EFM8 Universal Bee Family
EFM8UB2 Reference Manual

The EFM8UB2, part of the Universal Bee family of MCUs, is a multi-purpose line of 8-bit microcontrollers with USB feature set.

These devices offer high value by integrating a USB peripheral interface with a high precision oscillator, clock recovery circuit, and integrated transceiver, making them ideal for any full speed USB applications with no external components required. With an efficient 8051 core and precision analog, the EFM8UB2 family is also optimal for embedded applications.

EFM8UB2 applications include the following:

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

KEY FEATURES

- Pipelined 8-bit 8051 MCU Core with 48 MHz maximum operating frequency
- Up to 40 multifunction I/O pins
- Crystal-less full speed/low speed USB 2.0 compliant controller with 1 KB buffer memory
- One differential 10-bit ADC and two analog comparators
- Internal 48 MHz oscillator with ±0.25% accuracy with USB clock recovery supports crystal-free USB and UART operation
- 2 UARTs, SPI, 2 SMBus/I2C serial communications

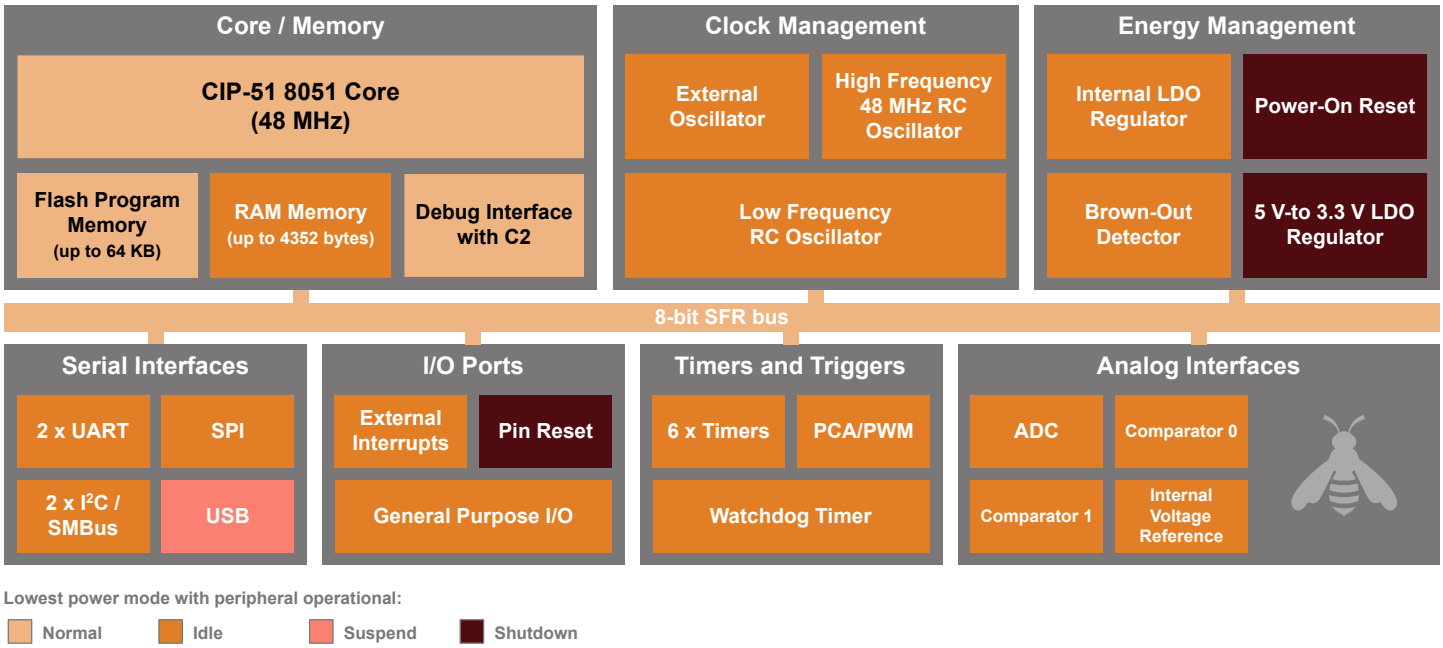


Table of Contents

1. System Overview

1.1 Introduction

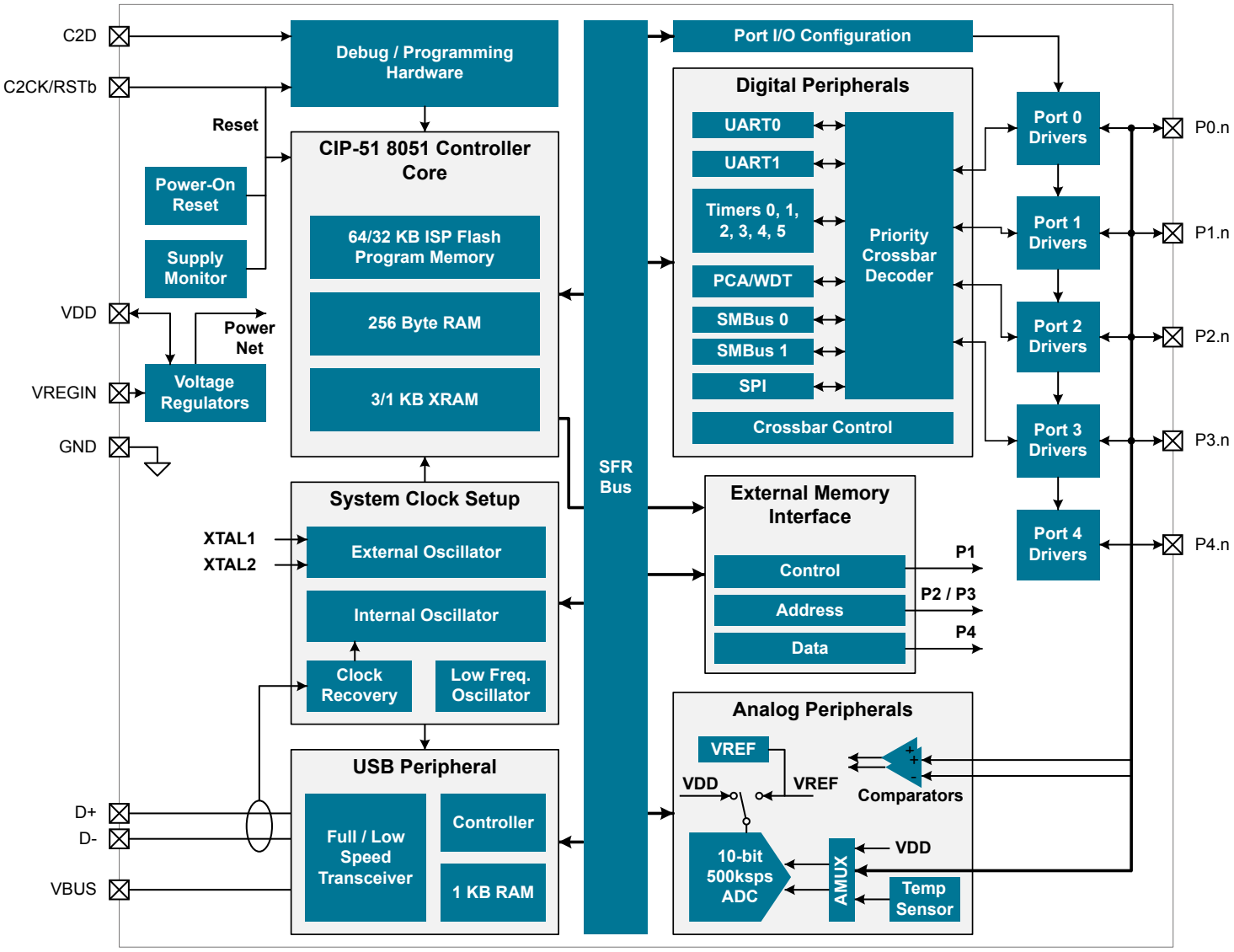


Figure 1.1. Detailed EFM8UB2 Block Diagram

This section describes the EFM8UB2 family at a high level. For more information on each module including register definitions, see the EFM8UB2 Reference Manual.

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 • Code resumes execution on wake event	1. Switch SYSCLK to HFOSC0 2. Set SUSPEND bit in HFO0CN	USB0 Bus Activity
Stop	• All internal power nets shut down • Pins retain state • Exit on any reset source	Set STOP bit in PCON0	Any reset source
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 REG01CN 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-P3.7 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 P4.0-P4.7 can be used as GPIO. Additionally, the C2 Interface Data signal (C2D) is shared with P3.0 on some packages.

The port control block offers the following features:

- Up to 40 multi-functions I/O pins, supporting digital and analog functions.
- Flexible priority crossbar decoder for digital peripheral assignment.
- Two direct-pin interrupt sources with dedicated interrupt vectors (INT0 and INT1) available on P0 pins.

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 48 MHz oscillator divided by 4, then divided by 8 (1.5 MHz).

- Provides clock to core and peripherals.
- 48 MHz internal oscillator (HFOSC0), accurate to ±1.5% over supply and temperature corners: accurate to +/- 0.25% when using USB clock recovery.
- 80 kHz low-frequency oscillator (LFOSC0).
- External RC, C, CMOS, and high-frequency crystal clock options (EXTCLK) for QFP48 packages.
- External CMOS clock option (EXTCLK) for QFP32 and QFN32 packages.
- Internal oscillator has clock divider with eight settings for flexible clock scaling: 1, 2, 4, or 8.

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 five independently-configurable channels
- 8- or 16-bit PWM modes (edge-aligned operation).
- Frequency output mode.
- Capture on rising, falling or any edge.
- Compare function for arbitrary waveform generation.
- Software timer (internal compare) mode.
- Integrated watchdog timer.

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

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

Timer 0 and Timer 1 include the following features:

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

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

- Clock sources include SYSCLK, SYSCLK divided by 12, or the External Clock divided by 8.
- 16-bit auto-reload timer mode
- Dual 8-bit auto-reload timer mode
- USB start-of-frame or falling edge of LFOSC0 capture (Timer 2 and Timer 3)

Watchdog Timer (WDT0)

The device includes a programmable watchdog timer (WDT) integrated within the PCA0 peripheral. 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. The state of the RSTb pin is unaffected by this reset.

The Watchdog Timer integrated in the PCA0 peripheral has the following features:

- Programmable timeout interval
- Runs from the selected PCA clock source
- Automatically enabled after any system reset

1.6 Communications and Other Digital Peripherals

Universal Serial Bus (USB0)

The USB0 module provides Full/Low Speed function for USB peripheral implementations. The USB function controller (USB0) consists of a Serial Interface Engine (SIE), USB transceiver (including matching resistors and configurable pull-up resistors), 1 KB FIFO block, and clock recovery mechanism for crystal-less operation. No external components are required. The USB0 module is Universal Serial Bus Specification 2.0 compliant.

The USB0 module includes the following features:

- Full and Low Speed functionality.
- Implements 4 bidirectional endpoints.
- 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.

Universal Asynchronous Receiver/Transmitter (UART0)

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

The UART module provides the following features:

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

Universal Asynchronous Receiver/Transmitter (UART1)

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

UART1 provides the following features:

- Asynchronous transmissions and receptions.
- Dedicated baud rate generator supports baud rates up to SYSCLK/2 (transmit) or SYSCLK/8 (receive)
- 5, 6, 7, 8, or 9 bit data.
- Automatic start and stop generation.
- Automatic parity generation and checking.
- Three byte FIFO on receive.

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.

The SPI module includes the following features:

- Supports 3- or 4-wire operation in master or slave modes.
- Supports external clock frequencies up to SYSCLK / 2 in master mode and SYSCLK / 10 in slave mode.
- Support for four clock phase and polarity options.
- 8-bit dedicated clock clock rate generator.
- Support for multiple masters on the same data lines.

System Management Bus / I2C (SMB0 and SMB1)

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 modules include 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.

External Memory Interface (EMIF0)

The External Memory Interface (EMIF) enables access of off-chip memories and memory-mapped devices connected to the GPIO ports. The external memory space may be accessed using the external move instruction (MOVX) with the target address specified in either 8-bit or 16-bit formats.

- Supports multiplexed and non-multiplexed memory access.
- Four external memory modes:
  - Internal only.
  - Split mode without bank select.
  - Split mode with bank select.
  - External only
- Configurable ALE (address latch enable) timing.
- Configurable address setup and hold times.
- Configurable write and read pulse widths.

1.7 Analog

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

The ADC is a successive-approximation-register (SAR) ADC with 10-bit mode, 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.

The ADC module is a Successive Approximation Register (SAR) Analog to Digital Converter (ADC). The key features of this ADC module are:

- Up to 32 external inputs.
- Differential or Single-ended 10-bit operation.
- Supports an output update rate of 500 ksps samples per second.
- Asynchronous hardware conversion trigger, selectable between software, external I/O and internal timer sources.
- Output data window comparator allows automatic range checking.
- Two tracking mode options with programmable tracking time.
- Conversion complete and window compare interrupts supported.
- Flexible output data formatting.
- Voltage reference selectable from external reference pin, on-chip precision reference (driven externally on reference pin), or VDD supply.
- 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 module includes the following features:

- Up to 5 external positive inputs.
- Up to 5 external negative inputs.
- 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.

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 EFM8UB2 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 the last three pages of code flash, which includes the code security page; it can be erased if it is not needed. In applications where code security is a concern, Silicon Labs recommends the bootloader be disabled and the flash memory locked after the production programming step. More information about the factory bootloader protocol, usage, customization, and best practices can be found in *AN945: EFM8 Factory Bootloader User's Guide*.

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

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

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

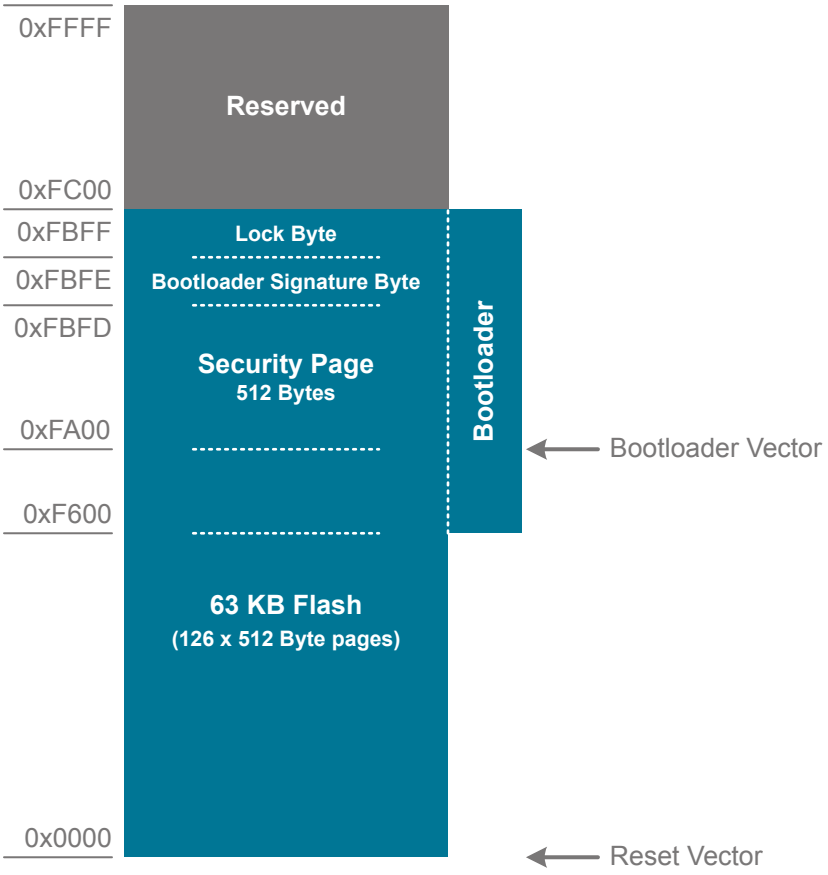


Figure 1.2. Flash Memory Map with Bootloader—64 KB Devices

Table 1.2. Summary of Pins for Bootloader Communication

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

Bootloader	Pins for Bootload Communication
USB	VBUS
	D+
	D-

Table 1.3. Summary of Pins for Bootload Mode Entry

Device Package	Pin for Bootload Mode Entry
QFN48	P3.7
QFP32	P3.0 / C2D
QFN32	P3.0 / C2D

