF x 512/1024	0x0800	0x2000	0x8000
VREF x 511/1024	0x07FC	0x1FF0	0x7FC0
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. Based on the principles of oversampling and averaging, the effective ADC resolution increases by 1 bit each time the oversampling rate is increased by a factor of 4. The example below shows how to increase the effective ADC resolution by 1, 2, and 3 bits to obtain an effective ADC resolution of 11- bit, 12-bit, or 13-bit respectively without CPU intervention.

Table 12.4. Using ADSJST for Output Formatting

Input Voltage	Repeat Count = 4 Shift Right = 1 11-Bit Result	Repeat Count = 16 Shift Right = 2 12-Bit Result	Repeat Count = 64 Shift Right = 3 12-Bit Result
VREF x 1023/1024	0x07F7	0x0FFC	0x1FF8
VREF x 512/1024	0x0400	0x0800	0x1000
VREF x 511/1024	0x03FE	0x04FC	0x0FF8
0	0x0000	0x0000	0x0000

12.3.11 Power Considerations

The ADC has several power-saving features which can help the user optimize power consumption according to the needs of the application. The most efficient way to use the ADC for slower sample rates is by using burst mode. Burst mode dynamically controls power to the ADC and (if used) the internal voltage reference. By completely powering off these circuits when the ADC is not tracking or converting, the average supply current required for lower sampling rates is reduced significantly.

The ADC also provides low power options that allow reduction in operating current when operating at low SAR clock frequencies or with longer tracking times. The internal common-mode buffer can be configured for low power mode by setting the ADLPM bit in ADC0PWR to 1. Two other fields in the ADC0PWR register (ADBIAS and ADMXLP) may be used together to adjust the power consumed by the ADC and its multiplexer and reference buffers, respectively. In general, these options are used together, when operating with a SAR conversion clock frequency of 4 MHz.

Table 12.5. ADC Optimal Power Configuration (8- and 10-bit Mode)

Required Throughput	Reference Source	Mode Configuration	SAR Clock Speed	Other Register Field Settings
325-800 ksps	Any	Always-On (ADEN = 1 ADBMEN = 0)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x40 ADC0TK = N/A ADRPT = 0
0-325 ksps	External	Burst Mode (ADEN = 0 ADBMEN = 1)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x44 ADC0TK = 0x3A ADRPT = 0
250-325 ksps	Internal	Burst Mode (ADEN = 0 ADBMEN = 1)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x44 ADC0TK = 0x3A ADRPT = 0
200-250 ksps	Internal	Burst Mode (ADEN = 0 ADBMEN = 1)	4.08 MHz (ADSC = 5)	ADC0PWR = 0xF0 ADC0TK = N/A ADRPT = 0
0-200 ksps	Internal	Burst Mode (ADEN = 0 ADBMEN = 1)	4.08 MHz (ADSC = 5)	ADC0PWR = 0xF4 ADC0TK = 0x34 ADRPT = 0

Notes:
1. For always-on configuration, ADSC settings assume SYSCLK is the internal 24.5 MHz high-frequency oscillator. Adjust ADSC as needed if using a different source for SYSCLK.
2. ADRPT reflects the minimum setting for this bit field. When using the ADC in Burst Mode, up to 64 samples may be auto-accumulated per conversion start by adjusting ADRPT.

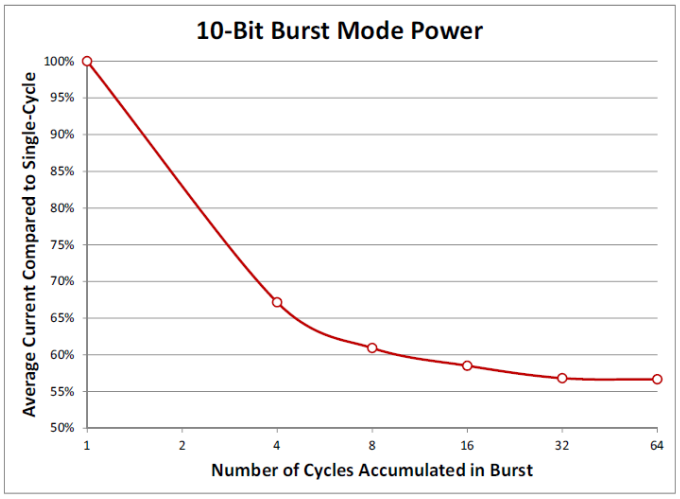
Table 12.6. ADC Optimal Power Configuration (12-bit Mode)

Required Throughput	Reference Source	Mode Configuration	SAR Clock Speed	Other Register Field Settings
180-200 ksps	Any	Always-On + Burst Mode (ADEN = 1 ADBMEN = 1)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x40 ADC0TK = 0xBF ADRPT = 1

Required Throughput	Reference Source	Mode Configuration	SAR Clock Speed	Other Register Field Settings
125-180 ksps	Any	Always-On + Burst Mode (ADEN = 1 ADBMEN = 1)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x40 ADC0TK = 0x3A ADRPT = 1
0-125 ksps	External	Burst Mode (ADEN = 0 ADBMEN = 1)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x44 ADC0TK = 0x3A ADRPT = 1
50-125 ksps	Internal	Burst Mode (ADEN = 0 ADBMEN = 1)	12.25 MHz (ADSC = 1)	ADC0PWR = 0x44 ADC0TK = 0x3A ADRPT = 1
0-50 ksps	Internal	Burst Mode (ADEN = 0 ADBMEN = 1)	4.08 MHz (ADSC = 5)	ADC0PWR = 0xF4 ADC0TK = 0x34 ADRPT = 1

Notes:
1. ADRPT reflects the minimum setting for this bit field. When using the ADC in burst mode, up to 64 samples may be auto-accumulated per conversion trigger by adjusting ADRPT.

For applications where burst mode is used to automatically accumulate multiple results, additional supply current savings can be realized. The length of time the ADC is active during each burst contains power-up time at the beginning of the burst as well as the conversion time required for each conversion in the burst. The power-on time is only required at the beginning of each burst. When compared with single-sample bursts to collect the same number of conversions, multi-sample bursts will consume significantly less power. For example, performing an eight-cycle burst of 10-bit conversions consumes about 61% of the power required to perform those same eight samples in single-cycle bursts. For 12-bit conversions, an eight-cycle burst results in about 85% of the equivalent single-cycle bursts. See the electrical characteristics tables for details on power consumption and the maximum clock frequencies allowed in each mode.



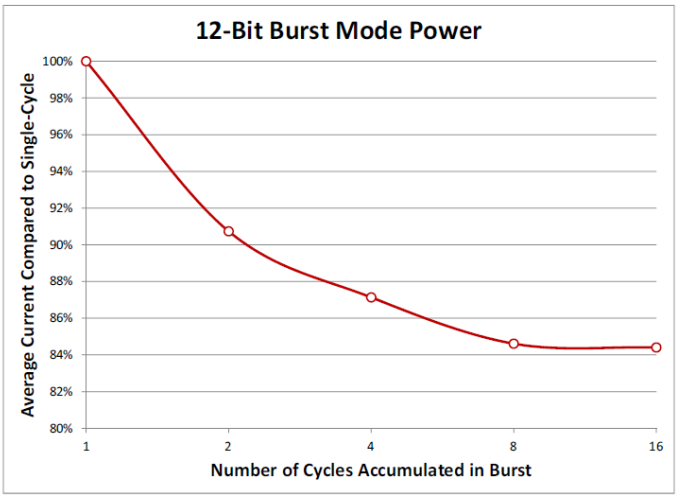


Figure 12.5. Burst Mode Accumulation Power Savings

12.3.12 Window Comparator

The ADC's programmable window detector continuously 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, beween, or outside of specific values.

Table 12.7. ADC Window Comparator Example (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 12.8. ADC Window Comparator Example (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 12.9. ADC Window Comparator Example (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 12.10. ADC Window Comparator Example (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	

12.3.13 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 12.6 Temperature Sensor Transfer Function on page 126. The output voltage (V_{TEMP}) is the positive ADC input when the ADC multiplexer is set correctly. The TEMPE bit in register REF0CN 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.

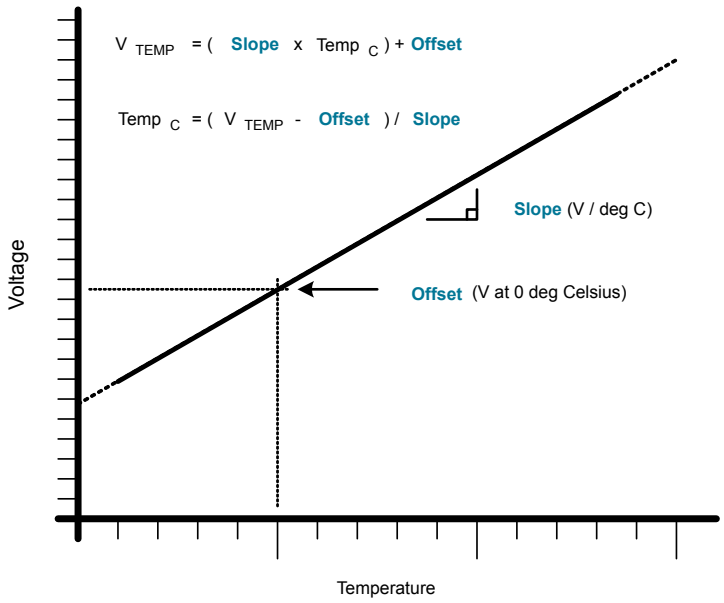


Figure 12.6. Temperature Sensor Transfer Function

12.3.13.1 Temperature Sensor Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements. For absolute temperature measurements, offset and/or gain calibration is recommended. 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 ADC0 Control Registers

12.4.1 ADC0CN0: ADC0 Control 0

Bit	7	6	5	4	3	2	1	0
Name	ADEN	ADBMEN	ADINT	ADBUSY	ADWINT	ADCM		
Access	RW	RW	RW	RW	RW	RW		
Reset	0	0	0	0	0	0x0		
SFR Page = 0x0, 0x10; 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	ADBMEN	0	RW	**Burst Mode Enable.**
	Value	Name		Description
	0	BURST_DISABLED		Disable ADC0 burst mode.
	1	BURST_ENABLED		Enable ADC0 burst mode.
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.			
2:0	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	CEX2		ADC0 conversion initiated on rising edge of CEX2.
	0x6	GATED_TIMER2		ADC0 conversion initiated on overflow of Timer 2 when CEX2 is logic high.
	0x7	TIMER4		ADC0 conversion initiated on overflow of Timer 4.

12.4.2 ADC0CN1: ADC0 Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved							ADCMBE
Access	R							RW
Reset	0x00							1
SFR Page = 0x0, 0x10; SFR Address: 0xB2								

Bit	Name	Reset	Access	Description
7:1	*Reserved*	*Must write reset value.*		
0	ADCMBE	1	RW	**Common Mode Buffer Enable.**

Value	Name	Description
0	CM_BUFFER_DISABLED	Disable the common mode buffer. This setting should be used only if the tracking time of the signal is greater than 1.5 us.
1	CM_BUFFER_ENABLED	Enable the common mode buffer. This setting should be used in most cases, and will give the best dynamic ADC performance. The common mode buffer must be enabled if signal tracking time is less than or equal to 1.5 us.

12.4.3 ADC0CF: ADC0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	ADSC					AD8BE	ADTM	ADGN
Access	RW					RW	RW	RW
Reset	0x1F					0	0	0
SFR Page = 0x0, 0x10; 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 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: Fclksar = (Fadcclk) / (ADSC + 1) F_{ADCCLK} is equal to the selected SYSCLK when ADBMEN is 0 and the high-frequency oscillator when ADBMEN is 1.
2	AD8BE	0	RW	**8-Bit Mode Enable.**
	Value	Name		Description
	0	NORMAL		ADC0 operates in 10-bit or 12-bit mode (normal operation).
	1	8_BIT		ADC0 operates in 8-bit mode.
1	ADTM	0	RW	**Track Mode.** Selects between Normal or Delayed Tracking Modes.
	Value	Name		Description
	0	TRACK_NORMAL		Normal Track Mode. When ADC0 is enabled, conversion begins immediately following the start-of-conversion signal.
	1	TRACK_DELAYED		Delayed Track Mode. When ADC0 is enabled, conversion begins 4 SAR clock cycles following the start-of-conversion signal. The ADC is allowed to track during this time.
0	ADGN	0	RW	**Gain Control.**
	Value	Name		Description
	0	GAIN_0P5		The on-chip PGA gain is 0.5.
	1	GAIN_1		The on-chip PGA gain is 1.

12.4.4 ADC0AC: ADC0 Accumulator Configuration

Bit	7	6	5	4	3	2	1	0
Name	AD12BE	ADAE	ADSJST			ADRPT		
Access	RW	RW	RW			RW		
Reset	0	0	0x0			0x0		
SFR Page = 0x0, 0x10; SFR Address: 0xB3								

Bit	Name	Reset	Access	Description
7	AD12BE	0	RW	**12-Bit Mode Enable.**

Enables 12-bit mode. In 12-bit mode, the ADC throughput is reduced by a factor of 4.

Value	Name	Description
0	12_BIT_DISABLED	Disable 12-bit mode.
1	12_BIT_ENABLED	Enable 12-bit mode.

Bit	Name	Reset	Access	Description
6	ADAE	0	RW	**Accumulate Enable.**

Enables multiple conversions to be accumulated when burst mode is disabled.

Value	Name	Description
0	ACC_DISABLED	ADC0H:ADC0L contain the result of the latest conversion when Burst Mode is disabled.
1	ACC_ENABLED	ADC0H:ADC0L contain the accumulated conversion results when Burst Mode is disabled. Firmware must write 0x0000 to ADC0H:ADC0L to clear the accumulated result.

Bit	Name	Reset	Access	Description
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.
0x4	LEFT_NO_SHIFT	Left justified. No shifting applied.

Bit	Name	Reset	Access	Description
2:0	ADRPT	0x0	RW	**Repeat Count.**

Selects the number of conversions to perform and accumulate in Burst Mode. This bit field must be set to 000 if Burst Mode is disabled.

Value	Name	Description
0x0	ACC_1	Perform and Accumulate 1 conversion (not used in 12-bit mode).
0x1	ACC_4	Perform and Accumulate 4 conversions (1 conversion in 12-bit mode).
0x2	ACC_8	Perform and Accumulate 8 conversions (2 conversions in 12-bit mode).
0x3	ACC_16	Perform and Accumulate 16 conversions (4 conversions in 12-bit mode).
0x4	ACC_32	Perform and Accumulate 32 conversions (8 conversions in 12-bit mode).

Bit	Name	Reset	Access	Description
	0x5	ACC_64		Perform and Accumulate 64 conversions (16 conversions in 12-bit mode).

12.4.5 ADC0PWR: ADC0 Power Control

Bit	7	6	5	4	3	2	1	0
Name	ADBIAS		ADMXLP	ADLPM	ADPWR			
Access	RW		RW	RW	RW			
Reset	0x0		0	0	0xF			
SFR Page = 0x0, 0x10; SFR Address: 0xDF								

Bit	Name	Reset	Access	Description
7:6	ADBIAS	0x0	RW	**Bias Power Select.** This field can be used to adjust the ADC's power consumption based on the conversion speed. Higher bias currents allow for faster conversion times.

Value	Name	Description
0x0	MODE0	Select bias current mode 0. Recommended to use modes 1, 2, or 3.
0x1	MODE1	Select bias current mode 1 (SARCLK <= 16 MHz).
0x2	MODE2	Select bias current mode 2.
0x3	MODE3	Select bias current mode 3 (SARCLK <= 4 MHz).

Bit	Name	Reset	Access	Description
5	ADMXLP	0	RW	**Mux and Reference Low Power Mode Enable.** Enables low power mode operation for the multiplexer and voltage reference buffers.

Value	Name	Description
0	LP_MUX_VREF_DISABLED	Low power mode disabled.
1	LP_MUX_VREF_ENABLED	Low power mode enabled (SAR clock < 4 MHz).

Bit	Name	Reset	Access	Description
4	ADLPM	0	RW	**Low Power Mode Enable.** This bit can be used to reduce power to the ADC's internal common mode buffer. It can be set to 1 to reduce power when tracking times in the application are longer (slower sample rates).

Value	Name	Description
0	LP_BUFFER_DISABLED	Disable low power mode.
1	LP_BUFFER_ENABLED	Enable low power mode (requires extended tracking time).

Bit	Name	Reset	Access	Description
3:0	ADPWR	0xF	RW	**Burst Mode Power Up Time.** This field sets the time delay allowed for the ADC to power up from a low power state. When ADTM is set, an additional 4 SARCLKs are added to this time. Tpwrtime = (8 * ADPWR) / (Fhfosc)

12.4.6 ADC0TK: ADC0 Burst Mode Track Time

Bit	7	6	5	4	3	2	1	0
Name	AD12SM	Reserved	ADTK					
Access	RW	RW	RW					
Reset	0	0	0x1E					
SFR Page = 0x0, 0x10; SFR Address: 0xB9								

Bit	Name	Reset	Access	Description
7	AD12SM	0	RW	**12-Bit Sampling Mode.** This bit controls the way that the ADC samples the input when in 12-bit mode. When the ADC is configured for multiple 12-bit conversions in burst mode, the AD12SM bit should be cleared to 0.

Value	Name	Description
0	SAMPLE_FOUR	The ADC will re-track and sample the input four times during a 12-bit conversion.
1	SAMPLE_ONCE	The ADC will sample the input once at the beginning of each 12-bit conversion. The ADTK field can be set to 63 to maximize throughput.

6	*Reserved*	*Must write reset value.*		
5:0	ADTK	0x1E	RW	**Burst Mode Tracking Time.** This field sets the time delay between consecutive conversions performed in Burst Mode. When ADTM is set, an additional 4 SARCLKs are added to this time. Tbmtk = (64 - ADTK) / (Fhfosc) The Burst Mode track delay is not inserted prior to the first conversion. The required tracking time for the first conversion should be defined with the ADPWR field.

12.4.7 ADC0H: ADC0 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0H							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; 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.				

12.4.8 ADC0L: ADC0 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0L							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x10; 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.				

12.4.9 ADC0GTH: ADC0 Greater-Than High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTH							
Access	RW							
Reset	0xFF							
SFR Page = 0x0, 0x10; 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, 0x10; 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, 0x10; 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, 0x10; 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, 0x10; 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 REF0CN: Voltage Reference Control

Bit	7	6	5	4	3	2	1	0
Name	IREFLVL	Reserved	GNDSL	REFSL		TEMPE	Reserved	
Access	RW	R	RW	RW		RW	R	
Reset	0	0	0	0x3		0	0x0	
SFR Page = 0x0, 0x10; SFR Address: 0xD1								

Bit	Name	Reset	Access	Description
7	IREFLVL	0	RW	**Internal Voltage Reference Level.**
	Sets the voltage level for the internal reference source.			
	Value	Name		Description
	0	1P65		The internal reference operates at 1.65 V nominal.
	1	2P4		The internal reference operates at 2.4 V nominal.
6	*Reserved*	*Must write reset value.*		
5	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 P0.1/AGND pin.
4:3	REFSL	0x3	RW	**Voltage Reference Select.**
	Selects the ADC0 voltage reference.			
	Value	Name		Description
	0x0	VREF_PIN		The ADC0 voltage reference is the P0.0/VREF pin.
	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.
2	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.
1:0	*Reserved*	*Must write reset value.*		

13. Comparators (CMP0 and CMP1)

13.1 Introduction

Analog comparators are 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.

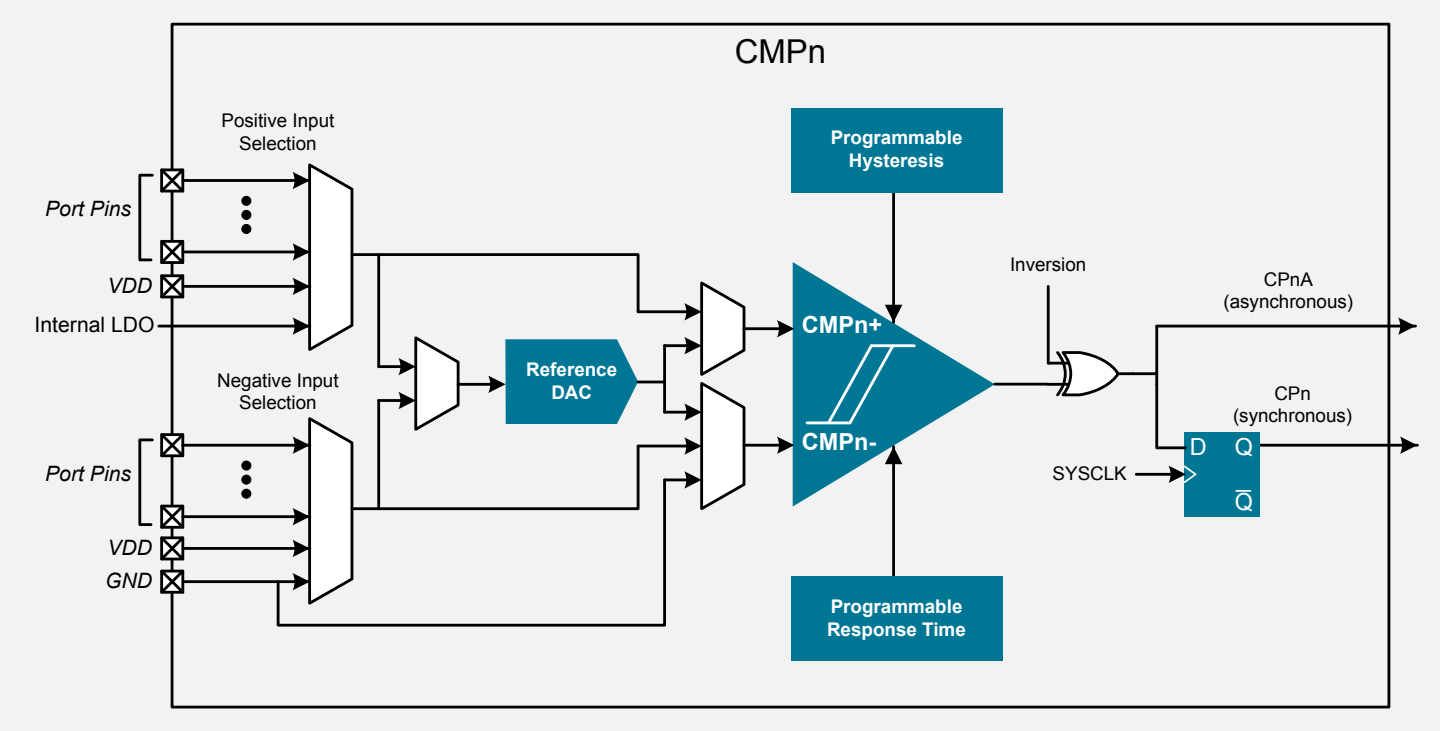


Figure 13.1. Comparator Block Diagram

13.2 Features

The comparator includes the following features:

- Up to 10 (CMP0) or 12 (CMP1) external positive inputs
- Up to 10 (CMP0) or 12 (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.

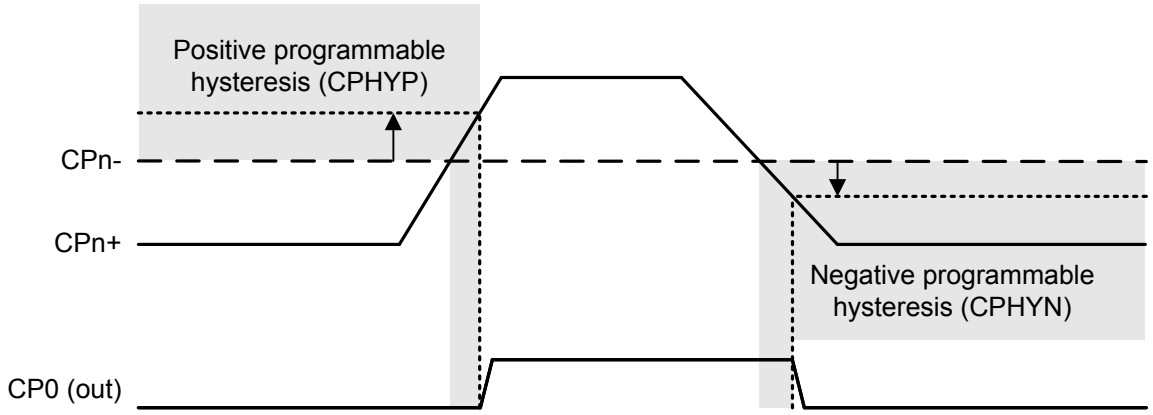


Figure 13.2. Comparator Hysteresis Plot

13.3.3 Input Selection

Comparator inputs may be routed to port I/O pins or internal signals. When connected externally, the comparator inputs can be driven from –0.25 V to (VDD) +0.25 V without damage or upset. 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	QFN28 Pin Name	QSOP24 Pin Name	QFN20 Pin Name
0000	CMP0P.0	CMP0P0	P0.0	P0.0	P0.0
0001	CMP0P.1	CMP0P1	P0.1	P0.1	P0.1
0010	CMP0P.2	CMP0P2	P0.2	P0.2	P0.2
0011	CMP0P.3	CMP0P3	P0.3	P0.3	P0.3
0100	CMP0P.4	CMP0P4	P0.4	P0.4	P0.4
0101	CMP0P.5	CMP0P5	P0.5	P0.5	P0.5
0110	CMP0P.6	CMP0P6	P0.6	P0.6	P0.6
0111	CMP0P.7	CMP0P7	P0.7	P0.7	P0.7
1000	CMP0P.8	LDO_OUT	Internal 1.8V LDO output		
1001	CMP0P.9	CMP0P9	P1.0	Reserved	Reserved
1010	CMP0P.10	CMP0P10	P1.1	Reserved	Reserved
1011-1110	CMP0P.11 - CMP0P.14		No connection / Reserved		
1111	CMP0P.15	VDD	VDD Supply Pin		

Table 13.2. CMP0 Negative Input Multiplexer Channels

CMXN Setting in Register CMP0MX	Signal Name	Enumeration Name	QFN28 Pin Name	QSOP24 Pin Name	QFN20 Pin Name
0000	CMP0N.0	CMP0N0	P0.0	P0.0	P0.0
0001	CMP0N.1	CMP0N1	P0.1	P0.1	P0.1
0010	CMP0N.2	CMP0N2	P0.2	P0.2	P0.2
0011	CMP0N.3	CMP0N3	P0.3	P0.3	P0.3
0100	CMP0N.4	CMP0N4	P0.4	P0.4	P0.4
0101	CMP0N.5	CMP0N5	P0.5	P0.5	P0.5
0110	CMP0N.6	CMP0N6	P0.6	P0.6	P0.6
0111	CMP0N.7	CMP0N7	P0.7	P0.7	P0.7
1000	CMP0N.8	GND	GND Supply Pin		
1001	CMP0N.9	CMP0N9	P1.0	Reserved	Reserved
1010	CMP0N.10	CMP0N10	P1.1	Reserved	Reserved
1011-1110	CMP0N.11 - CMP0N.14		No connection / Reserved		
1111	CMP0N.15	VDD	VDD Supply Pin		

Table 13.3. CMP1 Positive Input Multiplexer Channels

CMXP Setting in Register CMP1MX	Signal Name	Enumeration Name	QFN28 Pin Name	QSOP24 Pin Name	QFN20 Pin Name
0000	CMP1P.0	CMP1P0	P1.0	P0.6	P0.6
0001	CMP1P.1	CMP1P1	P1.1	P0.7	P0.7
0010	CMP1P.2	CMP1P2	P1.2	P1.0	P1.0
0011	CMP1P.3	CMP1P3	P1.3	P1.1	P1.1
0100	CMP1P.4	CMP1P4	P1.4	P1.2	P1.2
0101	CMP1P.5	CMP1P5	P1.5	P1.3	Reserved
0110	CMP1P.6	CMP1P6	P1.6	P1.4	Reserved
0111	CMP1P.7	CMP1P7	P1.7	P1.5	Reserved
1000	CMP1P.8	LDO_OUT	Internal 1.8V LDO output		
1001	CMP1P.9	CMP1P9	P2.0	P1.6	Reserved
1010	CMP1P.10	CMP1P10	P2.1	Reserved	Reserved
1011	CMP1P.11	CMP1P11	P2.2	Reserved	Reserved
1100	CMP1P.12	CMP1P12	P2.3	Reserved	Reserved
1101-1110	CMP1P.13 - CMP1P.14		No connection / Reserved		
1111	CMP1P.15	VDD	VDD Supply Pin		

Table 13.4. CMP1 Negative Input Multiplexer Channels

CMXN Setting in Register CMP1MX	Signal Name	Enumeration Name	QFN28 Pin Name	QSOP24 Pin Name	QFN20 Pin Name
0000	CMP1N.0	CMP1N0	P1.0	P0.6	P0.6
0001	CMP1N.1	CMP1N1	P1.1	P0.7	P0.7
0010	CMP1N.2	CMP1N2	P1.2	P1.0	P1.0
0011	CMP1N.3	CMP1N3	P1.3	P1.1	P1.1
0100	CMP1N.4	CMP1N4	P1.4	P1.2	P1.2
0101	CMP1N.5	CMP1N5	P1.5	P1.3	Reserved
0110	CMP1N.6	CMP1N6	P1.6	P1.4	Reserved
0111	CMP1N.7	CMP1N7	P1.7	P1.5	Reserved
1000	CMP1N.8	GND	GND Supply Pin		
1001	CMP1N.9	CMP1N9	P2.0	P1.6	Reserved
1010	CMP1N.10	CMP1N10	P2.1	Reserved	Reserved
1011	CMP1N.11	CMP1N11	P2.2	Reserved	Reserved
1100	CMP1N.12	CMP1N12	P2.3	Reserved	Reserved
1101-1110	CMP1N.13 - CMP1N.14		No connection / Reserved		
1111	CMP1N.15	VDD	VDD Supply Pin		

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.

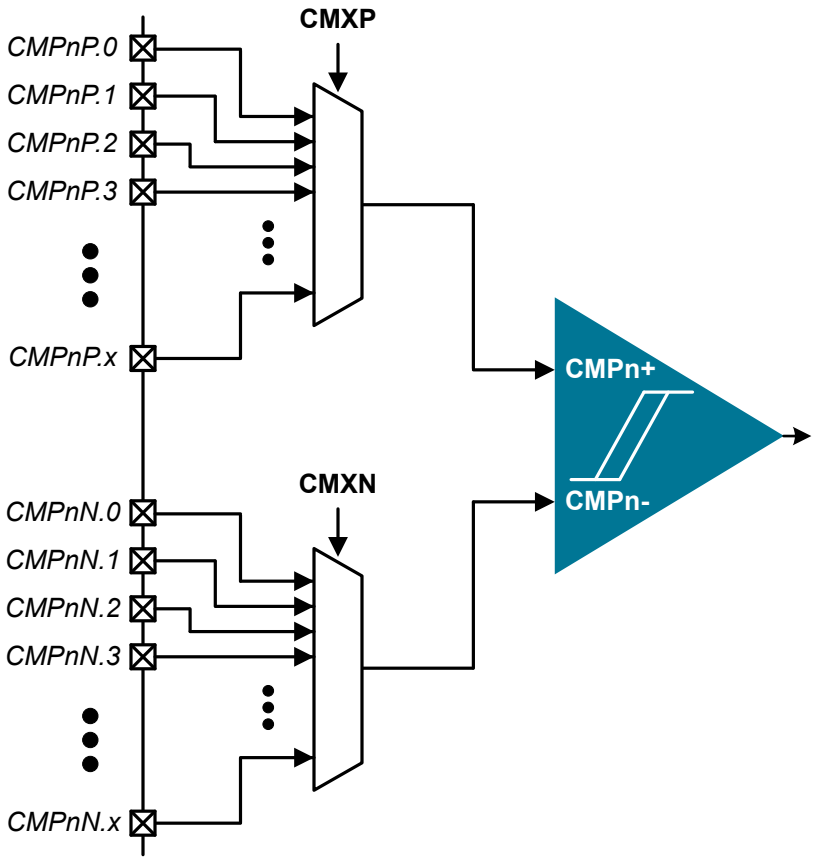


Figure 13.3. Direct Input Connection

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.

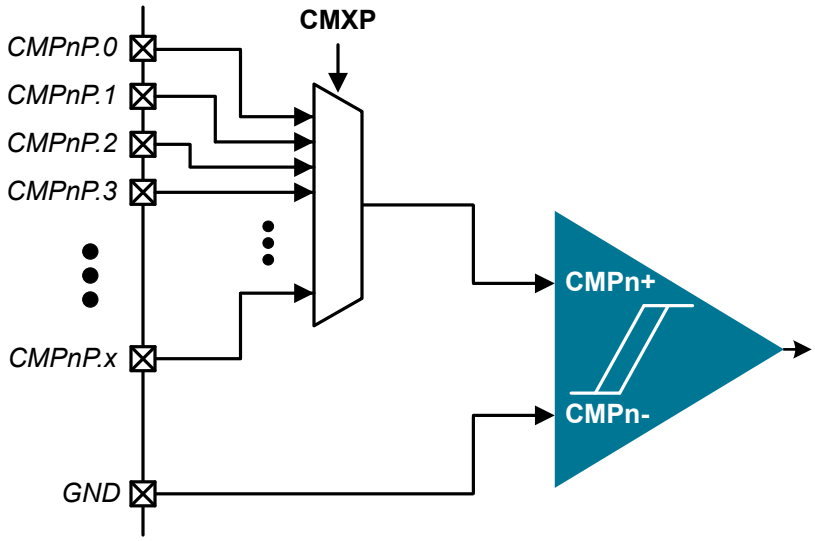


Figure 13.4. Negative Input Ground Connection

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.