2. Memory Organization

2.1 Memory Organization

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

2.2 Program Memory

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

MOVX Instruction and Program Memory

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

2.3 Data Memory

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

Internal RAM

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

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

General Purpose Registers

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

Bit Addressable Locations

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

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

```
Mov  C, 22.3h
```

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

Stack

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

External RAM

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

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

2.4 Memory Map

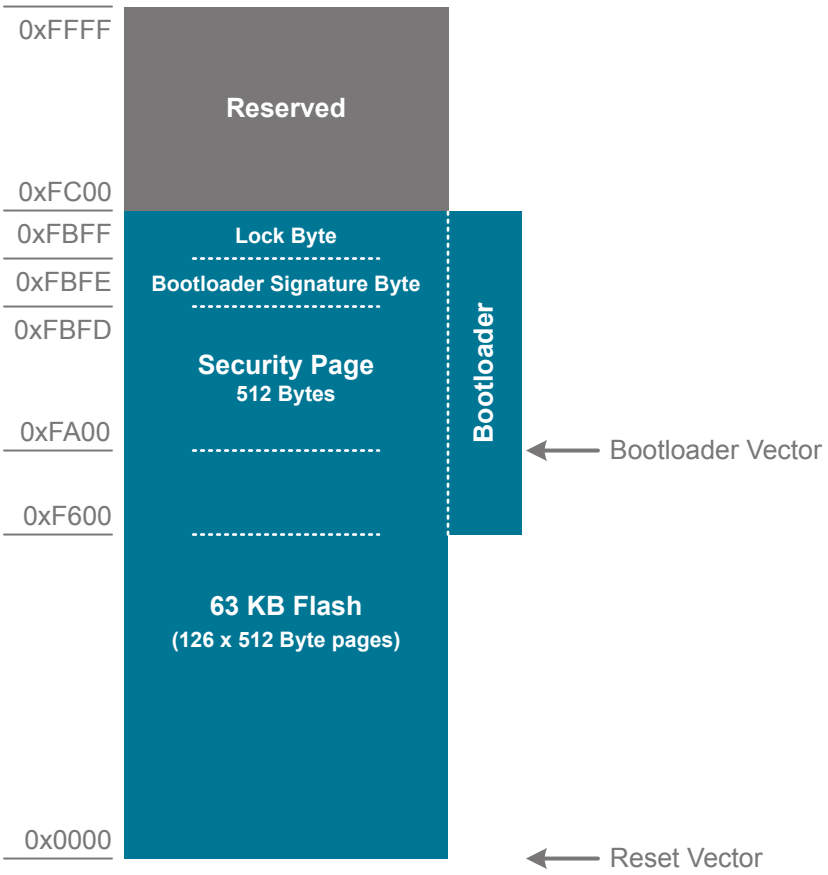


Figure 2.1. Flash Memory Map — 64 KB Devices

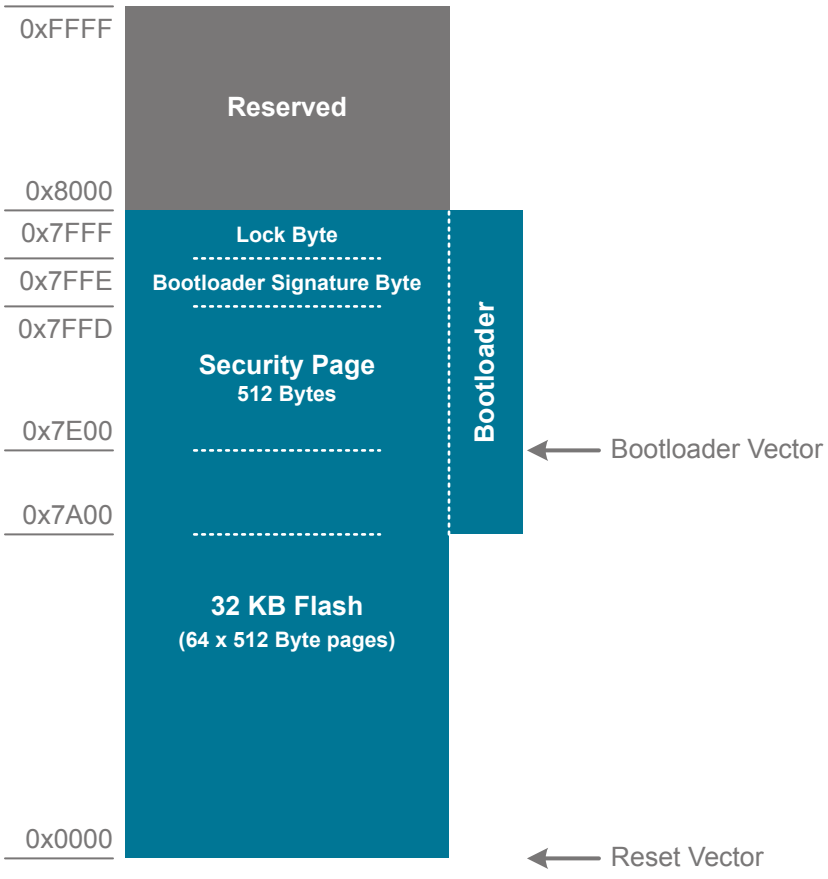


Figure 2.2. Flash Memory Map — 32 KB Devices

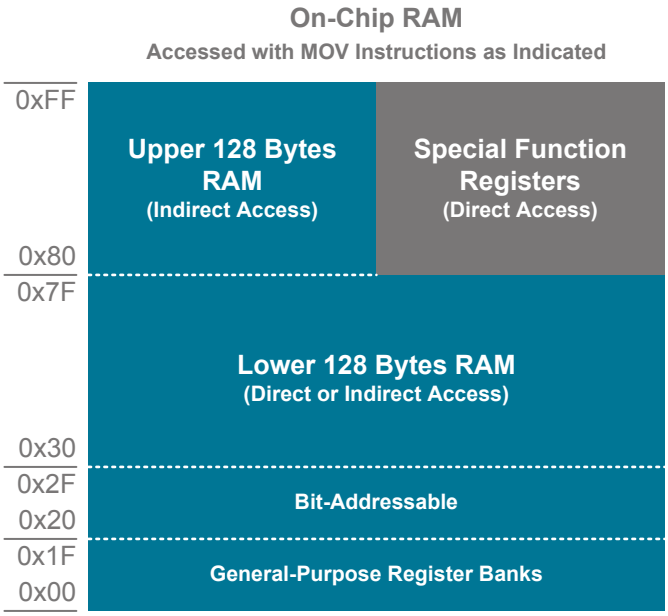


Figure 2.3. Direct / Indirect RAM Memory

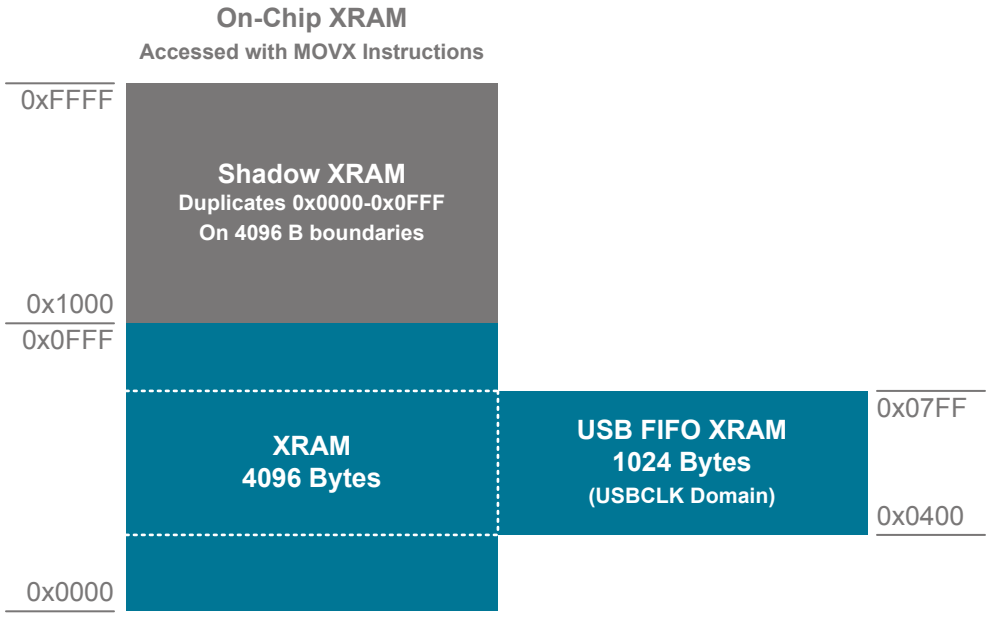


Figure 2.4. XRAM Memory

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 EFM8UB2 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. It is good practice inside of interrupt service routines to save the current SFRPAGE at the beginning of the ISR and restore this value at the end.

Interrupts and SFR Paging

In any system which changes the SFRPAGE while interrupts are active, it is good practice to save the current SFRPAGE value upon ISR entry, and then restore the SFRPAGE before exiting the ISR. This ensures that SFRPAGE will remain at the desired setting when returning from the ISR.

3.2 Special Function Register Memory Map

Table 3.1. Special Function Registers by Address

Address (*bit-addressable)	SFR Page 0x00	SFR Page 0x0F	Address (*bit-addressable)	SFR Page 0x00	SFR Page 0x0F
0x80*	P0		0xC0*	SMB0CN0	SMB1CN0
0x81	SP		0xC1	SMB0CF	SMB1CF
0x82	DPL		0xC2	SMB0DAT	SMB1DAT
0x83	DPH		0xC3	ADC0GTL	
0x84	EMI0TC		0xC4	ADC0GTH	
0x85	EMI0CF		0xC5	ADC0LTL	
0x86	LFO0CN		0xC6	ADC0LTH	
0x87	PCON0		0xC7	P4	
0x88*	TCON		0xC8*	TMR2CN0	TMR5CN0
0x89	TMOD		0xC9	REG01CN	
0x8A	TL0		0xCA	TMR2RLL	TMR5RLL
0x8B	TL1		0xCB	TMR2RLH	TMR5RLH
0x8C	TH0		0xCC	TMR2L	TMR5L
0x8D	TH1		0xCD	TMR2H	TMR5H
0x8E	CKCON0		0xCE	SMB0ADM	SMB1ADM
0x8F	PSCTL		0xCF	SMB0ADR	SMB1ADR
0x90*	P1		0xD0*	PSW	
0x91	TMR3CN0	TMR4CN0	0xD1	REF0CN	
0x92	TMR3RLL	TMR4RLL	0xD2	SCON1	
0x93	TMR3RLH	TMR4RLH	0xD3	SBUF1	
0x94	TMR3L	TMR4L	0xD4	P0SKIP	
0x95	TMR3H	TMR4H	0xD5	P1SKIP	
0x96	USB0ADR		0xD6	P2SKIP	
0x97	USB0DAT		0xD7	USB0XCN	
0x98*	SCON0		0xD8*	PCA0CN0	
0x99	SBUF0		0xD9	PCA0MD	
0x9A	CMP1CN0		0xDA	PCA0CPM0	
0x9B	CMP0CN0		0xDB	PCA0CPM1	
0x9C	CMP1MD		0xDC	PCA0CPM2	
0x9D	CMP0MD		0xDD	PCA0CPM3	
0x9E	CMP1MX		0xDE	PCA0CPM4	
0x9F	CMP0MX		0xDF	P3SKIP	
0xA0*	P2		0xE0*	ACC	

Address (*bit-addressable)	SFR Page 0x00	SFR Page 0x0F	Address (*bit-addressable)	SFR Page 0x00	SFR Page 0x0F
0xA1	SPI0CFG		0xE1	XBR0	
0xA2	SPI0CKR		0xE2	XBR1	
0xA3	SPI0DAT		0xE3	XBR2	
0xA4	P0MDOUT		0xE4	IT01CF	CKCON1
0xA5	P1MDOUT		0xE5	SMOD1	
0xA6	P2MDOUT		0xE6	EIE1	
0xA7	P3MDOUT		0xE7	EIE2	
0xA8*	IE		0xE8*	ADC0CN0	
0xA9	CLKSEL		0xE9	PCA0CPL1	
0xAA	EMI0CN		0xEA	PCA0CPH1	
0xAB	-		0xEB	PCA0CPL2	
0xAC	SBCON1		0xEC	PCA0CPH2	
0xAD	-		0xED	PCA0CPL3	
0xAE	P4MDOUT		0xEE	PCA0CPH3	
0xAF	PFE0CN		0xEF	RSTSRC	
0xB0*	P3		0xF0*	B	
0xB1	XOSC0CN		0xF1	P0MDIN	
0xB2	HFO0CN		0xF2	P1MDIN	
0xB3	HFO0CAL		0xF3	P2MDIN	
0xB4	SBRLL1		0xF4	P3MDIN	
0xB5	SBRLH1		0xF5	P4MDIN	
0xB6	FLSCL		0xF6	EIP1	
0xB7	FLKEY		0xF7	EIP2	
0xB8*	IP		0xF8*	SPI0CN0	
0xB9	-	SMBTC	0xF9	PCA0L	
0xBA	AMX0N		0xFA	PCA0H	
0xBB	AMX0P		0xFB	PCA0CPL0	
0xBC	ADC0CF		0xFC	PCA0CPH0	
0xBD	ADC0L		0xFD	PCA0CPL4	
0xBE	ADC0H		0xFE	PCA0CPH4	
0xBF	SFRPAGE		0xFF	VDM0CN	

Table 3.2. Special Function Registers by Name

Register	Address	SFR Pages	Description
ACC	0xE0	ALL	Accumulator

Register	Address	SFR Pages	Description
ADC0CF	0xBC	ALL	ADC0 Configuration
ADC0CN0	0xE8	ALL	ADC0 Control
ADC0GTH	0xC4	ALL	ADC0 Greater-Than High Byte
ADC0GTL	0xC3	ALL	ADC0 Greater-Than Low Byte
ADC0H	0xBE	ALL	ADC0 Data Word High Byte
ADC0L	0xBD	ALL	ADC0 Data Word Low Byte
ADC0LTH	0xC6	ALL	ADC0 Less-Than High Byte
ADC0LTL	0xC5	ALL	ADC0 Less-Than Low Byte
AMX0N	0xBA	ALL	AMUX0 Negative Multiplexer Selection
AMX0P	0xBB	ALL	AMUX0 Positive Multiplexer Selection
B	0xF0	ALL	B Register
CKCON0	0x8E	ALL	Clock Control 0
CKCON1	0xE4	0x0F	Clock Control 1
CLKSEL	0xA9	ALL	Clock Select
CMP0CN0	0x9B	ALL	Comparator 0 Control 0
CMP0MD	0x9D	ALL	Comparator 0 Mode
CMP0MX	0x9F	ALL	Comparator 0 Multiplexer Selection
CMP1CN0	0x9A	ALL	Comparator 1 Control 0
CMP1MD	0x9C	ALL	Comparator 1 Mode
CMP1MX	0x9E	ALL	Comparator 1 Multiplexer Selection
DPH	0x83	ALL	Data Pointer High
DPL	0x82	ALL	Data Pointer Low
EIE1	0xE6	ALL	Extended Interrupt Enable 1
EIE2	0xE7	ALL	Extended Interrupt Enable 2
EIP1	0xF6	ALL	Extended Interrupt Priority 1
EIP2	0xF7	ALL	Extended Interrupt Priority 2
EMI0CF	0x85	ALL	External Memory Configuration
EMI0CN	0xAA	ALL	External Memory Interface Control
EMI0TC	0x84	ALL	External Memory Timing Control
FLKEY	0xB7	ALL	Flash Lock and Key
FLSCL	0xB6	ALL	Flash Scale
HFO0CAL	0xB3	ALL	High Frequency Oscillator Calibration
HFO0CN	0xB2	ALL	High Frequency Oscillator Control
IE	0xA8	ALL	Interrupt Enable
IP	0xB8	ALL	Interrupt Priority
IT01CF	0xE4	0x00	INT0/INT1 Configuration
LFO0CN	0x86	ALL	Low Frequency Oscillator Control

Register	Address	SFR Pages	Description
P0	0x80	ALL	Port 0 Pin Latch
P0MDIN	0xF1	ALL	Port 0 Input Mode
P0MDOUT	0xA4	ALL	Port 0 Output Mode
P0SKIP	0xD4	ALL	Port 0 Skip
P1	0x90	ALL	Port 1 Pin Latch
P1MDIN	0xF2	ALL	Port 1 Input Mode
P1MDOUT	0xA5	ALL	Port 1 Output Mode
P1SKIP	0xD5	ALL	Port 1 Skip
P2	0xA0	ALL	Port 2 Pin Latch
P2MDIN	0xF3	ALL	Port 2 Input Mode
P2MDOUT	0xA6	ALL	Port 2 Output Mode
P2SKIP	0xD6	ALL	Port 2 Skip
P3	0xB0	ALL	Port 3 Pin Latch
P3MDIN	0xF4	ALL	Port 3 Input Mode
P3MDOUT	0xA7	ALL	Port 3 Output Mode
P3SKIP	0xDF	ALL	Port 3 Skip
P4	0xC7	ALL	Port 4 Pin Latch
P4MDIN	0xF5	ALL	Port 4 Input Mode
P4MDOUT	0xAE	ALL	Port 4 Output Mode
PCA0CN0	0xD8	ALL	PCA Control 0
PCA0CPH0	0xFC	ALL	PCA Channel 0 Capture Module High Byte
PCA0CPH1	0xEA	ALL	PCA Channel 1 Capture Module High Byte
PCA0CPH2	0xEC	ALL	PCA Channel 2 Capture Module High Byte
PCA0CPH3	0xEE	ALL	PCA Channel 3 Capture Module High Byte
PCA0CPH4	0xFE	ALL	PCA Channel 4 Capture Module High Byte
PCA0CPL0	0xFB	ALL	PCA Channel 0 Capture Module Low Byte
PCA0CPL1	0xE9	ALL	PCA Channel 1 Capture Module Low Byte
PCA0CPL2	0xEB	ALL	PCA Channel 2 Capture Module Low Byte
PCA0CPL3	0xED	ALL	PCA Channel 3 Capture Module Low Byte
PCA0CPL4	0xFD	ALL	PCA Channel 4 Capture Module Low Byte
PCA0CPM0	0xDA	ALL	PCA Channel 0 Capture/Compare Mode
PCA0CPM1	0xDB	ALL	PCA Channel 1 Capture/Compare Mode
PCA0CPM2	0xDC	ALL	PCA Channel 2 Capture/Compare Mode
PCA0CPM3	0xDD	ALL	PCA Channel 3 Capture/Compare Mode
PCA0CPM4	0xDE	ALL	PCA Channel 4 Capture/Compare Mode
PCA0H	0xFA	ALL	PCA Counter/Timer High Byte
PCA0L	0xF9	ALL	PCA Counter/Timer Low Byte

Register	Address	SFR Pages	Description
PCA0MD	0xD9	ALL	PCA Mode
PCON0	0x87	ALL	Power Control
PFE0CN	0xAF	ALL	Prefetch Engine Control
PSCTL	0x8F	ALL	Program Store Control
PSW	0xD0	ALL	Program Status Word
REF0CN	0xD1	ALL	Voltage Reference Control
REG01CN	0xC9	ALL	Voltage Regulator Control
RSTSRC	0xEF	ALL	Reset Source
SBCON1	0xAC	ALL	UART1 Baud Rate Generator Control
SBRLH1	0xB5	ALL	UART1 Baud Rate Generator High Byte
SBRLL1	0xB4	ALL	UART1 Baud Rate Generator Low Byte
SBUF0	0x99	ALL	UART0 Serial Port Data Buffer
SBUF1	0xD3	ALL	UART1 Serial Port Data Buffer
SCON0	0x98	ALL	UART0 Serial Port Control
SCON1	0xD2	ALL	UART1 Serial Port Control
SFRPAGE	0xBF	ALL	SFR Page
SMB0ADM	0xCE	0x00	SMBus 0 Slave Address Mask
SMB0ADR	0xCF	0x00	SMBus 0 Slave Address
SMB0CF	0xC1	0x00	SMBus 0 Configuration
SMB0CN0	0xC0	0x00	SMBus 0 Control
SMB0DAT	0xC2	0x00	SMBus 0 Data
SMB1ADM	0xCE	0x0F	SMBus 1 Slave Address Mask
SMB1ADR	0xCF	0x0F	SMBus 1 Slave Address
SMB1CF	0xC1	0x0F	SMBus 1 Configuration
SMB1CN0	0xC0	0x0F	SMBus 1 Control
SMB1DAT	0xC2	0x0F	SMBus 1 Data
SMBTC	0xB9	0x0F	SMBus Timing and Pin Control
SMOD1	0xE5	ALL	UART1 Mode
SP	0x81	ALL	Stack Pointer
SPI0CFG	0xA1	ALL	SPI0 Configuration
SPI0CKR	0xA2	ALL	SPI0 Clock Rate
SPI0CN0	0xF8	ALL	SPI0 Control
SPI0DAT	0xA3	ALL	SPI0 Data
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

Register	Address	SFR Pages	Description
TL1	0x8B	ALL	Timer 1 Low Byte
TMOD	0x89	ALL	Timer 0/1 Mode
TMR2CN0	0xC8	0x00	Timer 2 Control 0
TMR2H	0xCD	0x00	Timer 2 High Byte
TMR2L	0xCC	0x00	Timer 2 Low Byte
TMR2RLH	0xCB	0x00	Timer 2 Reload High Byte
TMR2RLL	0xCA	0x00	Timer 2 Reload Low Byte
TMR3CN0	0x91	0x00	Timer 3 Control 0
TMR3H	0x95	0x00	Timer 3 High Byte
TMR3L	0x94	0x00	Timer 3 Low Byte
TMR3RLH	0x93	0x00	Timer 3 Reload High Byte
TMR3RLL	0x92	0x00	Timer 3 Reload Low Byte
TMR4CN0	0x91	0x0F	Timer 4 Control 0
TMR4H	0x95	0x0F	Timer 4 High Byte
TMR4L	0x94	0x0F	Timer 4 Low Byte
TMR4RLH	0x93	0x0F	Timer 4 Reload High Byte
TMR4RLL	0x92	0x0F	Timer 4 Reload Low Byte
TMR5CN0	0xC8	0x0F	Timer 5 Control 0
TMR5H	0xCD	0x0F	Timer 5 High Byte
TMR5L	0xCC	0x0F	Timer 5 Low Byte
TMR5RLH	0xCB	0x0F	Timer 5 Reload High Byte
TMR5RLL	0xCA	0x0F	Timer 5 Reload Low Byte
USB0ADR	0x96	ALL	USB0 Indirect Address
USB0DAT	0x97	ALL	USB0 Data
USB0XCN	0xD7	ALL	USB0 Transceiver Control
VDM0CN	0xFF	ALL	Supply Monitor Control
XBR0	0xE1	ALL	Port I/O Crossbar 0
XBR1	0xE2	ALL	Port I/O Crossbar 1
XBR2	0xE3	ALL	Port I/O Crossbar 2
XOSC0CN	0xB1	ALL	External Oscillator Control

3.3 SFR Access Control Registers

3.3.1 SFRPAGE: SFR Page

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

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.

4. Flash Memory

4.1 Introduction

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

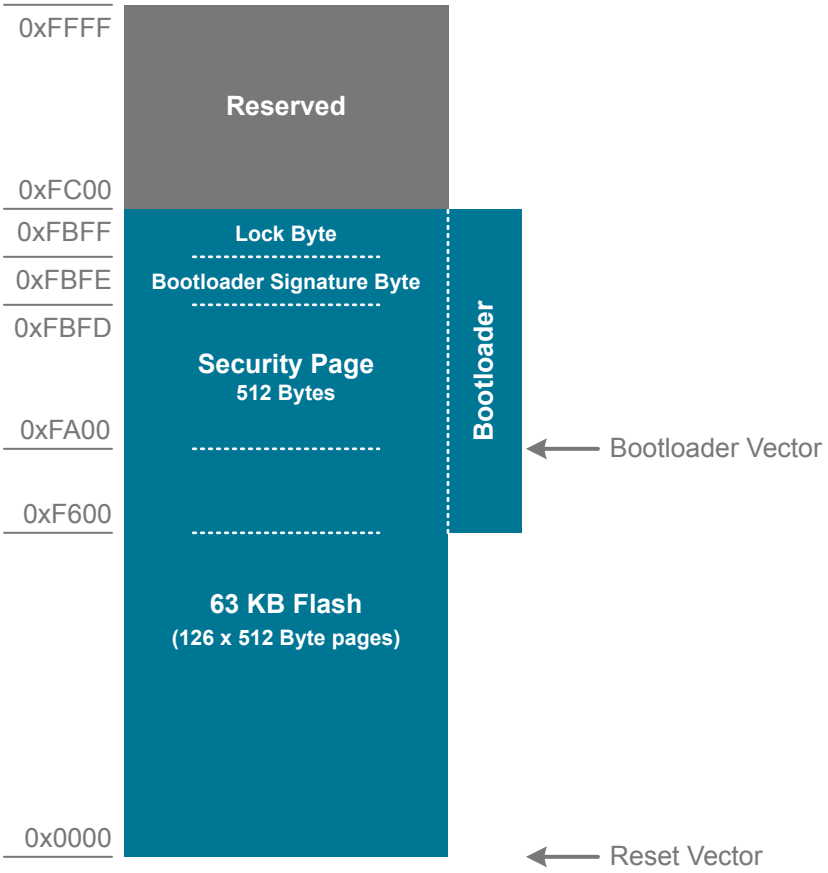


Figure 4.1. Flash Memory Map — 64 KB Devices

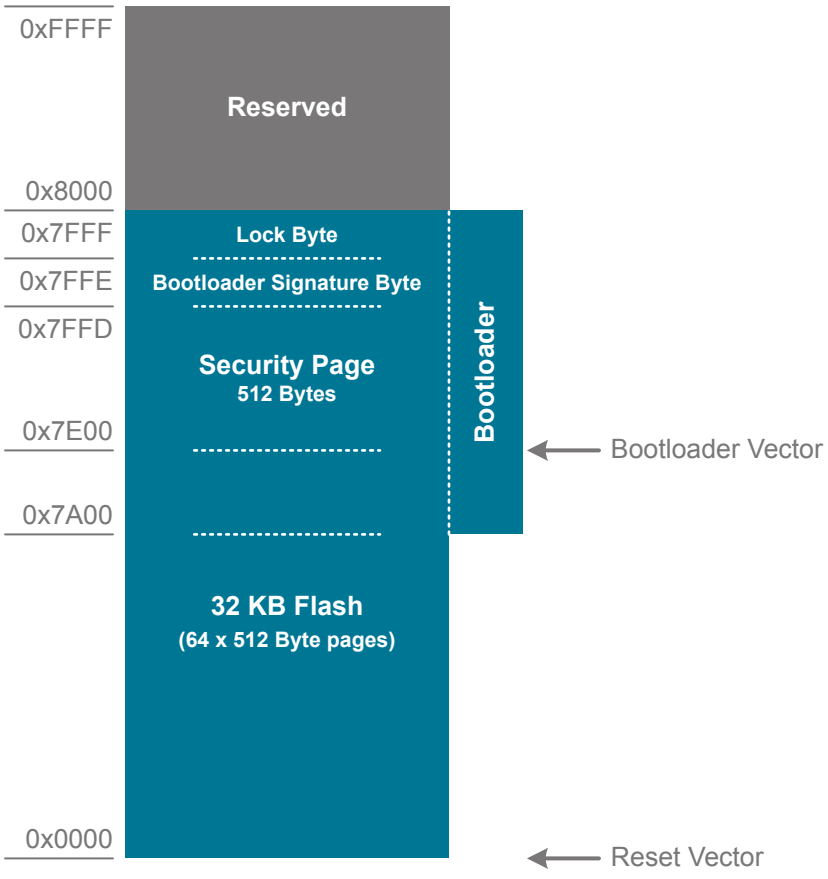


Figure 4.2. Flash Memory Map — 32 KB Devices

4.2 Features

The flash memory has the following features:

- Up to 64 KB organized in 512-byte sectors.
- In-system programmable from user firmware.
- Security lock to prevent unwanted read/write/erase access.

4.3 Functional Description

4.3.1 Security Options

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

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

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

Table 4.1. Security Byte Decoding

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

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

Table 4.2. Flash Security Summary—Firmware Permissions

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

Table 4.3. Flash Security Summary—C2 Permissions

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

4.3.2 Programming the Flash Memory

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

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

4.3.2.1 Flash Lock and Key Functions

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

4.3.2.2 Flash Page Erase Procedure

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

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

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

4.3.2.3 Flash Byte Write Procedure

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

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

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

4.3.3 Flash Write and Erase Precautions

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

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

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

Voltage Supply Maintenance and the Supply Monitor

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

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

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

PSWE Maintenance

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

System Clock

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

4.4 Flash Control Registers

4.4.1 PSCTL: Program Store Control

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

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

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

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

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

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

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

4.4.2 FLKEY: Flash Lock and Key

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

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

4.4.3 FLSCL: Flash Scale

Bit	7	6	5	4	3	2	1	0
Name	FOSE	Reserved		FLRT	Reserved			
Access	RW	RW		RW	RW			
Reset	1	0x0		0	0x0			
SFR Page = ALL; SFR Address: 0xB6								

Bit	Name	Reset	Access	Description
7	FOSE	1	RW	**Flash One-Shot Enable.**
	This bit enables the flash read one-shot (recommended). If the flash one-shot is disabled, the flash sense amps are enabled for a full clock cycle during flash reads, increasing the device power consumption.			
	Value	Name		Description
	0	DISABLED		Disable the flash one-shot.
	1	ENABLED		Enable the flash one-shot (recommended).
6:5	*Reserved*	*Must write reset value.*		
4	FLRT	0	RW	**Flash Read Timing.**
	This bit should be programmed to the smallest allowed value, according to the system clock speed.			
	Value	Name		Description
	0	SYSCLK_BELOW_25_MHZ		SYSCLK <= 25 MHz.
	1	SYSCLK_BELOW_48_MHZ		SYSCLK <= 48 MHz.
3:0	*Reserved*	*Must write reset value.*		

5. Device Identification

5.1 Unique Identifier

A 128-bit unique identifier (UID) is pre-loaded upon device reset into the last bytes of the XRAM area on all devices. The value assigned to a device is random and not sequential, but it is guaranteed unique. The UID can be read by firmware using MOVX instructions and through the debug port.

As the UID appears in RAM, firmware can overwrite the UID during normal operation. The bytes in memory will be automatically reinitialized with the UID value after any device reset. Firmware using this area of memory should always initialize the memory to a known value, as any previous data stored at these locations will be overwritten and not retained through a reset.

Table 5.1. UID Location in Memory

Device	XRAM Addresses
EFM8UB20F64G	(MSB) 0x0FFF, 0x0FFE, 0x0FFD, 0x0FFC, 0x0FFB, 0x0FFA, 0x0FF9, 0x0FF8, 0x0FF7, 0x0FF6, 0x0FF5, 0x0FF4, 0x0FF3, 0x0FF2, 0x0FF1, 0x0FF0 (LSB)
EFM8UB20F32G	(MSB) 0x07FF, 0x07FE, 0x07FD, 0x07FC, 0x07FB, 0x07FA, 0x07F9, 0x07F8, 0x07F7, 0x07F6, 0x07F5, 0x07F4, 0x07F3, 0x07F2, 0x07F1, 0x07F0 (LSB)

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 two priority levels: low or high. A low priority interrupt service routine can be preempted by a high priority interrupt. A high priority interrupt cannot be preempted. Each interrupt has an associated interrupt priority bit in the IP and EIPn registers, which are used to configure its priority level. Low priority is the default. If two interrupts are recognized simultaneously, the interrupt with the higher 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.

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 6.1. 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
UART 0	0x0023	4	IE_ES0	-	SCON0_RI SCON0_TI
Timer 2 Overflow	0x002B	5	IE_ET2	-	TMR2CN0_TF2H TMR2CN0_TF2L
SPI0	0x0033	6	IE_ESPI0	-	SPI0CN0_MODF SPI0CN0_RXOVRN SPI0CN0_SPIF SPI0CN0_WCOL
SMBus 0	0x003B	7	EIE1_ESMB0	-	SMB0CN0_SI
USB0	0x0043	8	EIE1_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
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 PCA0CPM3_ECCF PCA0CPM4_ECCF	PCA0CN0_CCF0 PCA0CN0_CCF1 PCA0CN0_CCF2 PCA0CN0_CCF3 PCA0CN0_CCF4 PCA0CN0_CF

Interrupt Source	Vector	Priority	Primary Enable	Auxiliary Enable(s)	Pending Flag(s)
Comparator 0	0x0063	12	EIE1_ECP0	CMP0MD_CPFIE CMP0MD_CPRIE	CMP0CN0_CPFIF CMP0CN0_CPRIF
Comparator 1	0x006B	13	EIE1_ECP1	CMP1MD_CPFIE CMP1MD_CPRIE	CMP1CN0_CPFIF CMP1CN0_CPRIF
Timer 3 Overflow	0x0073	14	EIE1_ET3	-	TMR3CN0_TF3H TMR3CN0_TF3L
VBUS Level	0x007B	15	EIE2_EVBUS	-	-
UART 1	0x0083	16	EIE2_ES1	-	SCON1_RI SCON1_TI
Reserved	0x008B	17	-	-	-
SMBus 1	0x0093	18	EIE2_ESMB1	-	SMB1CN0_SI
Timer 4 Overflow	0x009B	19	EIE2_ET4	-	TMR4CN0_TF4H TMR4CN0_TF4L
Timer 5 Overflow	0x00A3	20	EIE2_ET5	-	TMR5CN0_TF5H TMR5CN0_TF5L

6.3 Interrupt Control Registers

6.3.1 IE: Interrupt Enable

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

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

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

6.3.2 IP: Interrupt Priority

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

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	PSPI0	0	RW	**Serial Peripheral Interface (SPI0) Interrupt Priority Control.** This bit sets the priority of the SPI0 interrupt.
	Value	Name		Description
	0	LOW		SPI0 interrupt set to low priority level.
	1	HIGH		SPI0 interrupt set to high priority level.
5	PT2	0	RW	**Timer 2 Interrupt Priority Control.** This bit sets the priority of the Timer 2 interrupt.
	Value	Name		Description
	0	LOW		Timer 2 interrupt set to low priority level.
	1	HIGH		Timer 2 interrupt set to high priority level.
4	PS0	0	RW	**UART0 Interrupt Priority Control.** This bit sets the priority of the UART0 interrupt.
	Value	Name		Description
	0	LOW		UART0 interrupt set to low priority level.
	1	HIGH		UART0 interrupt set to high priority level.
3	PT1	0	RW	**Timer 1 Interrupt Priority Control.** This bit sets the priority of the Timer 1 interrupt.
	Value	Name		Description
	0	LOW		Timer 1 interrupt set to low priority level.
	1	HIGH		Timer 1 interrupt set to high priority level.
2	PX1	0	RW	**External Interrupt 1 Priority Control.** This bit sets the priority of the External Interrupt 1 interrupt.
	Value	Name		Description
	0	LOW		External Interrupt 1 set to low priority level.
	1	HIGH		External Interrupt 1 set to high priority level.

Bit	Name	Reset	Access	Description
1	PT0	0	RW	**Timer 0 Interrupt Priority Control.**

This bit sets the priority of the Timer 0 interrupt.

Value	Name	Description
0	LOW	Timer 0 interrupt set to low priority level.
1	HIGH	Timer 0 interrupt set to high priority level.

Bit	Name	Reset	Access	Description
0	PX0	0	RW	**External Interrupt 0 Priority Control.**

This bit sets the priority of the External Interrupt 0 interrupt.

Value	Name	Description
0	LOW	External Interrupt 0 set to low priority level.
1	HIGH	External Interrupt 0 set to high priority level.

6.3.3 EIE1: Extended Interrupt Enable 1

Bit	7	6	5	4	3	2	1	0
Name	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	EUSB0	ESMB0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xE6								

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

Bit	Name	Reset	Access	Description
2	EWADC0	0	RW	**ADC0 Window Comparison Interrupt Enable.**
	This bit sets the masking of ADC0 Window Comparison interrupt.			
	Value	Name		Description
	0	DISABLED		Disable ADC0 Window Comparison interrupt.
	1	ENABLED		Enable interrupt requests generated by ADC0 Window Compare flag (ADWINT).
1	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.
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.

6.3.4 EIP1: Extended Interrupt Priority 1

Bit	7	6	5	4	3	2	1	0
Name	PT3	PCP1	PCP0	PPCA0	PADC0	PWADC0	PUSB0	PSMB0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xF6								

Bit	Name	Reset	Access	Description
7	PT3	0	RW	**Timer 3 Interrupt Priority Control.**
	This bit sets the priority of the Timer 3 interrupt.			
	Value	Name		Description
	0	LOW		Timer 3 interrupts set to low priority level.
	1	HIGH		Timer 3 interrupts set to high priority level.
6	PCP1	0	RW	**Comparator1 (CP1) Interrupt Priority Control.**
	This bit sets the priority of the CP1 interrupt.			
	Value	Name		Description
	0	LOW		CP1 interrupt set to low priority level.
	1	HIGH		CP1 interrupt set to high priority level.
5	PCP0	0	RW	**Comparator0 (CP0) Interrupt Priority Control.**
	This bit sets the priority of the CP0 interrupt.			
	Value	Name		Description
	0	LOW		CP0 interrupt set to low priority level.
	1	HIGH		CP0 interrupt set to high priority level.
4	PPCA0	0	RW	**Programmable Counter Array (PCA0) Interrupt Priority Control.**
	This bit sets the priority of the PCA0 interrupt.			
	Value	Name		Description
	0	LOW		PCA0 interrupt set to low priority level.
	1	HIGH		PCA0 interrupt set to high priority level.
3	PADC0	0	RW	**ADC0 Conversion Complete Interrupt Priority Control.**
	This bit sets the priority of the ADC0 Conversion Complete interrupt.			
	Value	Name		Description
	0	LOW		ADC0 Conversion Complete interrupt set to low priority level.
	1	HIGH		ADC0 Conversion Complete interrupt set to high priority level.
2	PWADC0	0	RW	**ADC0 Window Comparator Interrupt Priority Control.**
	This bit sets the priority of the ADC0 Window interrupt.			

Bit	Name	Reset	Access	Description
	Value	Name		Description
	0	LOW		ADC0 Window interrupt set to low priority level.
	1	HIGH		ADC0 Window interrupt set to high priority level.
1	PUSB0	0	RW	**USB (USB0) Interrupt Priority Control.**
	This bit sets the priority of the USB0 interrupt.			
	Value	Name		Description
	0	LOW		USB0 interrupt set to low priority level.
	1	HIGH		USB0 interrupt set to high priority level.
0	PSMB0	0	RW	**SMBus (SMB0) Interrupt Priority Control.**
	This bit sets the priority of the SMB0 interrupt.			
	Value	Name		Description
	0	LOW		SMB0 interrupt set to low priority level.
	1	HIGH		SMB0 interrupt set to high priority level.

6.3.5 EIE2: Extended Interrupt Enable 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved		ET5	ET4	ESMB1	Reserved	ES1	EVBUS
Access	R		RW	RW	RW	RW	RW	RW
Reset	0x0		0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xE7								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	ET5	0	RW	**Timer 5 Interrupt Enable.**
	This bit sets the masking of the Timer 5 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable Timer 5 interrupts.
	1	ENABLED		Enable interrupt requests generated by the TF5L or TF5H flags.
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 4interrupts.
	1	ENABLED		Enable interrupt requests generated by the TF4L or TF4H flags.
3	ESMB1	0	RW	**SMBus1 Interrupt Enable.**
	This bit sets the masking of the SMB1 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all SMB1 interrupts.
	1	ENABLED		Enable interrupt requests generated by SMB1.
2	*Reserved*	*Must write reset value.*		
1	ES1	0	RW	**UART1 Interrupt Enable.**
	This bit sets the masking of the UART1 interrupt.			
	Value	Name		Description
	0	DISABLED		Disable UART1 interrupt.
	1	ENABLED		Enable UART1 interrupt.
0	EVBUS	0	RW	**VBUS Level Interrupt Enable.**
	This bit sets the masking of the VBUS interrupt.			
	Value	Name		Description
	0	DISABLED		Disable all VBUS interrupts.
	1	ENABLED		Enable interrupt requests generated by VBUS level sense.

6.3.6 EIP2: Extended Interrupt Priority 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved		PT5	PT4	PSMB1	Reserved	PS1	PVBUS
Access	R		RW	RW	RW	RW	RW	RW
Reset	0x0		0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xF7								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	PT5	0	RW	**Timer 5 Interrupt Priority Control.**
	This bit sets the priority of the Timer 5 interrupt.			
	Value	Name		Description
	0	LOW		Timer 5 interrupt set to low priority level.
	1	HIGH		Timer 5 interrupt set to high priority level.
4	PT4	0	RW	**Timer 4 Interrupt Priority Control.**
	This bit sets the priority of the Timer 4 interrupt.			
	Value	Name		Description
	0	LOW		Timer 4 interrupt set to low priority level.
	1	HIGH		Timer 4 interrupt set to high priority level.
3	PSMB1	0	RW	**SMBus1 Interrupt Priority Control.**
	This bit sets the priority of the SMB1 interrupt.			
	Value	Name		Description
	0	LOW		SMB1 interrupt set to low priority level.
	1	HIGH		SMB1 interrupt set to high priority level.
2	*Reserved*	*Must write reset value.*		
1	PS1	0	RW	**UART1 Interrupt Priority Control.**
	This bit sets the priority of the UART1 interrupt.			
	Value	Name		Description
	0	LOW		UART1 interrupt set to low priority level.
	1	HIGH		UART1 interrupt set to high priority level.
0	PVBUS	0	RW	**VBUS Level Interrupt Priority Control.**
	This bit sets the priority of the VBUS interrupt.			
	Value	Name		Description
	0	LOW		VBUS interrupt set to low priority level.
	1	HIGH		VBUS interrupt set to high priority level.

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.

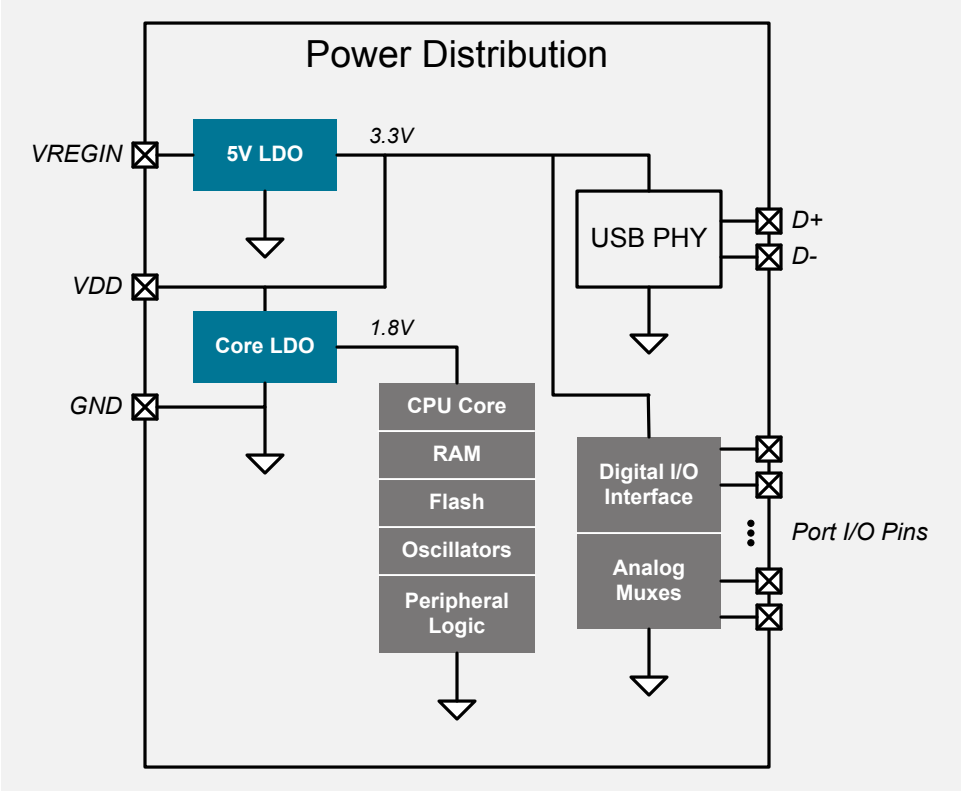


Figure 7.1. Power System Block Diagram

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 • Code resumes execution on wake event	1. Switch SYSCLK to HFOSC0 2. Set SUSPEND bit in HFO0CN	USB0 Bus Activity
Stop	• All internal power nets shut down • Pins retain state • Exit on any reset source	Set STOP bit in PCON0	Any reset source

Power Mode	Details	Mode Entry	Wake-Up Sources
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 REG01CN 2. Set STOP bit in PCON0	• RSTb pin reset • Power-on reset

7.2 Features

- Supports four 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: May be used with USB0 peripheral to provide low current in USB suspend. Very fast wake-up time and code resumes execution at the next instruction.
  4. Shutdown mode: Lowest power state. Device is off, drawing very little current, and waiting ror a pin reset or power-on reset.

  Note: Legacy 8051 Stop mode is also supported, where the internal LDO remains active, but a device reset is required to wake.
- Fully internal core LDO supplies power to majority of blocks.
- 5-to-3.3 V Regulator:
  - Allows direct connection to USB supply net.
  - Provides up to 100 mA for system-level use.
  - Low power consumption in Suspend 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 the high-frequency internal oscillator and goes into a low power state as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data.

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

7.6 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.

7.7 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.