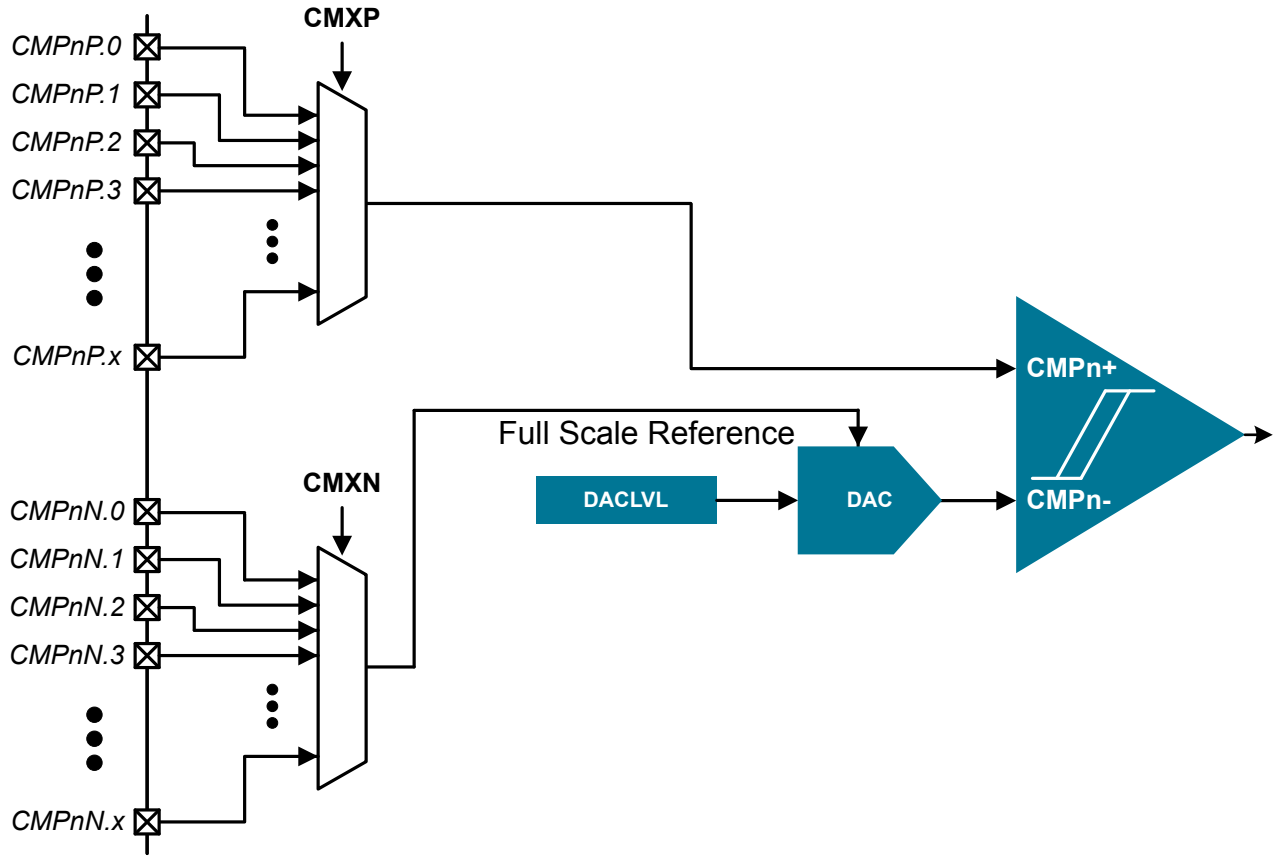


Figure 13.5. Negative Input DAC Connection

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.

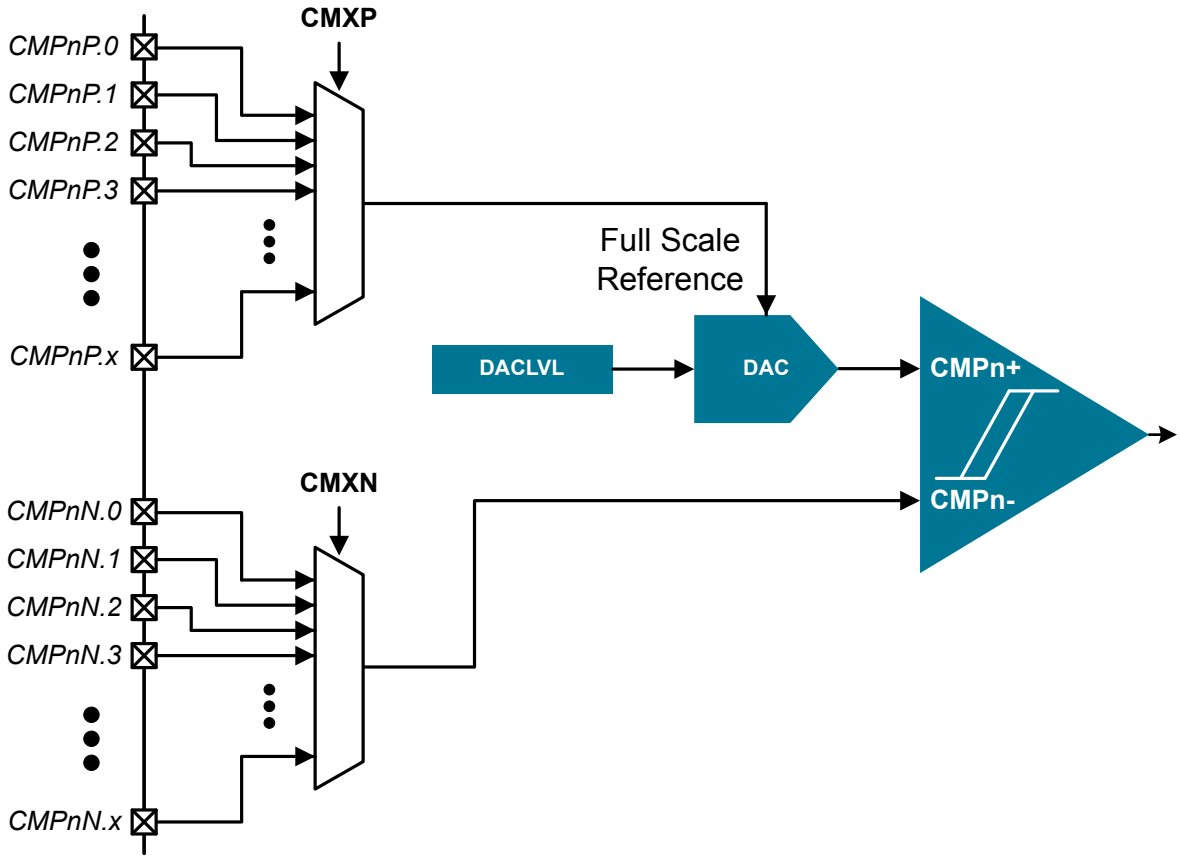


Figure 13.6. Positive Input DAC Connection

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. CMP0 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, 0x10; 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).
1:0	CPHYN	0x0	RW	**Comparator Negative Hysteresis Control.**

Bit	Name	Reset	Access	Description
	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, 0x10; 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.

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 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, 0x10; 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 = 0x10; 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, 0x10; 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).
1:0	CPHYN	0x0	RW	**Comparator Negative Hysteresis Control.**

Bit	Name	Reset	Access	Description
	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, 0x10; 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, 0x10; 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 = 0x10; 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.

Bit	Name	Reset	Access	Description
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. Cyclic Redundancy Check (CRC0)

14.1 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.

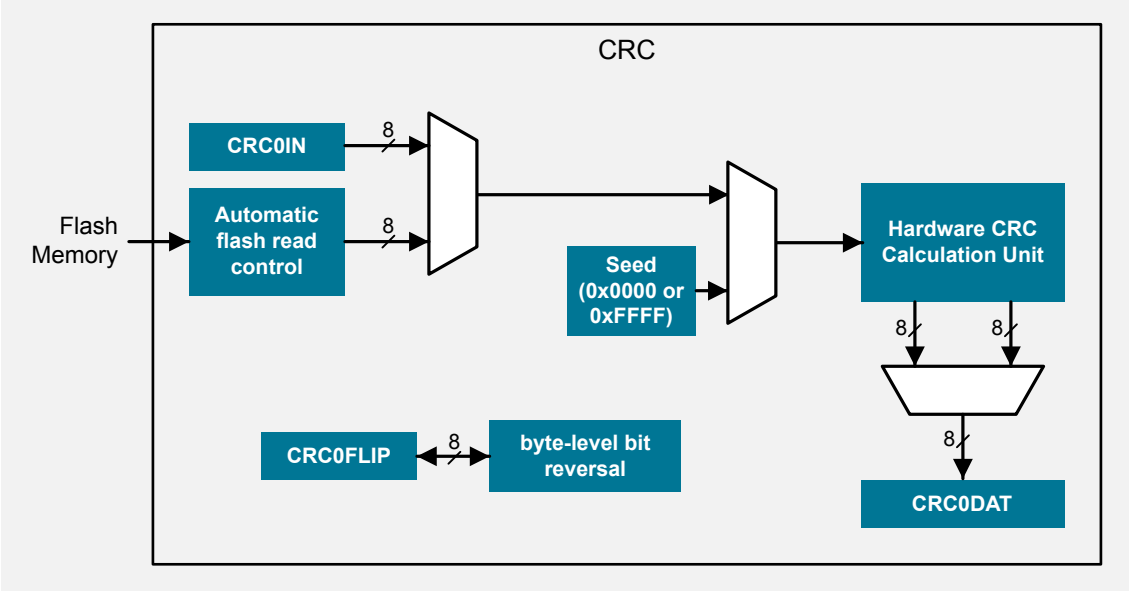


Figure 14.1. CRC Functional Block Diagram

14.2 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

14.3 Functional Description

14.3.1 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 14.1. Example 16-bit CRC Outputs

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

14.3.2 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 sream:

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.

14.3.3 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.

14.3.4 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.

14.4 CRC0 Control Registers

14.4.1 CRC0CN0: CRC0 Control 0

Bit	7	6	5	4	3	2	1	0
Name	Reserved				CRCINIT	CRCVAL	Reserved	CRCPNT
Access	R				RW	RW	R	RW
Reset	0x1				0	0	0	0
SFR Page = 0x0, 0x20; SFR Address: 0xCE								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	CRCINIT	0	RW	**CRC Initialization Enable.** Writing a 1 to this bit initializes the entire CRC result based on CRCVAL.
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.

14.4.2 CRC0IN: CRC0 Data Input

Bit	7	6	5	4	3	2	1	0
Name	CRC0IN							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x20; SFR Address: 0xDD								

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.

14.4.3 CRC0DAT: CRC0 Data Output

Bit	7	6	5	4	3	2	1	0
Name	CRC0DAT							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 0x20; SFR Address: 0xDE								

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.				

14.4.4 CRC0ST: CRC0 Automatic Flash Sector Start

Bit	7	6	5	4	3	2	1	0
Name	CRCST							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 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.

14.4.5 CRC0CNT: CRC0 Automatic Flash Sector Count

Bit	7	6	5	4	3	2	1	0
Name	CRCCNT							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 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.

14.4.6 CRC0FLIP: CRC0 Bit Flip

Bit	7	6	5	4	3	2	1	0
Name	CRC0FLIP							
Access	RW							
Reset	0x00							
SFR Page = 0x0, 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.

14.4.7 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 = 0x0, 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.*		

15. I2C Slave (I2CSLAVE0)

15.1 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.

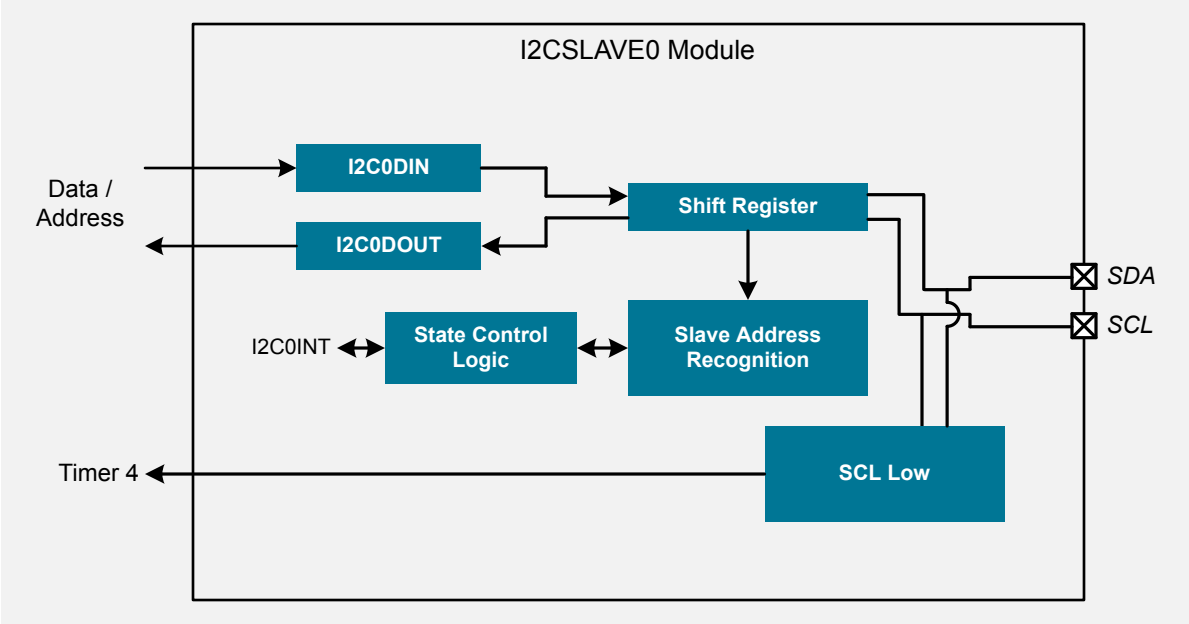


Figure 15.1. I2CSLAVE0 Block Diagram

15.2 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

15.3 Functional Description

15.3.1 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.

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 I2C0CNTL to enable automatic ACK response.
2. Write the slave address to I2C0SLAD.
3. Set the I2C0SEL bit in I2C0CNTL to 1 to enable the SCL and SDA pins.
4. Set the I2C0EN bit in I2C0CNTL to 1 to enable the I2CSLAVE0 peripheral.

15.3.2 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.

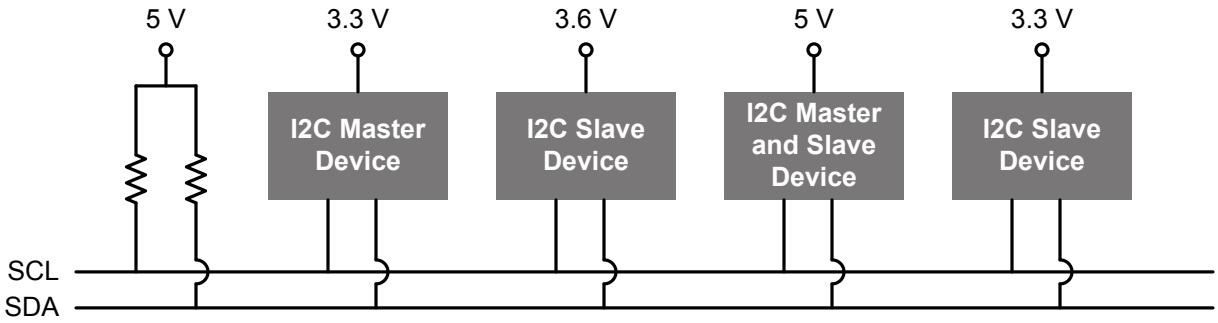


Figure 15.2. Typical I2C System Connection

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 15.3 I2C Transaction on page 162). 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 15.3 I2C Transaction on page 162 illustrates a typical I2C transaction.

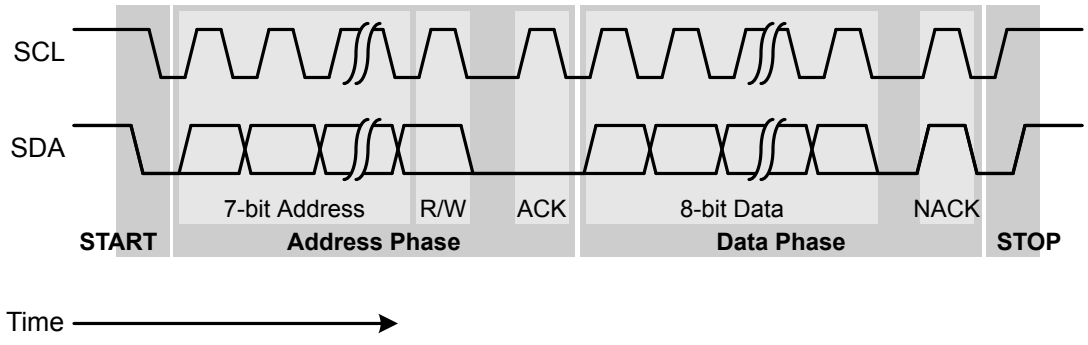


Figure 15.3. I2C Transaction

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.

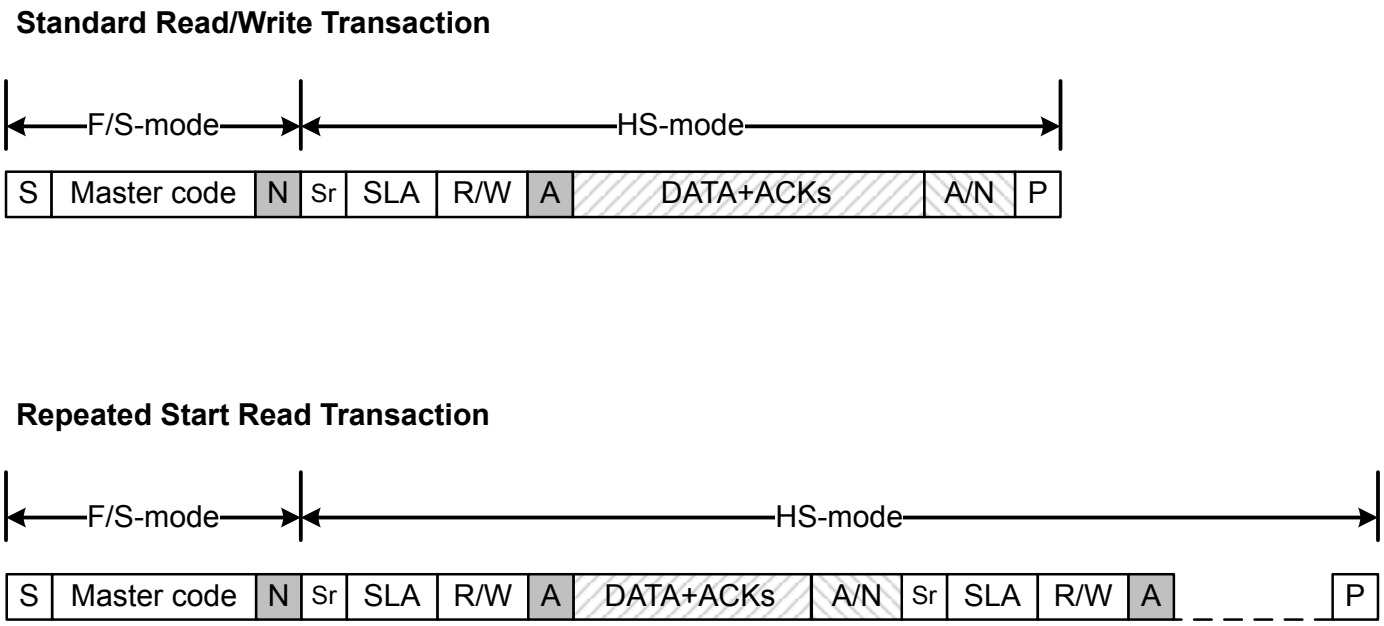


Figure 15.4. Fast-Mode to High-Speed Mode Transition

15.3.3 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 15.6 I2C Write Flow Diagram with the I2C Slave Peripheral on page 166.
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 15.5 Example I2C Write Sequence with the I2C Slave Peripheral on page 165 demonstrates an example sequence, including a repeated start, and Figure 15.6 I2C Write Flow Diagram with the I2C Slave Peripheral on page 166 describes the I2C Write sequence and firmware actions in each interrupt.

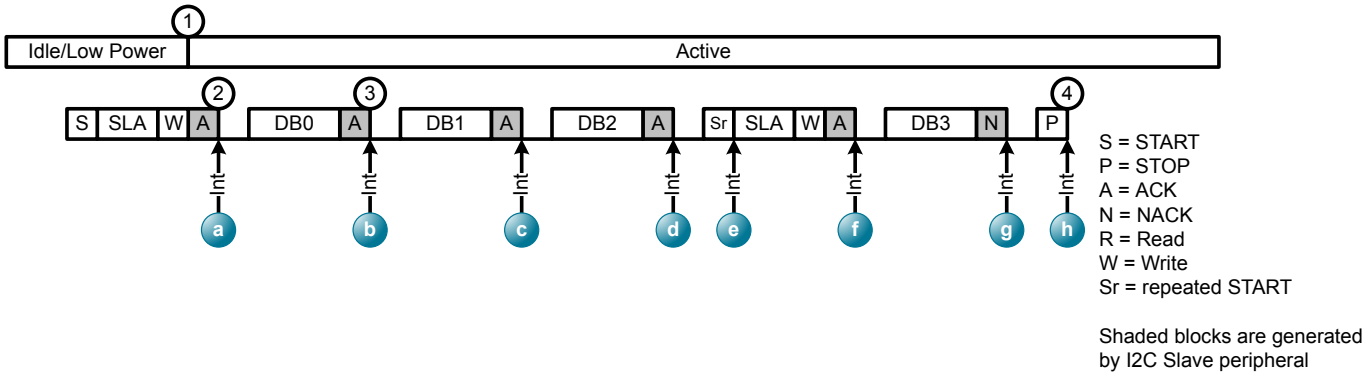


Figure 15.5. Example I2C Write Sequence with the I2C Slave Peripheral

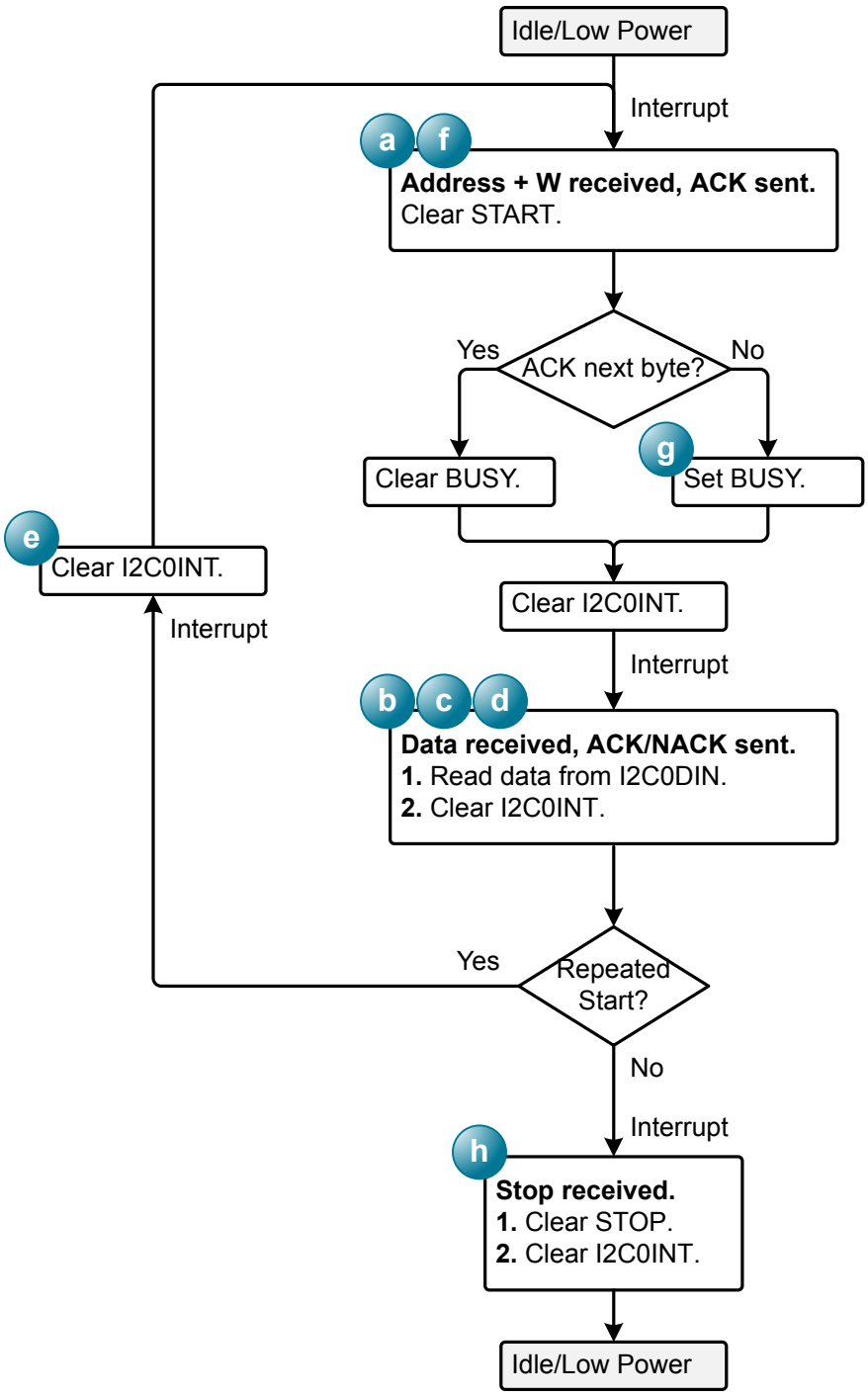


Figure 15.6. I2C Write Flow Diagram with the I2C Slave Peripheral

Note: Firmware can leave the BUSY bit as 0 in step F in the Figure 15.5 Example I2C Write Sequence with the I2C Slave Peripheral on page 165 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 15.8 I2C Read Flow Diagram with the I2C Slave Peripheral on page 168.
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 15.7 Example I2C Read Sequence with the I2C Slave Peripheral on page 167 demonstrates an example sequence, including a repeated start, and Figure 15.8 I2C Read Flow Diagram with the I2C Slave Peripheral on page 168 describes the I2C Read sequence and firmware actions in each interrupt.

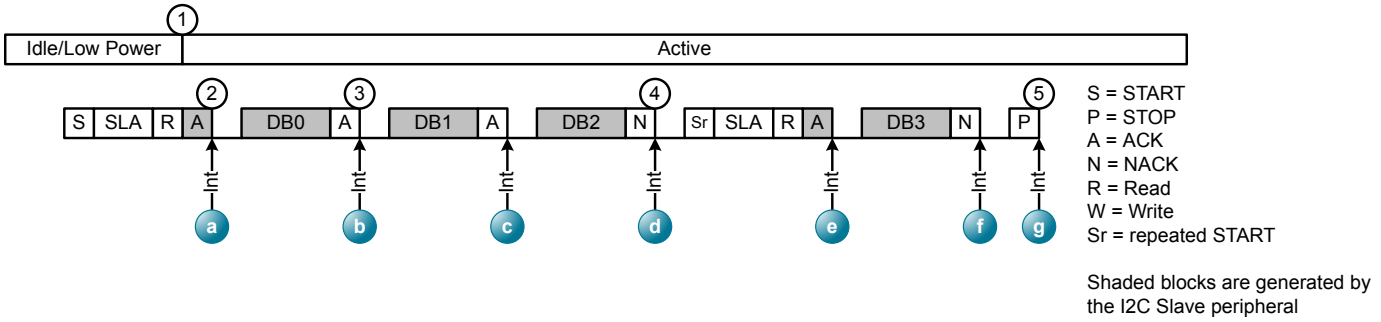


Figure 15.7. Example I2C Read Sequence with the I2C Slave Peripheral

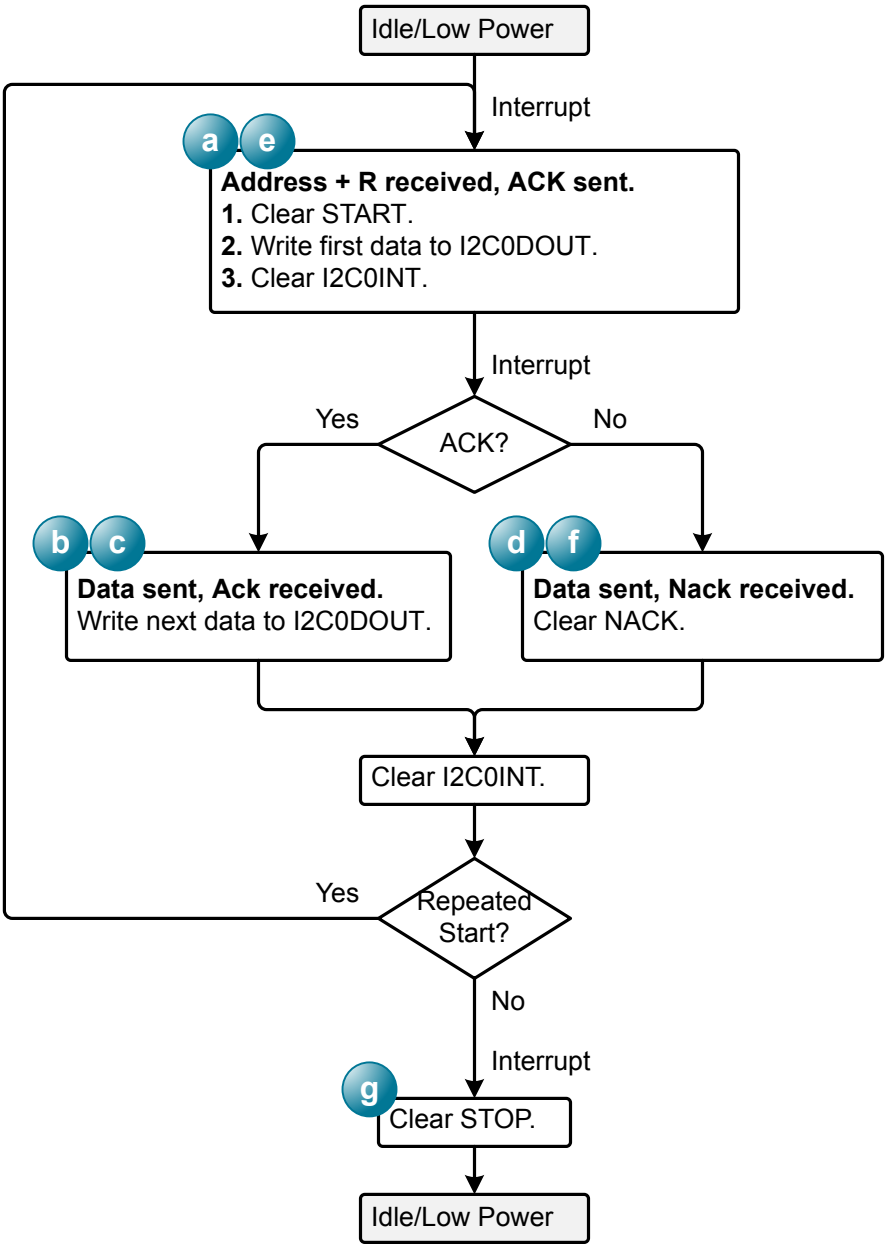


Figure 15.8. I2C Read Flow Diagram with the I2C Slave Peripheral

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.

15.3.4 Status Decoding

The current I2C status can be easily decoded using the I2C0STAT register. Table 15.1 I2C Status Decoding on page 169 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.

Table 15.1. 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 cclear 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	

15.4 I2C0 Slave Control Registers

15.4.1 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.

15.4.2 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.

15.4.3 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 defines the I2C0 Slave Address for automatic hardware acknowledgement. When the received I2C address matches this field, hardware sets the I2C0INT bit in the I2C0STAT register.

15.4.4 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 and the 8th bit was a WRITE bit (0). 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, 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 and the 8th bit was a READ bit (1). This bit will set the I2C0INT bit and generate an interrupt, if enabled. Software must clear this bit.

15.4.5 I2C0CN0: I2C0 Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved		PINDRV	PINMD	TIMEOUT	PRELOAD	I2C0EN	BUSY
Access	R		RW	RW	RW	RW	RW	RW
Reset	0x0		0	0	0	1	0	1
SFR Page = 0x20; SFR Address: 0xBA								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
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: 0 – LOW_DRIVE – SDA and SCL will use low drive strength. Value: 1 – HIGH_DRIVE – SDA and SCL will use high drive strength.
4	PINMD	0	RW	**Pin Mode Enable.** Value: 0 – GPIO_MODE – Set the I2C0 Slave pins in GPIO mode. Value: 1 – I2C_MODE – Set the I2C0 Slave pins in I2C mode.
3	TIMEOUT	0	RW	**SCL Low Timeout Enable.** When this bit is set, 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: 0 – DISABLED – Disable I2C SCL low timeout detection using Timer 4. Value: 1 – ENABLED – Enable I2C SCL low timeout detection using Timer 4.
2	PRELOAD	1	RW	**Preload Disable.** Value: 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. Value: 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: 0 – DISABLED – Disable the I2C0 Slave module. Value: 1 – ENABLED – Enable the I2C0 Slave module.

Bit	Name	Reset	Access	Description
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.

15.4.6 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.

15.4.7 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.*		

15.4.8 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.

16. Programmable Counter Array (PCA0)

16.1 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.