7.8 Power Management Control Registers

7.8.1 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.			

7.8.2 REG01CN: Voltage Regulator Control

Bit	7	6	5	4	3	2	1	0
Name	REG0DIS	VBSTAT	Reserved	REG0MD	STOPCF	Reserved	REG1MD	Reserved
Access	RW	R	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xC9								

Bit	Name	Reset	Access	Description
7	REG0DIS	0	RW	**Voltage Regulator (REG0) Disable.**
	This bit enables or disables the VREG0 Voltage Regulator.			
	Value	Name		Description
	0	ENABLED		Enable the VREG0 Voltage Regulator.
	1	DISABLED		Disable the VREG0 Voltage Regulator.
6	VBSTAT	0	R	**VBUS Signal Status.**
	This bit indicates whether the device is connected to a USB network.			
	Value	Name		Description
	0	NOT_SET		VBUS signal currently absent (device not attached to USB network).
	1	SET		VBUS signal currently present (device attached to USB network).
5	*Reserved*	*Must write reset value.*		
4	REG0MD	0	RW	**VREG0 Voltage Regulator Mode.**
	This bit selects the Voltage Regulator mode for VREG0. When REG0MD is set to 1, the VREG0 voltage regulator operates in lower power (suspend) mode.			
	Value	Name		Description
	0	NORMAL		VREG0 Voltage Regulator in normal mode.
	1	LOW_POWER		VREG0 Voltage Regulator in low power mode.
3	STOPCF	0	RW	**VREG1 Stop Mode Configuration.**
	This bit configures the VREG1 regulator's behavior when the device enters stop mode.			
	Value	Name		Description
	0	ACTIVE		VREG1 Regulator is still active in stop mode. Any enabled reset source will reset the device.
	1	SHUTDOWN		VREG1 Regulator is shut down in stop mode. Only the RSTb pin or power cycle can reset the device.
2	*Reserved*	*Must write reset value.*		
1	REG1MD	0	RW	**VREG1 Voltage Regulator Mode.**
	This bit selects the Voltage Regulator mode for VREG1. When REG1MD is set to 1, the VREG1 voltage regulator operates in lower power mode. This bit should not be set to 1 if the VREG0 Voltage Regulator is disabled.			

Bit	Name	Reset	Access	Description
	Value	Name		Description
	0	NORMAL		VREG1 Voltage Regulator in normal mode.
	1	LOW_POWER		VREG1 Voltage Regulator in low power mode.
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 48 MHz oscillator divided by 4, then divided by 8 (1.5 MHz).

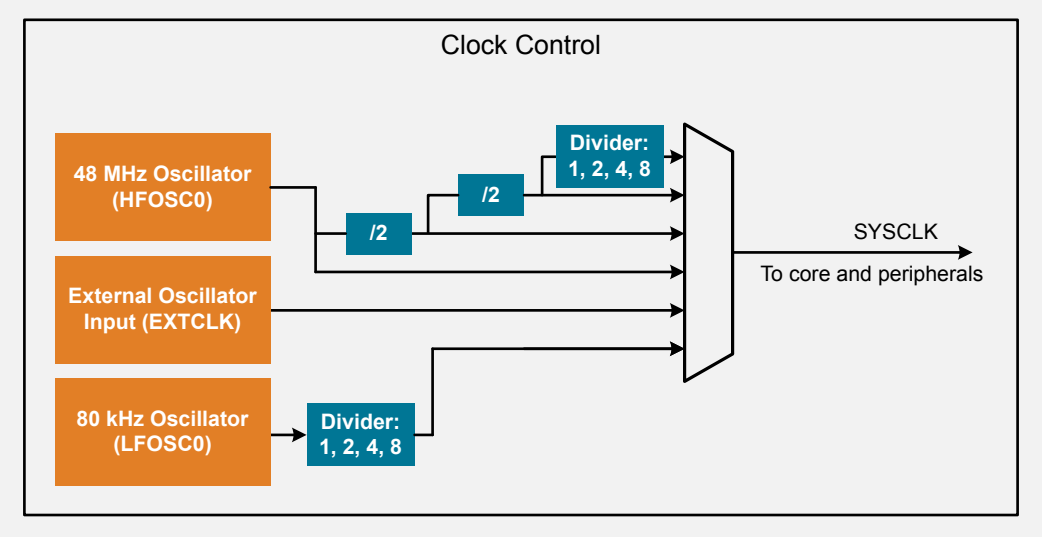


Figure 8.1. Clock Control Block Diagram

8.2 Features

- Provides clock to core and peripherals.
- 48 MHz internal oscillator (HFOSC0), accurate to ±1.5% over supply and temperature corners: accurate to +/- 0.25% when using USB clock recovery.
- 80 kHz low-frequency oscillator (LFOSC0).
- External RC, C, CMOS, and high-frequency crystal clock options (EXTCLK) for QFP48 packages.
- External CMOS clock option (EXTCLK) for QFP32 and QFN32 packages.
- Internal oscillator has clock divider with eight settings for flexible clock scaling: 1, 2, 4, or 8.

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 48 MHz Internal Oscillator

HFOSC0 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 HFO0CAL register to obtain other frequencies.

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

8.3.3 LFOSC0 80 kHz Internal Oscillator

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

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

Calibrating LFOSC0

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

8.3.4 External Crystal

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

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

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

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

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

Figure 8.2. External Oscillator Load Capacitance

Where:

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

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

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

Figure 8.3. External Oscillator Load Capacitance with Equal Capacitors

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

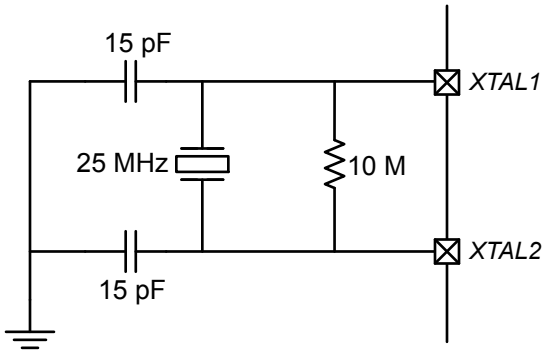


Figure 8.4. 25 MHz External Crystal Example

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

Table 8.1. Recommended XFCN Settings for Crystal Mode

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

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

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

8.3.5 External RC and C Modes

External RC Example

An RC network connected to the XTAL2 pin can be used as a basic oscillator. XTAL1 is not affected in RC mode.

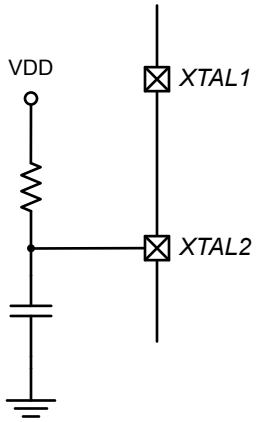


Figure 8.5. External RC Oscillator Configuration

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

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

Figure 8.6. RC Mode Oscillator Frequency

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

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

Figure 8.7. RC Mode Oscillator Example

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

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

1. Configure XTAL2 for analog I/O and disable the digital output drivers.
2. Configure and enable the external oscillator.
3. Poll for XCLKVLD = 1.
4. Switch the system clock to the external oscillator.

External Capacitor Example

If a capacitor is used as the external oscillator, the circuit should be configured as shown in . The capacitor should be added to XTAL2, and XTAL2 should be configured for analog I/O with the digital output drivers disabled. XTAL1 is not affected in C mode.

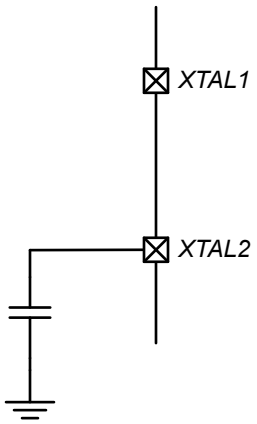


Figure 8.8. External Capacitor Oscillator Configuration

The capacitor should be no greater than 100 pF; however, for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. The oscillation frequency and the required XFCN field value determined by the following equation, where f is the frequency in MHz, C is the capacitor value on XTAL2 in pF, and V_{DD} is the power supply voltage in Volts:

$$f = \frac{KF}{C \times V_{DD}}$$

Figure 8.9. C Mode Oscillator Frequency

For example, assume VDD = 3.0 V and f = 150 kHz. Since a frequency of roughly 150 kHz is desired, select the K Factor from as KF = 22:

$$f = \frac{KF}{C \times V_{DD}}$$

$$0.150\ \text{MHz} = \frac{22}{C \times 3.0}$$

$$C = \frac{22}{0.150\ \text{MHz} \times 3.0}$$

$$C = 48.8\ \text{pF}$$

Figure 8.10. C Mode Oscillator Example

Therefore, the XFCN value to use in this example is 011 and C is approximately 50 pF. The recommended startup procedure for C mode is the same as RC mode.

Recommended XFCN Settings for RC and C Modes

Table 8.2. Recommended XFCN Settings for RC and C Modes

XFCN Field Setting	Approximate Frequency Range	K Factor (C Mode)	Actual Measured Frequency (C Mode)
000	f ≤ 25 kHz	K Factor = 0.87	f = 11 kHz, C = 33 pF
001	25 kHz < f ≤ 50 kHz	K Factor = 2.6	f = 33 kHz, C = 33 pF
010	50 kHz < f ≤ 100 kHz	K Factor = 7.7	f = 98 kHz, C = 33 pF
011	100 kHz < f ≤ 200 kHz	K Factor = 22	f = 270 kHz, C = 33 pF
100	200 kHz < f ≤ 400 kHz	K Factor = 65	f = 310 kHz, C = 46 pF
101	400 kHz < f ≤ 800 kHz	K Factor = 180	f = 890 kHz, C = 46 pF
110	800 kHz < f ≤ 1.6 MHz	K Factor = 664	f = 2.0 MHz, C = 46 pF
111	1.6 MHz < f ≤ 3.2 MHz	K Factor = 1590	f = 6.8 MHz, C = 46 pF

8.3.6 External CMOS

An external CMOS clock source is also supported as a core clock source. The XTAL2/EXTCLK pin on the device serves as the external clock input when running in this mode. When not selected as the SYSCLK source, the EXTCLK input is always re-synchronized to SYSCLK. XTAL1 is not used in external CMOS clock mode.

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

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

8.3.7 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 CLKSEL register. A typical full speed application would configure the USB clock to run directly from the HFOSC0 oscillator, while a typical low speed application would configure the clock for HFOSC0/8. 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 HFOSC0 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.

8.4 Clocking and Oscillator Control Registers

8.4.1 CLKSEL: Clock Select

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

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	USBCLK	0x0	RW	**USB Clock Source Select Bits.**
	Value	Name		Description
	0x0	HFOSC		USB clock (USBCLK) derived from the Internal High-Frequency Oscillator.
	0x1	HFOSC_DIV_8		USB clock (USBCLK) derived from the Internal High-Frequency Oscillator / 8.
	0x2	EXTOSC		USB clock (USBCLK) derived from the External Oscillator.
	0x3	EXTOSC_DIV_2		USB clock (USBCLK) derived from the External Oscillator / 2.
	0x4	EXTOSC_DIV_3		USB clock (USBCLK) derived from the External Oscillator / 3.
	0x5	EXTOSC_DIV_4		USB clock (USBCLK) derived from the External Oscillator / 4.
	0x6	LFOSC		USB clock (USBCLK) derived from the Internal Low-Frequency Oscillator.
3	OUTCLK	0	RW	**Crossbar Clock Out Select.**
	If the SYSCLK signal is enabled on the Crossbar, this bit selects between outputting SYSCLK and SYSCLK synchronized with the Port I/O pins.			
	Value	Name		Description
	0	SYSCLK		Enabling the Crossbar SYSCLK signal outputs SYSCLK.
	1	SYSCLK_SYNC_IO		Enabling the Crossbar SYSCLK signal outputs SYSCLK synchronized with the Port I/O.
2:0	CLKSL	0x0	RW	**System Clock Source Select Bits.**
	Value	Name		Description
	0x0	DIVIDED_HFOSC_DIV_4		Clock (SYSCLK) derived from the Internal High-Frequency Oscillator / 4 and scaled per the IFCN bits in register OSCICN.
	0x1	EXTOSC		Clock (SYSCLK) derived from the External Oscillator circuit.
	0x2	HFOSC_DIV_2		Clock (SYSCLK) derived from the Internal High-Frequency Oscillator / 2.
	0x3	HFOSC		Clock (SYSCLK) derived from the Internal High-Frequency Oscillator.

Bit	Name	Reset	Access	Description
	0x4	LFOSC		Clock (SYSCLK) derived from the Internal Low-Frequency Oscillator and scaled per the OSCLD bits in register OSCLCN.
Prior to switching to a system clock frequency > 25 MHz, ensure that the PFEN bit in the PFE0CN register and the FLRT bit in the FLSCL register have been set appropriately to ensure proper flash reads.				

8.4.2 HFO0CAL: High Frequency Oscillator Calibration

Bit	7	6	5	4	3	2	1	0
Name	Reserved	OSCICL						
Access	R	RW						
Reset	0	Varies						
SFR Page = ALL; SFR Address: 0xB3								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:0	OSCICL	Varies	RW	**Internal Oscillator Calibration.** These bits determine the internal oscillator period. When set to 0000000b, the oscillator operates at its fastest setting. When set to 1111111b, the oscillator operates at its slowest setting. The reset value is factory calibrated to generate an internal oscillator frequency of 48 MHz. OSCICL should only be changed by firmware when the oscillator is disabled (IOSCEN = 0).
The contents of this register are undefined when USB clock recovery is enabled.				

8.4.3 HFO0CN: High Frequency Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	IOSCEN	IFRDY	SUSPEND	Reserved			IFCN	
Access	RW	R	RW	R			RW	
Reset	1	1	0	0x0			0x0	
SFR Page = ALL; SFR Address: 0xB2								

Bit	Name	Reset	Access	Description
7	IOSCEN	1	RW	**Oscillator Enable.**
	Value	Name		Description
	0	DISABLED		Disable the High Frequency Oscillator.
	1	ENABLED		Enable the High Frequency Oscillator.
6	IFRDY	1	R	**Oscillator Frequency Ready Flag.**
	Value	Name		Description
	0	NOT_SET		The Internal High Frequency Oscillator is not running at the programmed frequency.
	1	SET		The Internal High Frequency Oscillator is running at the programmed frequency.
5	SUSPEND	0	RW	**Oscillator Suspend Enable.**
	Setting this bit to logic 1 places the internal oscillator in suspend mode. The internal oscillator resumes operation when one of the suspend mode awakening events occurs.			
4:2	*Reserved*	*Must write reset value.*		
1:0	IFCN	0x0	RW	**Oscillator Frequency Divider Control.**
	The Internal High Frequency Oscillator is divided by the IFCN bit setting after a divide-by-4 stage.			
	Value	Name		Description
	0x0	SYSCLK_DIV_8		SYSCLK can be derived from Internal H-F Oscillator divided by 8 (1.5 MHz).
	0x1	SYSCLK_DIV_4		SYSCLK can be derived from Internal H-F Oscillator divided by 4 (3 MHz).
	0x2	SYSCLK_DIV_2		SYSCLK can be derived from Internal H-F Oscillator divided by 2 (6 MHz).
	0x3	SYSCLK_DIV_1		SYSCLK can be derived from Internal H-F Oscillator divided by 1 (12 MHz).

8.4.4 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 = ALL; SFR Address: 0x86								

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

8.4.5 XOSC0CN: External Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	XCLKVLD	XOSCMD			Reserved	XFCN		
Access	R	RW			RW	RW		
Reset	0	0x0			0	0x0		
SFR Page = ALL; SFR Address: 0xB1								

Bit	Name	Reset	Access	Description
7	XCLKVLD	0	R	**External Oscillator Valid Flag.**
	Provides External Oscillator status and is valid at all times for all modes of operation except External CMOS Clock Mode and External CMOS Clock Mode with divide by 2. In these modes, XCLKVLD always returns 0.			
	Value	Name		Description
	0	NOT_SET		External Oscillator is unused or not yet stable.
	1	SET		External Oscillator is running and stable.
6:4	XOSCMD	0x0	RW	**External Oscillator Mode.**
	Value	Name		Description
	0x0	DISABLED		External Oscillator circuit disabled.
	0x2	CMOS		External CMOS Clock Mode.
	0x3	CMOS_DIV_2		External CMOS Clock Mode with divide by 2 stage.
	0x4	RC_DIV_2		RC Oscillator Mode with divide by 2 stage.
	0x5	C_DIV_2		Capacitor Oscillator Mode with divide by 2 stage.
	0x6	CRYSTAL		Crystal Oscillator Mode.
	0x7	CRYSTAL_DIV_2		Crystal Oscillator Mode with divide by 2 stage.
3	*Reserved*	*Must write reset value.*		
2:0	XFCN	0x0	RW	**External Oscillator Frequency Control.**
	Controls the external oscillator bias current. The value selected for this field depends on the frequency range of the external oscillator.			

9. Reset Sources and Power Supply Monitor

9.1 Introduction

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

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

All registers are reset to the predefined values noted in the register descriptions unless the bits only reset with a power-on reset. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost. 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.

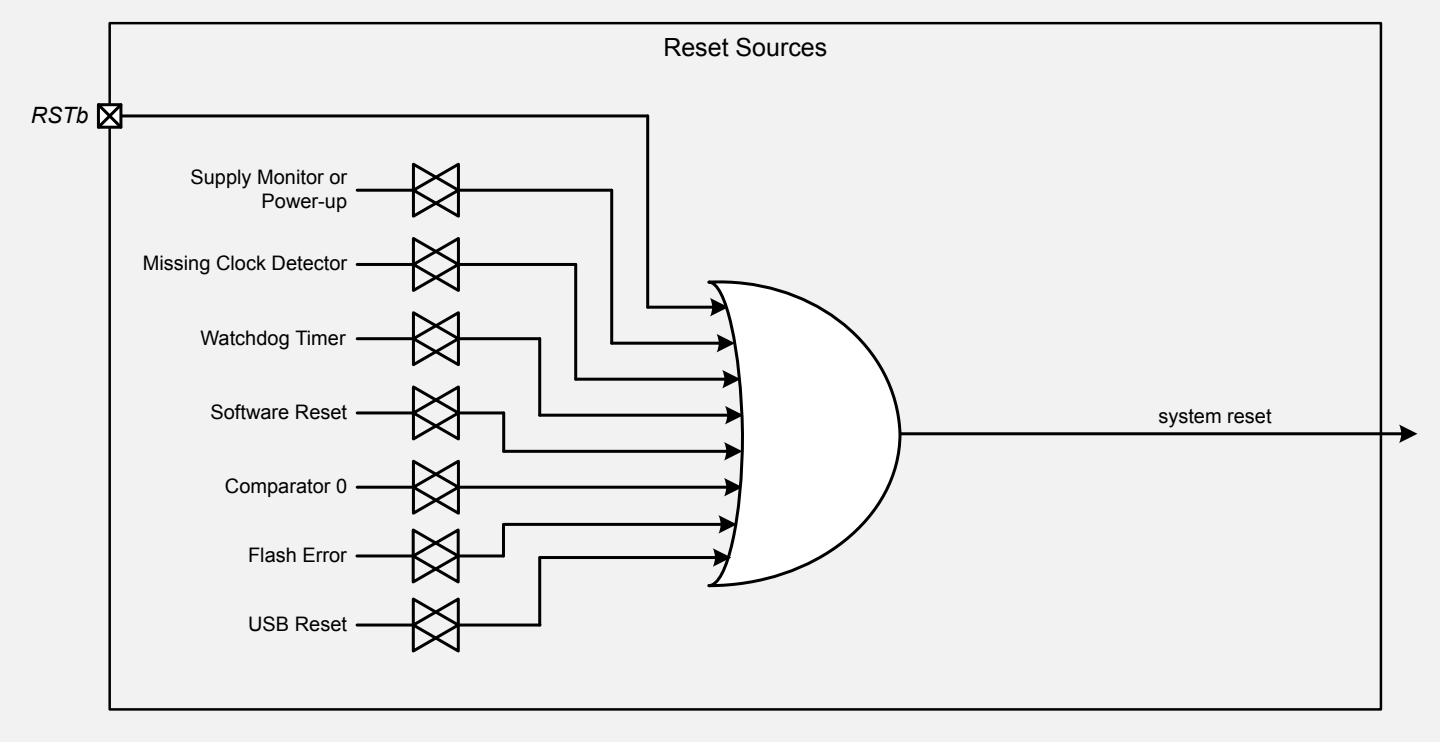


Figure 9.1. Reset Sources Block Diagram

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

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

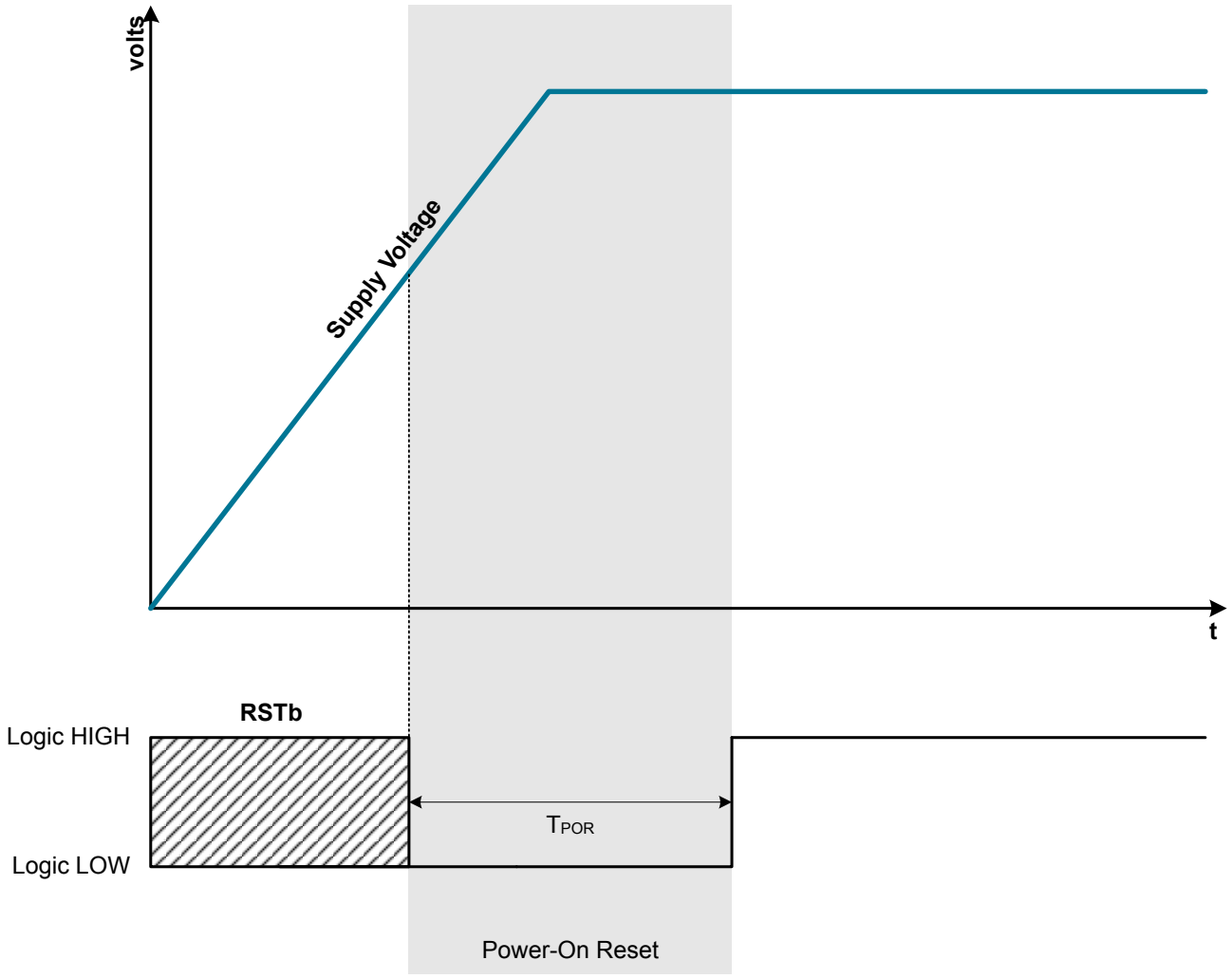


Figure 9.2. Power-On Reset Timing

9.3.3 Supply Monitor Reset

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

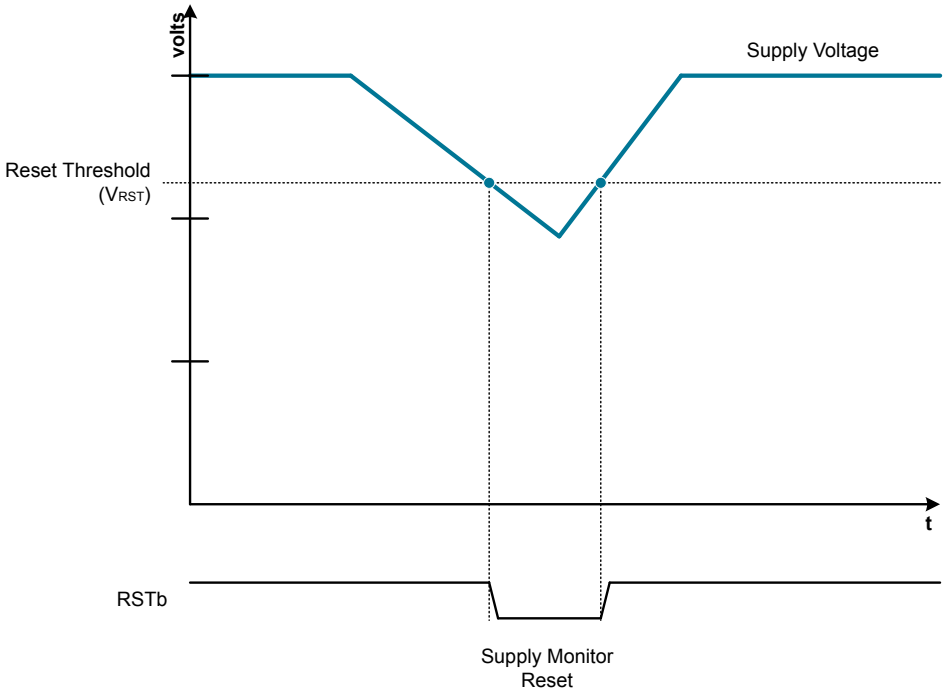


Figure 9.3. Reset Sources

9.3.4 External Reset

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

9.3.5 Missing Clock Detector Reset

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

9.3.6 Comparator (CMP0) Reset

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

9.3.7 PCA Watchdog Timer Reset

The programmable watchdog timer (WDT) function of the programmable counter array (PCA) can be used to prevent software from running out of control during a system malfunction. The PCA WDT function can be enabled or disabled by software as described in the PCA documentation. The WDT is enabled and clocked by SYSCLK/12 following any reset. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit in RSTSRC 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 = ALL; 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 = ALL; 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.

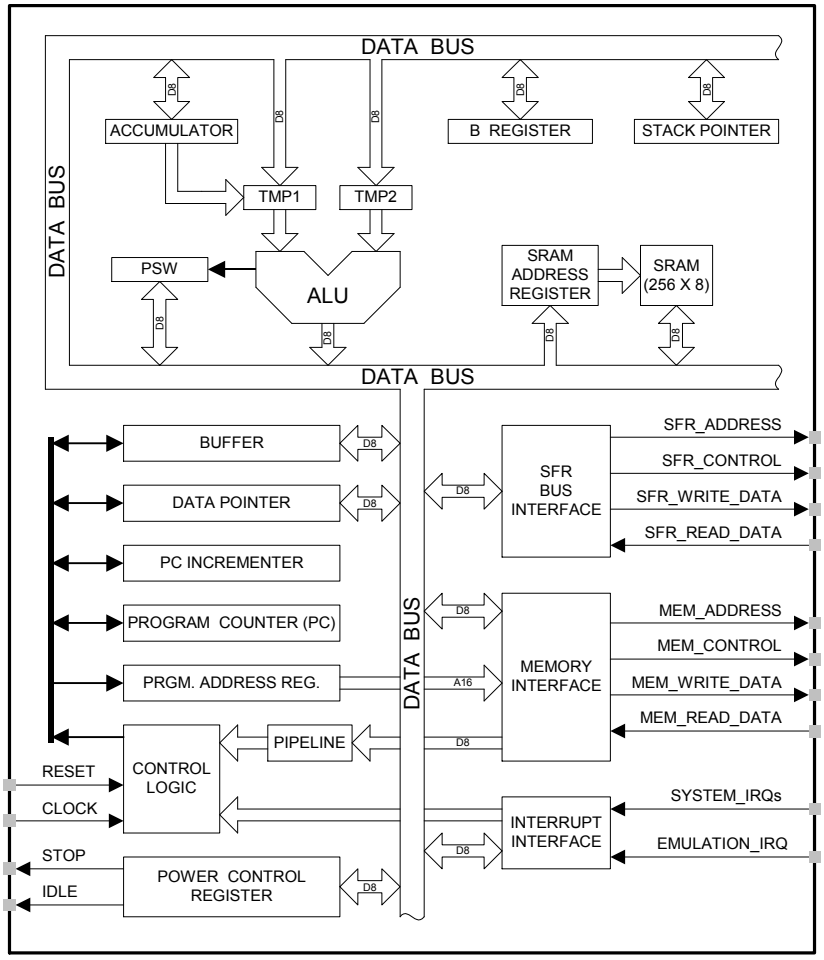


Figure 10.1. CIP-51 Block Diagram

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 48 MHz operating clock frequency.
- 48 MIPS peak throughput with 48 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 or two less clock cycles 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.

Note: It is recommended that the prefetch be used for optimal code execution timing. However, the prefetch can be disabled when the device is in Suspend mode to save power.

Table 10.2. CIP-51 Instruction Set Summary

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch on SYSCLK = 12 MHz FLRT = 0)	prefetch on SYSCLK = 48 MHz FLRT = 1)
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

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch on SYSCLK = 12 MHz FLRT = 0)	prefetch on SYSCLK = 48 MHz FLRT = 1)
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
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

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch on SYSCLK = 12 MHz FLRT = 0)	prefetch on SYSCLK = 48 MHz FLRT = 1)
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
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

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch on SYSCLK = 12 MHz FLRT = 0)	prefetch on SYSCLK = 48 MHz FLRT = 1)
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 4	2 or 6
JNC rel	Jump if Carry is not set	2	2 or 4	2 or 5
JB bit, rel	Jump if direct bit is set	3	3 or 5	3 or 7
JNB bit, rel	Jump if direct bit is not set	3	3 or 5	3 or 6
JBC bit, rel	Jump if direct bit is set and clear bit	3	3 or 5	3 or 7
Program Branching				
ACALL addr11	Absolute subroutine call	2	4	6
LCALL addr16	Long subroutine call	3	5	7
RET	Return from subroutine	1	6	8
RETI	Return from interrupt	1	6	7
AJMP addr11	Absolute jump	2	4	6
LJMP addr16	Long jump	3	4	6
SJMP rel	Short jump (relative address)	2	4	6
JMP @A+DPTR	Jump indirect relative to DPTR	1	3	5
JZ rel	Jump if A equals zero	2	2 or 4	2 or 5
JNZ rel	Jump if A does not equal zero	2	2 or 4	2 or 5
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	4 or 6	4 or 7
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3 or 5	3 or 6
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3 or 5	3 or 6
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4 or 6	4 or 7
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2 or 4	2 or 5
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3 or 5	3 or 7
NOP	No operation	1	1	1

Mnemonic	Description	Bytes	Clock Cycles	
			prefetch on SYSCLK = 12 MHz FLRT = 0)	prefetch on SYSCLK = 48 MHz FLRT = 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

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	Reserved				FLBWE
Access	R		RW	R				RW
Reset	0x0		0	0x0				0
SFR Page = ALL; SFR Address: 0xAF								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	PFEN	1	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
4:1	*Reserved*	*Must write reset value.*		
0	FLBWE	0	RW	**Flash Block Write Enable.** This bit allows block writes to Flash memory from firmware.

Value	Name	Description
0	BLOCK_WRITE_DISABLED	Each byte of a firmware flash write is written individually.
1	BLOCK_WRITE_ENABLED	Flash bytes are written in groups of two.

11. Port I/O, Crossbar and External Interrupts

11.1 Introduction

Digital and analog resources are externally available on the device's multi-purpose I/O pins. Port pins P0.0-P3.7 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 P4.0-P4.7 can be used as GPIO. Additionally, the C2 Interface Data signal (C2D) is shared with P3.0 on some packages.

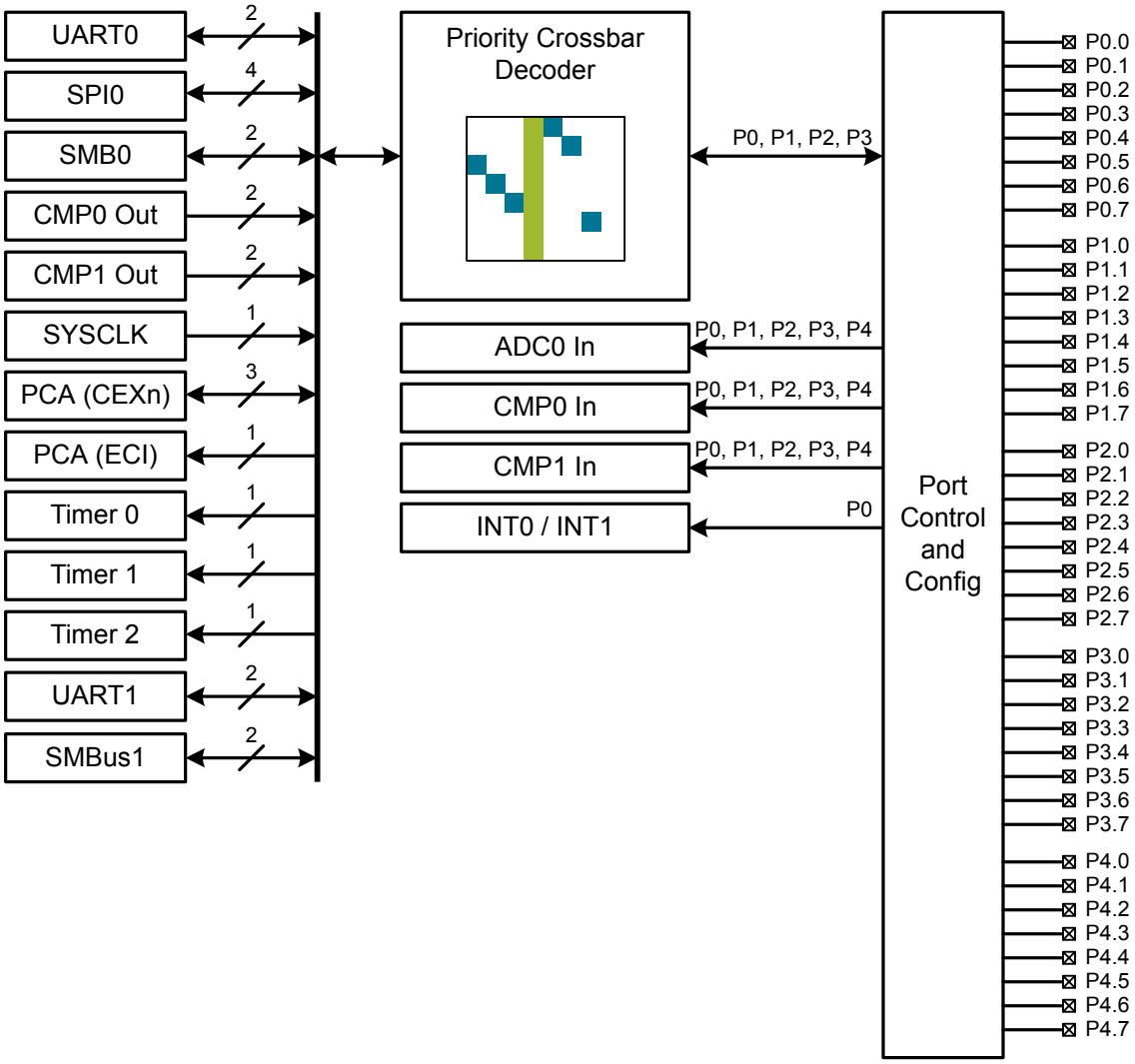


Figure 11.1. Port I/O Block Diagram

11.2 Features

The port control block offers the following features:

- Up to 40 multi-functions I/O pins, supporting digital and analog functions.
- Flexible priority crossbar decoder for digital peripheral assignment.
- Two direct-pin interrupt sources with dedicated interrupt vectors (INT0 and INT1) available on P0 pins.

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.

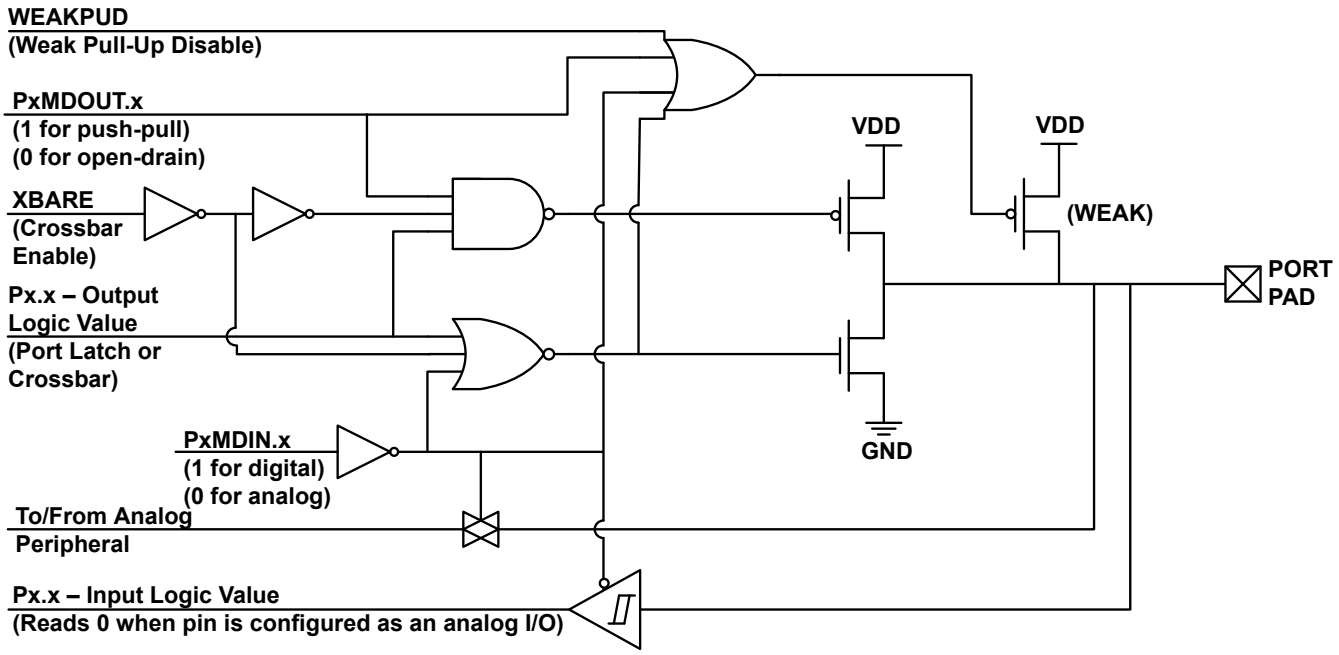


Figure 11.2. Port I/O Cell Block Diagram

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.

11.3.2 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 – P0.7, P1.2 – P1.4	ADC0MX, PnSKIP, PnMDIN
Comparator 0 Input	P1.0, P0.1	CMP0MX, PnSKIP, PnMDIN
Voltage Reference (VREF)	P1.5 (TQFP48) P0.7 (LQFP32, QFN32)	REF0CN, PnSKIP, PnMDIN
External Oscillator Input (XTAL2)	P0.7 (TQFP48) Not available (LQFP32, QFN32)	HFO0CN, PnSKIP, PnMDIN
External Oscillator Output (XTAL1)	P0.6 (TQFP48) Not available (LQFP32, QFN32)	HFO0CN, 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, SPI0, SMB0, CP0, CP0A, CP1, CP1A, SYSCLK, PCA0 (CEX0-4 and ECI), T0, T1, UART1, SMBus1	Any port pin available for assignment by the crossbar. This includes P0.0 – P3.7 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
External Interrupt 0, External Interrupt 1	P0.0 – P0.7	IT01CF
Conversion Start (CNVSTR)	P1.4 (TQFP48) P0.6 (LQFP32, QFN32)	ADC0CN0
External Clock Input (EXTCLK)	P0.3 (LQFP32, QFN32)	CLKSEL
Any pin used for GPIO	P0.0 – P3.7	P0SKIP, P1SKIP, P2SKIP, P3SKIP

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 INT0 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 94 shows an example of the resulting pin assignments of the device with UART0 and SPI0 enabled and P0.3 skipped (P0SKIP = 0x08). UART0 is the highest priority and it will be assigned first. The UART0 pins can only appear at fixed locations (in this example, P0.4 and P0.5), so it occupies those pins. The next-highest enabled peripheral is SPI0. P0.0, P0.1 and P0.2 are free, so SPI0 takes these three pins. The fourth pin, NSS, is routed to P0.6 because P0.3 is skipped and P0.4 and P0.5 are already occupied by the UART. Any other pins on the device are available for use as general-purpose digital I/O or analog functions.