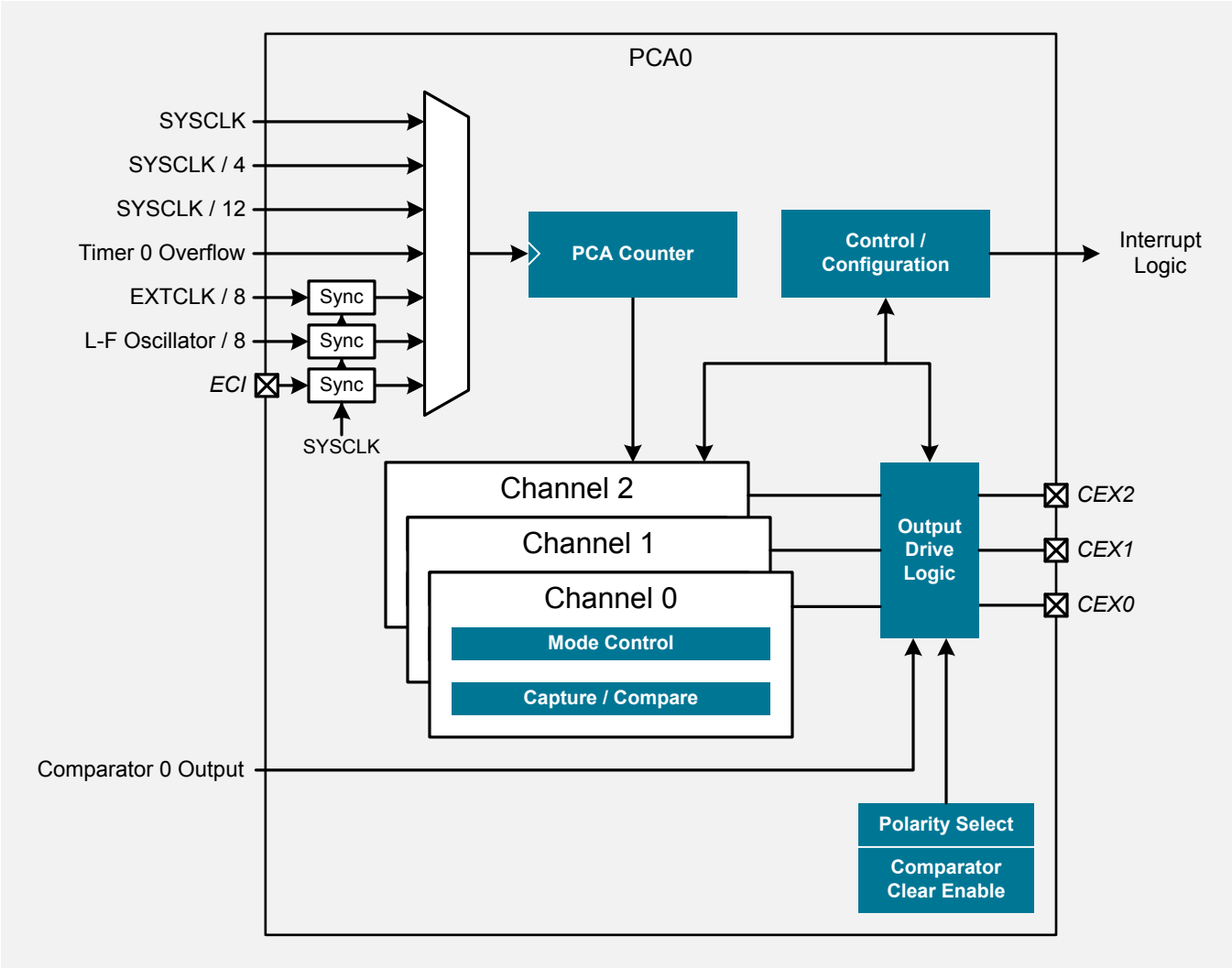


Figure 16.1. PCA Block Diagram

16.2 Features

- 16-bit time base
- Programmable clock divisor and clock source selection
- Up to three 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

16.3 Functional Description

16.3.1 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 16.1. 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.

16.3.2 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.

16.3.3 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 16.2 PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules on page 179 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 16.2. 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.

16.3.3.1 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.

16.3.4 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.

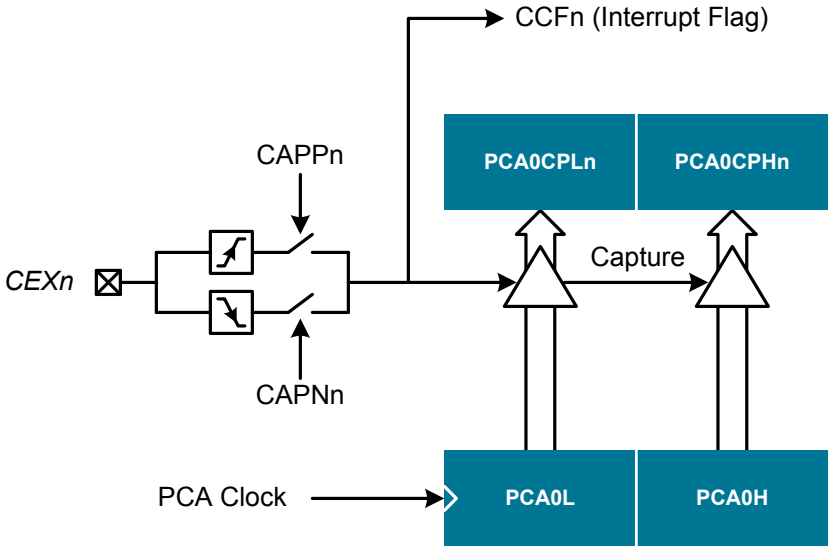


Figure 16.2. PCA Capture Mode Diagram

Note: The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

16.3.5 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.

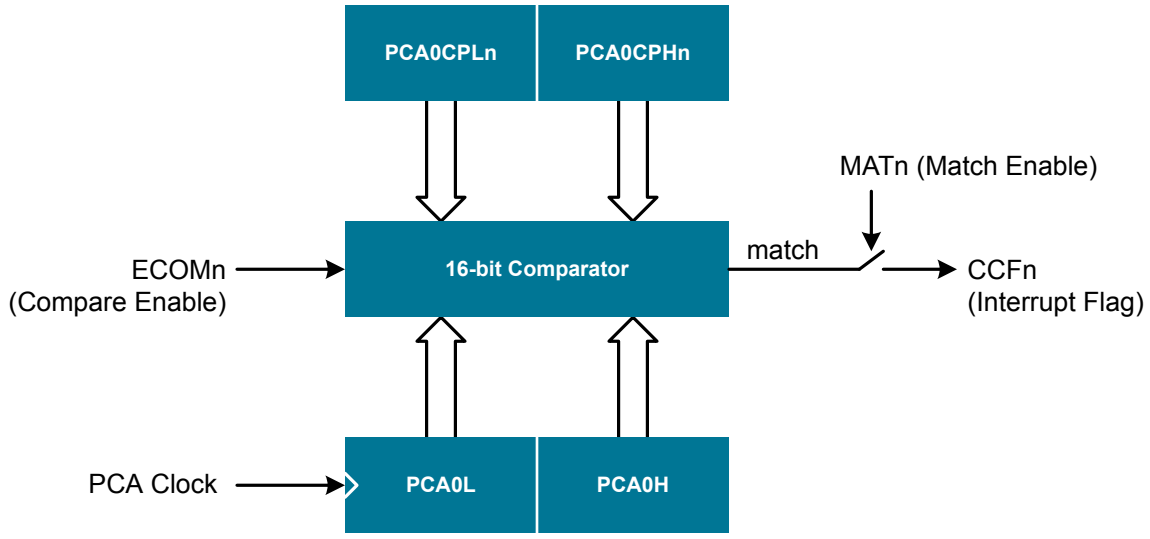


Figure 16.3. PCA Software Timer Mode Diagram

16.3.6 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 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.

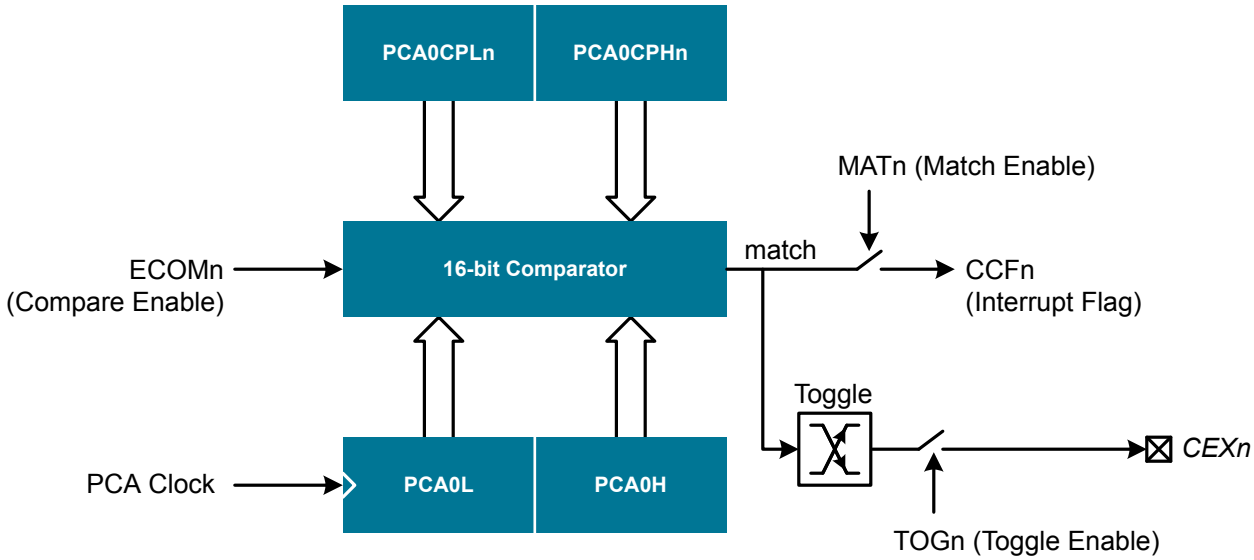


Figure 16.4. PCA High-Speed Output Mode Diagram

16.3.7 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, n 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.

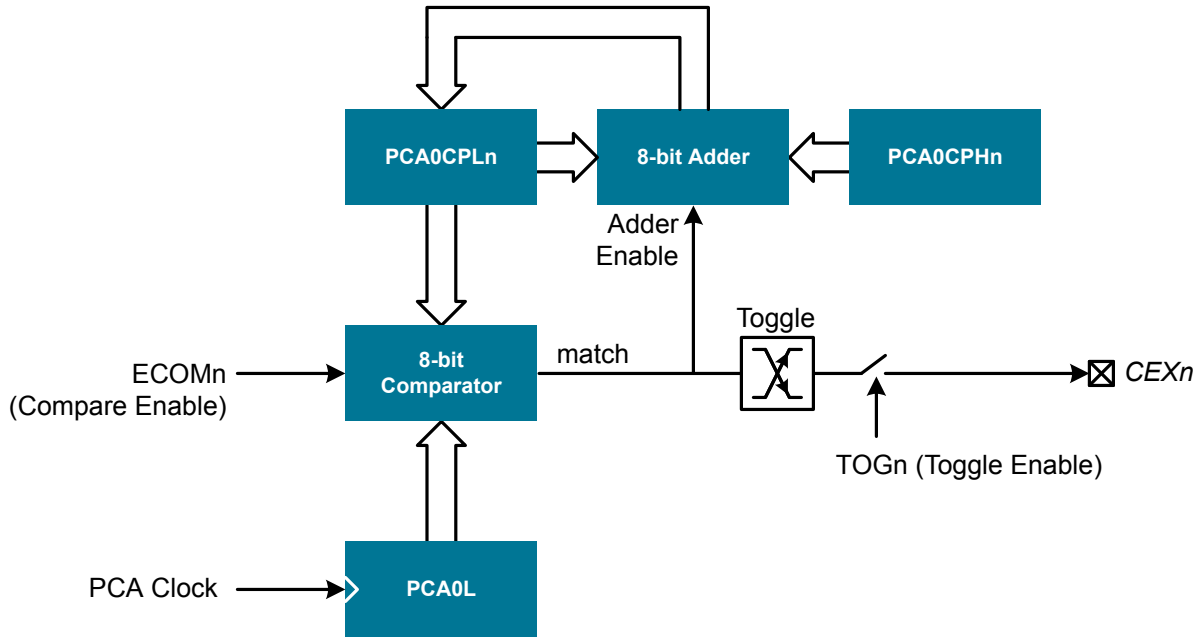


Figure 16.5. PCA Frequency Output Mode

16.3.8 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.

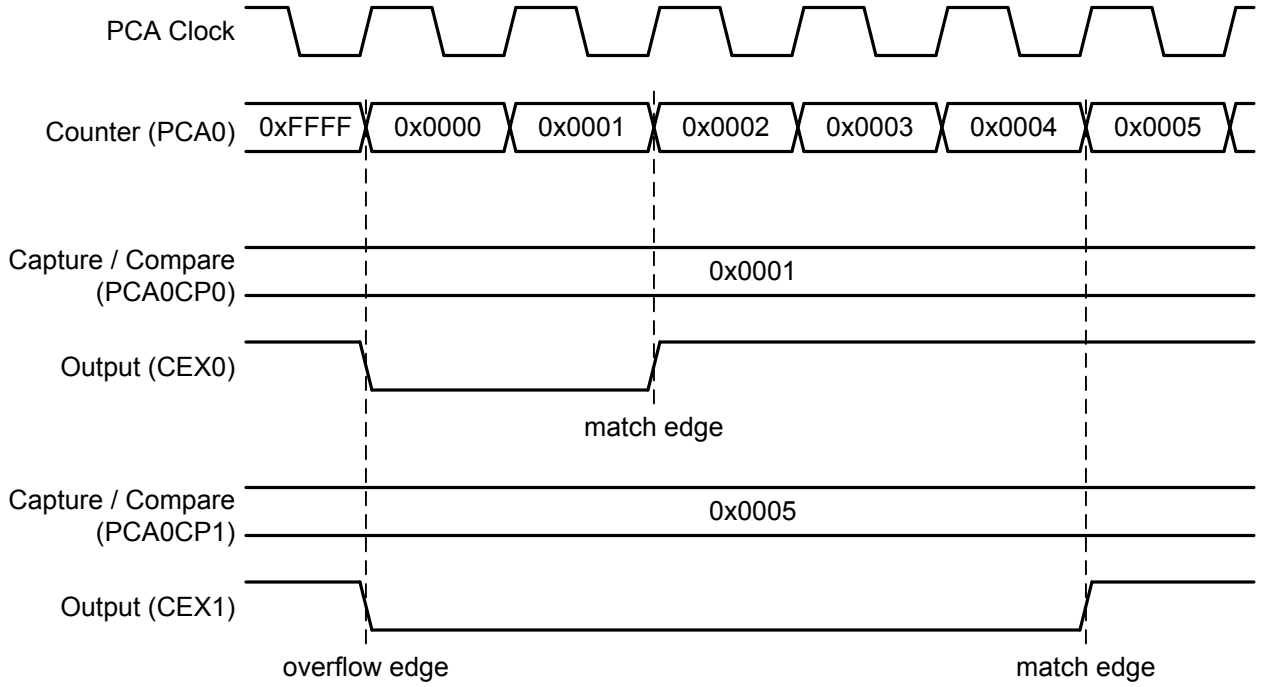


Figure 16.6. Edge-Aligned PWM Timing

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 16.7 N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution) on page 184describes the duty cycle when CEXnPOL in the PCA0POL regsiter is cleared to 0. Figure 16.8 N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution) on page 185 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 16.7. N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution)

$$\text{Duty Cycle} = \frac{\text{PCA0CPn}}{2^N}$$

Figure 16.8. 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.

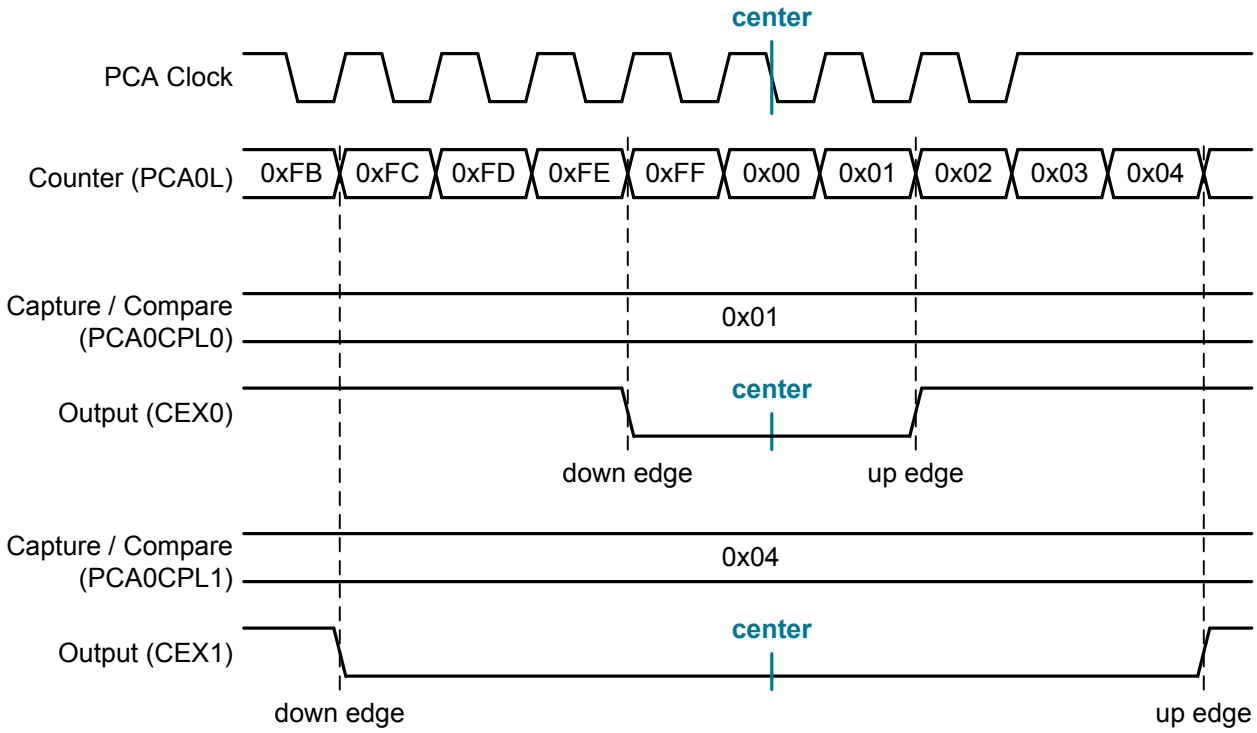


Figure 16.9. Center-Aligned PWM Timing

Figure 16.10 N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution) on page 187 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is cleared to 0. Figure 16.11 N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution) on page 187 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 16.10. 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 16.11. N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution)

16.3.8.1 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.

16.3.8.2 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.

16.3.8.3 Comparator Clear Function

In 8/9/10/11/16-bit PWM modes, the comparator clear function utilizes the Comparator0 output synchronized to the system clock to clear CEXn to logic low for the current PWM cycle. This comparator clear function can be enabled for each PWM channel by setting the CPCEn bits to 1 in the PCA0CLR SFR. When the comparator clear function is disabled, CEXn is unaffected.

The asynchronous Comparator 0 output is logic high when the voltage of CP0+ is greater than CP0– and logic low when the voltage of CP0+ is less than CP0–. The polarity of the Comparator 0 output is used to clear CEXn as follows: when CPCPOL = 0, CEXn is cleared on the falling edge of the Comparator0 output.

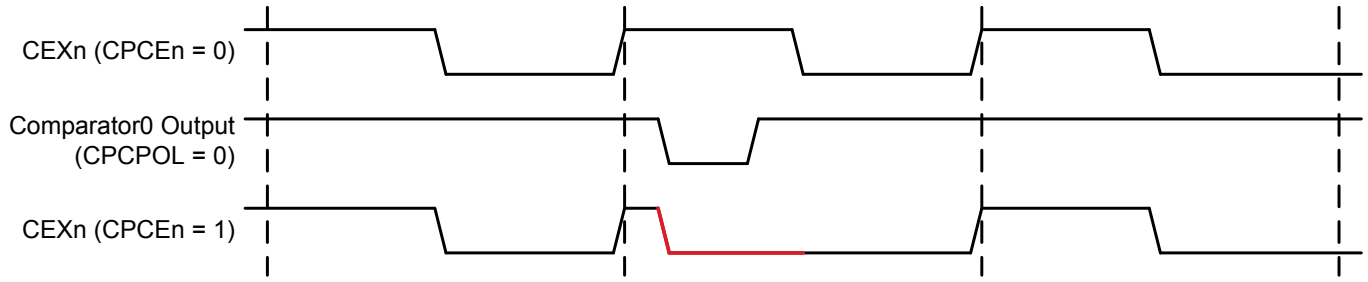


Figure 16.12. CEXn with CPCEn = 1, CPCPOL = 0

When CPCPOL = 1, CEXn is cleared on the rising edge of the Comparator0 output.

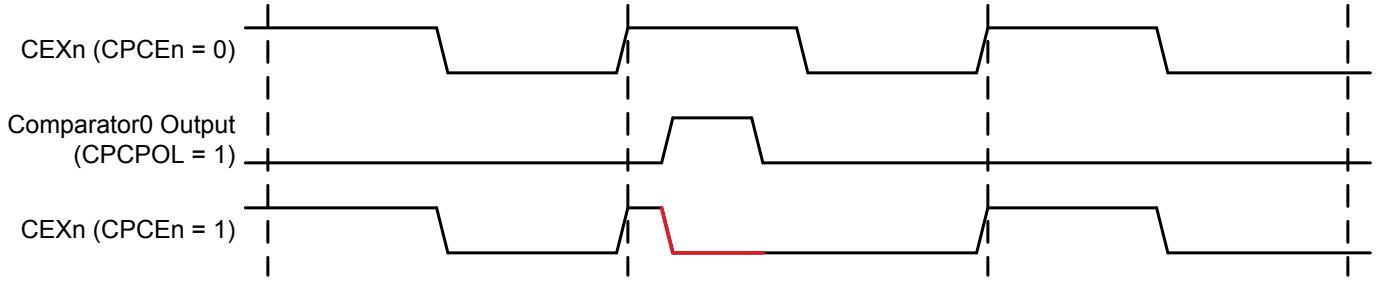


Figure 16.13. CEXn with CPCEn = 1, CPCPOL = 1

In the PWM cycle following the current cycle, should the Comparator 0 output remain logic low when CPCPOL = 0 or logic high when CPCPOL = 1, CEXn will continue to be cleared.

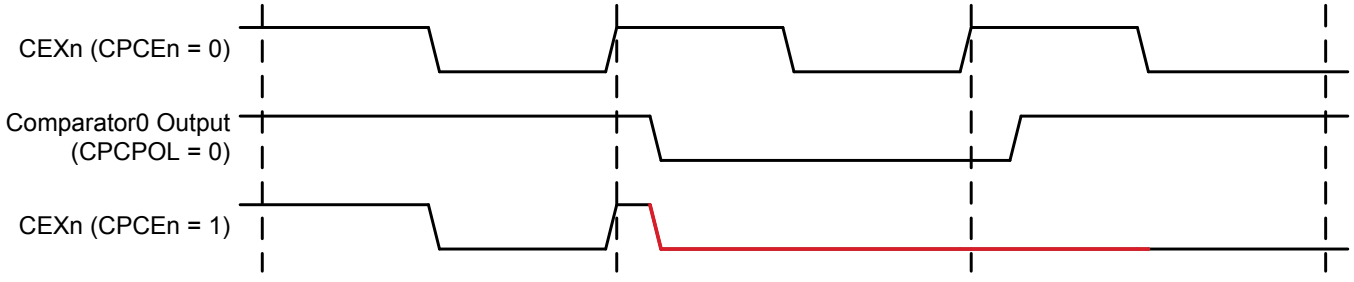


Figure 16.14. CEXn with CPCEn = 1, CPCPOL = 0

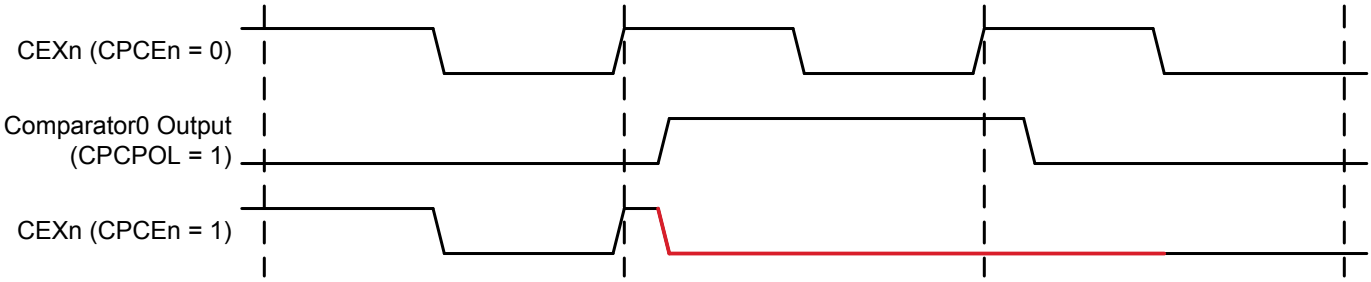


Figure 16.15. CEXn with CPCEn = 1, CPCPOL = 1

16.4 PCA0 Control Registers

16.4.1 PCA0CN0: PCA Control

Bit	7	6	5	4	3	2	1	0
Name	CF	CR	Reserved			CCF2	CCF1	CCF0
Access	RW	RW	R			RW	RW	RW
Reset	0	0	0x0			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 – Description: Stop the PCA Counter/Timer. Value: 1 – Name: RUN – Description: Start the PCA Counter/Timer running.
5:3	*Reserved*	*Must write reset value.*		
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.

16.4.2 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.

16.4.3 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.

16.4.4 PCA0CLR: PCA Comparator Clear Control

Bit	7	6	5	4	3	2	1	0
Name	CPCPOL	Reserved				CPCE2	CPCE1	CPCE0
Access	RW	R				RW	RW	RW
Reset	0	0x0				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.

6:3	*Reserved*	*Must write reset value.*		
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.

16.4.5 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.

16.4.6 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.

16.4.7 PCA0POL: PCA Output Polarity

Bit	7	6	5	4	3	2	1	0
Name	Reserved					CEX2POL	CEX1POL	CEX0POL
Access	R					RW	RW	RW
Reset	0x00					0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0x96								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
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.

Bit	Name	Reset	Access	Description
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.

16.4.8 PCA0CENT: PCA Center Alignment Enable

Bit	7	6	5	4	3	2	1	0
Name	Reserved					CEX2CEN	CEX1CEN	CEX0CEN
Access	R					RW	RW	RW
Reset	0x00					0	0	0
SFR Page = 0x0, 0x10; SFR Address: 0x9E								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
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	Name	Description
0	EDGE	Edge-aligned.
1	CENTER	Center-aligned.

Bit	Name	Reset	Access	Description
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	Name	Description
0	EDGE	Edge-aligned.
1	CENTER	Center-aligned.

Bit	Name	Reset	Access	Description
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.

16.4.9 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: 0 – Name: 8_BIT – Description: 8 to 11-bit PWM selected. Value: 1 – Name: 16_BIT – Description: 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: 0 – Name: DISABLED – Description: Disable CCF0 interrupts. Value: 1 – Name: ENABLED – Description: Enable a Capture/Compare Flag interrupt request when CCF0 is set.

16.4.10 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.				

16.4.11 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.				

16.4.12 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: 0 – Name: 8_BIT – Description: 8 to 11-bit PWM selected. Value: 1 – Name: 16_BIT – Description: 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: 0 – Name: DISABLED – Description: Disable CCF1 interrupts. Value: 1 – Name: ENABLED – Description: Enable a Capture/Compare Flag interrupt request when CCF1 is set.

16.4.13 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.				

16.4.14 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.				

16.4.15 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: 0 – Name: 8_BIT – Description: 8 to 11-bit PWM selected. Value: 1 – Name: 16_BIT – Description: 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: 0 – Name: DISABLED – Description: Disable CCF2 interrupts. Value: 1 – Name: ENABLED – Description: Enable a Capture/Compare Flag interrupt request when CCF2 is set.

16.4.16 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.				

16.4.17 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.				

17. Serial Peripheral Interface (SPI0)

17.1 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.

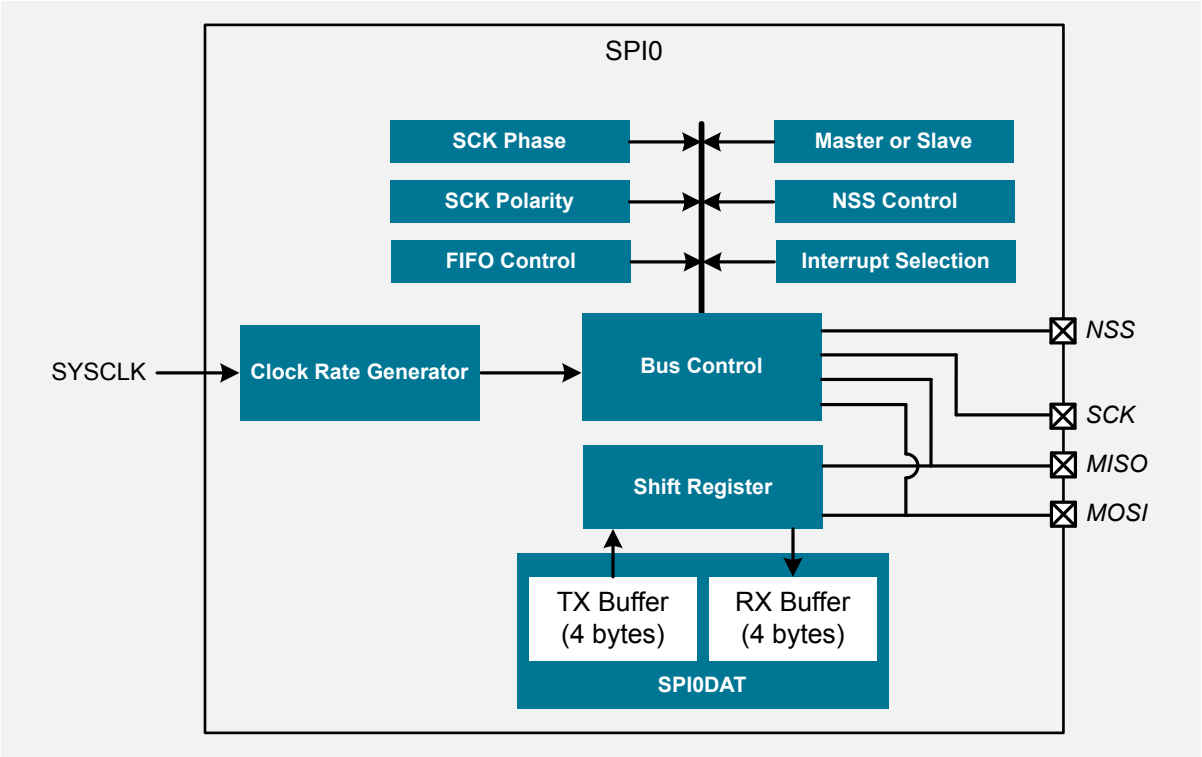


Figure 17.1. SPI Block Diagram

17.2 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).
- Four 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.

17.3 Functional Description

17.3.1 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.

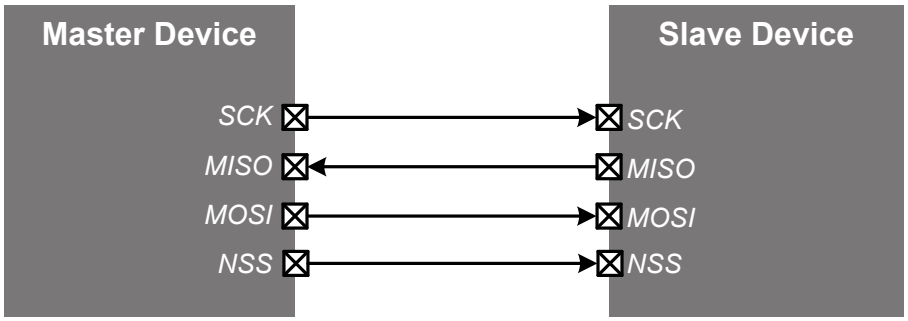


Figure 17.2. 4-Wire Connection Diagram

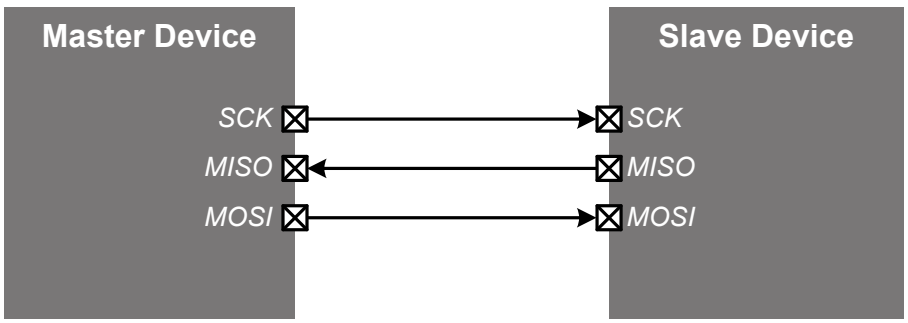


Figure 17.3. 3-Wire Connection Diagram

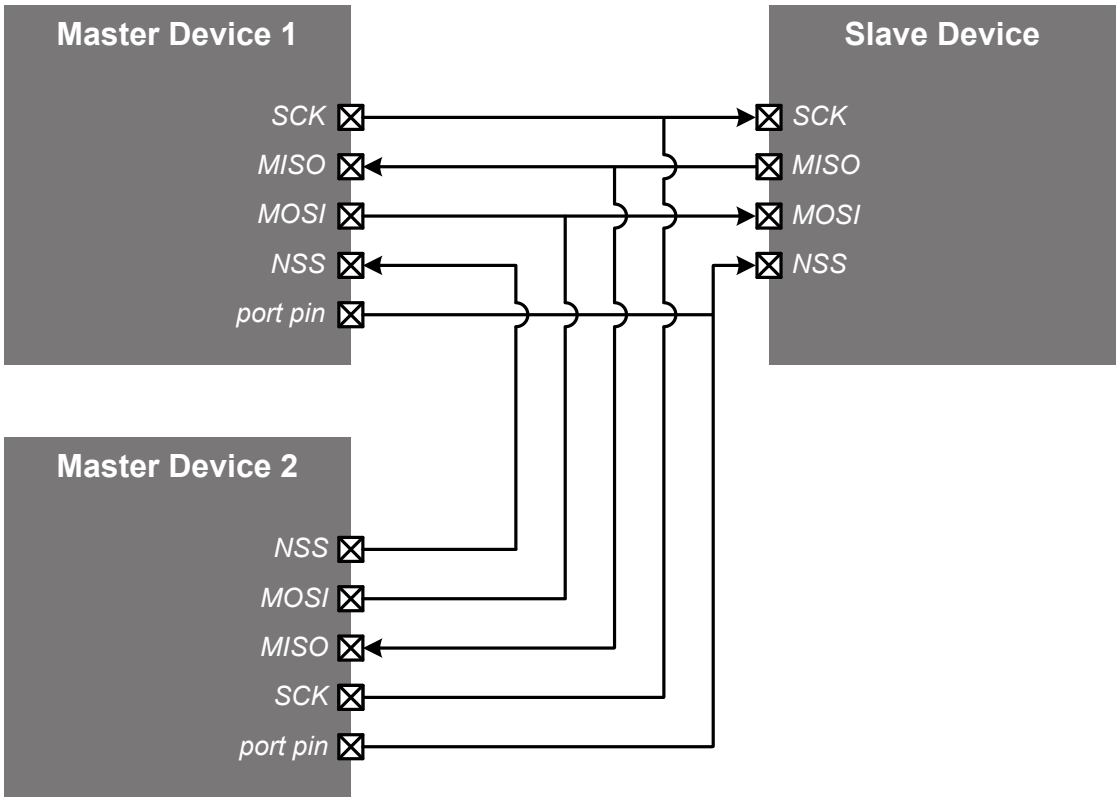


Figure 17.4. Multi-Master Connection Diagram

17.3.2 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.