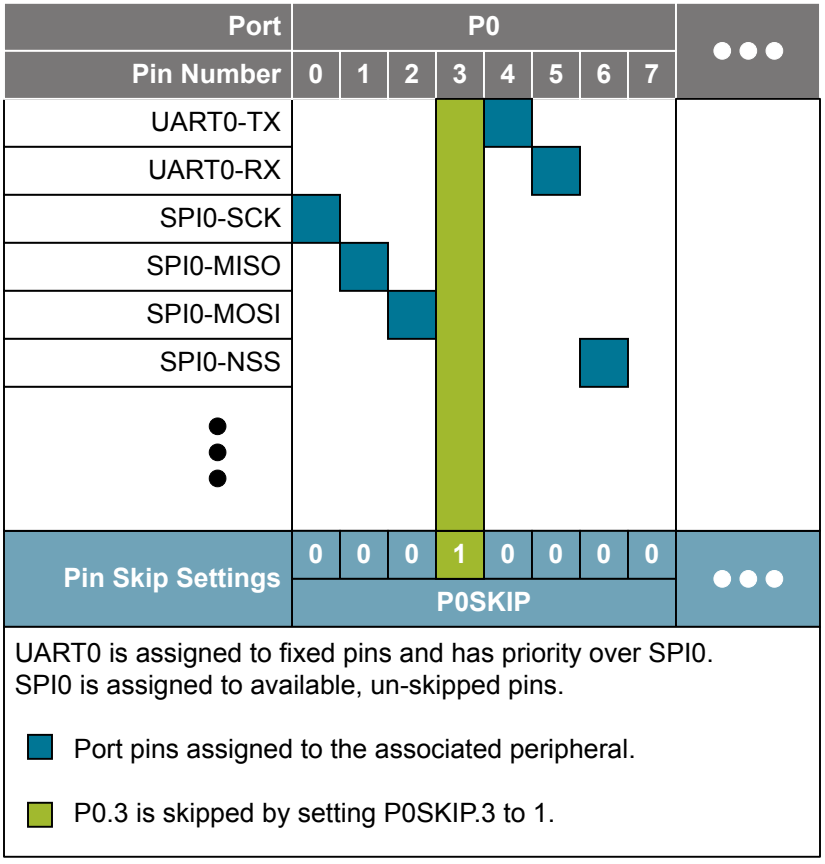


Figure 11.3. Crossbar Priority Decoder Example Assignments

11.3.3.1 Crossbar Functional Map

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

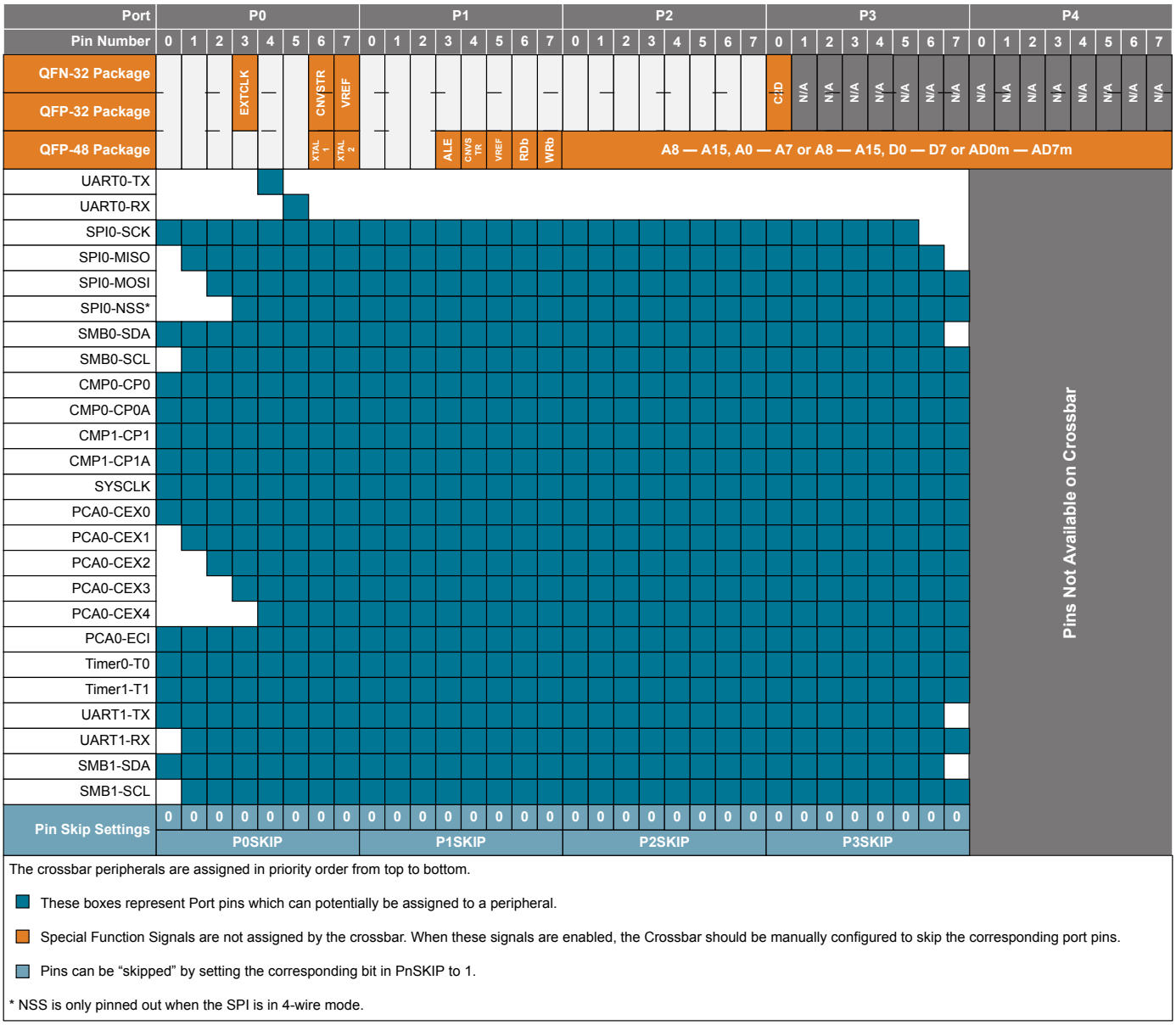


Figure 11.4. Full Crossbar Map

11.3.4 INT0 and INT1

Two direct-pin digital interrupt sources (INT0 and INT1) are included, which can be routed to port 0 pins. 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 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	CP1AE	CP1E	CP0AE	CP0E	SYSCKE	SMB0E	SPI0E	URT0E
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xE1								

Bit	Name	Reset	Access	Description
7	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.
6	CP1E	0	RW	**Comparator1 Output Enable.**
	Value	Name		Description
	0	DISABLED		CP1 unavailable at Port pin.
	1	ENABLED		CP1 routed to Port pin.
5	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.
4	CP0E	0	RW	**Comparator0 Output Enable.**
	Value	Name		Description
	0	DISABLED		CP0 unavailable at Port pin.
	1	ENABLED		CP0 routed to Port pin.
3	SYSCKE	0	RW	**SYSCLK Output Enable.**
	Value	Name		Description
	0	DISABLED		SYSCLK unavailable at Port pin.
	1	ENABLED		SYSCLK output 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 Output Enable.**
	Value	Name		Description
	0	DISABLED		UART0 I/O unavailable at Port pin.
	1	ENABLED		UART0 TX, RX 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	WEAKPUD	XBARE	T1E	T0E	ECIE	PCA0ME		
Access	RW	RW	RW	RW	RW	RW		
Reset	0	0	0	0	0	0x0		
SFR Page = ALL; SFR Address: 0xE2								

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	T1E	0	RW	**T1 Enable.**
	Value	Name		Description
	0	DISABLED		T1 unavailable at Port pin.
	1	ENABLED		T1 routed to Port pin.
4	T0E	0	RW	**T0 Enable.**
	Value	Name		Description
	0	DISABLED		T0 unavailable at Port pin.
	1	ENABLED		T0 routed to Port pin.
3	ECIE	0	RW	**PCA0 External Counter Input Enable.**
	Value	Name		Description
	0	DISABLED		ECI unavailable at Port pin.
	1	ENABLED		ECI routed to Port pin.
2:0	PCA0ME	0x0	RW	**PCA Module I/O Enable.**
	Value	Name		Description
	0x0	DISABLED		All PCA I/O unavailable at Port pins.
	0x1	CEX0		CEX0 routed to Port pin.
	0x2	CEX0_CEX1		CEX0, CEX1 routed to Port pins.
	0x3	CEX0_CEX1_CEX2		CEX0, CEX1, CEX2 routed to Port pins.

Bit	Name	Reset	Access	Description
	0x4	CEX0_CEX1_CEX2_CEX3		CEX0, CEX1, CEX2, CEX3 routed to Port pins.
	0x5	CEX0_CEX1_CEX2_CEX3_CEX4		CEX0, CEX1, CEX2, CEX3, CEX4 routed to Port pins.

11.4.3 XBR2: Port I/O Crossbar 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved						SMB1E	URT1E
Access	RW						RW	RW
Reset	0x00						0	0
SFR Page = ALL; SFR Address: 0xE3								

Bit	Name	Reset	Access	Description
7:2	*Reserved*	*Must write reset value.*		
1	SMB1E	0	RW	**SMBus1 I/O Enable.**
	Value	Name		Description
	0	DISABLED		SMBus1 I/O unavailable at Port pins.
	1	ENABLED		SMBus1 I/O routed to Port pins.
0	URT1E	0	RW	**UART1 I/O Output Enable.**
	Value	Name		Description
	0	DISABLED		UART1 I/O unavailable at Port pin.
	1	ENABLED		UART1 TX, RX routed to Port pins.

11.4.4 P0: Port 0 Pin Latch

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

SFR Page = ALL; SFR Address: 0x80 (bit-addressable)

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

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

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

11.4.5 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 = ALL; SFR Address: 0xF1								

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

11.4.6 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 = ALL; 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			

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

11.4.8 P1: Port 1 Pin Latch

Bit	7	6	5	4	3	2	1	0
Name	B7	B6	B5	B4	B3	B2	B1	B0
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	1	1	1	1	1	1	1	1
SFR Page = ALL; SFR Address: 0x90 (bit-addressable)								

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

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

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

11.4.9 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 = ALL; SFR Address: 0xF2								

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

11.4.10 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 = ALL; 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			

11.4.11 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 = ALL; SFR Address: 0xD5								

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

11.4.12 P2: Port 2 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: 0xA0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 2 Bit 7 Latch.**
	Value	Name		Description
	0	LOW		P2.7 is low. Set P2.7 to drive low.
	1	HIGH		P2.7 is high. Set P2.7 to drive or float high.
6	B6 See bit 7 description	1	RW	**Port 2 Bit 6 Latch.**
5	B5 See bit 7 description	1	RW	**Port 2 Bit 5 Latch.**
4	B4 See bit 7 description	1	RW	**Port 2 Bit 4 Latch.**
3	B3 See bit 7 description	1	RW	**Port 2 Bit 3 Latch.**
2	B2 See bit 7 description	1	RW	**Port 2 Bit 2 Latch.**
1	B1 See bit 7 description	1	RW	**Port 2 Bit 1 Latch.**
0	B0 See bit 7 description	1	RW	**Port 2 Bit 0 Latch.**

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

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

11.4.13 P2MDIN: Port 2 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 = ALL; SFR Address: 0xF3								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 2 Bit 7 Input Mode.**
	Value	Name		Description
	0	ANALOG		P2.7 pin is configured for analog mode.
	1	DIGITAL		P2.7 pin is configured for digital mode.
6	B6 See bit 7 description	1	RW	**Port 2 Bit 6 Input Mode.**
5	B5 See bit 7 description	1	RW	**Port 2 Bit 5 Input Mode.**
4	B4 See bit 7 description	1	RW	**Port 2 Bit 4 Input Mode.**
3	B3 See bit 7 description	1	RW	**Port 2 Bit 3 Input Mode.**
2	B2 See bit 7 description	1	RW	**Port 2 Bit 2 Input Mode.**
1	B1 See bit 7 description	1	RW	**Port 2 Bit 1 Input Mode.**
0	B0 See bit 7 description	1	RW	**Port 2 Bit 0 Input Mode.**
Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.				

11.4.14 P2MDOUT: Port 2 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 = ALL; SFR Address: 0xA6								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 2 Bit 7 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P2.7 output is open-drain.
	1	PUSH_PULL		P2.7 output is push-pull.
6	B6	0	RW	**Port 2 Bit 6 Output Mode.**
	See bit 7 description			
5	B5	0	RW	**Port 2 Bit 5 Output Mode.**
	See bit 7 description			
4	B4	0	RW	**Port 2 Bit 4 Output Mode.**
	See bit 7 description			
3	B3	0	RW	**Port 2 Bit 3 Output Mode.**
	See bit 7 description			
2	B2	0	RW	**Port 2 Bit 2 Output Mode.**
	See bit 7 description			
1	B1	0	RW	**Port 2 Bit 1 Output Mode.**
	See bit 7 description			
0	B0	0	RW	**Port 2 Bit 0 Output Mode.**
	See bit 7 description			

11.4.15 P2SKIP: Port 2 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 = ALL; SFR Address: 0xD6								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 2 Bit 7 Skip.**
	Value	Name		Description
	0	NOT_SKIPPED		P2.7 pin is not skipped by the crossbar.
	1	SKIPPED		P2.7 pin is skipped by the crossbar.
6	B6 See bit 7 description	0	RW	**Port 2 Bit 6 Skip.**
5	B5 See bit 7 description	0	RW	**Port 2 Bit 5 Skip.**
4	B4 See bit 7 description	0	RW	**Port 2 Bit 4 Skip.**
3	B3 See bit 7 description	0	RW	**Port 2 Bit 3 Skip.**
2	B2 See bit 7 description	0	RW	**Port 2 Bit 2 Skip.**
1	B1 See bit 7 description	0	RW	**Port 2 Bit 1 Skip.**
0	B0 See bit 7 description	0	RW	**Port 2 Bit 0 Skip.**

11.4.16 P3: Port 3 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: 0xB0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 3 Bit 7 Latch.**
	Value	Name		Description
	0	LOW		P3.7 is low. Set P3.7 to drive low.
	1	HIGH		P3.7 is high. Set P3.7 to drive or float high.
6	B6 See bit 7 description	1	RW	**Port 3 Bit 6 Latch.**
5	B5 See bit 7 description	1	RW	**Port 3 Bit 5 Latch.**
4	B4 See bit 7 description	1	RW	**Port 3 Bit 4 Latch.**
3	B3 See bit 7 description	1	RW	**Port 3 Bit 3 Latch.**
2	B2 See bit 7 description	1	RW	**Port 3 Bit 2 Latch.**
1	B1 See bit 7 description	1	RW	**Port 3 Bit 1 Latch.**
0	B0 See bit 7 description	1	RW	**Port 3 Bit 0 Latch.**

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

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

Port 3 consists of 8 bits (P3.0-P3.7) on TQFP48 packages and 1 bit (P3.0) on LQFP32 and QFN32 packages.

11.4.17 P3MDIN: Port 3 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 = ALL; SFR Address: 0xF4								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 3 Bit 7 Input Mode.**
	Value	Name		Description
	0	ANALOG		P3.7 pin is configured for analog mode.
	1	DIGITAL		P3.7 pin is configured for digital mode.
6	B6 See bit 7 description	1	RW	**Port 3 Bit 6 Input Mode.**
5	B5 See bit 7 description	1	RW	**Port 3 Bit 5 Input Mode.**
4	B4 See bit 7 description	1	RW	**Port 3 Bit 4 Input Mode.**
3	B3 See bit 7 description	1	RW	**Port 3 Bit 3 Input Mode.**
2	B2 See bit 7 description	1	RW	**Port 3 Bit 2 Input Mode.**
1	B1 See bit 7 description	1	RW	**Port 3 Bit 1 Input Mode.**
0	B0 See bit 7 description	1	RW	**Port 3 Bit 0 Input Mode.**

Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.

Port 3 consists of 8 bits (P3.0-P3.7) on TQFP48 packages and 1 bit (P3.0) on LQFP32 and QFN32 packages.

11.4.18 P3MDOUT: Port 3 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 = ALL; SFR Address: 0xA7								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 3 Bit 7 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P3.7 output is open-drain.
	1	PUSH_PULL		P3.7 output is push-pull.
6	B6	0	RW	**Port 3 Bit 6 Output Mode.**
	See bit 7 description			
5	B5	0	RW	**Port 3 Bit 5 Output Mode.**
	See bit 7 description			
4	B4	0	RW	**Port 3 Bit 4 Output Mode.**
	See bit 7 description			
3	B3	0	RW	**Port 3 Bit 3 Output Mode.**
	See bit 7 description			
2	B2	0	RW	**Port 3 Bit 2 Output Mode.**
	See bit 7 description			
1	B1	0	RW	**Port 3 Bit 1 Output Mode.**
	See bit 7 description			
0	B0	0	RW	**Port 3 Bit 0 Output Mode.**
	See bit 7 description			
Port 3 consists of 8 bits (P3.0-P3.7) on TQFP48 packages and 1 bit (P3.0) on LQFP32 and QFN32 packages.				

11.4.19 P3SKIP: Port 3 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 = ALL; SFR Address: 0xDF								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 3 Bit 7 Skip.**
	Value	Name		Description
	0	NOT_SKIPPED		P3.7 pin is not skipped by the crossbar.
	1	SKIPPED		P3.7 pin is skipped by the crossbar.
6	B6 See bit 7 description	0	RW	**Port 3 Bit 6 Skip.**
5	B5 See bit 7 description	0	RW	**Port 3 Bit 5 Skip.**
4	B4 See bit 7 description	0	RW	**Port 3 Bit 4 Skip.**
3	B3 See bit 7 description	0	RW	**Port 3 Bit 3 Skip.**
2	B2 See bit 7 description	0	RW	**Port 3 Bit 2 Skip.**
1	B1 See bit 7 description	0	RW	**Port 3 Bit 1 Skip.**
0	B0 See bit 7 description	0	RW	**Port 3 Bit 0 Skip.**
Port 3 consists of 8 bits (P3.0-P3.7) on TQFP48 packages and 1 bit (P3.0) on LQFP32 and QFN32 packages.				

11.4.20 P4: Port 4 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: 0xC7								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 4 Bit 7 Latch.**
	Value	Name		Description
	0	LOW		P4.7 is low. Set P4.7 to drive low.
	1	HIGH		P4.7 is high. Set P4.7 to drive or float high.
6	B6 See bit 7 description	1	RW	**Port 4 Bit 6 Latch.**
5	B5 See bit 7 description	1	RW	**Port 4 Bit 5 Latch.**
4	B4 See bit 7 description	1	RW	**Port 4 Bit 4 Latch.**
3	B3 See bit 7 description	1	RW	**Port 4 Bit 3 Latch.**
2	B2 See bit 7 description	1	RW	**Port 4 Bit 2 Latch.**
1	B1 See bit 7 description	1	RW	**Port 4 Bit 1 Latch.**
0	B0 See bit 7 description	1	RW	**Port 4 Bit 0 Latch.**

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

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

Port 4 consists of 8 bits (P4.0-P4.7) on TQFP48 packages and is unavailable on LQFP32 and QFN32 packages.

11.4.21 P4MDIN: Port 4 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 = ALL; SFR Address: 0xF5								

Bit	Name	Reset	Access	Description
7	B7	1	RW	**Port 4 Bit 7 Input Mode.**
	Value	Name		Description
	0	ANALOG		P4.7 pin is configured for analog mode.
	1	DIGITAL		P4.7 pin is configured for digital mode.
6	B6 See bit 7 description	1	RW	**Port 4 Bit 6 Input Mode.**
5	B5 See bit 7 description	1	RW	**Port 4 Bit 5 Input Mode.**
4	B4 See bit 7 description	1	RW	**Port 4 Bit 4 Input Mode.**
3	B3 See bit 7 description	1	RW	**Port 4 Bit 3 Input Mode.**
2	B2 See bit 7 description	1	RW	**Port 4 Bit 2 Input Mode.**
1	B1 See bit 7 description	1	RW	**Port 4 Bit 1 Input Mode.**
0	B0 See bit 7 description	1	RW	**Port 4 Bit 0 Input Mode.**

Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.

Port 4 consists of 8 bits (P4.0-P4.7) on TQFP48 packages and is unavailable on LQFP32 and QFN32 packages.