17.3.3 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.

17.3.4 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.

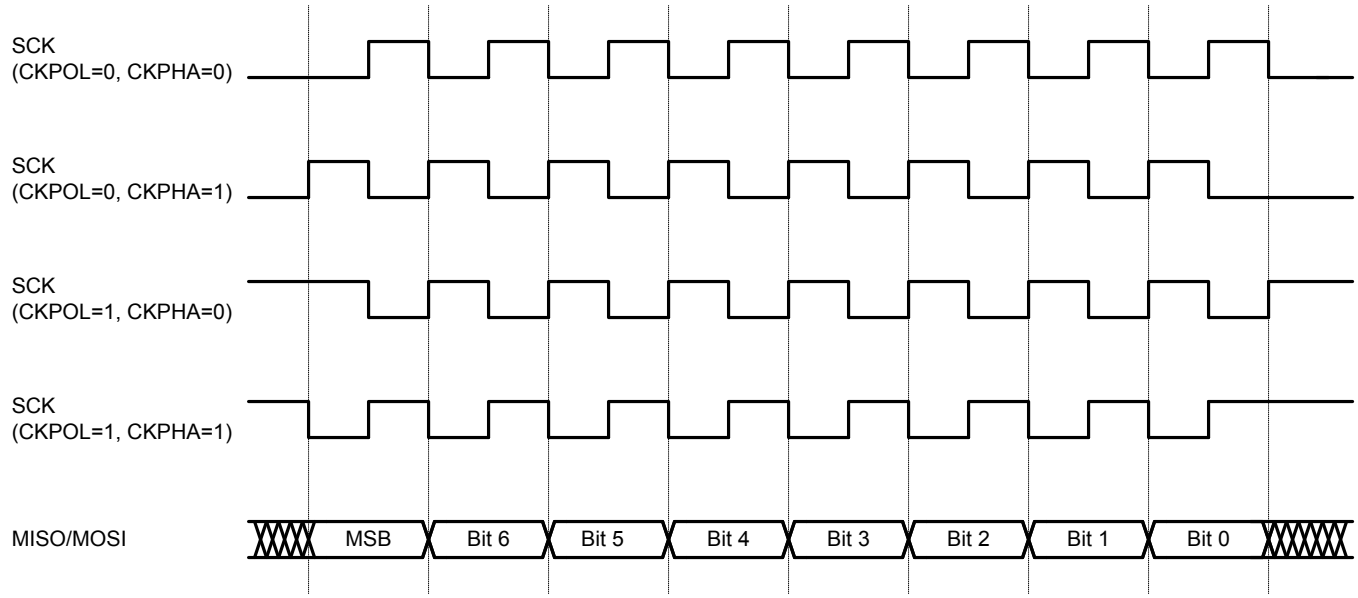


Figure 17.5. Master Mode Data/Clock Timing

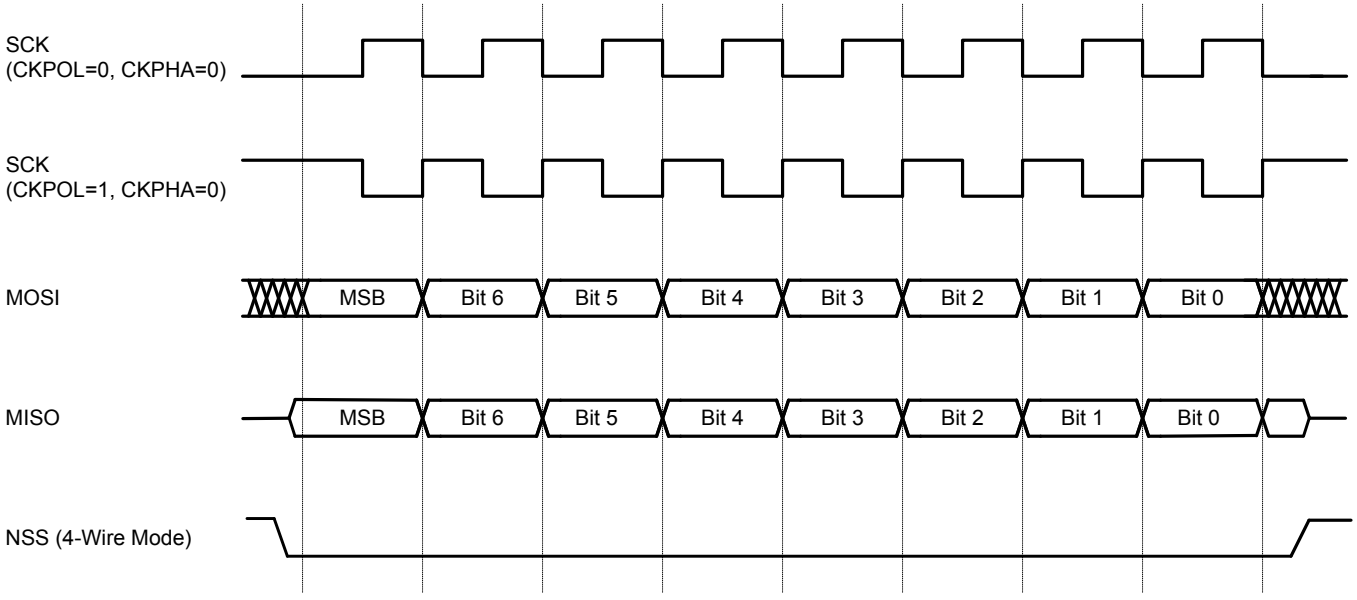


Figure 17.6. Slave Mode Data/Clock Timing (CKPHA = 0)

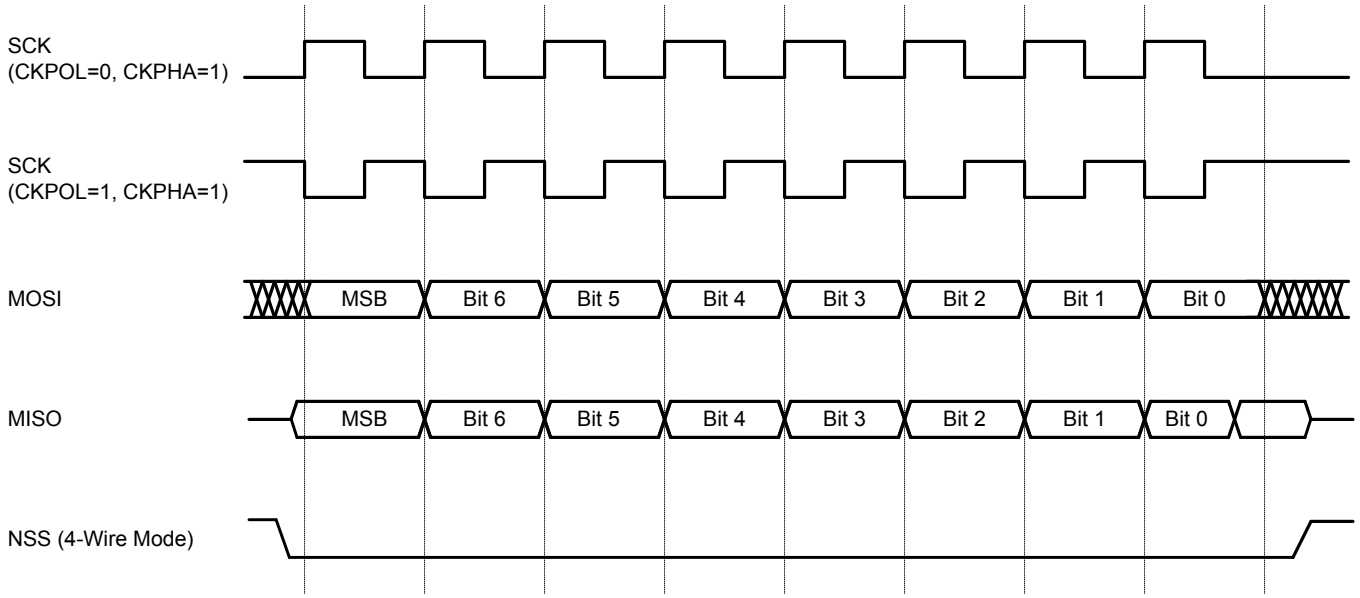


Figure 17.7. Slave Mode Data/Clock Timing (CKPHA = 1)

17.3.5 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.

17.3.6 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 17.3.5 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

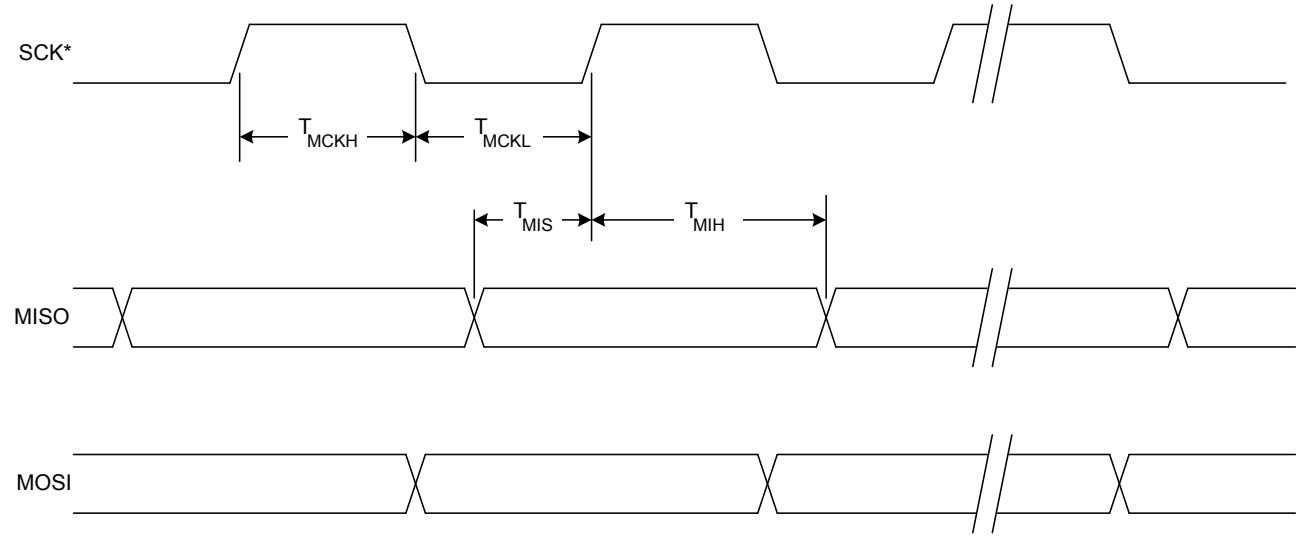
When acting as a SPI slave using RFRQ interrupts and with the RXTH field set to a value greater than 0, it is possible for the external master to write too few bytes to the device to immediately generate an interrupt. To avoid leaving lingering bytes in the receive FIFO, the slave receiver timeout feature 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 RFTH 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.

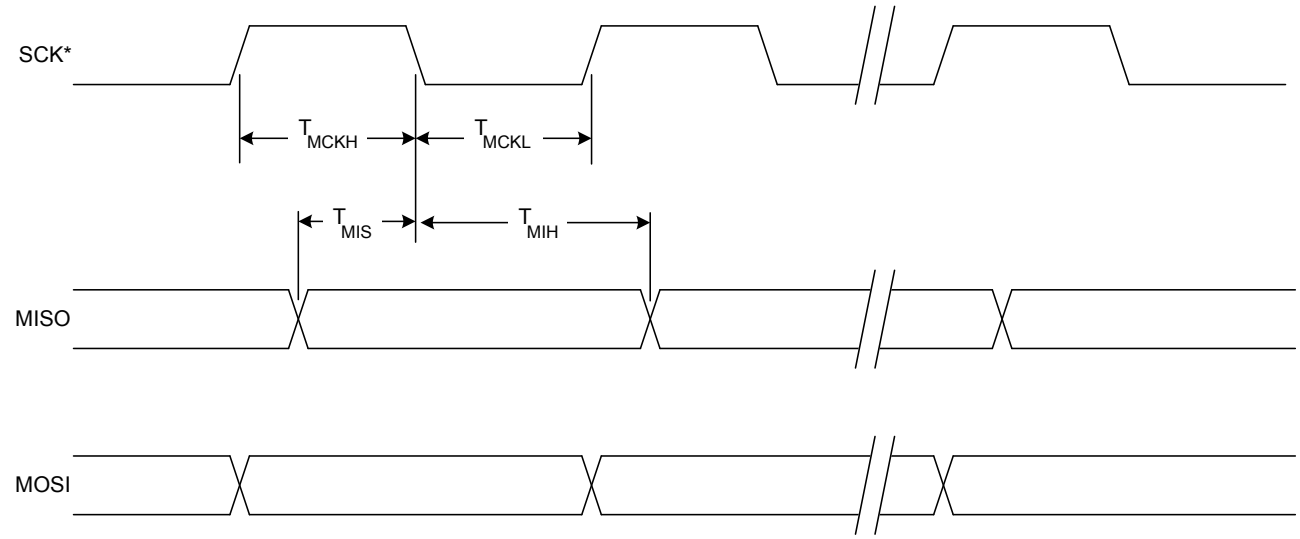
If the internal counter runs out, a SPI interrupt will be generated, allowing firmware to read any bytes remaining in the receive FIFO.

17.3.7 SPI Timing Diagrams



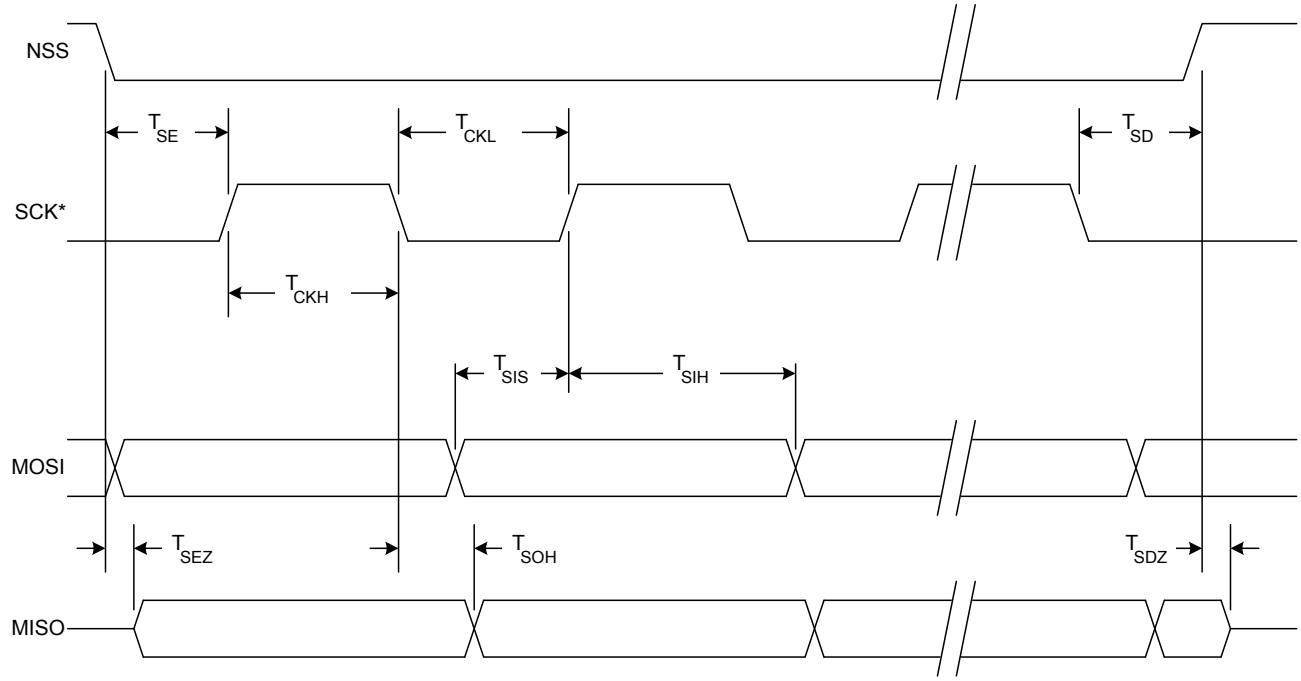
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 17.8. SPI Master Timing (CKPHA = 0)



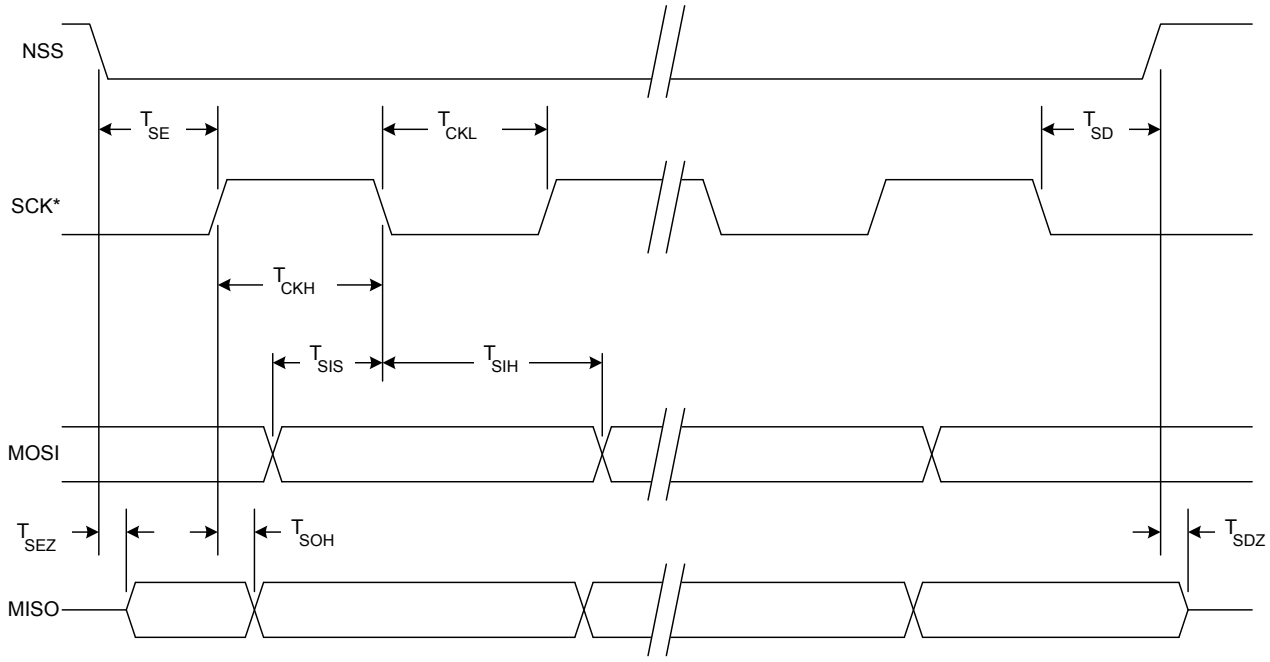
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 17.9. SPI Master Timing (CKPHA = 1)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 17.10. SPI Slave Timing (CKPHA = 0)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 17.11. SPI Slave Timing (CKPHA = 1)

Table 17.1. 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).

17.4 SPI0 Control Registers

17.4.1 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.

17.4.2 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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

17.4.3 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

17.4.4 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.

17.4.5 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.

Bit	Name	Reset	Access	Description
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).

Bit	Name	Reset	Access	Description
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.
0x2	TWO	TFRQ will be set when the TX FIFO contains two or fewer bytes.
0x3	THREE	TFRQ will be set when the TX FIFO contains three or fewer bytes.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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).

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
	0x2	TWO		RFRQ will be set if the RX FIFO contains more than two bytes.
	0x3	THREE		RFRQ will be set if the RX FIFO contains more than three bytes.

17.4.6 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.

17.4.7 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.			

18. System Management Bus / I2C (SMB0)

18.1 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.

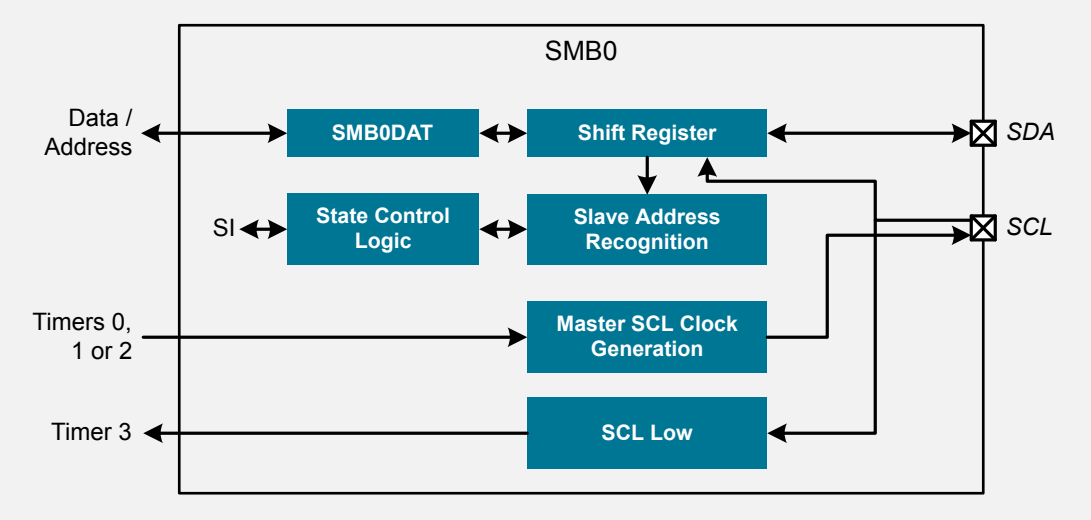


Figure 18.1. SMBus 0 Block Diagram

18.2 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 buffers to help increase throughput in faster applications

18.3 Functional Description

18.3.1 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.

18.3.2 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.

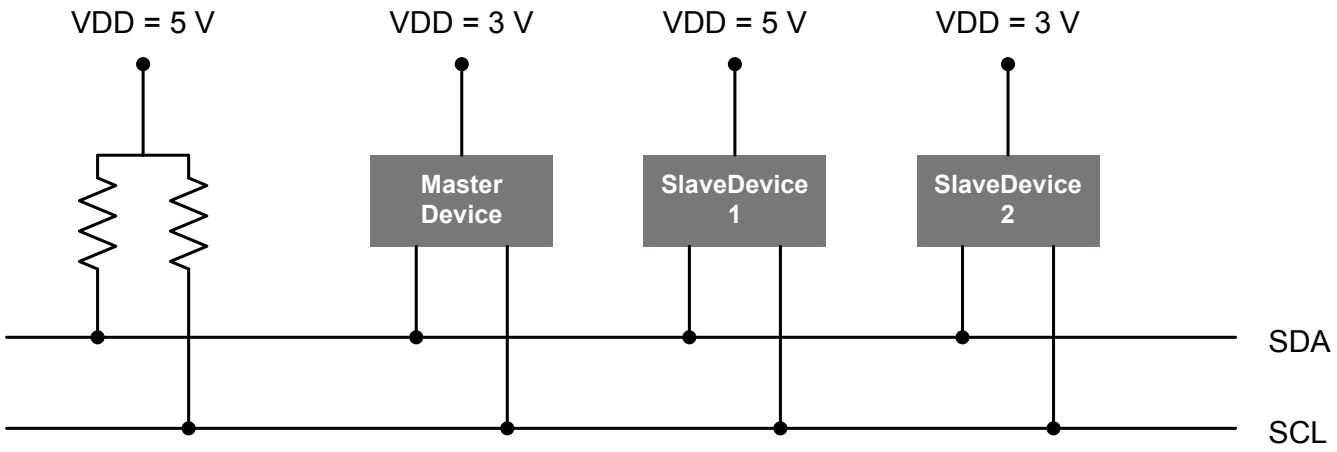


Figure 18.2. Typical SMBus System Connection

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 18.3 SMBus Transaction on page 222). 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 18.3 SMBus Transaction on page 222 illustrates a typical SMBus transaction.

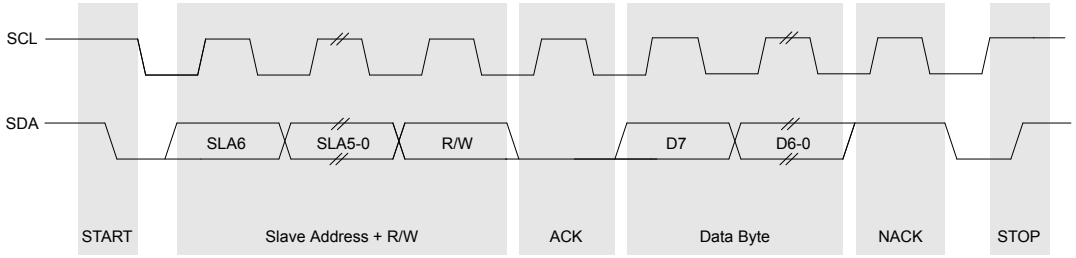


Figure 18.3. SMBus Transaction

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 222). 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.

18.3.3 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.

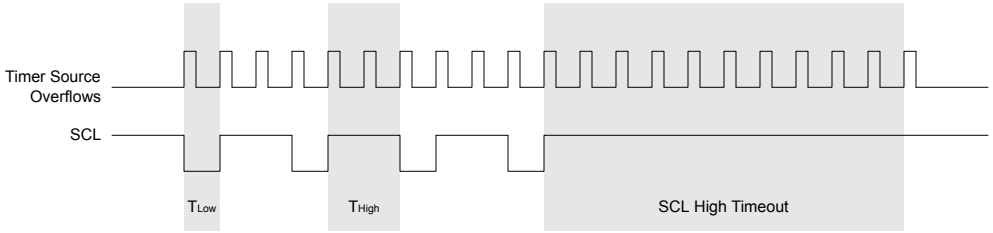


Figure 18.4. Typical SMBus SCL Generation

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 18.1. 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 SMBTC 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 SMBTC register is used to restrict the detection of a START condition under certain circumstances. In some systems where there is significant mismatch between the impedance or the capacitance on the SDA and SCL lines, it may be possible for SCL to fall after SDA during an address or data transfer. Such an event can cause a false START detection on the bus. These kind 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.

By default, if the SCL falling edge is detected after the falling edge of SDA (i.e., one SYSCLK cycle or more), the device will detect this as a START condition. The SDD field is used to increase the amount of hold time that is required between SDA and SCL falling before a START is recognized. An additional 2, 4, or 8 SYSCLKs can be added to prevent false START detection in systems where the bus conditions warrant this.

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 18.2. 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. 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.

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 18.3. 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.

18.3.4 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 18.5 Typical Master Write Sequence on page 228 shows a typical master write sequence as it appears on the bus, and Figure 18.6 Master Write Sequence State Diagram (EHACK = 1) on page 229 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.

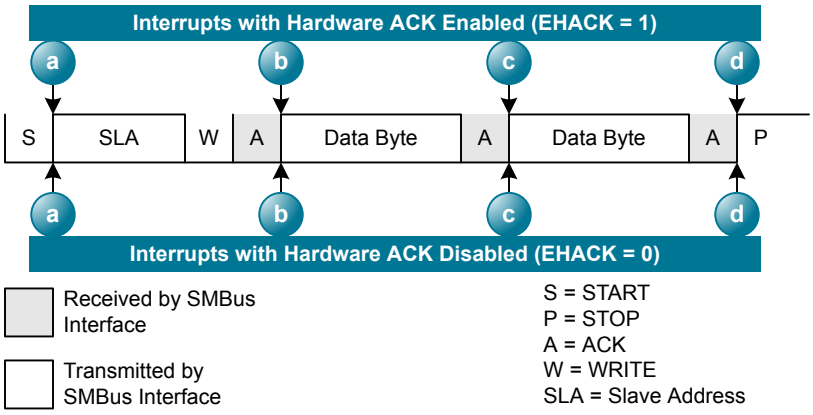


Figure 18.5. Typical Master Write Sequence

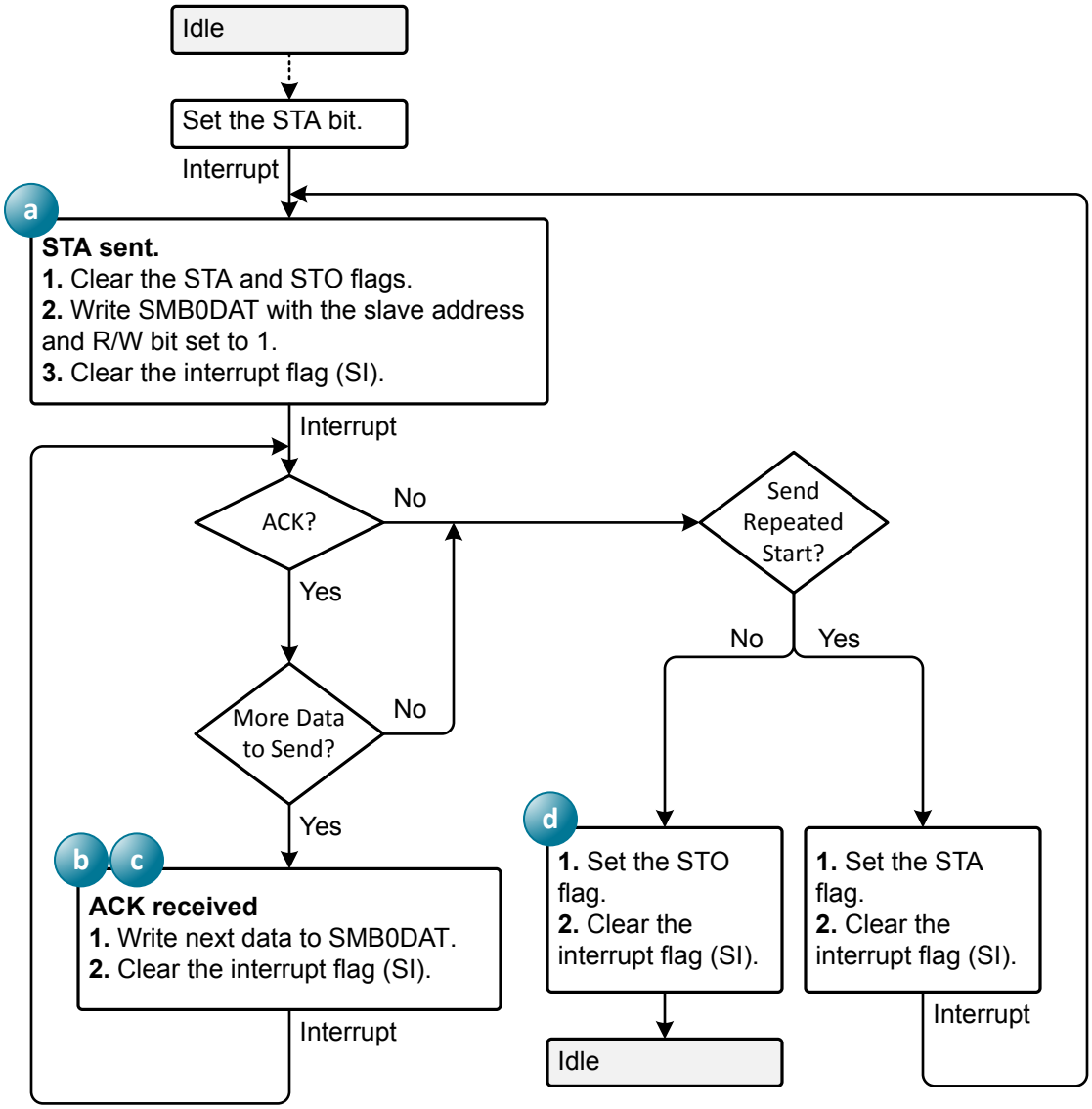


Figure 18.6. Master Write Sequence State Diagram (EHACK = 1)

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 18.7 Typical Master Read Sequence on page 230 shows a typical master read sequence as it appears on the bus, and Figure 18.8 Master Read Sequence State Diagram (EHACK = 1) on page 231 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.

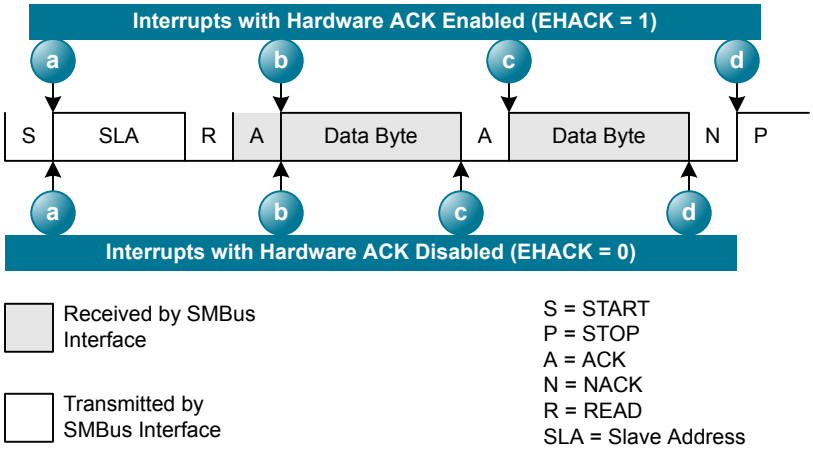


Figure 18.7. Typical Master Read Sequence

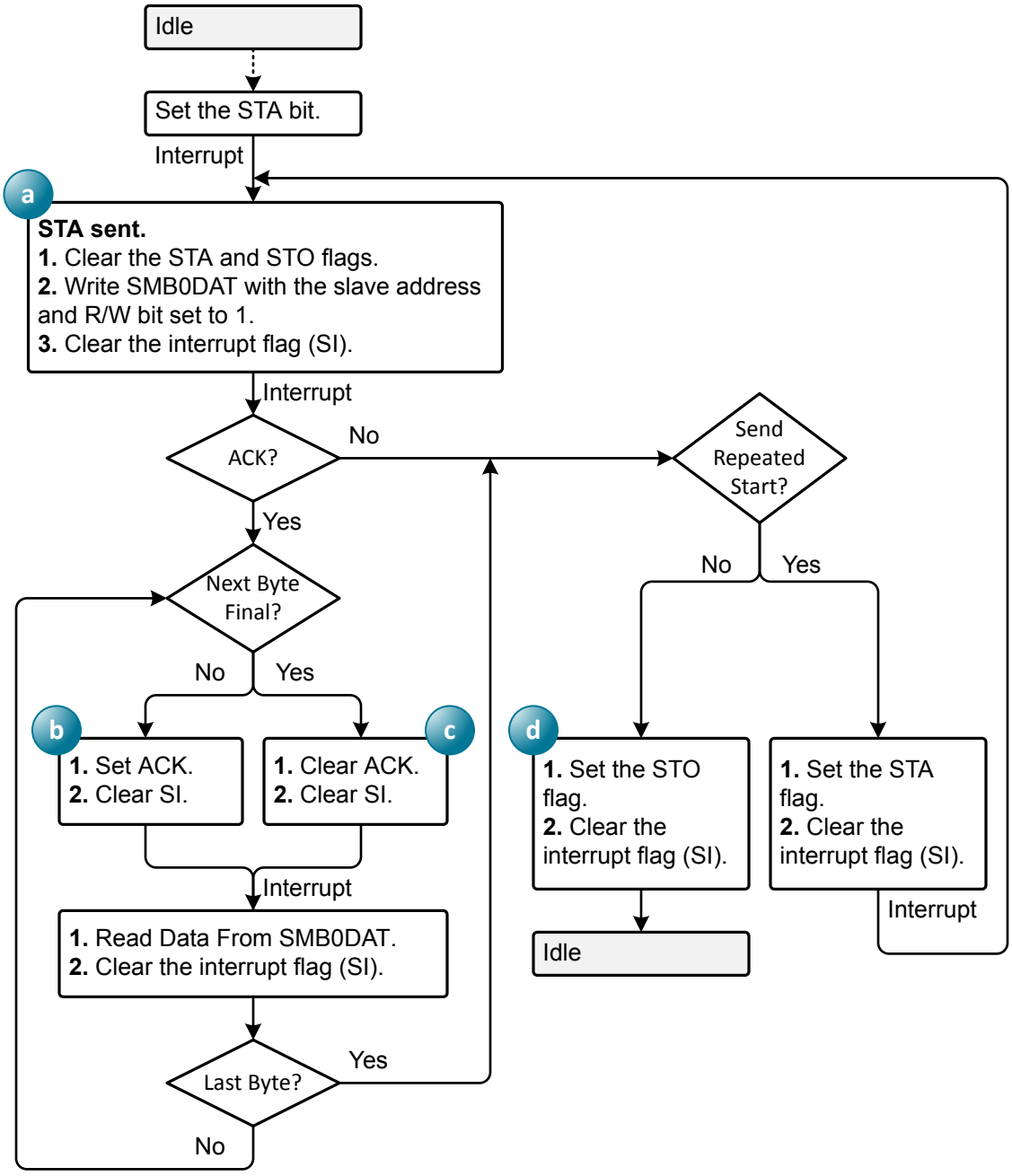


Figure 18.8. Master Read Sequence State Diagram (EHACK = 1)

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 18.9 Typical Slave Write Sequence on page 232 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 18.10 Slave State Diagram (EHACK = 1) on page 233. 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.

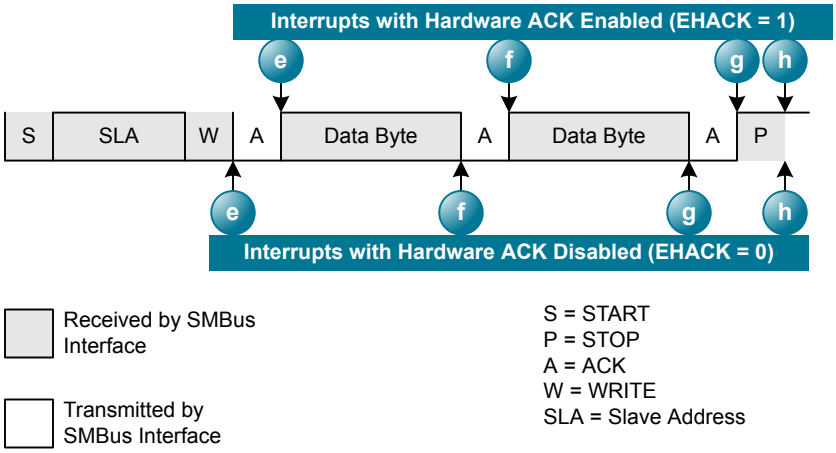


Figure 18.9. Typical Slave Write Sequence

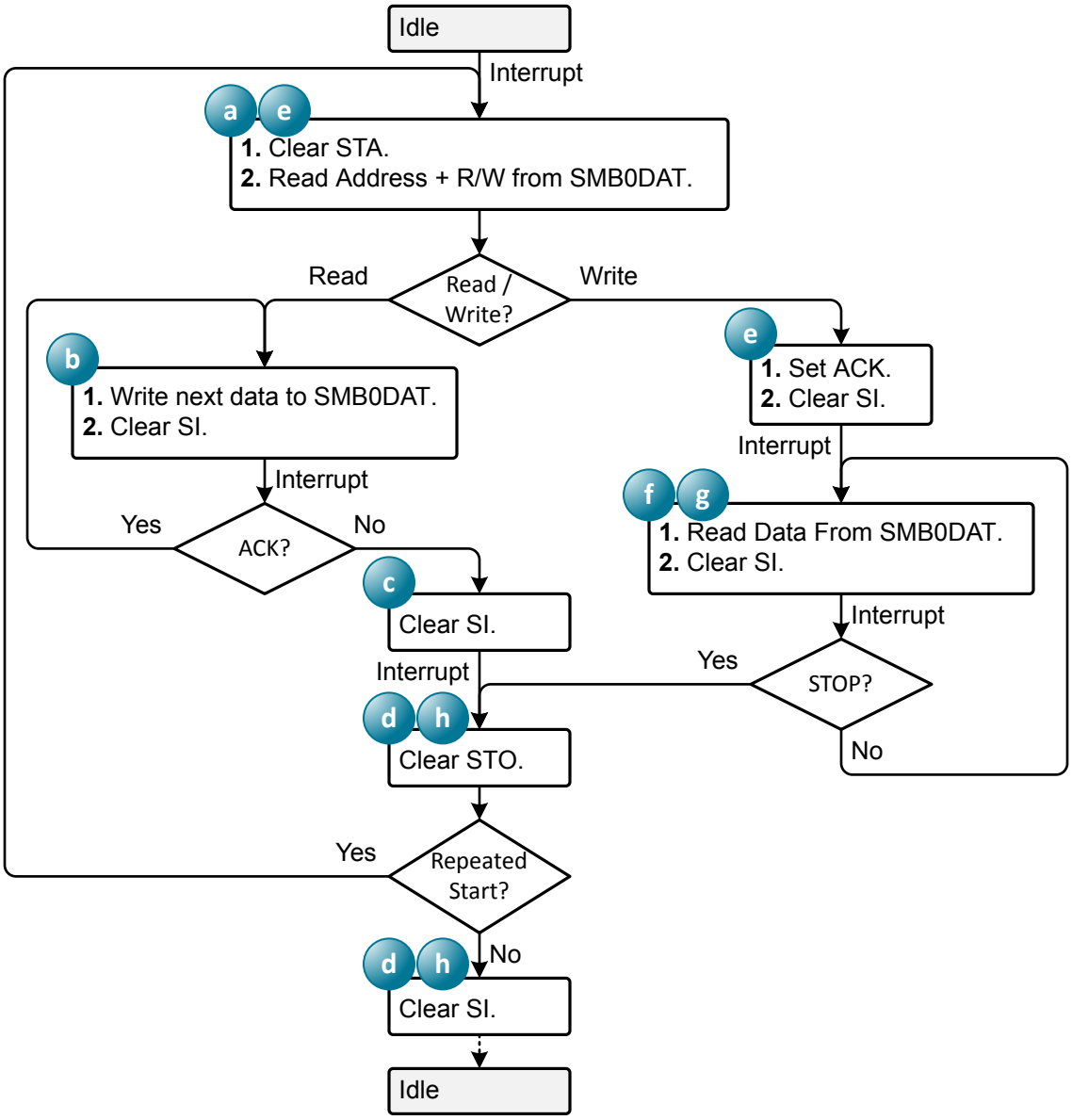


Figure 18.10. Slave State Diagram (EHACK = 1)

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 18.11 Typical Slave Read Sequence on page 234 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 18.10 Slave State Diagram (EHACK = 1) on page 233. 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.

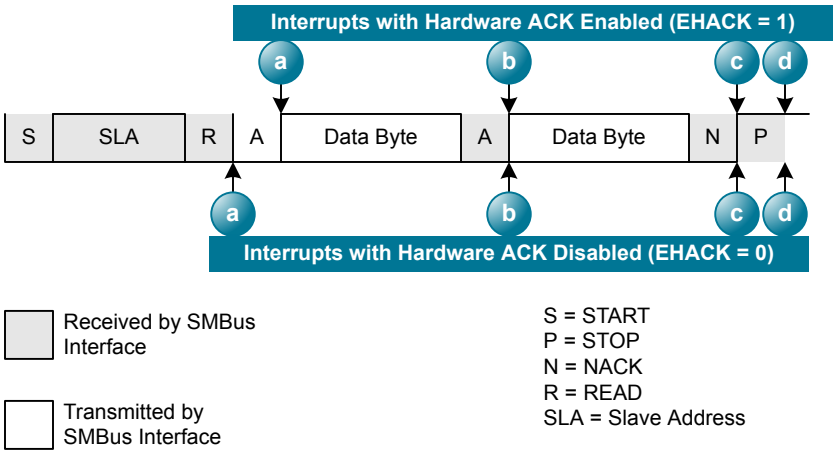


Figure 18.11. Typical Slave Read Sequence

18.4 SMB0 Control Registers

18.4.1 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 controls 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, 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.

18.4.2 SMB0TC: SMBus 0 Timing and Pin Control

Bit	7	6	5	4	3	2	1	0
Name	SWAP	Reserved					SDD	
Access	RW	R					RW	
Reset	0	0x00					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.

Bit	Name	Reset	Access	Description
6:2	*Reserved*	*Must write reset value.*		
1:0	SDD	0x0	RW	**SMBus Start Detection Window.**

These bits increase the hold time requirement between SDA falling and SCL falling for START detection.

Value	Name	Description
0x0	NONE	No additional hold time window (0-1 SYSCLK).
0x1	ADD_2_SYSCLKS	Increase hold time window to 2-3 SYSCLKs.
0x2	ADD_4_SYSCLKS	Increase hold time window to 4-5 SYSCLKs.
0x3	ADD_8_SYSCLKS	Increase hold time window to 8-9 SYSCLKs.

18.4.3 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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

18.4.4 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.

18.4.5 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.

18.4.6 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.

18.4.7 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).

18.4.8 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.*		

18.4.9 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.				

18.4.10 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.

19. Timers (Timer0, Timer1, Timer2, Timer3, and Timer4)

19.1 Introduction

Five counter/timers ar included in the device: two are 16-bit counter/timers compatible with those found in the standard 8051, and three 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, and Timer 4 are also similar, and offer both 16-bit and split 8-bit timer functionality with auto-reload capabilities. Timer 2, 3, and 4 offer capture functions that may be selected from several on-chip sources or an external pin.

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, and 4 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 19.1. Timer Modes

Timer 0 and Timer 1 Modes	Timer 2 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

19.2 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 and Timer 4 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.
- USB Start-of-Frame (SOF) capture.

19.3 Functional Description

19.3.1 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 19.2. 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
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]	
T2 Input Capture Pin					Yes			Yes			Yes
LFOSC0 Capture					Yes			Yes			Yes
Comparator 0 Output Capture					Yes			Yes			Yes
USB Start-of-Frame Capture					Yes			Yes			Yes

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.

19.3.2 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.

19.3.2.1 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:

$$F_{TIMER0} = \frac{F_{Input\ Clock}}{2^{13} - TH0{:}TL0} = \frac{F_{Input\ Clock}}{8192 - TH0{:}TL0}$$

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 19.3. 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.

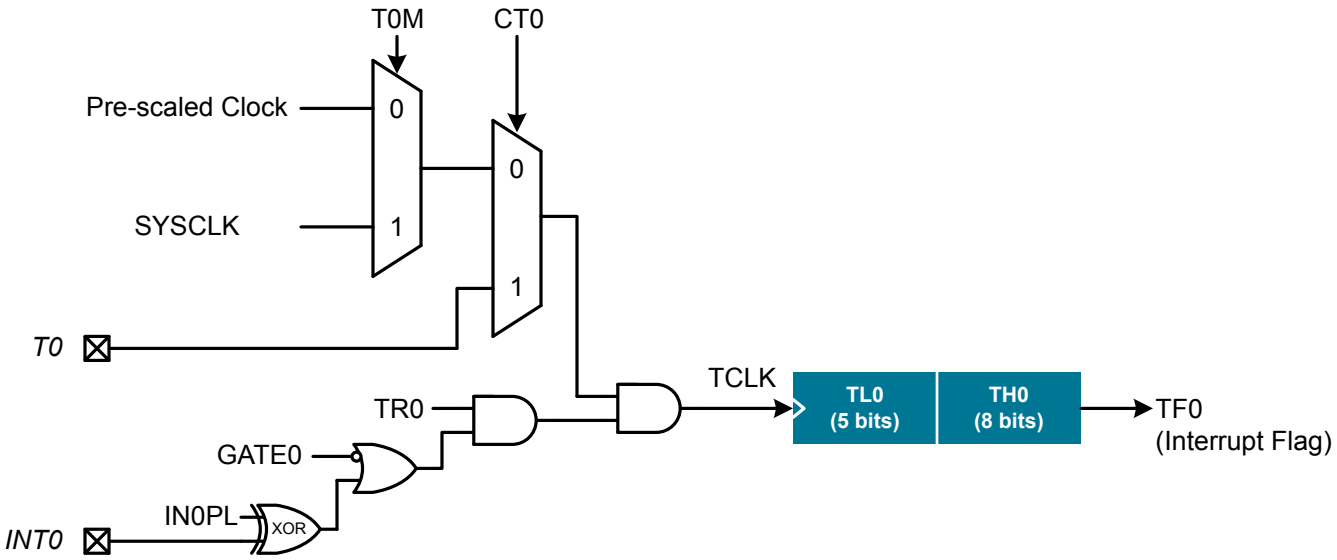


Figure 19.1. T0 Mode 0 Block Diagram

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.

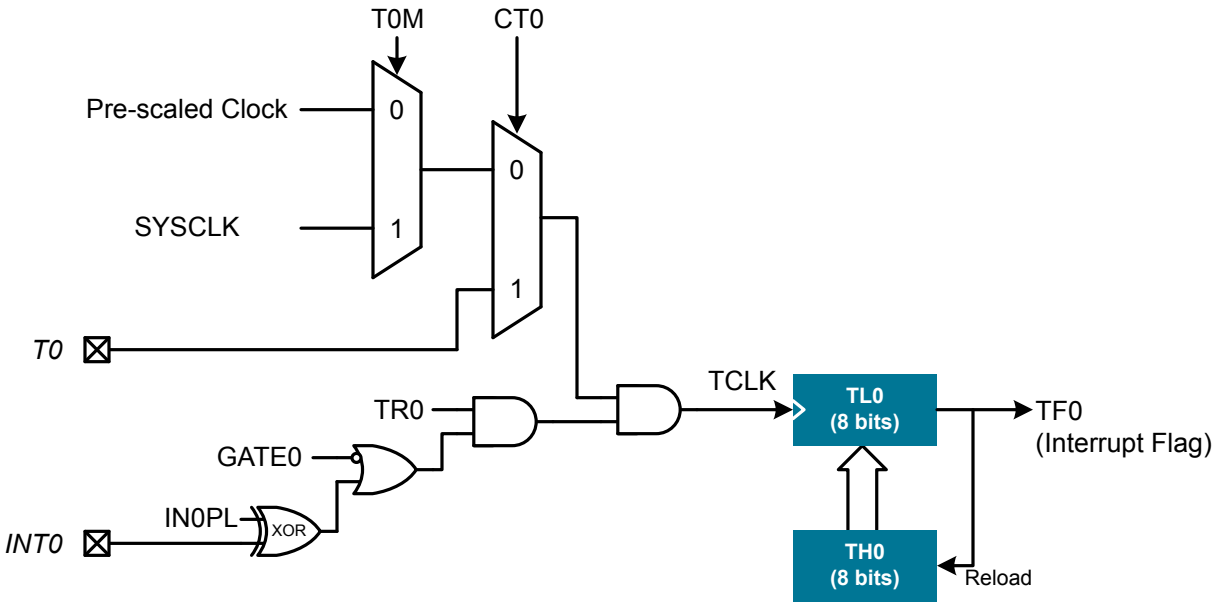


Figure 19.2. T0 Mode 2 Block Diagram

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.

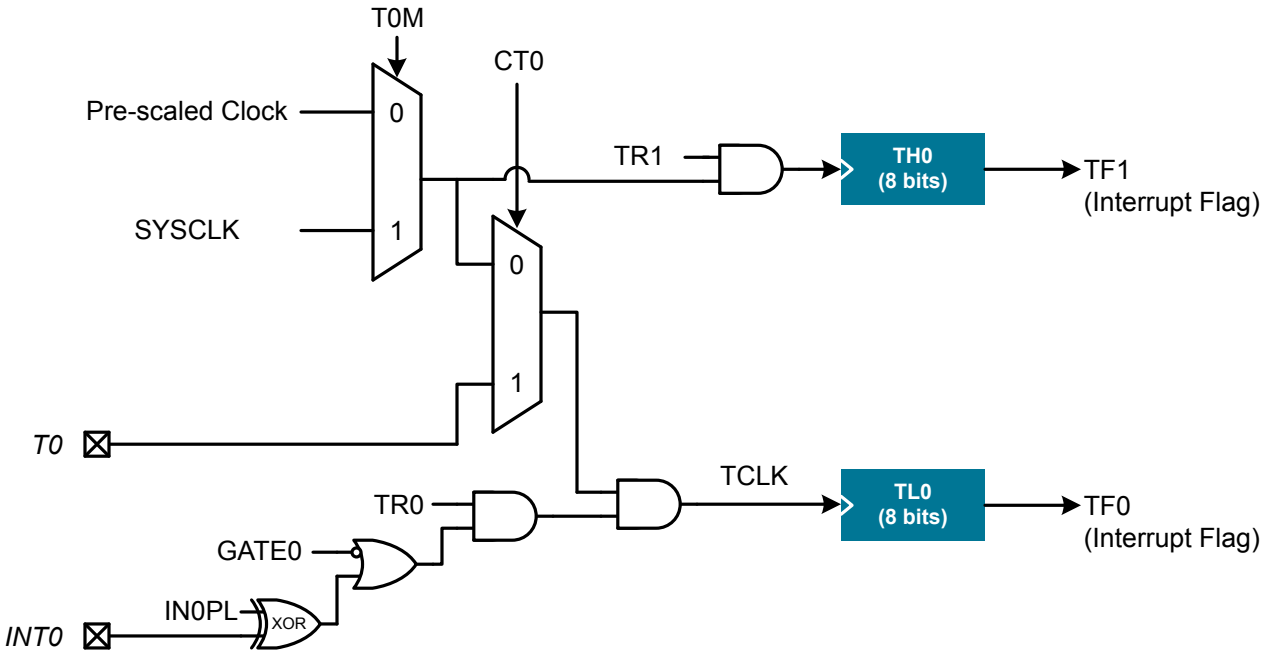


Figure 19.3. T0 Mode 3 Block Diagram

19.3.3 Timer 2, Timer 3, and Timer 4

Timer 2, Timer 3, and Timer 4 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.

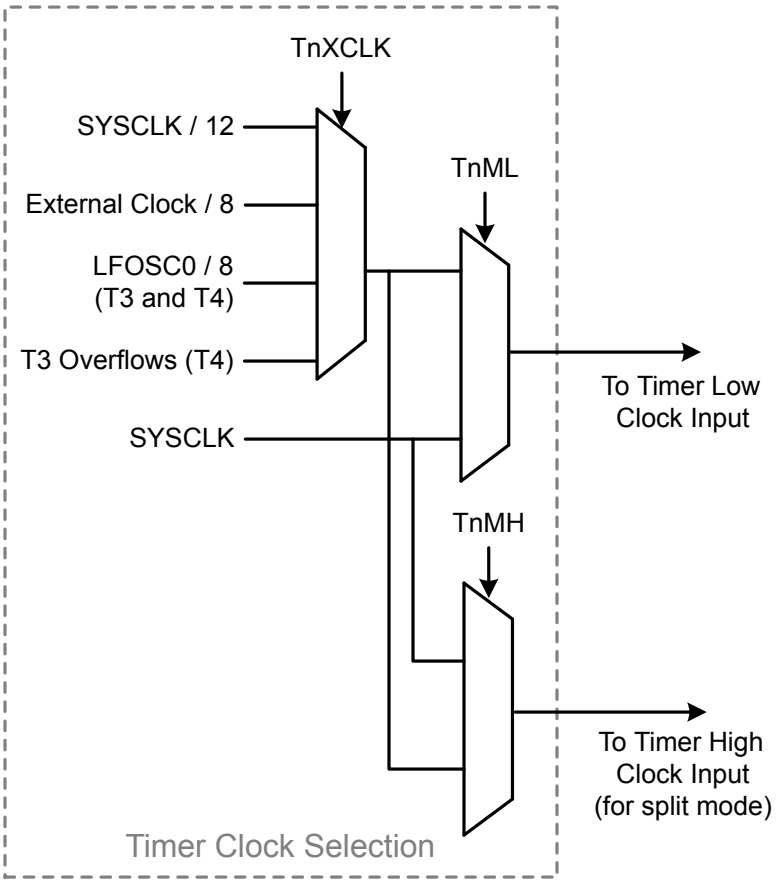


Figure 19.4. Timer 2, 3, and 4 Clock Source Selection

Capture Source Selection

Capture mode allows an external input, the low-frequency oscillator clock, comparator 0, or USB start-of-frame (SOF) 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, or USB start-of-frame (SOF) events. The capture input signal for the timer is selected using the TnCSEL field in the TMRnCN1 register.

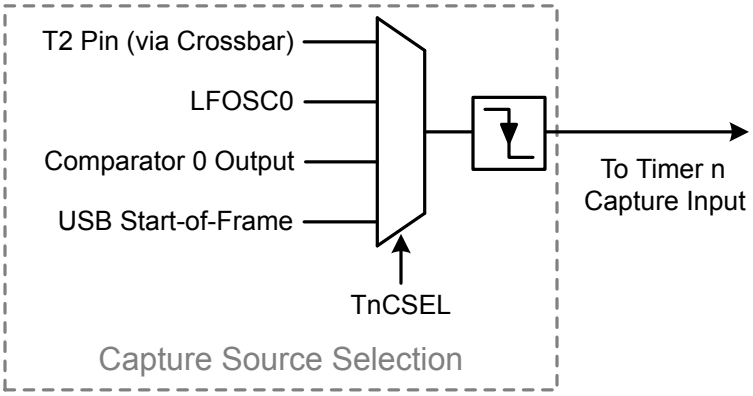


Figure 19.5. Timer 2, 3, and 4 Capture Source Selection

19.3.3.1 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.

The overflow rate of the timer in split 16-bit auto-reload mode is:

$$F_{TIMERn} = \frac{F_{Input\ Clock}}{2^{16} - TMRnRLH{:}TMRnRLL} = \frac{F_{Input\ Clock}}{65536 - TMRnRLH{:}TMRnRLL}$$

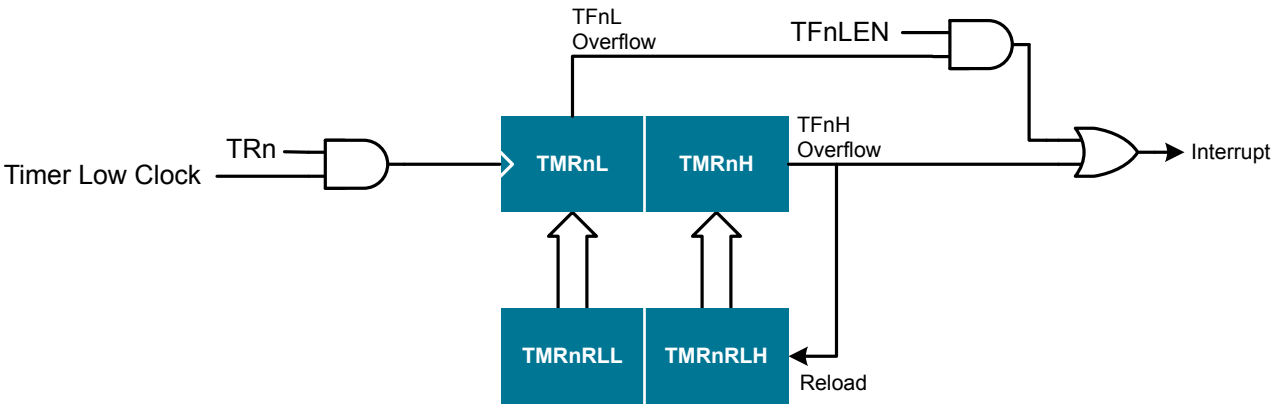


Figure 19.6. 16-Bit Mode Block Diagram

19.3.3.2 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.

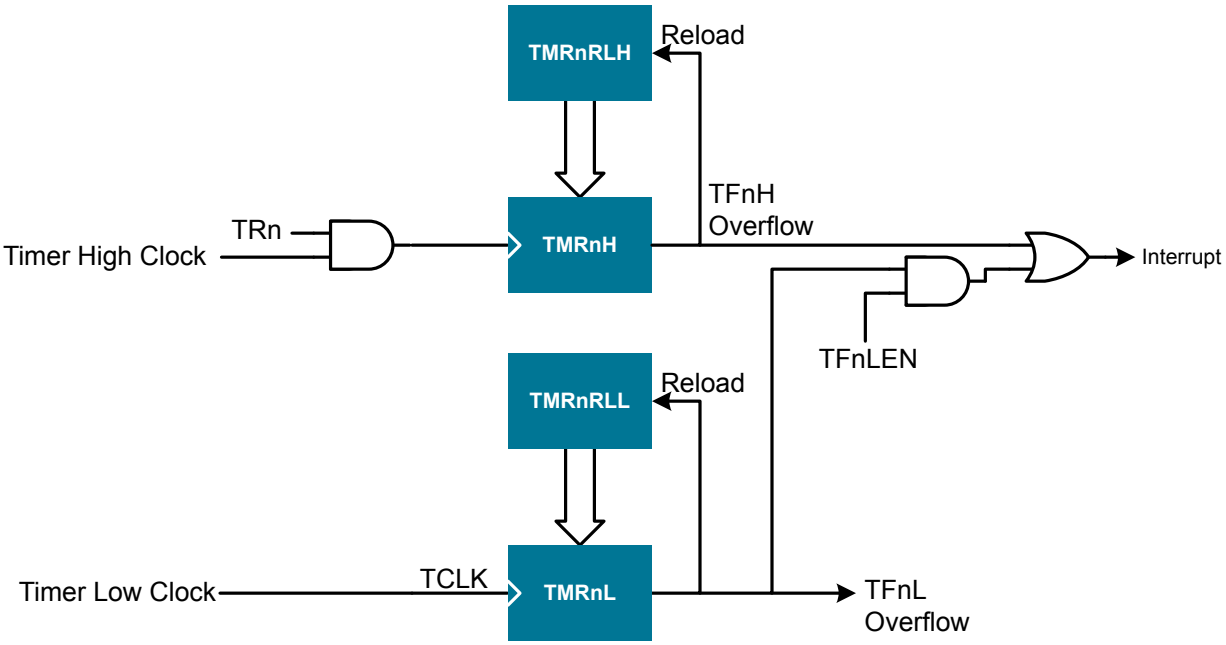


Figure 19.7. 8-Bit Split Mode Block Diagram

19.3.3.3 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.

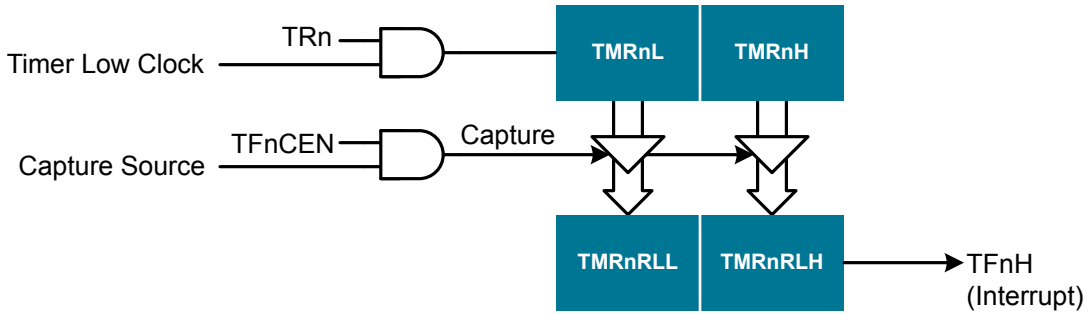


Figure 19.8. Capture Mode Block Diagram

19.3.3.4 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.

19.4 Timer 0, 1, 2, 3, and 4 Control Registers

19.4.1 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 = ALL; 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.
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).

19.4.2 CKCON1: Clock Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved						T4MH	T4ML
Access	R						RW	RW
Reset	0x00						0	0
SFR Page = 0x10; SFR Address: 0xA6								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
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.
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.

19.4.3 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 = ALL; 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.

19.4.4 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 = ALL; 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

19.4.5 TL0: Timer 0 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL0							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

19.4.6 TL1: Timer 1 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL1							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

19.4.7 TH0: Timer 0 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH0							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

19.4.8 TH1: Timer 1 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH1							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

19.4.9 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.** T2XCLK selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, T2XCLK 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).

19.4.10 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.

19.4.11 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.

19.4.12 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: 0xCC								

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.

19.4.13 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: 0xCD								

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.

19.4.14 TMR2CN1: Timer 2 Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved					T2CSEL		
Access	R					RW		
Reset	0x00					0x0		
SFR Page = 0x10; SFR Address: 0xFD								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
2:0	T2CSEL	0x0	RW	**Timer 2 Capture Select.** When used in capture mode, the T2CSEL register 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.
0x3	USB_SOF	Capture USB start-of-frame (SOF) events.

19.4.15 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.

19.4.16 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.

19.4.17 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.

19.4.18 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.

19.4.19 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).

19.4.20 TMR3CN1: Timer 3 Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved					T3CSEL		
Access	RW					RW		
Reset	0x00					0x1		
SFR Page = 0x10; SFR Address: 0xFE								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
2:0	T3CSEL	0x1	RW	**Timer 3 Capture Select.** When used in capture mode, the T3CSEL register 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.
0x3	USB_SOF	Capture USB start-of-frame (SOF) events.

19.4.21 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.

19.4.22 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.

19.4.23 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.

19.4.24 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.

19.4.25 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).

19.4.26 TMR4CN1: Timer 4 Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved					T4CSEL		
Access	RW					RW		
Reset	0x00					0x1		
SFR Page = 0x10; SFR Address: 0xFF								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
2:0	T4CSEL	0x1	RW	**Timer 4 Capture Select.** When used in capture mode, the T4CSEL register 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.
0x3	USB_SOF	Capture USB start-of-frame (SOF) events.

20. Universal Asynchronous Receiver/Transmitter 0 (UART0)

20.1 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).

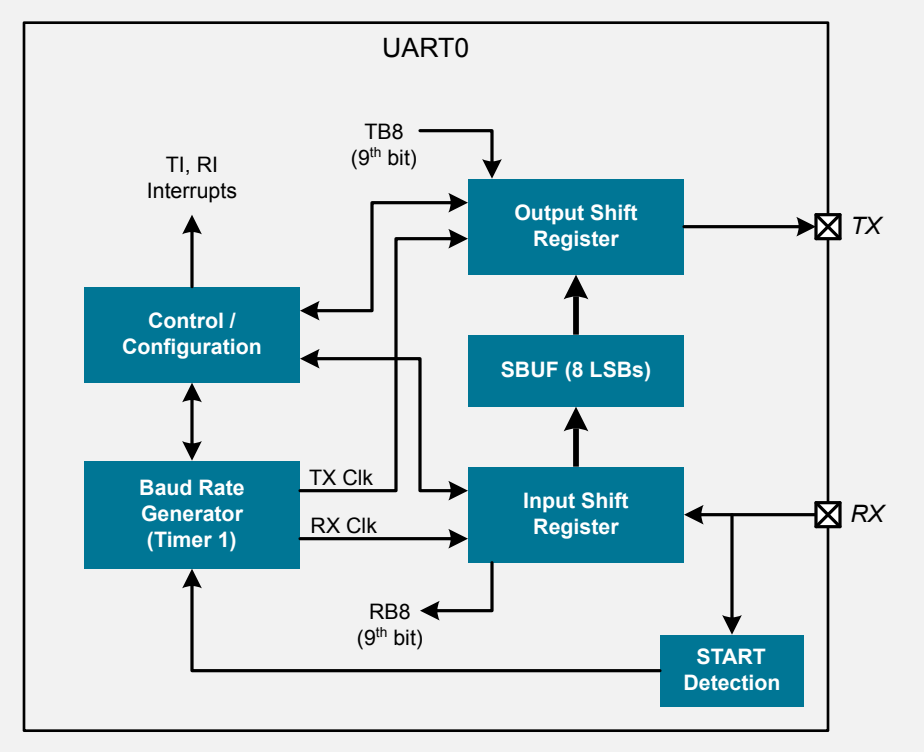


Figure 20.1. UART0 Block Diagram

20.2 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

20.3 Functional Description

20.3.1 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.