11.4.22 P4MDOUT: Port 4 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 = ALL; SFR Address: 0xAE								

Bit	Name	Reset	Access	Description
7	B7	0	RW	**Port 4 Bit 7 Output Mode.**
	Value	Name		Description
	0	OPEN_DRAIN		P4.7 output is open-drain.
	1	PUSH_PULL		P4.7 output is push-pull.
6	B6	0	RW	**Port 4 Bit 6 Output Mode.**
	See bit 7 description			
5	B5	0	RW	**Port 4 Bit 5 Output Mode.**
	See bit 7 description			
4	B4	0	RW	**Port 4 Bit 4 Output Mode.**
	See bit 7 description			
3	B3	0	RW	**Port 4 Bit 3 Output Mode.**
	See bit 7 description			
2	B2	0	RW	**Port 4 Bit 2 Output Mode.**
	See bit 7 description			
1	B1	0	RW	**Port 4 Bit 1 Output Mode.**
	See bit 7 description			
0	B0	0	RW	**Port 4 Bit 0 Output Mode.**
	See bit 7 description			
Port 4 consists of 8 bits (P4.0-P4.7) on TQFP48 packages and is unavailable on LQFP32 and QFN32 packages.				

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; 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.
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.
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.
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 10-bit mode, 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.

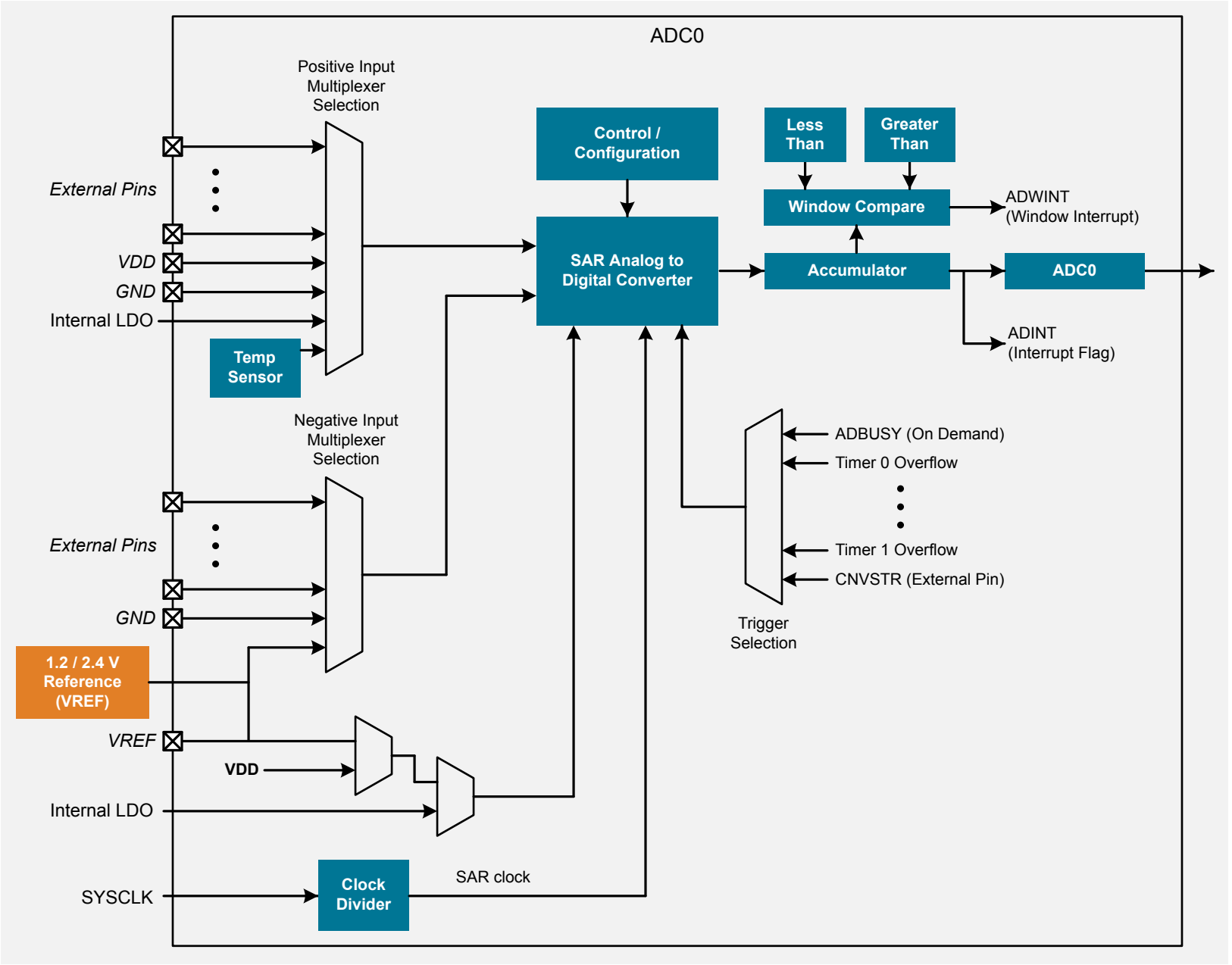


Figure 12.1. ADC Block Diagram

12.2 Features

The ADC module is a Successive Approximation Register (SAR) Analog to Digital Converter (ADC). The key features of this ADC module are:

- Up to 32 external inputs.
- Differential or Single-ended 10-bit operation.
- Supports an output update rate of 500 ksps samples per second.
- Asynchronous hardware conversion trigger, selectable between software, external I/O and internal timer sources.
- Output data window comparator allows automatic range checking.
- Two tracking mode options with programmable tracking time.
- Conversion complete and window compare interrupts supported.
- Flexible output data formatting.
- Voltage reference selectable from external reference pin, on-chip precision reference (driven externally on reference pin), or VDD supply.
- Integrated temperature sensor.

12.3 Functional Description

12.3.1 Clocking

The ADC is clocked by an adjustable conversion clock (SARCLK), which is a divided version of the selected system clock. The clock divide value is determined by the ADSC 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 for the ADC is configurable to use an externally connected voltage reference, the on-chip reference voltage generator routed to the VREF pin, the unregulated power supply voltage (VDD), or the regulated 1.8 V internal supply. The REFSL bit in the REF0CN register selects the reference source for the ADC. For an external source or the on-chip reference, REFSL should be set to 0 to select the VREF pin. To use VDD as the reference source, REFSL should be set to 1. To override this selection and use the internal regulator as the reference source, the REGOVR bit can be set to 1.

12.3.2.1 Internal Voltage Reference

The on-chip voltage reference circuit consists of a 1.2 V, temperature stable bandgap voltage reference generator and a selectable-gain output buffer amplifier. The buffer is configured for 1x or 2x gain using the REFBGS bit in register REF0CN. On the 1x gain setting the output voltage is nominally 1.2 V, and on the 2x gain setting the output voltage is nominally 2.4 V. The on-chip voltage reference can be driven on the VREF pin by setting the REFBE bit in register REF0CN to a 1. The maximum load seen by the VREF pin must be less than 200 µA to GND. Bypass capacitors of 0.1 µF and 4.7 µF are recommended from the VREF pin to GND, and a minimum of 0.1 µF is required. If the on-chip reference is not used, the REFBE bit should be cleared to 0.

Note: When using either an external voltage reference or the on-chip reference circuitry, the VREF pin should be configured as an analog pin and skipped by the Digital Crossbar.

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.

12.3.3 Input Selection

The ADC has analog multiplexers which allow selection of external pins, the on-chip temperature sensor, the internal regulated supply, VREF, the VDD supply, or GND for the positive and negative inputs. ADC input channels are selected using the AMX0P and AMX0N registers.

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 Positive Input Multiplexer Channels

AMX0P setting	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
000000	ADC0P.0	P2.0	P1.0	P1.0
000001	ADC0P.1	P2.1	P1.1	P1.1
000010	ADC0P.2	P2.2	P1.2	P1.2
000011	ADC0P.3	P2.3	P1.3	P1.3
000100	ADC0P.4	P2.5	P1.4	P1.4
000101	ADC0P.5	P2.6	P1.5	P1.5
000110	ADC0P.6	P3.0	P1.6	P1.6
000111	ADC0P.7	P3.1	P1.7	P1.7
001000	ADC0P.8	P3.4	P2.0	P2.0
001001	ADC0P.9	P3.5	P2.1	P2.1
001010	ADC0P.10	P3.7	P2.2	P2.2
001011	ADC0P.11	P4.0	P2.3	P2.3
001100	ADC0P.12	P4.3	P2.4	P2.4
001101	ADC0P.13	P4.4	P2.5	P2.5
001110	ADC0P.14	P4.5	P2.6	P2.6
001111	ADC0P.15	P4.6	P2.7	P2.7
010000	ADC0P.16	Reserved	P3.0	P3.0
010001	ADC0P.17	P0.3	P0.0	P0.0
010010	ADC0P.18	P0.4	P0.1	P0.1
010011	ADC0P.19	P1.1	P0.4	P0.4
010100	ADC0P.20	P1.2	P0.5	P0.5
010101	ADC0P.21	P1.0	Reserved	Reserved
010110	ADC0P.22	P1.3	Reserved	Reserved
010111	ADC0P.23	P1.6	Reserved	Reserved
011000	ADC0P.24	P1.7	Reserved	Reserved
011001	ADC0P.25	P2.4	Reserved	Reserved
011010	ADC0P.26	P2.7	Reserved	Reserved
011011	ADC0P.27	P3.2	Reserved	Reserved

AMX0P setting	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
011100	ADC0P.28	P3.3	Reserved	Reserved
011101	ADC0P.29	P3.6	Reserved	Reserved
011110	ADC0P.30	Internal Temperature Sensor		
011111	ADC0P.31	VDD Supply Pin		
100000	ADC0P.32	P4.1	Reserved	Reserved
100001	ADC0P.33	P4.2	Reserved	Reserved
100010	ADC0P.34	P4.7	Reserved	Reserved
100011 - 111111	ADC0P.35 - ADC0P.63	Reserved	Reserved	Reserved

Table 12.2. ADC0 Negative Input Multiplexer Channels

AMX0N setting	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
000000	ADC0N.0	P2.0	P1.0	P1.0
000001	ADC0N.1	P2.1	P1.1	P1.1
000010	ADC0N.2	P2.2	P1.2	P1.2
000011	ADC0N.3	P2.3	P1.3	P1.3
000100	ADC0N.4	P2.5	P1.4	P1.4
000101	ADC0N.5	P2.6	P1.5	P1.5
000110	ADC0N.6	P3.0	P1.6	P1.6
000111	ADC0N.7	P3.1	P1.7	P1.7
001000	ADC0N.8	P3.4	P2.0	P2.0
001001	ADC0N.9	P3.5	P2.1	P2.1
001010	ADC0N.10	P3.7	P2.2	P2.2
001011	ADC0N.11	P4.0	P2.3	P2.3
001100	ADC0N.12	P4.3	P2.4	P2.4
001101	ADC0N.13	P4.4	P2.5	P2.5
001110	ADC0N.14	P4.5	P2.6	P2.6
001111	ADC0N.15	P4.6	P2.7	P2.7
010000	ADC0N.16	Reserved	P3.0	P3.0
010001	ADC0N.17	P0.3	P0.0	P0.0
010010	ADC0N.18	P0.4	P0.1	P0.1
010011	ADC0N.19	P1.1	P0.4	P0.4
010100	ADC0N.20	P1.2	P0.5	P0.5
010101	ADC0N.21	P1.0	Reserved	Reserved
010110	ADC0N.22	P1.3	Reserved	Reserved
010111	ADC0N.23	P1.6	Reserved	Reserved
011000	ADC0N.24	P1.7	Reserved	Reserved
011001	ADC0N.25	P2.4	Reserved	Reserved

AMX0N setting	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
011010	ADC0N.26	P2.7	Reserved	Reserved
011011	ADC0N.27	P3.2	Reserved	Reserved
011100	ADC0N.28	P3.3	Reserved	Reserved
011101	ADC0N.29	P3.6	Reserved	Reserved
011110	ADC0N.30	Internal VREF		
011111	ADC0N.31	Ground		
100000	ADC0N.32	P4.1	Reserved	Reserved
100001	ADC0N.33	P4.2	Reserved	Reserved
100010	ADC0N.34	P4.7	Reserved	Reserved
100011 - 111111	ADC0N.35 - ADC0N.63	Reserved	Reserved	Reserved

12.3.4 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.

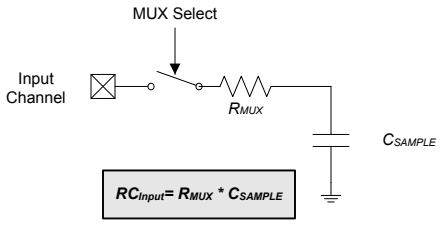
Note: The CNVSTR pin is a multi-function GPIO pin. When the CNVSTR input is used as the ADC conversion source, the associated port pin should be skipped in the crossbar settings.

12.3.5 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 Equivalent Input Circuit

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

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

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

t is the required settling time in seconds

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

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

n is the ADC resolution in bits.

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

Configuring the Tracking Time

The 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 3 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.

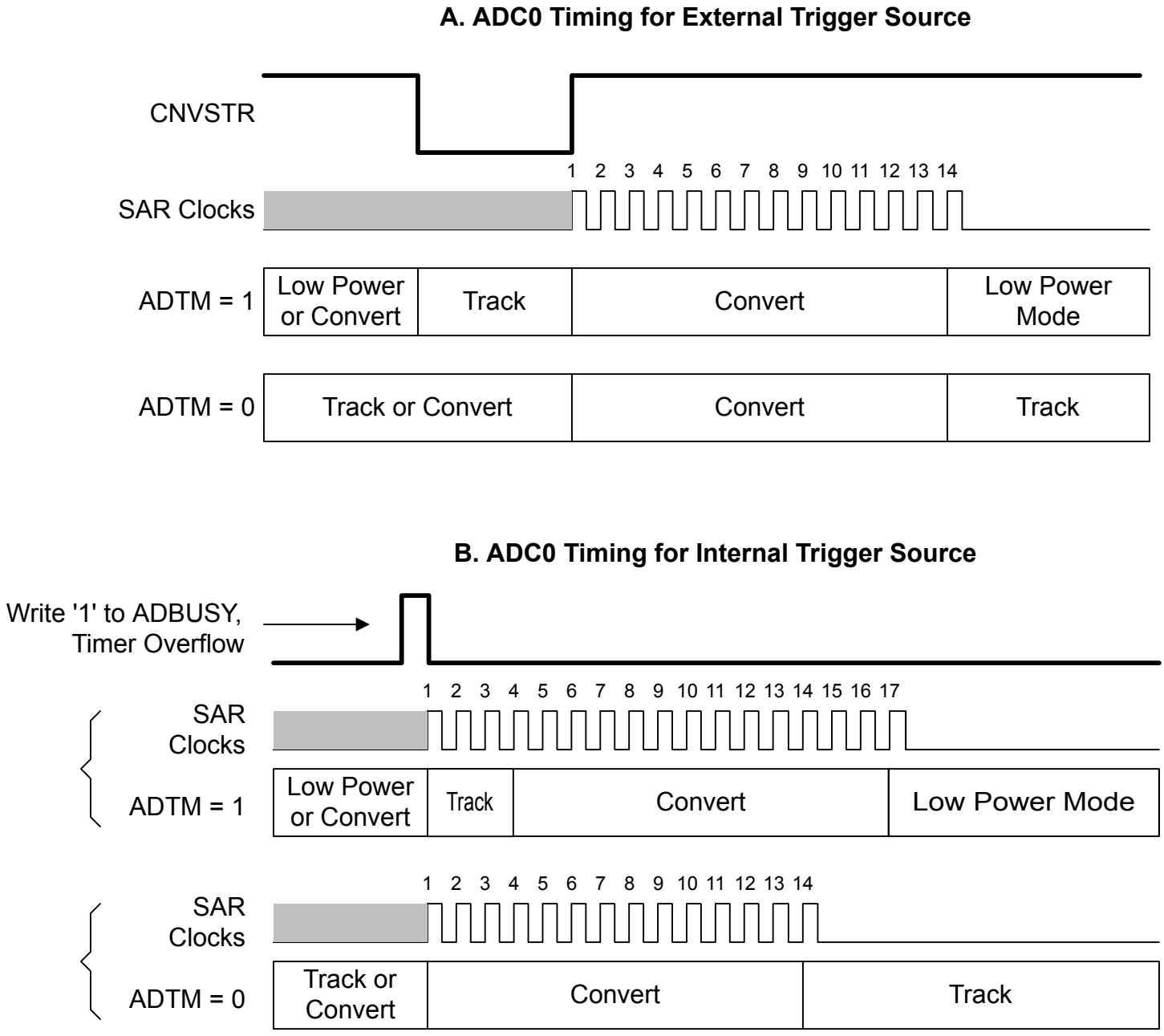


Figure 12.3. Track and Conversion Example Timing (Normal, Non-Burst Operation)

12.3.6 Output Formatting

The conversion code format differs between single-ended and differential modes. 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 ADLJST bit.

When in single-ended mode, conversion codes are represented as 10-bit unsigned integers. Inputs are measured from 0 to VREF x 1023/1024. Unused bits in the ADC0H and ADC0L registers are set to 0.

Table 12.3. Single-Ended Output Code Example

Input Voltage	Right-Justified (ADLJST = 0) ADC0H:L	Left-Justified (ADLJST = 1) ADC0H:L
VREF x 1023/1024	0x03FF	0xFFC0
VREF x 512/1024	0x0200	0x8000
VREF x 256/1024	0x0100	0x4000
0	0x0000	0x0000

When in differential mode, conversion codes are represented as 10-bit signed 2's complement numbers. Inputs are measured from –VREF to VREF x 511/512. For right-justified data, the unused MSBs of ADC0H are a sign-extension of the data word. For left-justified data, the unused LSBs in the ADC0L register are set to 0.

Table 12.4. Differential Output Code Example

Input Voltage	Right-Justified (ADLJST = 0) ADC0H:L	Left-Justified (ADLJST = 1) ADC0H:L
VREF x 511/512	0x01FF	0x7FC0
VREF x 256/512	0x0100	0x4000
0	0x0000	0x0000
–VREF x 256/512	0xFF00	0xC000
–VREF	0xFE00	0x8000

12.3.7 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 12.5. 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 12.6. 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 12.7. 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 12.8. 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	

12.3.8 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 12.4 Temperature Sensor Transfer Function on page 132. 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.

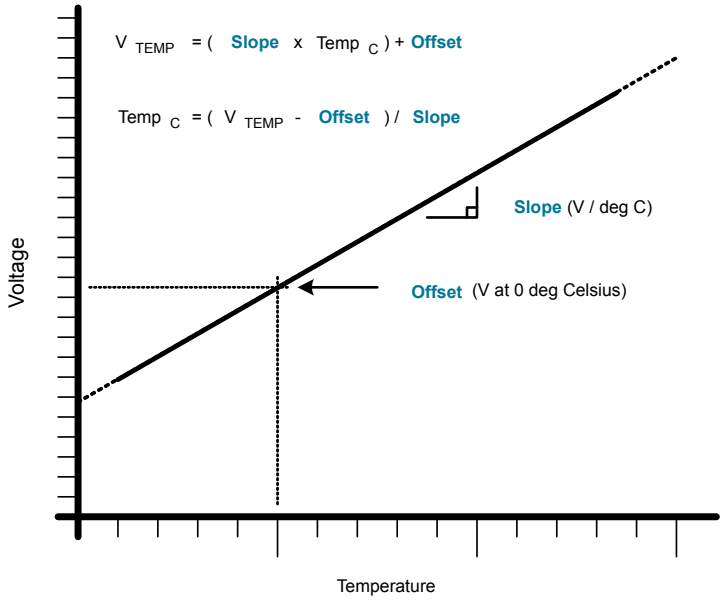


Figure 12.4. Temperature Sensor Transfer Function

12.3.8.1 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 ADC0CF: ADC0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	ADSC					ADLJST	Reserved	
Access	RW					RW	RW	
Reset	0x1F					0	0x0	
SFR Page = ALL; 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 = (Fsysclk) / (ADSC + 1)
2	ADLJST	0	RW	**ADC0 Left Justify Select.**
	Value	Name		Description
	0	RIGHT_JUSTIFIED		Data in the ADC0H:ADC0L registers is right-justified.
	1	LEFT_JUSTIFIED		Data in the ADC0H:ADC0L registers is left-justified.
1:0	*Reserved*	*Must write reset value.*		

12.4.2 ADC0H: ADC0 Data Word High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0H							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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 ADC data holding register (ADC0H:L) formatted according to the settings in ADLJST. Any unused bits for right-justified results will be zeroes.

12.4.3 ADC0L: ADC0 Data Word Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0L							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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 ADC data holding register (ADC0H:L) formatted according to the settings in ADLJST. Any unused bits for left-justified results will be zeroes.