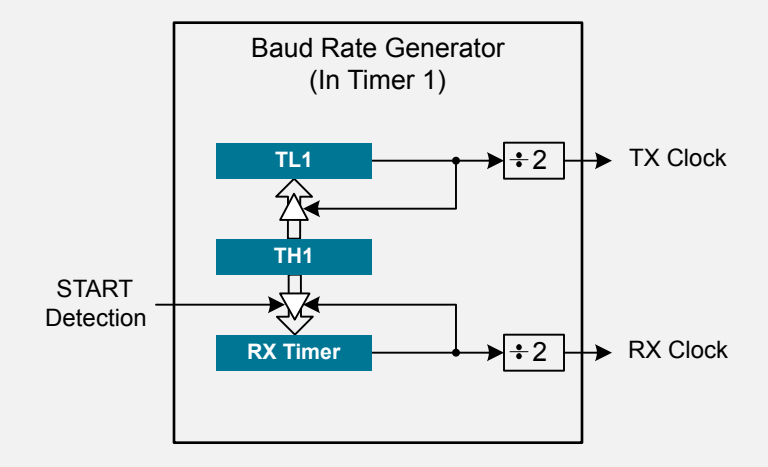


Figure 20.2. UART0 Baud Rate Logic Block Diagram

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.

20.3.2 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.

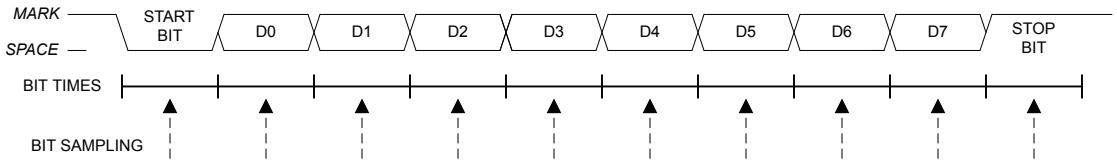


Figure 20.3. 8-Bit Data Transfer

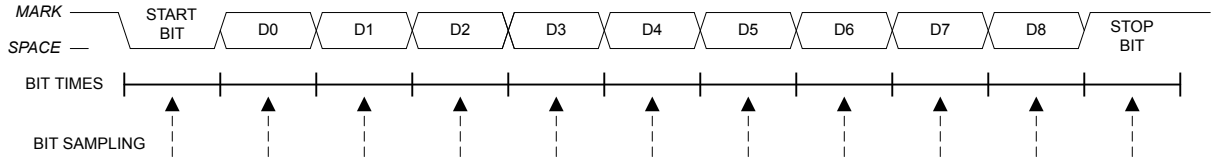


Figure 20.4. 9-Bit Data Transfer

20.3.3 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.

20.3.4 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).

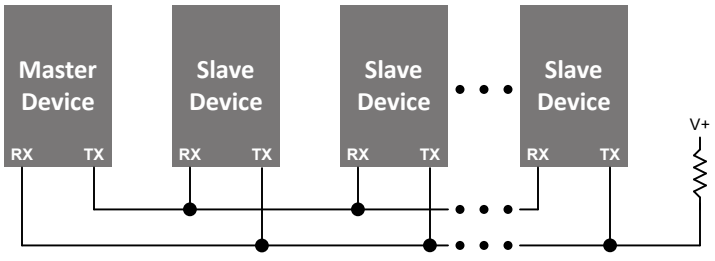


Figure 20.5. Multi-Processor Mode Interconnect Diagram

20.4 UART0 Control Registers

20.4.1 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).

Bit	Name	Reset	Access	Description
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).

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

20.4.2 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.

21. Universal Asynchronous Receiver/Transmitter 1 (UART1)

21.1 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.

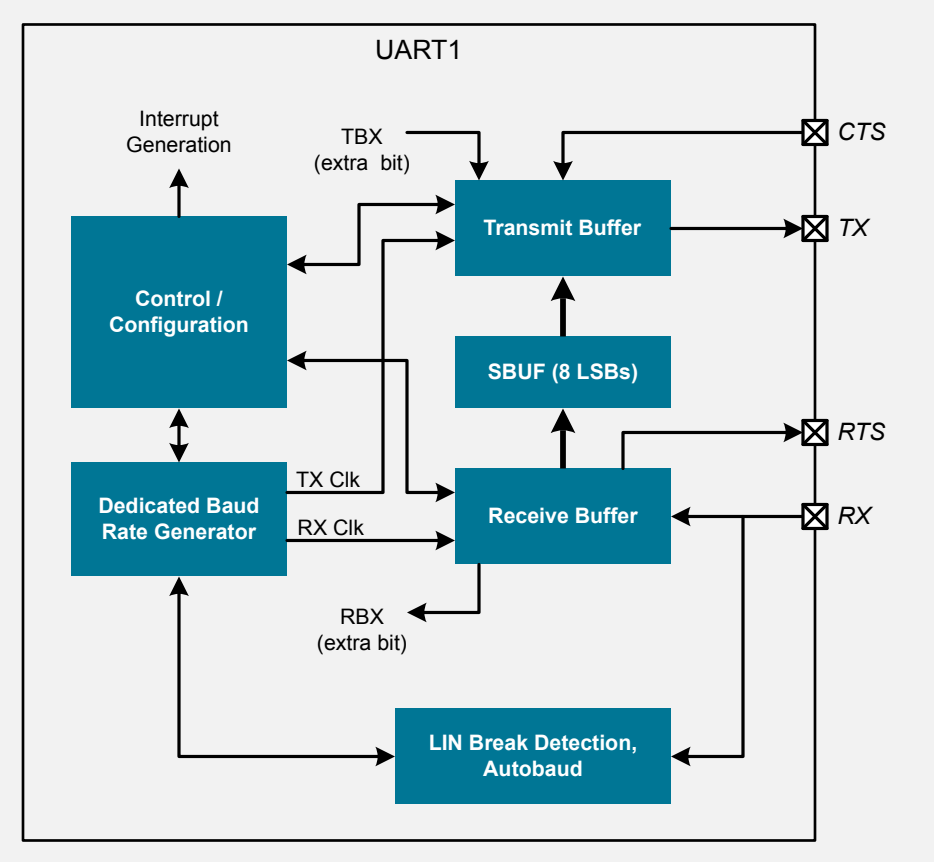


Figure 21.1. UART 1 Block Diagram

21.2 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.
- Four byte FIFO on transmit and receive.
- Auto-baud detection.
- LIN break and sync field detection.
- CTS / RTS hardware flow control.

21.3 Functional Description

21.3.1 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.

Figure 21.2. Baud Rate Generation

21.3.2 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.

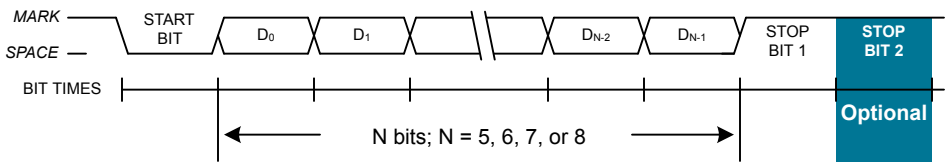


Figure 21.3. UART1 Timing Without Parity or Extra Bit

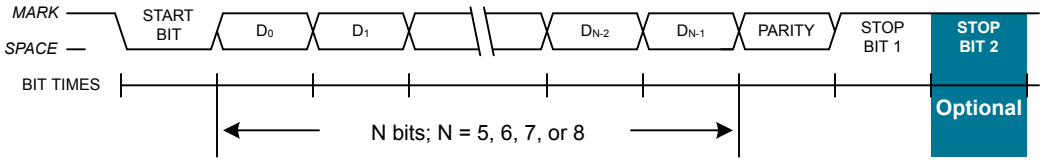


Figure 21.4. UART1 Timing With Parity

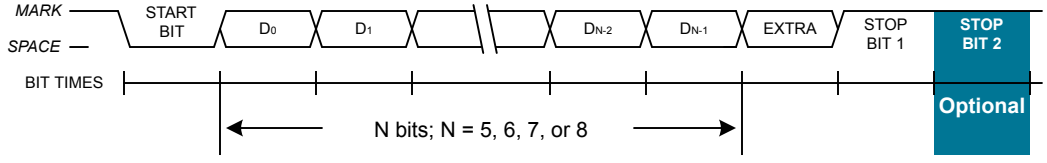


Figure 21.5. UART1 Timing With Extra Bit

21.3.3 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.

21.3.4 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.

21.3.5 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.

21.3.6 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).

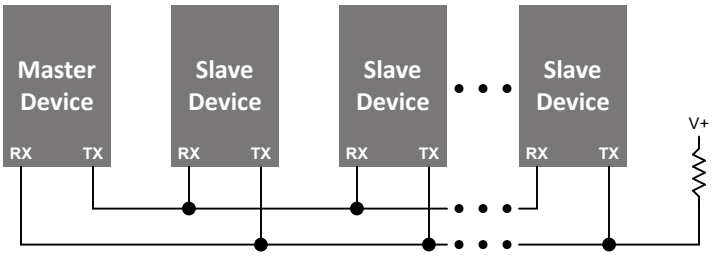


Figure 21.6. Multi-Processor Mode Interconnect Diagram

21.3.7 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.

21.3.8 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.

21.4 UART1 Control Registers

21.4.1 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.

21.4.2 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 if the stop bits are 1.
1	MULTI_ENABLED	RI will be activated if the stop bits and extra bit are 1. The extra bit must be enabled using XBE.

Bit	Name	Reset	Access	Description
6:5	SPT	0x0	RW	**Parity Type.**

Value	Name	Description
0x0	ODD_PARTY	Odd.
0x1	EVEN_PARITY	Even.
0x2	MARK_PARITY	Mark.
0x3	SPACE_PARITY	Space.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.
1	ENABLED	Enable the extra bit.

Bit	Name	Reset	Access	Description
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).

21.4.3 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.

21.4.4 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.

21.4.5 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))			

21.4.6 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))

21.4.7 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.

Bit	Name	Reset	Access	Description
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).

Bit	Name	Reset	Access	Description
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.
0x2	TWO	TFRQ will be set when the TX FIFO contains two or fewer bytes.
0x3	THREE	TFRQ will be set when the TX FIFO contains three or fewer bytes.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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).

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
	0x2	TWO		RFRQ will be set if the RX FIFO contains more than two bytes.
	0x3	THREE		RFRQ will be set if the RX FIFO contains more than three bytes.

21.4.8 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.

Bit	Name	Reset	Access	Description
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.

21.4.9 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.

21.4.10 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	BREAK-DNIE	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.

22. Universal Serial Bus (USB0)

22.1 Introduction

The USB0 peripheral provides a full-speed USB 2.0 compliant device controller and PHY with additional Low Energy USB features. The device supports both full-speed (12MBit/s) and low speed (1.5MBit/s) operation, and includes a dedicated USB oscillator with clock recovery mechanism for crystal-free operation. No external components are required. The USB function controller (USB0) consists of a Serial Interface Engine (SIE), USB transceiver (including matching resistors and configurable pull-up resistors), and 1 KB FIFO block. The Low Energy Mode ensures the current consumption is optimized and enables USB communication on a strict power budget.

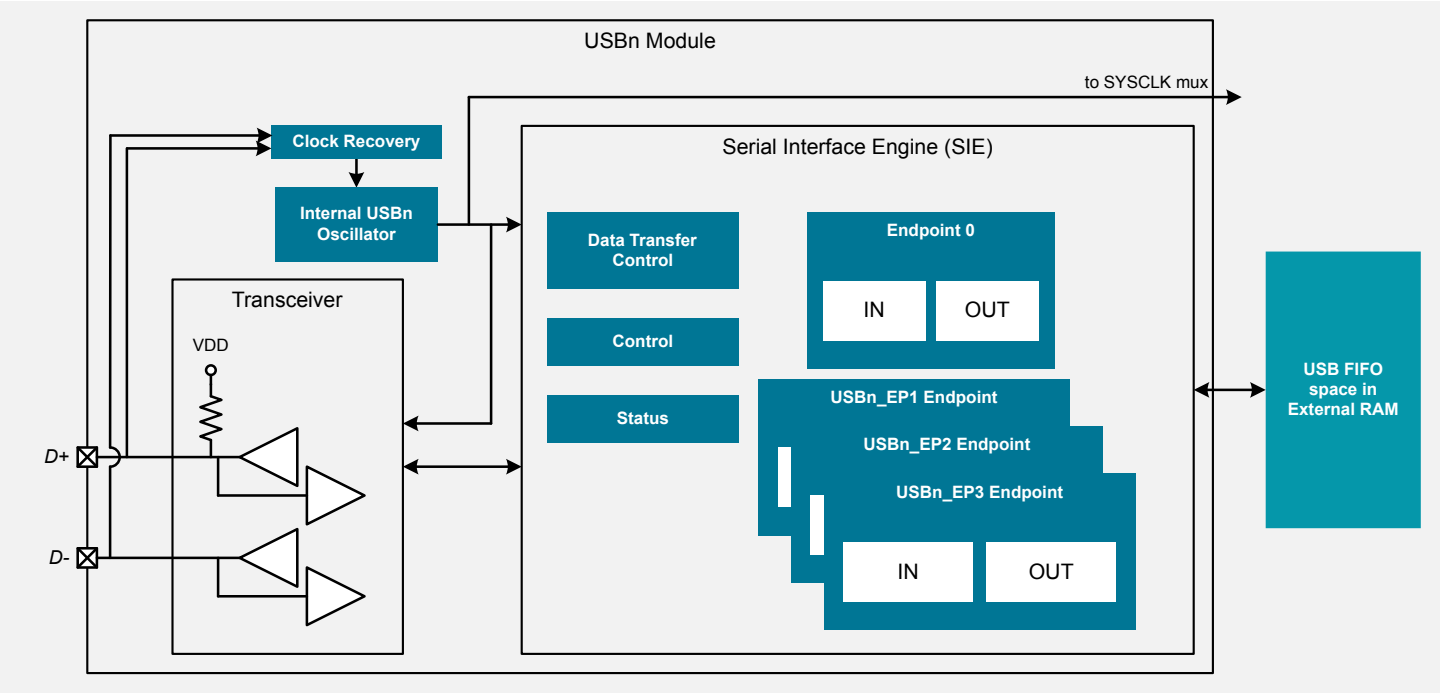


Figure 22.1. USB Block Diagram

22.2 Features

The USB0 module includes the following features:

- Full and Low Speed functionality.
- Implements 4 bidirectional endpoints.
- Low Energy Mode to reduce active supply current based on bus bandwidth.
- USB 2.0 compliant USB peripheral support (no host capability).
- Direct module access to 1 KB of RAM for FIFO memory.
- Clock recovery to meet USB clocking requirements with no external components.
- Charger detection circuitry with automatic detection of SDP, CDP, and DCP interfaces.
- D+ and D- can be routed to ADC input to support ACM and proprietary charger architectures.

22.3 Functional Description

22.3.1 Endpoint Addressing

A total of eight endpoint pipes are available. The control endpoint (Endpoint0) always functions as a bi-directional IN/OUT endpoint. The other endpoints are implemented as three pairs of IN/OUT endpoint pipes.

Table 22.1. Endpoint Addressing Scheme

Endpoint	Associated Pipes	USB Protocol Address
Endpoint 0	Endpoint 0 IN	0x00
	Endpoint 0 OUT	0x00
Endpoint 1	Endpoint 1 IN	0x81
	Endpoint 1 OUT	0x01
Endpoint 2	Endpoint 2 IN	0x82
	Endpoint 2 OUT	0x02
Endpoint 3	Endpoint 3 IN	0x83
	Endpoint 3 OUT	0x03

22.3.2 Transceiver Control

The USB Transceiver is configured via the USB0XCN register. This configuration includes transceiver enable/disable, pull-up resistor enable/disable, and device speed selection (full or low speed). When bit SPEED = 1, USB0 operates as a full speed USB function, and the on-chip pull-up resistor (if enabled) appears on the D+ pin. When bit SPEED = 0, USB0 operates as a low speed USB function, and the on-chip pull-up resistor (if enabled) appears on the D- pin. The PHYTST bits can be used for transceiver testing. The pull-up resistor is enabled only when VBUS is present.
Note: The USB clock should be active before the transceiver is enabled.

22.3.3 Clock Configuration

The USB module is capable of communication as a full or low speed USB function. Communication speed is selected via the SPEED bit in USB0XCN. When operating as a low speed function, the USB clock must be 6 MHz. When operating as a full speed function, the USB clock must be 48 MHz. The USB clock is selected using the USBCLK bit field in the USB0CF register. A typical full speed application would configure the USB clock to run directly from the HFOSC1 oscillator, while a typical low speed application would configure the clock for HFOSC1/8. The USB clock may also be derived from an external CMOS clock with various divider options. By default, the clock to the USB module is turned off to save power.

Clock Recovery circuitry uses the incoming USB data stream to adjust the internal oscillator; this allows the internal oscillator to meet the requirements for USB clock tolerance. Clock Recovery should always be used any time the USB block is clocked from the internal HFOSC1 clock in full speed applications. When operating the USB module as a low speed function with Clock Recovery, software must write 1 to the CRLOW bit to enable low speed Clock Recovery. Clock Recovery is typically not necessary in low speed mode. Single Step Mode can be used to help the Clock Recovery circuitry to lock when high noise levels are present on the USB network. This mode is not required (or recommended) in typical USB environments.

22.3.4 VBUS Control

In a self-powered system, it is generally desirable to be able to detect the presence of VBUS. VBUS indicates when a host device has been connected to or disconnected from the USB peripheral. The VBUS signal may be enabled on a port pin and configured to generate system interrupts if the state changes.

The VBUS control bits are found int he USB0CF register. VBUSEN enables the VBUS pin as an input to the USB module, while the VBUSIE bit enables the associated interrupt. VBUSI will be set any time the state of VBUS changes, and firmware may then read the state of the VBUS pin and act accordingly.

22.3.5 Register Access

Many of the USB0 controller registers are accessed indirectly through two SFRs: USB0 Address (USB0ADR) and USB0 Data (USB0DAT). The USB0ADR register selects which USB register is targeted by reads/writes of the USB0DAT register. Endpoint control/status registers are accessed by first writing the USB register INDEX with the target endpoint number. Once the target endpoint number is written to the INDEX register, the control/status registers associated with the target endpoint may be accessed.

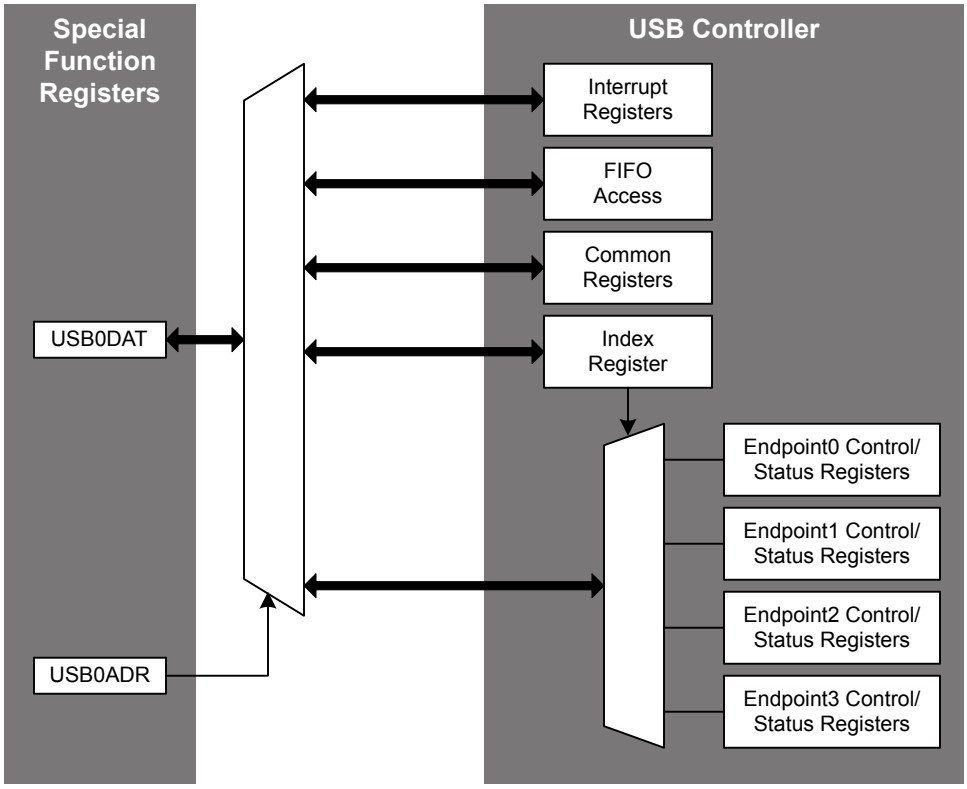


Figure 22.2. USB Indirect Register Access

Note: The USB clock must be active when accessing indirect USB registers.

Table 22.2. USB Indirect Registers

USB Register Name	USB Register Address	Description
Interrupt Registers		
IN1INT	0x02	Endpoint0 and Endpoints1-3 IN Interrupt Flags
OUT1INT	0x04	Endpoints1-3 OUT Interrupt Flags
CMINT	0x06	Common USB Interrupt Flags
IN1IE	0x07	Endpoint0 and Endpoints1-3 IN Interrupt Enables
OUT1IE	0x09	Endpoints1-3 OUT Interrupt Enables
CMIE	0x0B	Common USB Interrupt Enables
Common Registers		
FADDR	0x00	Function Address
POWER	0x01	Power Management

USB Register Name	USB Register Address	Description
FRAMEL	0x0C	Frame Number Low Byte
FRAMEH	0x0D	Frame Number High Byte
INDEX	0x0E	Endpoint Index Selection
CLKREC	0x0F	Clock Recovery Control
EENABLE	0x1E	Endpoint Enable
FIFOn	0x20-0x23	Endpoints0-3 FIFOs
Indexed Registers		
E0CSR	0x11	Endpoint0 Control / Status
EINCSRL		Endpoint IN Control / Status Low Byte
EINCSRH	0x12	Endpoint IN Control / Status High Byte
EOUTCSRL	0x14	Endpoint OUT Control / Status Low Byte
EOUTCSRH	0x15	Endpoint OUT Control / Status High Byte
E0CNT	0x16	Number of Received Bytes in Endpoint0 FIFO
EOUTCNTL		Endpoint OUT Packet Count Low Byte
EOUTCNTH	0x17	Endpoint OUT Packet Count High Byte

22.3.6 FIFO Management

1024 bytes of on-chip XRAM are used as FIFO space for the USB block. This FIFO space is split between Endpoints0-3. Endpoint0 is 64 bytes long, Endpoint1 is 128 bytes long, Endpoint2 is 256 bytes long, and Endpoint3 is 512 bytes long. FIFO space allocated for Endpoints1-3 is also configurable as IN, OUT, or both (split mode: half IN, half OUT).

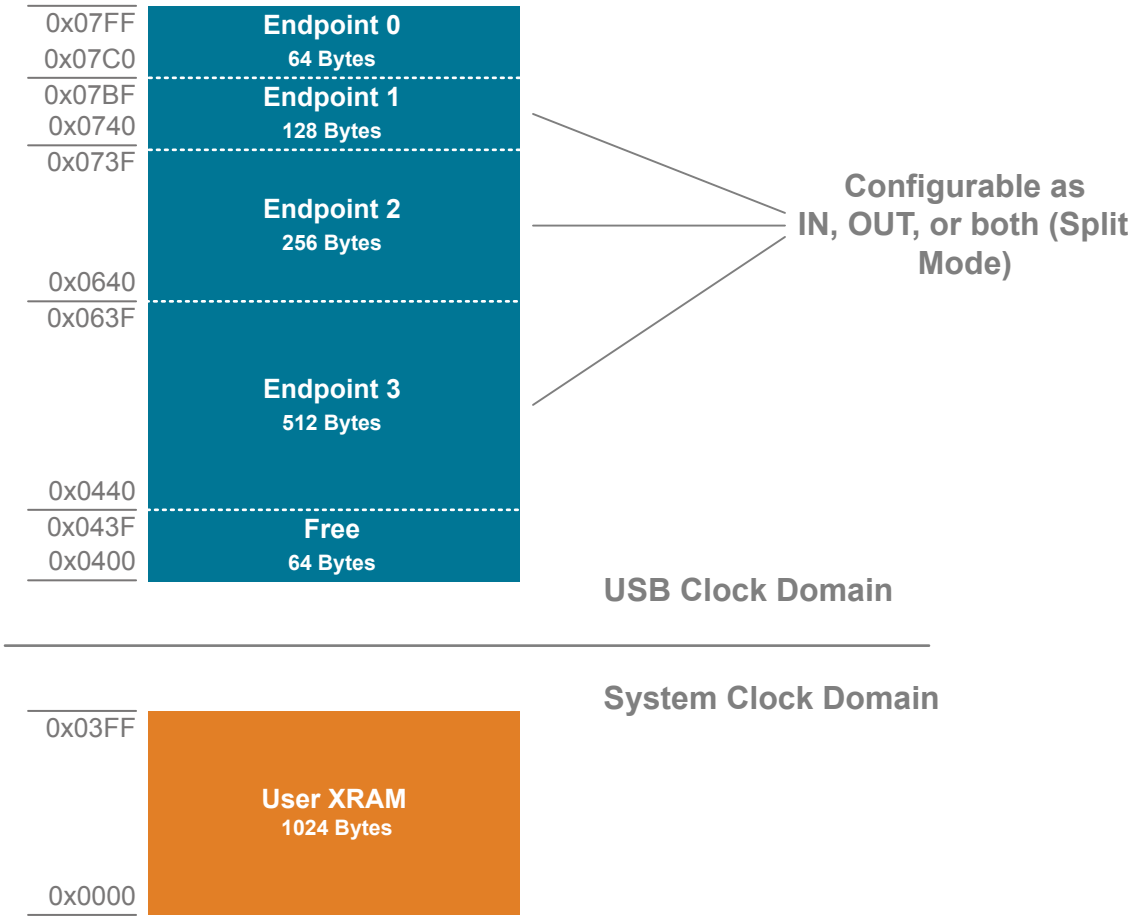


Figure 22.3. FIFO Memory Map

FIFO Split Mode

The FIFO space for Endpoints1-3 can be split such that the upper half of the FIFO space is used by the IN endpoint, and the lower half is used by the OUT endpoint. For example: if the Endpoint3 FIFO is configured for Split Mode, the upper 256 bytes are used by Endpoint3 IN and the lower 256 bytes are used by Endpoint3 OUT.

If an endpoint FIFO is not configured for split mode, that endpoint IN/OUT pair's FIFOs are combined to form a single IN or OUT FIFO. In this case only one direction of the endpoint IN/OUT pair may be used at a time. The endpoint direction (IN/OUT) is determined by the DIRSEL bit in the corresponding endpoint's EINCSRH register.

FIFO Double Buffering

FIFO slots for Endpoints1-3 can be configured for double-buffered mode. In this mode, the maximum packet size is halved and the FIFO may contain two packets at a time. This mode is available for Endpoints1-3. When an endpoint is configured for Split Mode, double buffering may be enabled for the IN Endpoint and/or the OUT endpoint. When split mode is not enabled, double-buffering may be enabled for the entire endpoint FIFO.

Table 22.3. FIFO Configuration

Endpoint Number	Split Mode Enabled?	Maximum IN Packet Size (Single Buffer / Double Buffer)	Maximum OUT Packet Size (Single Buffer / Double Buffer)
0	n/a	64	
1	N	128 / 64	
	Y	64 / 32	64 / 32
2	N	256 / 128	
	Y	128 / 64	128 / 64
3	N	512 / 256	
	Y	256 / 128	256 / 128

FIFO Access

Each endpoint FIFO is accessed through a corresponding FIFOn register. A read of an endpoint FIFOn register unloads one byte from the FIFO; a write of an endpoint FIFOn register loads one byte into the endpoint FIFO. When an endpoint FIFO is configured for Split Mode, a read of the endpoint FIFOn register unloads one byte from the OUT endpoint FIFO; a write of the endpoint FIFOn register loads one byte into the IN endpoint FIFO.

Accessing the Unused FIFO Memory

Unused areas of the USB FIFO space may be used as general purpose XRAM, if necessary. The FIFO block operates on the USB clock domain; thus, the USB clock must be active when accessing FIFO space. Note that the number of SYSCLK cycles required by the MOVX instruction is increased when accessing USB FIFO space.
Note: The USB clock must be active when accessing FIFO space.

22.3.7 Function Addressing

The FADDR register holds the current USB function address. Software should write the host-assigned 7-bit function address to the FADDR register when received as part of a SET_ADDRESS command. A new address written to FADDR will not take effect (USB will not respond to the new address) until the end of the current transfer, typically following the status phase of the SET_ADDRESS command transfer. The UPDATE bit is set to 1 by hardware when software writes a new address to the FADDR register. Hardware clears the UPDATE bit when the new address takes effect.

22.3.8 Function Configuration and Control

The USB register POWER is used to configure and control the USB block at the device level (enable/disable, Reset/Suspend/Resume handling, etc.).

USB Reset: The USBRST bit is set to 1 by hardware when Reset signaling is detected on the bus. Upon this detection, the following occur:

1. The USB0 Address is reset (FADDR = 0x00).
2. Endpoint FIFOs are flushed.
3. Control/status registers are reset to 0x00 (E0CSR, EINCSRL, EINCSRH, EOUTCSRL, EOUTCSRH).
4. USB register INDEX is reset to 0x00.
5. All USB interrupts (excluding the Suspend interrupt) are enabled and their corresponding flags cleared.
6. A USB Reset interrupt is generated if enabled.

Writing a 1 to the USBRST bit will generate an asynchronous USB reset. All USB registers are reset to their default values following this asynchronous reset.

Suspend Mode: With Suspend detection enabled (SUSEN = 1), USB0 will enter suspend mode when Suspend signaling is detected on the bus. An interrupt will be generated if enabled (SUSINTE = 1). The Suspend interrupt service routine (ISR) should perform application-specific configuration tasks such as disabling appropriate peripherals and/or configuring clock sources for low power modes.

The USB module exits Suspend mode when any of the following occur:

- Resume signaling is detected or generated
- Reset signaling is detected
- A device or USB reset occurs

If the device itself is in suspend mode, the internal oscillator will also exit suspend mode upon any of the above listed events.

Resume Signaling: The USB module exits Suspend mode if Resume signaling is detected on the bus. A Resume interrupt will be generated upon detection if enabled (RESINTE = 1). Software may force a Remote Wakeup by writing 1 to the RESUME bit (POWER.2). When forcing a Remote Wakeup, software should write RESUME = 0 to end Resume signaling 10-15 ms after the Remote Wakeup is initiated (RESUME = 1).

ISO Update: When software writes 1 to the ISOUP bit, the isochronous update function is enabled. With isochronous update enabled, new packets written to an isochronous IN endpoint will not be transmitted until a new Start-Of-Frame (SOF) is received. If the isochronous IN endpoint receives an IN token before a SOF, the USB interface will transmit a zero-length packet. When ISOUP = 1, isochronous update is enabled for all isochronous endpoints.

USB Enable: The USB module is disabled following a power-on-reset (POR). USB is enabled by clearing the USBINH bit. Once written to 0, the USBINH can only be set to 1 by a POR or an asynchronous USB reset generated by writing 1 to the USBRST bit.

Software should perform all USB configuration before enabling the USB module. The configuration sequence should be performed as follows:

1. Select and enable the USB clock source.
2. Reset the USB block by writing USBRST= 1.
3. Configure and enable the USB Transceiver.
4. Perform any USB function configuration (interrupts, Suspend detect, power mode configuration).
5. Enable USB by writing USBINH = 0.

22.3.9 Interrupts

The read-only USB interrupt flags are located in the USB registers shown in IN1INT, OUT1INT, and CMINT. The associated interrupt enable bits are located in the USB registers IN1IE, OUT1IE, and CMIE. A USB interrupt is generated when any of the USB interrupt flags is set to 1.
Note: Reading a USB interrupt flag register resets all flags in that register to 0.

22.3.10 Serial Interface Engine

The serial interface engine (SIE) performs all low level USB protocol tasks, interrupting the processor when data has successfully been transmitted or received. When receiving data, the SIE will interrupt the processor when a complete data packet has been received; appropriate handshaking signals are automatically generated by the SIE. When transmitting data, the SIE will interrupt the processor when a complete data packet has been transmitted and the appropriate handshake signal has been received.

The SIE will not interrupt the processor when corrupted/erroneous packets are received.

22.3.11 Endpoint 0

Endpoint0 is managed through the USB register E0CSR. The INDEX register must be loaded with 0x00 to access the E0CSR register. An Endpoint0 interrupt is generated when one of the following occurs:

- A data packet (OUT or SETUP) has been received and loaded into the Endpoint0 FIFO.
- The OPRDY bit is set to 1 by hardware.
- An IN data packet has successfully been unloaded from the Endpoint0 FIFO and transmitted to the host; INPRDY is reset to 0 by hardware.
- An IN transaction is completed (this interrupt generated during the status stage of the transaction).
- Hardware sets the STSTL bit after a control transaction ended due to a protocol violation.
- Hardware sets the SUEND bit because a control transfer ended before firmware set the DATAEND bit.

The E0CNT register holds the number of received data bytes in the Endpoint0 FIFO. Hardware will automatically detect protocol errors and send a STALL condition in response. Firmware may force a STALL condition to abort the current transfer. When a STALL condition is generated, the STSTL bit will be set to 1 and an interrupt generated. The following conditions will cause hardware to generate a STALL condition:

- The host sends an OUT token during a OUT data phase after the DATAEND bit has been set to 1.
- The host sends an IN token during an IN data phase after the DATAEND bit has been set to 1.
- The host sends a packet that exceeds the maximum packet size for Endpoint0.
- The host sends a non-zero length DATA1 packet during the status phase of an IN transaction.
- Firmware sets the SDSTL bit to 1.

Endpoint0 SETUP Transactions

All control transfers must begin with a SETUP packet. SETUP packets are similar to OUT packets, containing an 8-byte data field sent by the host. Any SETUP packet containing a command field of anything other than 8 bytes will be automatically rejected by USB0. An Endpoint0 interrupt is generated when the data from a SETUP packet is loaded into the Endpoint0 FIFO. Software should unload the command from the Endpoint0 FIFO, decode the command, perform any necessary tasks, and set the SOPRDY bit to indicate that it has serviced the OUT packet.

Endpoint0 IN Transactions

When a SETUP request is received that requires the USB interface to transmit data to the host, one or more IN requests will be sent by the host. For the first IN transaction, firmware should load an IN packet into the Endpoint0 FIFO, and set the INPRDY bit. An interrupt will be generated when an IN packet is transmitted successfully. Note that no interrupt will be generated if an IN request is received before firmware has loaded a packet into the Endpoint0 FIFO. If the requested data exceeds the maximum packet size for Endpoint0 (as reported to the host), the data should be split into multiple packets; each packet should be of the maximum packet size excluding the last (residual) packet. If the requested data is an integer multiple of the maximum packet size for Endpoint0, the last data packet should be a zero-length packet signaling the end of the transfer. Firmware should set the DATAEND bit to 1 after loading into the Endpoint0 FIFO the last data packet for a transfer.

Upon reception of the first IN token for a particular control transfer, Endpoint0 is said to be in Transmit Mode. In this mode, only IN tokens should be sent by the host to Endpoint0. The SUEND bit is set to 1 if a SETUP or OUT token is received while Endpoint0 is in Transmit Mode. Endpoint0 will remain in Transmit Mode until any of the following occur:

- The USB interface receives an Endpoint0 SETUP or OUT token.
- Firmware sends a packet less than the maximum Endpoint0 packet size.
- Firmware sends a zero-length packet.

Firmware should set the DATAEND bit to 1 when sending a zero-length packet or sending a packet less than the maximum Endpoint0 size. The SIE will transmit a NAK in response to an IN token if there is no packet ready in the IN FIFO (INPRDY = 0).

Endpoint0 OUT Transactions

When a SETUP request is received that requires the host to transmit data to USB0, one or more OUT requests will be sent by the host. When an OUT packet is successfully received by USB0, hardware will set the OPRDY bit to 1 and generate an Endpoint0 interrupt. Following this interrupt, firmware should unload the OUT packet from the Endpoint0 FIFO and set the SOPRDY bit to 1.

If the amount of data required for the transfer exceeds the maximum packet size for Endpoint0, the data will be split into multiple packets. If the requested data is an integer multiple of the maximum packet size for Endpoint0 (as reported to the host), the host will send a zero-length data packet signaling the end of the transfer.

Upon reception of the first OUT token for a particular control transfer, Endpoint0 is said to be in Receive Mode. In this mode, only OUT tokens should be sent by the host to Endpoint0. The SUEND bit is set to 1 if a SETUP or IN token is received while Endpoint0 is in Receive Mode. Endpoint0 will remain in Receive mode until one of the following occurs:

- The SIE receives a SETUP or IN token.
- The host sends a packet less than the maximum Endpoint0 packet size.
- The host sends a zero-length packet.

Firmware should set the DATAEND bit to 1 when the expected amount of data has been received. The SIE will transmit a STALL condition if the host sends an OUT packet after the DATAEND bit has been set by firmware. An interrupt will be generated with the STSTL bit set to 1 after the STALL is transmitted.

22.3.12 Endpoints 1, 2, and 3

Endpoints 1-3 are configured and controlled through their own sets of the following control/status registers: IN registers EINCSRL and EINCSRH, and OUT registers EOUTCSRL and EOUTCSRH. Only one set of endpoint control/status registers is mapped into the USB register address space at a time, defined by the contents of the INDEX register.

Endpoints 1-3 can be configured as IN, OUT, or both IN/OUT (Split Mode). The endpoint mode (Split/Normal) is selected via the SPLIT bit in register EINCSRH. When SPLIT = 1, the corresponding endpoint FIFO is split, and both IN and OUT pipes are available. When SPLIT = 0, the corresponding endpoint functions as either IN or OUT; the endpoint direction is selected by the DIRSEL bit in register EINCSRH. Endpoints 1-3 can be disabled individually by the corresponding bits in the ENABLE register. When an Endpoint is disabled, it will not respond to bus traffic or stall the bus. All Endpoints are enabled by default.

Endpoint 1-3 IN General Control

Endpoints 1-3 IN are managed via USB registers EINCSRL and EINCSRH. All IN endpoints can be used for Interrupt, Bulk, or Isochronous transfers. Isochronous (ISO) mode is enabled by writing 1 to the ISO bit in register EINCSRH. Bulk and Interrupt transfers are handled identically by hardware. An Endpoint 1-3 IN interrupt is generated by any of the following conditions:

- An IN packet is successfully transferred to the host.
- Software writes 1 to the FLUSH bit when the target FIFO is not empty.
- Hardware generates a STALL condition.

Operating Endpoints 1-3 as IN Interrupt or Bulk Endpoints

When the ISO bit = 0 the target endpoint operates in Bulk or Interrupt Mode. Once an endpoint has been configured to operate in Bulk/Interrupt IN mode (typically following an Endpoint0 SET_INTERFACE command), firmware should load an IN packet into the endpoint IN FIFO and set the INPRDY bit. Upon reception of an IN token, hardware will transmit the data, clear the INPRDY bit, and generate an interrupt.

Writing 1 to INPRDY without writing any data to the endpoint FIFO will cause a zero-length packet to be transmitted upon reception of the next IN token. A Bulk or Interrupt pipe can be shut down (or Halted) by writing 1 to the SDSTL bit (EINCSRL.4). While SDSTL = 1, hardware will respond to all IN requests with a STALL condition. Each time hardware generates a STALL condition, an interrupt will be generated and the STSTL bit set to 1. The STSTL bit must be reset to 0 by firmware.

Hardware will automatically reset INPRDY to 0 when a packet slot is open in the endpoint FIFO. If double buffering is enabled for the target endpoint, it is possible for firmware to load two packets into the IN FIFO at a time. In this case, hardware will reset INPRDY to 0 immediately after firmware loads the first packet into the FIFO and sets INPRDY to 1. An interrupt will not be generated in this case; an interrupt will only be generated when a data packet is transmitted.

When firmware writes 1 to the FCDT bit, the data toggle for each IN packet will be toggled continuously, regardless of the handshake received from the host. This feature is typically used by Interrupt endpoints functioning as rate feedback communication for Isochronous endpoints. When FCDT = 0, the data toggle bit will only be toggled when an ACK is sent from the host in response to an IN packet.

Operating Endpoints 1-3 as IN Isochronous Endpoints

When the ISO bit is set to 1, the target endpoint operates in Isochronous (ISO) mode. Once an endpoint has been configured for ISO IN mode, the host will send one IN token (data request) per frame; the location of data within each frame may vary. Because of this, it is recommended that double buffering be enabled for ISO IN endpoints.

Hardware will automatically reset INPRDY to 0 when a packet slot is open in the endpoint FIFO. Note that if double buffering is enabled for the target endpoint, it is possible for firmware to load two packets into the IN FIFO at a time. In this case, hardware will reset INPRDY to 0 immediately after firmware loads the first packet into the FIFO and sets INPRDY to 1. An interrupt will not be generated in this case; an interrupt will only be generated when a data packet is transmitted.

If there is not a data packet ready in the endpoint FIFO when USB0 receives an IN token from the host, USB0 will transmit a zero-length data packet and set the UNDRUN bit to 1.

The ISO Update feature can be useful in starting a double buffered ISO IN endpoint. If the host has already set up the ISO IN pipe (has begun transmitting IN tokens) when firmware writes the first data packet to the endpoint FIFO, the next IN token may arrive and the first data packet sent before firmware has written the second (double buffered) data packet to the FIFO. The ISO Update feature ensures that any data packet written to the endpoint FIFO will not be transmitted during the current frame; the packet will only be sent after a SOF signal has been received.

Endpoint 1-3 OUT General Control

Endpoints 1-3 OUT are managed via USB registers EOUTCSRL and EOUTCSRH. All OUT endpoints can be used for Interrupt, Bulk, or Isochronous transfers. Isochronous (ISO) mode is enabled by writing 1 to the ISO bit in register EOUTCSRH. Bulk and Interrupt transfers are handled identically by hardware. An Endpoint 1-3 OUT interrupt may be generated by the following:

- Hardware sets the OPRDY bit to 1.
- Hardware generates a STALL condition.

Operating Endpoints 1-3 as OUT Interrupt or Bulk Endpoints

When the ISO bit = 0 the target endpoint operates in Bulk or Interrupt mode. Once an endpoint has been configured to operate in Bulk/Interrupt OUT mode (typically following an Endpoint0 SET_INTERFACE command), hardware will set the OPRDY bit to 1 and generate an interrupt upon reception of an OUT token and data packet. The number of bytes in the current OUT data packet (the packet ready to be unloaded from the FIFO) is given in the EOUTCNTH and EOUTCNTL registers. In response to this interrupt, firmware should unload the data packet from the OUT FIFO and reset the OPRDY bit to 0.

A Bulk or Interrupt pipe can be shut down (or Halted) by writing 1 to the SDSTL bit. While SDSTL = 1, hardware will respond to all OUT requests with a STALL condition. Each time hardware generates a STALL condition, an interrupt will be generated and the STSTL bit set to 1. The STSTL bit must be reset to 0 by firmware.

Hardware will automatically set OPRDY when a packet is ready in the OUT FIFO. Note that if double buffering is enabled for the target endpoint, it is possible for two packets to be ready in the OUT FIFO at a time. In this case, hardware will set OPRDY to 1 immediately after firmware unloads the first packet and resets OPRDY to 0. A second interrupt will be generated in this case.

Operating Endpoints 1-3 as OUT Isochronous Endpoints

When the ISO bit is set to 1, the target endpoint operates in Isochronous (ISO) mode. Once an endpoint has been configured for ISO OUT mode, the host will send exactly one data per USB frame; the location of the data packet within each frame may vary, however. Because of this, it is recommended that double buffering be enabled for ISO OUT endpoints.

Each time a data packet is received, hardware will load the received data packet into the endpoint FIFO, set the OPRDY bit to 1, and generate an interrupt (if enabled). Firmware would typically use this interrupt to unload the data packet from the endpoint FIFO and reset the OPRDY bit to 0.

If a data packet is received when there is no room in the endpoint FIFO, an interrupt will be generated and the OVRUN bit set to 1. If USB0 receives an ISO data packet with a CRC error, the data packet will be loaded into the endpoint FIFO, OPRDY will be set to 1, an interrupt (if enabled) will be generated, and the DATAERR bit will be set to 1. Software should check the DATAERR bit each time a data packet is unloaded from an ISO OUT endpoint FIFO.

22.3.13 Low Energy Mode

The USB module has controls for automatically optimizing the power used by the block according to the current bus activity. The affected portions of the hardware and when the hardware uses the low energy mode are both configured using fields in the USB0AEC register.

The USB module can enter a low energy mode in response to different events on the USB bus based on the LEMCN bit field setting. By default, the USB block never enters low energy mode. Firmware may configure APMMD to instruct the block to enter low energy mode during an idle bus (no USB traffic), during NAKed OUT packets, or under both circumstances.

The amount of supply current used by the block in low energy mode can be adjusted in two ways: by lowering the supply current to the transceiver, or by gating the clock to the USB logic. These options are controlled by the XCVRMD and OSCMD bit fields. In addition to gating the USB clock, if the rest of the system besides USB operates from the HFOSC0 oscillator, the HFOSC1 oscillator may be dynamically turned off in low energy mode.

For the most efficient energy consumption, the following USB0AEC register configurations are recommended:

- LEMCN = 0x3 to enable low energy mode during idle bus times and NAKed OUT packets.
- OSCMD = 0x0 to dynamically gate the USB clock source, and disable HFOSC1 if possible.
- XCVRMD = 0x0 to dynamically adjust the transceiver supply current when possible.

The selections for low energy operation should be configured before enabling the transceiver.

22.3.14 Charger Detect Function

The USB block contains a charger detection circuit which is compliant with the *USB-IF Battery Charging Specification, Revision 1.2*. Upon establishing a physical connection to a USB host, the peripheral can distinguish between a standard downstream port (SDP), dedicated charging port (DCP), or a charging downstream port (CDP). ADC multiplexer connections to the USB D+ and D– pins are also provided internally for detecting the presence of non-standard charging hardware. Firmware may optionally implement algorithms to detect ACA or non-compliant charger hardware.
Note: The USB charger detect function only distinguishes between the various types of USB ports outlined in the specification. The device itself does not contain direct battery management or battery charging circuitry.

Firmware interfaces to the USB charger detection hardware through three special function registers: USB0CDCF, USB0CDCN, and USB0CDSTA. The USB0CDCF and USB0CDCN registers configure and control the hardware, while USB0CDSTA provides status information. The charger detection hardware shares an interrupt with the VBUS detection interrupt, allowing firmware to use the same interrupt service routine to handle all of the USB charger detect functions. Interrupts may be generated on the following events:

- VBUS detection (see VBUS configuration section)
- VBUS removal (generates an error interrupt)
- Completion of data contact detection (DCD) phase
- Completion of primary detection (PD) phase
- Completion of secondary detection (SD) phase

Additionally, the charger detection block allows firmware to selectively choose which functions will be performed when charge detection is enabled. Data contact detection (DCD), DCD timeout, primary detection (PD), and secondary detection (SD) may all be enabled individually. Hardware does not perform any of these operations until the charger detection function is enabled using the CHDEN bit and the hardware detects a valid VBUS signal. If VBUS is not enabled, it is assumed to be present by the hardware. Once DCDEN is enabled, the hardware proceeds through the selected functions in the following order, skipping any that are not enabled:

1. Data Contact Detection
2. Primary Detection
3. Secondary Detection

As each function completes, the hardware sets the associated interrupt flag and clears the enable flag. If VBUS is removed at any time while the charger detection circuit is enabled, the current function aborts, and the hardware sets the error flag (ERR).

Detection of SDP, DCP, and CDP

The most common and straightforward usage of the charger detection block is to determine the type of USB port to which the device has been connected. Each type of port has different load profile, maximum current, and communications capabilities, per the specification. To use the charger detection block for this purpose:

1. Enable VBUS detection on the VBUS pin (must be connected to USB VBUS).
2. Optionally, enable the PD and/or SD interrupts with the corresponding enable bits.
3. Set DCDEN to "Full Detection" (0x3) to enable data contact detection and the associated timeout circuit.
4. Set PDEN to enable Primary Detection.
5. Set CHDEN to begin the charge detect sequence.
6. Wait for Primary Detection to complete (PDI = 1, or service the interrupt).
7. The SDP bit will indicate if a Standard Downstream Port is detected.
8. If the application requires further differentiation between DCP and CDP, set SDEN to enable Secondary Detection, and set CHDEN to begin this sequence.
9. Wait for Secondary Detection to complete (SDI = 1, or service the interrupt).
10. The DCP and CDP bits will indicate if a Dedicated Charging Port or Charging Downstream Port has been detected.

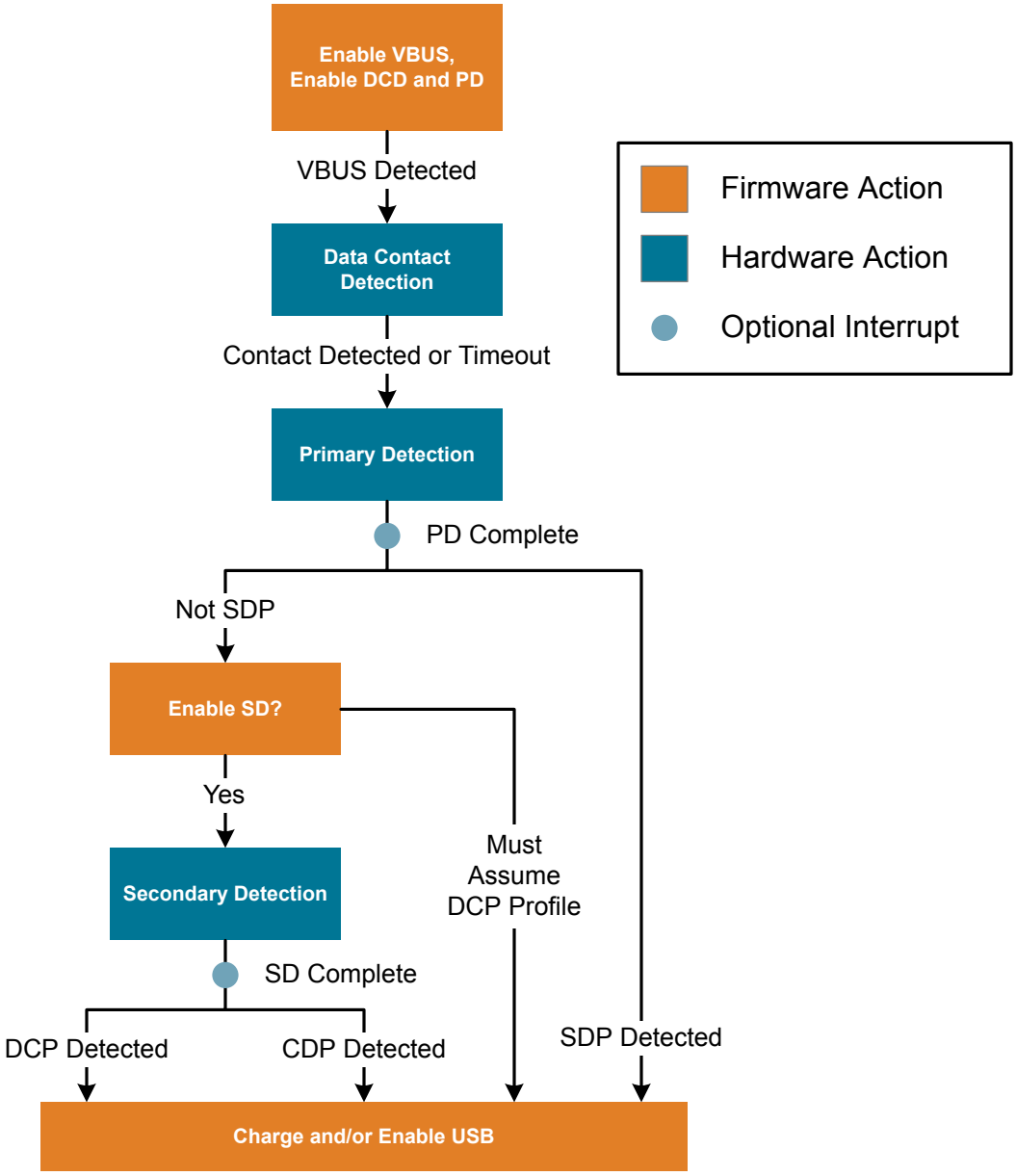


Figure 22.4. Basic USB Charger Detection Flow Diagram

Atypical Charger Detection

It is possible to detect ACA chargers, as well as certain chargers that do not comply with the USB specification, using additional resources on the device. Accessory charging adapters (ACA) chargers use a resistor to ground on a special ID pin and a specific voltage on the USB D- pin to encode the type of ACA and its capabilities. If ACA detection is required, the ID pin signal should be connected to any GPIO on the device which supports ADC input, and an external current source or pullup resistor must be provided. The ADC may be used to measure the voltage on the ID signal and the voltage on D- to distinguish between different ACA options. Applications needing to determine ACA ports should check for ACA after primary detection is complete and, optionally, after data contact detection is complete.

Many dedicated charging units pre-date the USB Battery Charging Specification or do not comply with this specification for other reasons, such as additional supply current capabilities . Most of these cases implement resistive voltage dividers to produce very specific voltages on the D+ and D- pins. In this case, the D+ and D- pins may be measured directly using the ADC to determine the voltage levels and whether such a charger is attached. Normally, this would be performed after VBUS is detected and before going through the data contact detection sequence.

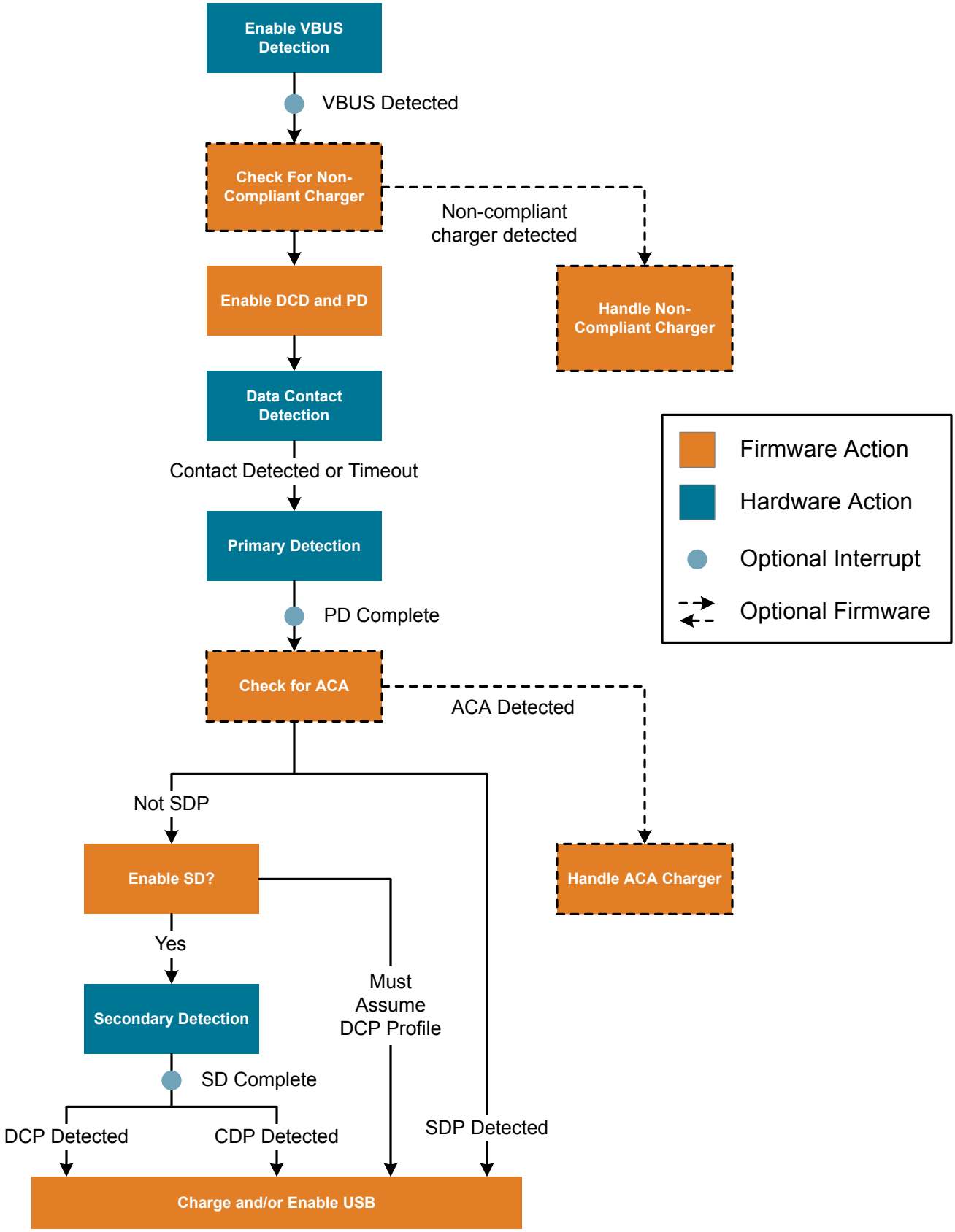


Figure 22.5. USB Charger Detection Flow Diagram with ACA and Non-Compliant Charger

22.4 USB0 Control Registers

22.4.1 USB0XCN: USB0 Transceiver Control

Bit	7	6	5	4	3	2	1	0
Name	PREN	PHYEN	SPEED	PHYTST		DFREC	Dp	Dn
Access	RW	RW	RW	RW		R	R	R
Reset	0	0	0	0x0		0	0	0
SFR Page = 0x20; SFR Address: 0xB3								

Bit	Name	Reset	Access	Description
7	PREN	0	RW	**Internal Pull-up Resistor Enable.** The location of the pull-up resistor (D+ or D-) is determined by the SPEED bit.

Value	Name	Description
0	PULL_UP_DISABLED	Internal pull-up resistor disabled (device effectively detached from USB network).
1	PULL_UP_ENABLED	Internal pull-up resistor enabled when VBUS is present (device attached to the USB network).

Bit	Name	Reset	Access	Description
6	PHYEN	0	RW	**Physical Layer Enable.**

Value	Name	Description
0	DISABLED	Disable the USB0 physical layer transceiver (suspend).
1	ENABLED	Enable the USB0 physical layer transceiver (normal).

Bit	Name	Reset	Access	Description
5	SPEED	0	RW	**USB0 Speed Select.** This bit selects the USB0 speed.

Value	Name	Description
0	LOW_SPEED	USB0 operates as a Low Speed device. If enabled, the internal pull-up resistor appears on the D- line.
1	FULL_SPEED	USB0 operates as a Full Speed device. If enabled, the internal pull-up resistor appears on the D+ line.

Bit	Name	Reset	Access	Description
4:3	PHYTST	0x0	RW	**Physical Layer Test.**

Value	Name	Description
0x0	MODE0	Mode 0: Normal (non-test mode) (D+ = X, D- = X).
0x1	MODE1	Mode 1: Differential 1 forced (D+ = 1, D- = 0).
0x2	MODE2	Mode 2: Differential 0 forced (D+ = 0, D- = 1).
0x3	MODE3	Mode 3: Single-Ended 0 forced (D+ = 0, D- = 0).

Bit	Name	Reset	Access	Description
2	DFREC	0	R	**Differential Receiver.** The state of this bit indicates the current differential value present on the D+ and D- lines when PHYEN = 1.

Value	Name	Description
0	DIFFERENTIAL_ZERO	Differential 0 signalling on the bus.
1	DIFFERENTIAL_ONE	Differential 1 signalling on the bus.

Bit	Name	Reset	Access	Description
1	Dp	0	R	**D+ Signal Status.** This bit indicates the current logic level of the D+ pin.

Value	Name	Description
0	LOW	D+ signal currently at logic 0.
1	HIGH	D+ signal currently at logic 1.

Bit	Name	Reset	Access	Description
0	Dn	0	R	**D- Signal Status.** This bit indicates the current logic level of the D- pin.

Value	Name	Description
0	LOW	D- signal currently at logic 0.
1	HIGH	D- signal currently at logic 1.

22.4.2 USB0ADR: USB0 Indirect Address

Bit	7	6	5	4	3	2	1	0
Name	BUSY	AUTORD	USB0ADR					
Access	RW	RW	RW					
Reset	0	0	0x00					
SFR Page = ALL; SFR Address: 0xAE								

Bit	Name	Reset	Access	Description
7	BUSY	0	RW	**USB0 Register Read Busy Flag.** This bit is used during indirect USB0 register accesses.
6	AUTORD	0	RW	**USB0 Register Auto-Read Flag.** This bit is used for block FIFO reads.

Value	Name	Description
0	DISABLED	BUSY must be written manually for each USB0 indirect register read.
1	ENABLED	The next indirect register read will automatically be initiated when firmware reads USB0DAT (USBADDR bits will not be changed).

Bit	Name	Reset	Access	Description
5:0	USB0ADR	0x00	RW	**USB0 Indirect Register Address.** These bits hold a 6-bit address used to indirectly access the USB0 core registers. Reads and writes to USB0DAT will target the register indicated by the USBADDR bits.

22.4.3 USB0DAT: USB0 Data

Bit	7	6	5	4	3	2	1	0
Name	USB0DAT							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xAF								

Bit	Name	Reset	Access	Description
7:0	USB0DAT	0x00	RW	**USB0 Data.** This register is used to indirectly read and write the USB0 register targeted by USB0ADDR.

22.4.4 INDEX: USB0 Endpoint Index

Bit	7	6	5	4	3	2	1	0
Name	Reserved				EPSEL			
Access	R				RW			
Reset	0x0				0x0			
Indirect Address: 0x0E								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3:0	EPSEL	0x0	RW	**Endpoint Select Bits.** This field selects which endpoint is targeted when indexed USB0 registers are accessed.

Value	Name	Description
0x0	ENDPOINT_0	Endpoint 0.
0x1	ENDPOINT_1	Endpoint 1.
0x2	ENDPOINT_2	Endpoint 2.
0x3	ENDPOINT_3	Endpoint 3.

This register is accessed indirectly using the USB0ADR and USB0DAT registers.

22.4.5 CLKREC: USB0 Clock Recovery Control

Bit	7	6	5	4	3	2	1	0
Name	CRE	CRSSEN	CRLOW	Reserved				
Access	RW	RW	RW	RW				
Reset	0	0	0	0x0F				
Indirect Address: 0x0F								

Bit	Name	Reset	Access	Description
7	CRE	0	RW	**Clock Recovery Enable.**
	This bit enables/disables the USB clock recovery feature.			
	Value	Name		Description
	0	DISABLED		Disable clock recovery.
	1	ENABLED		Enable clock recovery.
6	CRSSEN	0	RW	**Clock Recovery Single Step.**
	This bit forces the oscillator calibration into single-step mode during clock recovery.			
	Value	Name		Description
	0	DISABLED		Disable single-step mode (normal calibration mode).
	1	ENABLED		Enable single-step mode.
5	CRLOW	0	RW	**Low Speed Clock Recovery Mode.**
	This bit must be set to 1 if clock recovery is used when operating as a Low Speed USB device.			
	Value	Name		Description
	0	FULL_SPEED		Full Speed Mode.
	1	LOW_SPEED		Low Speed Mode.
4:0	*Reserved*	*Must write reset value.*		
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.6 FIFO0: USB0 Endpoint 0 FIFO Access

Bit	7	6	5	4	3	2	1	0
Name	FIFODATA							
Access	RW							
Reset	0x00							
Indirect Address: 0x20								

Bit	Name	Reset	Access	Description
7:0	FIFODATA	0x00	RW	**Endpoint 0 FIFO Access.** Writing to this FIFO address loads data into the IN FIFO for Endpoint 0. Reading from the FIFO address reads data from the Endpoint 0 OUT FIFO.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.7 FIFO1: USB0 Endpoint 1 FIFO Access

Bit	7	6	5	4	3	2	1	0
Name	FIFODATA							
Access	RW							
Reset	0x00							
Indirect Address: 0x21								

Bit	Name	Reset	Access	Description
7:0	FIFODATA	0x00	RW	**Endpoint 1 FIFO Access.** Writing to this FIFO address loads data into the IN FIFO for Endpoint 1. Reading from the FIFO address reads data from the Endpoint 1 OUT FIFO.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.8 FIFO2: USB0 Endpoint 2 FIFO Access

Bit	7	6	5	4	3	2	1	0
Name	FIFODATA							
Access	RW							
Reset	0x00							
Indirect Address: 0x22								

Bit	Name	Reset	Access	Description
7:0	FIFODATA	0x00	RW	**Endpoint 2 FIFO Access.** Writing to this FIFO address loads data into the IN FIFO for Endpoint 2. Reading from the FIFO address reads data from the Endpoint 2 OUT FIFO.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.9 FIFO3: USB0 Endpoint 3 FIFO Access

Bit	7	6	5	4	3	2	1	0
Name	FIFODATA							
Access	RW							
Reset	0x00							
Indirect Address: 0x23								

Bit	Name	Reset	Access	Description
7:0	FIFODATA	0x00	RW	**Endpoint 3 FIFO Access.** Writing to this FIFO address loads data into the IN FIFO for Endpoint 3. Reading from the FIFO address reads data from the Endpoint 3 OUT FIFO.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.10 FADDR: USB0 Function Address

Bit	7	6	5	4	3	2	1	0
Name	UPDATE	FADDR						
Access	R	RW						
Reset	0	0x00						
Indirect Address: 0x00								

Bit	Name	Reset	Access	Description
7	UPDATE	0	R	**Function Address Update.** Set to 1 when firmware writes the FADDR register. USB0 clears this bit to 0 when the new address takes effect.

Value	Name	Description
0	NOT_SET	The last address written to FADDR is in effect.
1	SET	The last address written to FADDR is not yet in effect.

Bit	Name	Reset	Access	Description
6:0	FADDR	0x00	RW	**Function Address.** This field is the 7-bit function address for USB0. This address should be written by firmware when the SET_ADDRESS standard device request is received on Endpoint 0. The new address takes effect when the device request completes.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.11 POWER: USB0 Power

Bit	7	6	5	4	3	2	1	0
Name	ISOUD	Reserved		USBINH	USBRST	RESUME	SUSMD	SUSEN
Access	RW	RW		RW	RW	RW	R	RW
Reset	0	0x0		0	0	0	0	0
Indirect Address: 0x01								

Bit	Name	Reset	Access	Description
7	ISOUD	0	RW	**Isochronous Update Mode.** This bit affects all IN Isochronous endpoints.

Value	Name	Description
0	IN_TOKEN	When firmware writes INPRDY = 1, USB0 will send the packet when the next IN token is received.
1	SOF_TOKEN	When firmware writes INPRDY = 1, USB0 will wait for a SOF token before sending the packet. If an IN token is received before a SOF token, USB0 will send a zero-length data packet.

Bit	Name	Reset	Access	Description
6:5	*Reserved*	*Must write reset value.*		
4	USBINH	0	RW	**USB0 Inhibit.** This bit is set to 1 following a power-on reset (POR) or an asynchronous USB0 reset. Firmware should clear this bit after the USB0 transceiver initialization is complete. Firmware cannot set this bit to 1.

Value	Name	Description
0	ENABLED	USB0 enabled.
1	DISABLED	USB0 inhibited. All USB traffic is ignored.

Bit	Name	Reset	Access	Description
3	USBRST	0	RW	**Reset Detect.** This bit is set to 1 by hardware when reset signalling is detected on the bus. Upon this detection, the following occur: 1. The USB0 Address is reset (FADDR = 0x00). 2. Endpoint FIFOs are flushed. 3. Control/status registers are reset to 0x00 (E0CSR, EINCSRL, EINCSRH, EOUTCSRL, EOUTCSRH). 4. USB register INDEX is reset to 0x00. 5. All USB interrupts (excluding the suspend interrupt) are enabled and their corresponding flags cleared. 6. A USB Reset interrupt is generated, if enabled.
2	RESUME	0	RW	**Force Resume.** Writing a 1 to this bit while in suspend mode (SUSMD = 1) forces USB0 to generate resume signaling on the bus (a remote wakeup event). Firmware should clear RESUME to 0 after 10 to 15 ms to end the resume signaling. An interrupt is generated, and hardware clears SUSMD, when firmware writes RESUME to 0.
1	SUSMD	0	R	**Suspend Mode.** This bit is set to 1 by hardware when USB0 enters suspend mode. This bit is cleared by hardware when firmware writes RESUME = 0 (following a remote wakeup) or reads the CMINT register after detection of resume signaling on the bus.

Value	Name	Description
0	NOT_SUSPENDED	USB0 not in suspend mode.

Bit	Name	Reset	Access	Description
	1	SUSPENDED		USB0 in suspend mode.
0	SUSEN	0	RW	**Suspend Detection Enable.**
	Value	Name		Description
	0	DISABLED		Disable suspend detection. USB0 will ignore suspend signaling on the bus.
	1	ENABLED		Enable suspend detection. USB0 will enter suspend mode if it detects suspend signaling on the bus.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.12 FRAMEL: USB0 Frame Number Low

Bit	7	6	5	4	3	2	1	0
Name	FRMEL							
Access	R							
Reset	0x00							
Indirect Address: 0x0C								

Bit	Name	Reset	Access	Description
7:0	FRMEL	0x00	R	**Frame Number Low.** This register contains bits 7-0 of the last received frame number.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.13 FRAMEH: USB0 Frame Number High

Bit	7	6	5	4	3	2	1	0
Name	Reserved					FRMEH		
Access	R					R		
Reset	0x00					0x0		
Indirect Address: 0x0D								

Bit	Name	Reset	Access	Description
7:3	*Reserved*	*Must write reset value.*		
2:0	FRMEH	0x0	R	**Frame Number High.** This register contains bits 10-8 of the last received frame number.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.14 IN1INT: USB0 IN Endpoint Interrupt

Bit	7	6	5	4	3	2	1	0
Name	Reserved				IN3	IN2	IN1	EP0
Access	R				R	R	R	R
Reset	0x0				0	0	0	0
Indirect Address: 0x02								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	IN3	0	R	**IN Endpoint 3 Interrupt Flag.** This bit is cleared when firmware reads the IN1INT register.
	Value	Name		Description
	0	NOT_SET		IN Endpoint 3 interrupt inactive.
	1	SET		IN Endpoint 3 interrupt active.
2	IN2	0	R	**IN Endpoint 2 Interrupt Flag.** This bit is cleared when firmware reads the IN1INT register.
	Value	Name		Description
	0	NOT_SET		IN Endpoint 2 interrupt inactive.
	1	SET		IN Endpoint 2 interrupt active.
1	IN1	0	R	**IN Endpoint 1 Interrupt Flag.** This bit is cleared when firmware reads the IN1INT register.
	Value	Name		Description
	0	NOT_SET		IN Endpoint 1 interrupt inactive.
	1	SET		IN Endpoint 1 interrupt active.
0	EP0	0	R	**Endpoint 0 Interrupt Flag.** This bit is cleared when firmware reads the IN1INT register.
	Value	Name		Description
	0	NOT_SET		Endpoint 0 interrupt inactive.
	1	SET		Endpoint 0 interrupt active.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.15 OUT1INT: USB0 OUT Endpoint Interrupt

Bit	7	6	5	4	3	2	1	0
Name	Reserved				OUT3	OUT2	OUT1	Reserved
Access	R				R	R	R	R
Reset	0x0				0	0	0	0
Indirect Address: 0x04								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	OUT3	0	R	**OUT Endpoint 3 Interrupt Flag.** This bit is cleared when firmware reads the OUT1INT register. Value: 0 – NOT_SET – OUT Endpoint 3 interrupt inactive. Value: 1 – SET – OUT Endpoint 3 interrupt active.
2	OUT2	0	R	**OUT Endpoint 2 Interrupt Flag.** This bit is cleared when firmware reads the OUT1INT register. Value: 0 – NOT_SET – OUT Endpoint 2 interrupt inactive. Value: 1 – SET – OUT Endpoint 2 interrupt active.
1	OUT1	0	R	**OUT Endpoint 1 Interrupt Flag.** This bit is cleared when firmware reads the OUT1INT register. Value: 0 – NOT_SET – OUT Endpoint 1 interrupt inactive. Value: 1 – SET – OUT Endpoint 1 interrupt active.
0	*Reserved*	*Must write reset value.*		

This register is accessed indirectly using the USB0ADR and USB0DAT registers.

22.4.16 CMINT: USB0 Common Interrupt

Bit	7	6	5	4	3	2	1	0
Name	Reserved				SOF	RSTINT	RSUINT	SUSINT
Access	R				R	R	R	R
Reset	0x0				0	0	0	0
Indirect Address: 0x06								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	SOF	0	R	**Start of Frame Interrupt Flag.** This bit is set by hardware when a SOF token is received. This interrupt event is synthesized by hardware: an interrupt will be generated when hardware expects to receive a SOF event, even if the actual SOF signal is missed or corrupted. This bit is cleared when firmware reads the CMINT register. Value: 0 – NOT_SET – SOF interrupt inactive. Value: 1 – SET – SOF interrupt active.
2	RSTINT	0	R	**Reset Interrupt Flag.** Set by hardware when reset signaling is detected on the bus. This bit is cleared when firmware reads the CMINT register. Value: 0 – NOT_SET – Reset interrupt inactive. Value: 1 – SET – Reset interrupt active.
1	RSUINT	0	R	**Resume Interrupt Flag.** Set by hardware when resume signaling is detected on the bus while USB0 is in suspend mode. This bit is cleared when firmware reads the CMINT register. Value: 0 – NOT_SET – Resume interrupt inactive. Value: 1 – SET – Resume interrupt active.
0	SUSINT	0	R	**Suspend Interrupt Flag.** When suspend detection is enabled (bit SUSEN in register POWER), this bit is set by hardware when suspend signaling is detected on the bus. This bit is cleared when firmware reads the CMINT register. Value: 0 – NOT_SET – Suspend interrupt inactive. Value: 1 – SET – Suspend interrupt active.

This register is accessed indirectly using the USB0ADR and USB0DAT registers.

22.4.17 IN1IE: USB0 IN Endpoint Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	Reserved				IN3E	IN2E	IN1E	EP0E
Access	R				RW	RW	RW	RW
Reset	0x0				1	1	1	1
Indirect Address: 0x07								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	IN3E	1	RW	**IN Endpoint 3 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 3 IN interrupts.
	1	ENABLED		Enable Endpoint 3 IN interrupts.
2	IN2E	1	RW	**IN Endpoint 2 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 2 IN interrupts.
	1	ENABLED		Enable Endpoint 2 IN interrupts.
1	IN1E	1	RW	**IN Endpoint 1 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 1 IN interrupts.
	1	ENABLED		Enable Endpoint 1 IN interrupts.
0	EP0E	1	RW	**Endpoint 0 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 0 interrupts.
	1	ENABLED		Enable Endpoint 0 interrupts.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.18 OUT1IE: USB0 OUT Endpoint Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	Reserved				OUT3E	OUT2E	OUT1E	Reserved
Access	R				RW	RW	RW	R
Reset	0x0				1	1	1	0
Indirect Address: 0x09								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	OUT3E	1	RW	**OUT Endpoint 3 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 3 OUT interrupts.
	1	ENABLED		Enable Endpoint 3 OUT interrupts.
2	OUT2E	1	RW	**OUT Endpoint 2 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 2 OUT interrupts.
	1	ENABLED		Enable Endpoint 2 OUT interrupts.
1	OUT1E	1	RW	**OUT Endpoint 1 Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable Endpoint 1 OUT interrupts.
	1	ENABLED		Enable Endpoint 1 OUT interrupts.
0	*Reserved*	*Must write reset value.*		
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.19 CMIE: USB0 Common Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	Reserved				SOFE	RSTINTE	RSUINTE	SUSINTE
Access	R				RW	RW	RW	RW
Reset	0x0				0	1	1	0
Indirect Address: 0x0B								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	SOFE	0	RW	**Start of Frame Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable SOF interrupts.
	1	ENABLED		Enable SOF interrupts.
2	RSTINTE	1	RW	**Reset Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable reset interrupts.
	1	ENABLED		Enable reset interrupts.
1	RSUINTE	1	RW	**Resume Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable resume interrupts.
	1	ENABLED		Enable resume interrupts.
0	SUSINTE	0	RW	**Suspend Interrupt Enable.**
	Value	Name		Description
	0	DISABLED		Disable suspend interrupts.
	1	ENABLED		Enable suspend interrupts.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.20 E0CSR: USB0 Endpoint0 Control

Bit	7	6	5	4	3	2	1	0
Name	SSUEND	SOPRDY	SDSTL	SUEND	DATAEND	STSTL	INPRDY	OPRDY
Access	RW	RW	RW	R	RW	RW	RW	R
Reset	0	0	0	0	0	0	0	0
Indirect Address: 0x11								

Bit	Name	Reset	Access	Description
7	SSUEND	0	RW	**Serviced Setup End.** Firmware should set this bit to 1 after servicing a setup end (SUEND) event. Hardware clears the SUEND bit when firmware writes 1 to SSUEND.
6	SOPRDY	0	RW	**Serviced OPRDY.** Firmware should write 1 to this bit after servicing a received Endpoint 0 packet. The OPRDY bit will be cleared by a write of 1 to SOPRDY.
5	SDSTL	0	RW	**Send Stall.** Firmware can write 1 to this bit to terminate the current transfer (due to an error condition, unexpected transfer request, etc.). Hardware will clear this bit to 0 when the STALL handshake is transmitted.
4	SUEND	0	R	**Setup End.** Hardware sets this read-only bit to 1 when a control transaction ends before firmware has written 1 to the DATAEND bit. Hardware clears this bit when firmware writes 1 to SSUEND.
3	DATAEND	0	RW	**Data End.** Firmware should write 1 to this bit: 1. When writing 1 to INPRDY for the last outgoing data packet. 2. When writing 1 to INPRDY for a zero-length data packet. 3. When writing 1 to SOPRDY after servicing the last incoming data packet. This bit is automatically cleared by hardware.
2	STSTL	0	RW	**Sent Stall.** Hardware sets this bit to 1 after transmitting a STALL handshake signal. This flag must be cleared by firmware.
1	INPRDY	0	RW	**IN Packet Ready.** Firmware should write 1 to this bit after loading a data packet into the Endpoint 0 FIFO for transmit. Hardware clears this bit and generates an interrupt under one of the following conditions: 1. The packet is transmitted. 2. The packet is overwritten by an incoming SETUP packet. 3. The packet is overwritten by an incoming OUT packet.
0	OPRDY	0	R	**OUT Packet Ready.** Hardware sets this read-only bit and generates an interrupt when a data packet has been received. This bit is cleared only when firmware writes 1 to the SOPRDY bit.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.21 E0CNT: USB0 Endpoint0 Data Count

Bit	7	6	5	4	3	2	1	0
Name	Reserved	E0CNT						
Access	R	R						
Reset	0	0x00						
Indirect Address: 0x16								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:0	E0CNT	0x00	R	**Endpoint 0 Data Count.** This 7-bit number indicates the number of received data bytes in the Endpoint 0 FIFO. This number is only valid while OPRDY is 1.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.22 EENABLE: USB0 Endpoint Enable

Bit	7	6	5	4	3	2	1	0
Name	Reserved				EEN3	EEN2	EEN1	Reserved
Access	R				RW	RW	RW	RW
Reset	0x1				1	1	1	1
Indirect Address: 0x1E								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	EEN3	1	RW	**Endpoint 3 Enable.** This bit enables or disables Endpoint 3.
	Value	Name		Description
	0	DISABLED		Disable Endpoint 3 (no NACK, ACK, or STALL on the USB network).
	1	ENABLED		Enable Endpoint 3 (normal).
2	EEN2	1	RW	**Endpoint 2 Enable.** This bit enables or disables Endpoint 2.
	Value	Name		Description
	0	DISABLED		Disable Endpoint 2 (no NACK, ACK, or STALL on the USB network).
	1	ENABLED		Enable Endpoint 2 (normal).
1	EEN1	1	RW	**Endpoint 1 Enable.** This bit enables or disables Endpoint 1.
	Value	Name		Description
	0	DISABLED		Disable Endpoint 1 (no NACK, ACK, or STALL on the USB network).
	1	ENABLED		Enable Endpoint 1 (normal).
0	*Reserved*	*Must write reset value.*		
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.23 EINCSRL: USB0 IN Endpoint Control Low

Bit	7	6	5	4	3	2	1	0
Name	Reserved	CLRDT	STSTL	SDSTL	FLUSH	UNDRUN	FIFONE	INPRDY
Access	R	W	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
Indirect Address: 0x11								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	CLRDT	0	W	**Clear Data Toggle.**
5	STSTL	0	RW	**Sent Stall Flag.** Hardware sets this bit to 1 when a STALL handshake signal is transmitted. The FIFO is flushed, and the INPRDY bit cleared. This flag must be cleared by firmware.
4	SDSTL	0	RW	**Send Stall.** Firmware should set this bit to 1 to generate a STALL handshake in response to an IN token. Firmware should clear this bit to 0 to terminate the STALL signal. This bit has no effect in Isochronous mode.
3	FLUSH	0	RW	**FIFO Flush.** Writing a 1 to this bit flushes the next packet to be transmitted from the IN Endpoint FIFO. The FIFO pointer is reset and the INPRDY bit is cleared. If the FIFO contains multiple packets, firmware must write 1 to FLUSH for each packet. Hardware resets the FLUSH bit to 0 when the FIFO flush is complete.
2	UNDRUN	0	RW	**Data Underrun Flag.** The function of this bit depends on the IN Endpoint mode: Isochronous: Set when a zero-length packet is sent after an IN token is received while bit INPRDY = 0. Interrupt/Bulk: Set when a NAK is returned in response to an IN token. This bit must be cleared by firmware.
1	FIFONE	0	RW	**FIFO Not Empty.**
	Value	Name		Description
	0	EMPTY		The IN Endpoint FIFO is empty.
	1	NOT_EMPTY		The IN Endpoint FIFO contains one or more packets.
0	INPRDY	0	RW	**In Packet Ready.** Firmware should write 1 to this bit after loading a data packet into the IN Endpoint FIFO. Hardware clears INPRDY due to any of the following: 1. A data packet is transmitted. 2. Double buffering is enabled (DBIEN = 1) and there is an open FIFO packet slot. 3. If the endpoint is in Isochronous Mode (ISO = 1) and ISOUD = 1, INPRDY will read 0 until the next SOF is received. An interrupt (if enabled) will be generated when hardware clears INPRDY as a result of a packet being transmitted.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.24 EINCSRH: USB0 IN Endpoint Control High

Bit	7	6	5	4	3	2	1	0
Name	DBIEN	ISO	DIRSEL	Reserved	FCDT	SPLIT	Reserved	
Access	RW	RW	RW	R	RW	RW	R	
Reset	0	0	0	0	0	0	0x0	
Indirect Address: 0x12								

Bit	Name	Reset	Access	Description
7	DBIEN	0	RW	**IN Endpoint Double-Buffer Enable.**
	Value	Name		Description
	0	DISABLED		Disable double-buffering for the selected IN endpoint.
	1	ENABLED		Enable double-buffering for the selected IN endpoint.
6	ISO	0	RW	**Isochronous Transfer Enable.**
	This bit enables or disables Isochronous transfers on the current endpoint.			
	Value	Name		Description
	0	DISABLED		Endpoint configured for Bulk/Interrupt transfers.
	1	ENABLED		Endpoint configured for Isochronous transfers.
5	DIRSEL	0	RW	**Endpoint Direction Select.**
	This bit is valid only when the selected FIFO is not split (SPLIT = 0).			
	Value	Name		Description
	0	OUT		Endpoint direction selected as OUT.
	1	IN		Endpoint direction selected as IN.
4	*Reserved*	*Must write reset value.*		
3	FCDT	0	RW	**Force Data Toggle.**
	Value	Name		Description
	0	ACK_TOGGLE		Endpoint data toggle switches only when an ACK is received following a data packet transmission.
	1	ALWAYS_TOGGLE		Endpoint data toggle forced to switch after every data packet is transmitted, regardless of ACK reception.
2	SPLIT	0	RW	**FIFO Split Enable.**
	When this bit is set to 1, the selected endpoint FIFO is split. The upper half of the selected FIFO is used by the IN endpoint, and the lower half of the selected FIFO is used by the OUT endpoint.			
1:0	*Reserved*	*Must write reset value.*		
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.25 EOUTCSRL: USB0 OUT Endpoint Control Low

Bit	7	6	5	4	3	2	1	0
Name	CLRDT	STSTL	SDSTL	FLUSH	DATERR	OVRUN	FIFOFUL	OPRDY
Access	W	RW	RW	RW	R	RW	R	RW
Reset	0	0	0	0	0	0	0	0
Indirect Address: 0x14								

Bit	Name	Reset	Access	Description
7	CLRDT	0	W	**Clear Data Toggle.**
6	STSTL	0	RW	**Sent Stall Flag.** Hardware sets this bit to 1 when a STALL handshake signal is transmitted. This flag must be cleared by firmware.
5	SDSTL	0	RW	**Send Stall.** Firmware should set this bit to 1 to generate a STALL handshake. Firmware should clear this bit to 0 to terminate the STALL signal. This bit has no effect in Isochronous mode.
4	FLUSH	0	RW	**FIFO Flush.** Writing a 1 to this bit flushes the next packet to be read from the OUT endpoint FIFO. The FIFO pointer is reset and the OPRDY bit is cleared. Multiple packets must be flushed individually. Hardware resets the FLUSH bit to 0 when the flush is complete. If data for the current packet has already been read from the FIFO, the FLUSH bit should not be used to flush the packet. Instead, the FIFO should be read manually.
3	DATERR	0	R	**Data Error Flag.** In Isochronous mode, this bit is set by hardware if a received packet has a CRC or bit-stuffing error. It is cleared when firmware clears OPRDY. This bit is only valid in Isochronous mode.
2	OVRUN	0	RW	**Data Overrun Flag.** This bit is set by hardware when an incoming data packet cannot be loaded into the OUT Endpoint FIFO. This bit is only valid in Isochronous mode and must be cleared by firmware.

Value	Name	Description
0	NOT_SET	No data overrun.
1	SET	A data packet was lost because of a full FIFO since this flag was last cleared.

1	FIFOFUL	0	R	**OUT FIFO Full.** This bit indicates the contents of the OUT FIFO. If double buffering is enabled (DBIEN = 1), the FIFO is full when the FIFO contains two packets. If DBIEN = 0, the FIFO is full when the FIFO contains one packet.

Value	Name	Description
0	NOT_FULL	OUT endpoint FIFO is not full.
1	FULL	OUT endpoint FIFO is full.

0	OPRDY	0	RW	**OUT Packet Ready.** Hardware sets this bit to 1 and generates an interrupt when a data packet is available. Firmware should clear this bit after each data packet is read from the OUT endpoint FIFO.

This register is accessed indirectly using the USB0ADR and USB0DAT registers.

22.4.26 EOUTCSRH: USB0 OUT Endpoint Control High

Bit	7	6	5	4	3	2	1	0
Name	DBOEN	ISO	Reserved					
Access	RW	RW	R					
Reset	0	0	0x00					
Indirect Address: 0x15								

Bit	Name	Reset	Access	Description
7	DBOEN	0	RW	**Double-Buffer Enable.**
	Value	Name		Description
	0	DISABLED		Disable double-buffering for the selected OUT endpoint.
	1	ENABLED		Enable double-buffering for the selected OUT endpoint.
6	ISO	0	RW	**Isochronous Transfer Enable.**
	This bit enables or disables Isochronous transfers on the current endpoint.			
	Value	Name		Description
	0	DISABLED		Endpoint configured for Bulk/Interrupt transfers.
	1	ENABLED		Endpoint configured for Isochronous transfers.
5:0	*Reserved*	*Must write reset value.*		
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.27 EOUTCNTL: USB0 OUT Endpoint Count Low

Bit	7	6	5	4	3	2	1	0
Name	EOCL							
Access	R							
Reset	0x00							
Indirect Address: 0x16								

Bit	Name	Reset	Access	Description
7:0	EOCL	0x00	R	**OUT Endpoint Count Low.**
	EOCL holds the lower 8-bits of the 10-bit number of data bytes in the last received packet in the current OUT endpoint FIFO. This number is only valid while OPRDY = 1.			
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.28 EOUTCNTH: USB0 OUT Endpoint Count High

Bit	7	6	5	4	3	2	1	0
Name	Reserved						EOCH	
Access	R						R	
Reset	0x00						0x0	
Indirect Address: 0x17								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1:0	EOCH	0x0	R	**OUT Endpoint Count High.** EOCH holds the upper 2-bits of the 10-bit number of data bytes in the last received packet in the current OUT endpoint FIFO. This number is only valid while OPRDY = 1.
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

22.4.29 USB0CF: USB0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	VBUSEN	VBUSIE	VBUSI	Reserved		USBCLK		
Access	RW	RW	RW	R		RW		
Reset	0	0	0	0x0		0x7		
SFR Page = 0x20; SFR Address: 0xB5								

Bit	Name	Reset	Access	Description
7	VBUSEN	0	RW	**VBUS Sense Enable.** Enables the VBUS function on the VBUS I/O line.
	Value	Name		Description
	0	DISABLED		The VBUS pin can be used as GPIO.
	1	ENABLED		The VBUS pin is used to sense the USB VBUS signal voltage.
6	VBUSIE	0	RW	**VBUS Interrupt Enable.** Enables VBUS as an interrupt source.
	Value	Name		Description
	0	DISABLED		The VBUS sense signal will not generate interrupts.
	1	ENABLED		The VBUS sense signal may generate VBUS interrupts when a VBUS event is deteted. The VBUS interrupt must be enabled in the main interrupt controller for the interrupt to occur.
5	VBUSI	0	RW	**VBUS Interrupt.** The VBUS interrupt flag indicates when a VBUS sense event has occurred. If VBUS interrupts are enabled, an interrupt will be generated any time the VBUS line transitions from high to low or low to high. This bit must be cleared by firmware.
	Value	Name		Description
	0	NOT_SET		A VBUS event has not occurred.
	1	SET		A VBUS event has occurred.
4:3	*Reserved*	*Must write reset value.*		
2:0	USBCLK	0x7	RW	**USB0 Clock Source Select Bits.**
	Value	Name		Description
	0x0	HFOSC1		USB0 clock (USB0CLK) derived from High-Frequency Oscillator 1 (HFOSC1).
	0x1	HFOSC1_DIV_8		USB0 clock (USB0CLK) derived from High-Frequency Oscillator 1 / 8 (HFOSC1 / 8).
	0x2	EXTOSC		USB0 clock (USB0CLK) derived from the External Oscillator.
	0x3	EXTOSC_DIV_2		USB0 clock (USB0CLK) derived from the External Oscillator / 2.
	0x4	EXTOSC_DIV_3		USB0 clock (USB0CLK) derived from the External Oscillator / 3.
	0x5	EXTOSC_DIV_4		USB0 clock (USB0CLK) derived from the External Oscillator / 4.
	0x6	LFOSC		USB0 clock (USB0CLK) derived from the Internal Low-Frequency Oscillator.
	0x7	NOCLOCK		USB0 clock (USB0CLK) is turned off.

22.4.30 USB0AEC: USB0 Advanced Energy Control

Bit	7	6	5	4	3	2	1	0
Name	LEMSTA	XCVRMD	OSCMD		Reserved		LEMCN	
Access	R	RW	RW		R		RW	
Reset	0	0	0x0		0x0		0x0	
SFR Page = 0x20; SFR Address: 0xB2								

Bit	Name	Reset	Access	Description
7	LEMSTA	0	R	**Low Energy Mode Status.**
	This bit indicates whether low energy mode mode is active, due to current USB bus conditions.			
	Value	Name		Description
	0	OFF		The USB peripheral is in normal mode.
	1	ON		The USB peripheral is in a low energy mode.
6	XCVRMD	0	RW	**Transceiver Mode.**
	This bit controls how the USB transceiver hardware is affected by low energy mode.			
	Value	Name		Description
	0	LOW_POWER		The USB transceiver is selectively put into a lower power state when low energy mode is active.
	1	NORMAL_POWER		The USB transceiver is not affected by low energy mode.
5:4	OSCMD	0x0	RW	**Low Energy Mode Oscillator Control.**
	This field configures how LE mode affects USB clocking. It should be set to 00 in most applications.			
	Value	Name		Description
	0x0	OSC_GATED_SUSPEND		The USB clock source is selectively gated by LE mode, and the High-Frequency Oscillator (HFOSC1) is suspended if possible.
	0x1	OSC_GATED		The USB clock source is selectively gated by LE mode. There is no effect to HFOSC1.
	0x3	OSC_ON		LE mode has no effect on either the USB clock or HFOSC1.
3:2	*Reserved*	*Must write reset value.*		
1:0	LEMCN	0x0	RW	**Low Energy Mode Control.**
	The LEMCN bits control when Low Energy Mode is used. It is recommended that this field is set to 11 when LE mode is used.			
	Value	Name		Description
	0x0	DISABLED		LE mode is disabled.
	0x1	IDLE		LE takes effect only during idle bus times.
	0x2	NAK_OUT		LE takes effect only during NAKed OUT packets.
	0x3	ALL		LE takes effect during idle bus and NAKed OUT packets.

22.4.31 USB0CDCF: USB0 Charger Detect Configuration

Bit	7	6	5	4	3	2	1	0
Name	Reserved				SDIE	PDIE	DCDIE	Reserved
Access	R				RW	RW	RW	R
Reset	0x0				0	0	0	0
SFR Page = 0x20; SFR Address: 0xB6								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	SDIE	0	RW	**SD Interrupt Enable.** Enables the SDI flag as an interrupt source.
	Value	Name		Description
	0	DISABLED		SDI will not generate charger detect interrupts.
	1	ENABLED		SDI allowed to generate charger detect interrupts.
2	PDIE	0	RW	**PD Interrupt Enable.** Enables the PDI flag as an interrupt source.
	Value	Name		Description
	0	DISABLED		PDI will not generate charger detect interrupts.
	1	ENABLED		PDI allowed to generate charger detect interrupts.
1	DCDIE	0	RW	**DCD Interrupt Enable.** Enables the DCDI flag as an interrupt source.
	Value	Name		Description
	0	DISABLED		DCDI will not generate charger detect interrupts.
	1	ENABLED		DCDI allowed to generate charger detect interrupts.
0	*Reserved*	*Must write reset value.*		

22.4.32 USB0CDCN: USB0 Charger Detect Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved			CHDEN	SDEN	PDEN	DCDEN	
Access	R			RW	RW	RW	RW	
Reset	0x0			0	0	0	0x0	
SFR Page = 0x20; SFR Address: 0xBE								

Bit	Name	Reset	Access	Description
7:5	*Reserved*	*Must write reset value.*		
4	CHDEN	0	RW	**Charger Detection Enable.** This bit enables the charger detect circuitry. When CHDEN is set to 1, the circuit will perform any enabled functions in the sequence: DCD, PD, SD. Any disabled functions will be skipped. If VBUS is used as a separate signal, charger detect will not begin until VBUS is detected as high.
	Value	Name		Description
	0	DISABLED		Disable the charger detection block.
	1	ENABLED		Enable the charger detection block.
3	SDEN	0	RW	**Secondary Detection Enable.** This bit enables secondary detection (SD) when CHDEN is set to 1. SD will occur on completion of PD or when PDEN is 0 and DCDEN is 00. When SD finishes, the SDEN bit will return to 0, and the SDI flag will be asserted.
	Value	Name		Description
	0	DISABLED		Disable secondary detection.
	1	ENABLED		Enable secondary detection.
2	PDEN	0	RW	**Primary Detection Enable.** This bit enables primary detection (PD) when CHDEN is set to 1. PD will occur on completion of DCD or when DCDEN is 00. When PD finishes, the PDEN bit will return to 0, and the PDI flag will be asserted.
	Value	Name		Description
	0	DISABLED		Disable primary detection.
	1	ENABLED		Enable primary detection.
1:0	DCDEN	0x0	RW	**Data Contact Detection Enable.** This field enables and configures the data contact detection (DCD) feature when CHDEN is also set to 1. When DCD is complete, the DCDEN field will return to 00, and the DCDI flag will be asserted.
	Value	Name		Description
	0x0	DISABLED		DCD is disabled.
	0x2	TIMEOUT		Only DCD timeout will be initiated.
	0x3	ENABLED		Full DCD operation (physical contact and timeout) will be initiated.

22.4.33 USB0CDSTA: USB0 Charger Detect Status

Bit	7	6	5	4	3	2	1	0
Name	ERR	SDP	CDP	DCP	SDI	PDI	DCDI	DCDTO
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x20; SFR Address: 0xBF

Bit	Name	Reset	Access	Description
7	ERR	0	RW	**USB Charger Detection Error.**

This bit indicates that an error occurred during the charger detect sequence. This bit will be set if the VBUS signal is disconnected while the charger detect circuit is active.

Value	Name	Description
0	NO_ERROR	No error has occurred.
1	ERROR	An error has occurred. If enabled, a USB charger detect interrupt will be triggered. This bit must be cleared by firmware.

Bit	Name	Reset	Access	Description
6	SDP	0	RW	**Standard Downstream Port Detected.**

This bit is set at the completion of a primary detection phase if a Standard Downstream Port has been deteted.

Bit	Name	Reset	Access	Description
5	CDP	0	RW	**Charging Downstream Port Detected.**

This bit is set at the completion of a secondary detection phase if a Chargring Downstream Port has been deteted.

Bit	Name	Reset	Access	Description
4	DCP	0	RW	**Dedicated Charging Port Detected.**

This bit is set at the completion of a secondary detection phase if a Dedicated Chargring Port has been deteted.

Bit	Name	Reset	Access	Description
3	SDI	0	RW	**Secondary Detection Complete.**

This bit is set at the completion of a SD operation.

Value	Name	Description
0	NOT_SET	SD operation has not completed.
1	SET	SD operation has completed. If SDIE is set to 1 a charger detect interrupt may be generated. This flag must be cleared by firmware.

Bit	Name	Reset	Access	Description
2	PDI	0	RW	**Primary Detection Complete.**

This bit is set at the completion of a PD operation.

Value	Name	Description
0	NOT_SET	PD operation has not completed.
1	SET	PD operation has completed. If PDIE is set to 1 a charger detect interrupt may be generated. This flag must be cleared by firmware.

Bit	Name	Reset	Access	Description
1	DCDI	0	RW	**Data Contact Detect Complete.**

This bit is set at the completion of a DCD operation. The DCDTO bit will indicate whether the DCD operation timed out.

Value	Name	Description
0	NOT_SET	DCD operation has not completed.
1	SET	DCD operation has completed. If DCDIE is set to 1 a charger detect interrupt may be generated. This flag must be cleared by firmware.

Bit	Name	Reset	Access	Description
0	DCDTO	0	RW	**Data Contact Detection Timeout.**

This bit is set at the completion of a DCD operation if the operation was stopped due to DCD timeout.

Value	Name	Description
0	NO_TIMEOUT	A DCD timeout was not triggered.
1	TIMEOUT	A DCD timeout was triggered.

23. Watchdog Timer (WDT0)

23.1 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).

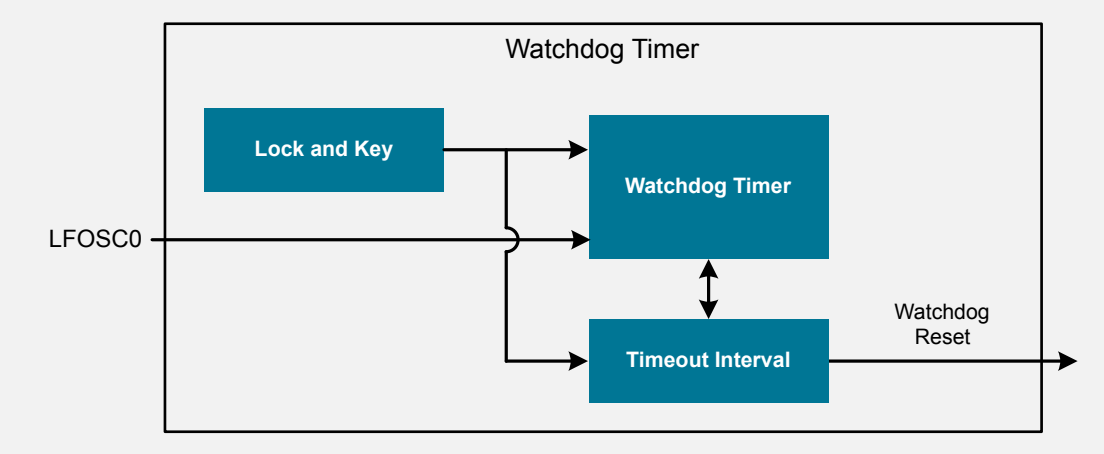


Figure 23.1. Watchdog Timer Block Diagram

23.2 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

23.3 Using the Watchdog Timer

Enabling/Resetting the WDT

The watchdog timer is both enabled and reset by 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 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
SETB EA             ; re-enable interrupts
```

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.

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.

23.4 WDT0 Control Registers

23.4.1 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.

24. C2 Debug and Programming Interface

24.1 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.

24.2 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.

24.3 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.

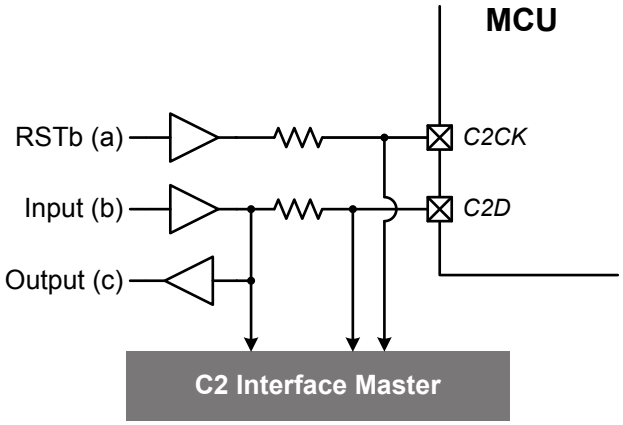


Figure 24.1. Typical C2 Pin Sharing

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.

24.4 C2 Interface Registers

24.4.1 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

24.4.2 C2DEVID: C2 Device ID

Bit	7	6	5	4	3	2	1	0
Name	C2DEVID							
Access	R							
Reset	0x32							
C2 Address: 0x00								

Bit	Name	Reset	Access	Description
7:0	C2DEVID	0x32	R	**Device ID.** This read-only register returns the 8-bit device ID.

24.4.3 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: 0x02 = Revision A.

24.4.4 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.

24.4.5 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

Table of Contents

Simpilcity Studio

One-click access to MCU tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

www.silabs.com/simplicity

MCU Portfolio
www.silabs.com/mcu

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com