12.4.4 ADC0GTH: ADC0 Greater-Than High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTH							
Access	RW							
Reset	0xFF							
SFR Page = ALL; 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.

12.4.5 ADC0GTL: ADC0 Greater-Than Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTL							
Access	RW							
Reset	0xFF							
SFR Page = ALL; 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.

12.4.6 ADC0LTH: ADC0 Less-Than High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTH							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

12.4.7 ADC0LTL: ADC0 Less-Than Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTL							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

12.4.8 ADC0CN0: ADC0 Control

Bit	7	6	5	4	3	2	1	0
Name	ADEN	ADTM	ADINT	ADBUSY	ADWINT	ADCM		
Access	RW	RW	RW	RW	RW	RW		
Reset	0	0	0	0	0	0x0		

SFR Page = ALL; SFR Address: 0xE8 (bit-addressable)

Bit	Name	Reset	Access	Description
7	ADEN	0	RW	**ADC Enable.**

Value	Name	Description
0	DISABLED	ADC0 Disabled (low-power shutdown).
1	ENABLED	ADC0 Enabled (active and ready for data conversions).

Bit	Name	Reset	Access	Description
6	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 3 SAR clock cycles following the start-of-conversion signal. The ADC is allowed to track during this time. Note that there is not a tracking delay when the external conversion start (CNVSTR) is used as the start-of-conversion source.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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.

Bit	Name	Reset	Access	Description
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	TIMER1	ADC0 conversion initiated on overflow of Timer 1.
0x4	CNVSTR	ADC0 conversion initiated on rising edge of CNVSTR.
0x5	TIMER3	ADC0 conversion initiated on overflow of Timer 3.

Bit	Name	Reset	Access	Description
	0x6	TIMER4		ADC0 conversion initiated on overflow of Timer 4.
	0x7	TIMER5		ADC0 conversion initiated on overflow of Timer 5.

12.4.9 AMX0P: AMUX0 Positive Multiplexer Selection

Bit	7	6	5	4	3	2	1	0
Name	Reserved		AMX0P					
Access	R		RW					
Reset	0x0		0x00					
SFR Page = ALL; SFR Address: 0xBB								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5:0	AMX0P	0x00	RW	**AMUX0 Positive Input Selection.** Selects the positive input channel for ADC0. For reserved bit combinations, no input is selected.

12.4.10 AMX0N: AMUX0 Negative Multiplexer Selection

Bit	7	6	5	4	3	2	1	0
Name	Reserved		AMX0N					
Access	R		RW					
Reset	0x0		0x00					
SFR Page = ALL; SFR Address: 0xBA								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5:0	AMX0N	0x00	RW	**AMUX0 Negative Input Selection.** Selects the negative input channel for ADC0. For reserved bit combinations, no input is selected.

12.4.11 REF0CN: Voltage Reference Control

Bit	7	6	5	4	3	2	1	0
Name	REFBGS	Reserved		REGOVR	REFSL	TEMPE	Reserved	REFBE
Access	RW	R		RW	RW	RW	RW	RW
Reset	0	0x0		0	0	0	0	0
SFR Page = ALL; SFR Address: 0xD1								

Bit	Name	Reset	Access	Description
7	REFBGS	0	RW	**Reference Buffer Gain Select.**
	This bit selects between 1x and 2x gain for the on-chip voltage reference buffer.			
	Value	Name		Description
	0	GAIN_2		The on-chip voltage reference buffer gain is 2.
	1	GAIN_1		The on-chip voltage reference buffer gain is 1.
6:5	*Reserved*	*Must write reset value.*		
4	REGOVR	0	RW	**Regulator Reference Override.**
	This bit overrides the REFSL bit and allows the internal regulator to be used as a reference source.			
	Value	Name		Description
	0	REFSL		The REFSL bit selects the voltage reference source.
	1	VREG		Use the output of the internal regulator as the voltage reference source.
3	REFSL	0	RW	**Voltage Reference Select.**
	This bit selects the ADC voltage reference.			
	Value	Name		Description
	0	VREF		Use the VREF pin as the voltage reference.
	1	VDD		Use VDD as the voltage reference.
2	TEMPE	0	RW	**Temperature Sensor Enable.**
	Enables/Disables the internal temperature sensor.			
	Value	Name		Description
	0	DISABLED		Disable the internal Temperature Sensor.
	1	ENABLED		Enable the internal Temperature Sensor.
1	*Reserved*	*Must write reset value.*		
0	REFBE	0	RW	**Internal Reference Buffer Enable.**
	Value	Name		Description
	0	DISABLED		Disable the internal reference buffer.
	1	ENABLED		Enable the internal reference buffer. The internal voltage reference is driven on the VREF pin.

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.

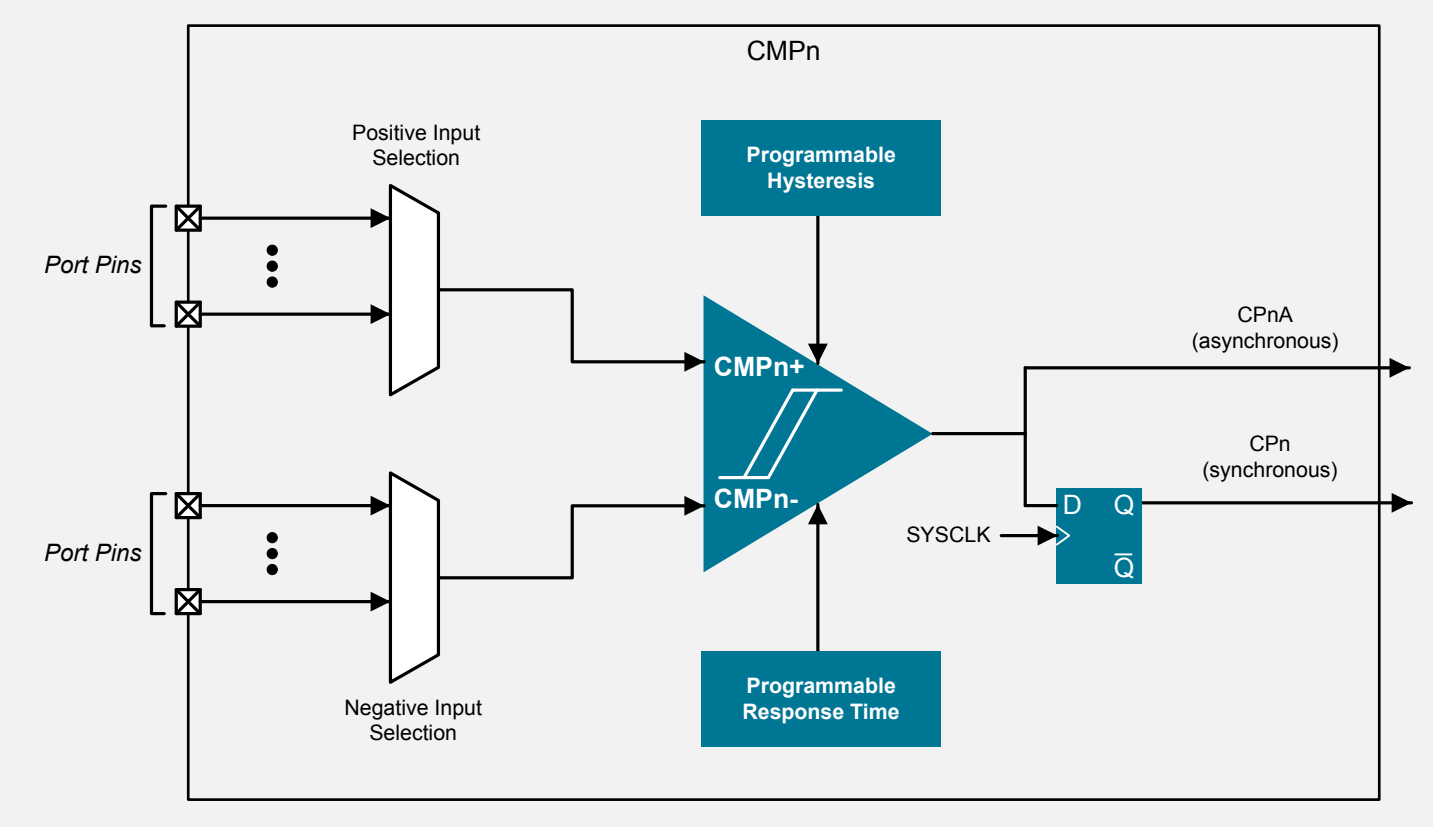


Figure 13.1. Comparator Block Diagram

13.2 Features

The comparator module includes the following features:

- Up to 5 external positive inputs.
- Up to 5 external negative inputs.
- 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.

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.

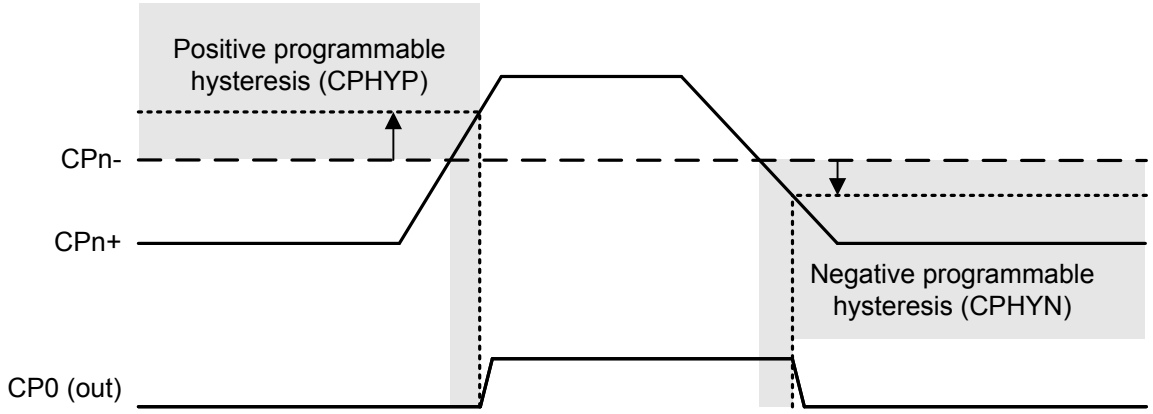


Figure 13.2. Comparator Hysteresis Plot

13.3.3 Input Selection

Comparator inputs may be routed to port I/O pins. 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	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
000	CMP0P.0	P2.0	P1.0	P1.0
001	CMP0P.1	P2.5	P1.4	P1.4
010	CMP0P.2	P3.4	P2.0	P2.0
011	CMP0P.3	P4.3	P2.4	P2.4
100	CMP0P.4	P0.3	P0.0	P0.0
101	CMP0P.5	Reserved	Reserved	Reserved
110	CMP0P.6	Reserved	Reserved	Reserved
111	CMP0P.7	Reserved	Reserved	Reserved

Table 13.2. CMP0 Negative Input Multiplexer Channels

CMXN Setting in Register CMP0MX	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
000	CMP0N.0	P2.1	P1.1	P1.1
001	CMP0N.1	P2.6	P1.5	P1.5
010	CMP0N.2	P3.5	P2.1	P2.1
011	CMP0N.3	P4.4	P2.5	P2.5
100	CMP0N.4	P0.4	P0.1	P0.1
101	CMP0N.5	Reserved	Reserved	Reserved
110	CMP0N.6	Reserved	Reserved	Reserved
111	CMP0N.7	Reserved	Reserved	Reserved

Table 13.3. CMP1 Positive Input Multiplexer Channels

CMXP Setting in Register CMP1MX	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
000	CMP1P.0	P2.2	P1.2	P1.2
001	CMP1P.1	P3.0	P1.6	P1.6
010	CMP1P.2	P3.7	P2.2	P2.2
011	CMP1P.3	P4.5	Reserved	Reserved
100	CMP1P.4	P1.1	P0.4	P0.4
101	CMP1P.5	Reserved	Reserved	Reserved
110	CMP1P.6	Reserved	Reserved	Reserved
111	CMP1P.7	Reserved	Reserved	Reserved

Table 13.4. CMP1 Negative Input Multiplexer Channels

CMXN Setting in Register CMP1MX	Signal Name	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
000	CMP1N.0	P2.3	P1.3	P1.3
001	CMP1N.1	P3.1	P1.7	P1.7
010	CMP1N.2	P4.0	P2.3	P2.3
011	CMP1N.3	P4.6	Reserved	Reserved
100	CMP1N.4	P1.2	P0.5	P0.5
101	CMP1N.5	Reserved	Reserved	Reserved
110	CMP1N.6	Reserved	Reserved	Reserved
111	CMP1N.7	Reserved	Reserved	Reserved

13.3.4 Output Routing

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

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

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

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

13.4 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 = ALL; SFR Address: 0x9B								

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

Bit	Name	Reset	Access	Description
1:0	CPHYN	0x0	RW	**Comparator Negative Hysteresis Control.**
	Value	Name		Description
	0x0	DISABLED		Negative Hysteresis disabled.
	0x1	ENABLED_MODE1		Negative Hysteresis = Hysteresis 1.
	0x2	ENABLED_MODE2		Negative Hysteresis = Hysteresis 2.
	0x3	ENABLED_MODE3		Negative Hysteresis = Hysteresis 3 (Maximum).

13.4.2 CMP0MD: Comparator 0 Mode

Bit	7	6	5	4	3	2	1	0
Name	Reserved		CPRIE	CPFIE	Reserved		CPMD	
Access	R		RW	RW	R		RW	
Reset	0x0		0	0	0x0		0x2	
SFR Page = ALL; SFR Address: 0x9D								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	CPRIE	0	RW	**Comparator Rising-Edge Interrupt Enable.**
	Value	Name		Description
	0	RISE_INT_DISABLED		Comparator rising-edge interrupt disabled.
	1	RISE_INT_ENABLED		Comparator rising-edge interrupt enabled.
4	CPFIE	0	RW	**Comparator Falling-Edge Interrupt Enable.**
	Value	Name		Description
	0	FALL_INT_DISABLED		Comparator falling-edge interrupt disabled.
	1	FALL_INT_ENABLED		Comparator falling-edge interrupt enabled.
3:2	*Reserved*	*Must write reset value.*		
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	Reserved	CMXN			Reserved	CMXP		
Access	RW	RW			RW	RW		
Reset	0	0x0			0	0x0		
SFR Page = ALL; SFR Address: 0x9F								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	CMXN	0x0	RW	**Comparator Negative Input MUX Selection.** This field selects the negative input for the comparator.
3	*Reserved*	*Must write reset value.*		
2:0	CMXP	0x0	RW	**Comparator Positive Input MUX Selection.** This field selects the positive input for the comparator.

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 = ALL; SFR Address: 0x9A								

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

Bit	Name	Reset	Access	Description
1:0	CPHYN	0x0	RW	**Comparator Negative Hysteresis Control.**
	Value	Name		Description
	0x0	DISABLED		Negative Hysteresis disabled.
	0x1	ENABLED_MODE1		Negative Hysteresis = Hysteresis 1.
	0x2	ENABLED_MODE2		Negative Hysteresis = Hysteresis 2.
	0x3	ENABLED_MODE3		Negative Hysteresis = Hysteresis 3 (Maximum).

13.5.2 CMP1MD: Comparator 1 Mode

Bit	7	6	5	4	3	2	1	0
Name	Reserved		CPRIE	CPFIE	Reserved		CPMD	
Access	R		RW	RW	R		RW	
Reset	0x0		0	0	0x0		0x2	
SFR Page = ALL; SFR Address: 0x9C								

Bit	Name	Reset	Access	Description
7:6	*Reserved*	*Must write reset value.*		
5	CPRIE	0	RW	**Comparator Rising-Edge Interrupt Enable.**
	Value	Name		Description
	0	RISE_INT_DISABLED		Comparator rising-edge interrupt disabled.
	1	RISE_INT_ENABLED		Comparator rising-edge interrupt enabled.
4	CPFIE	0	RW	**Comparator Falling-Edge Interrupt Enable.**
	Value	Name		Description
	0	FALL_INT_DISABLED		Comparator falling-edge interrupt disabled.
	1	FALL_INT_ENABLED		Comparator falling-edge interrupt enabled.
3:2	*Reserved*	*Must write reset value.*		
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	Reserved	CMXN			Reserved	CMXP		
Access	RW	RW			RW	RW		
Reset	0	0x0			0	0x0		
SFR Page = ALL; SFR Address: 0x9E								

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6:4	CMXN	0x0	RW	**Comparator Negative Input MUX Selection.** This field selects the negative input for the comparator.
3	*Reserved*	*Must write reset value.*		
2:0	CMXP	0x0	RW	**Comparator Positive Input MUX Selection.** This field selects the positive input for the comparator.

14. Programmable Counter Array (PCA0)

14.1 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.

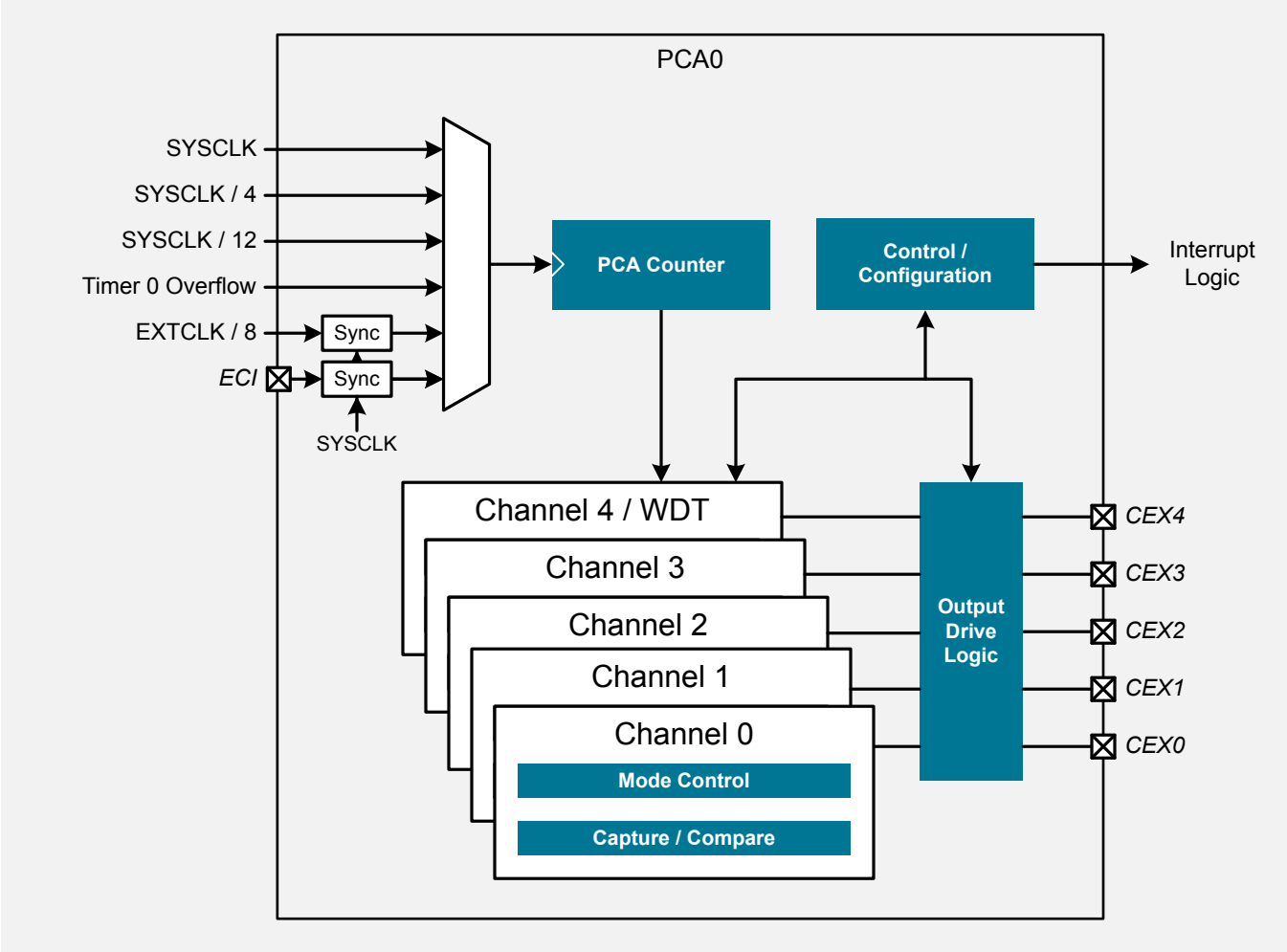


Figure 14.1. PCA Block Diagram

14.2 Features

- 16-bit time base.
- Programmable clock divisor and clock source selection.
- Up to five independently-configurable channels
- 8- or 16-bit PWM modes (edge-aligned operation).
- Frequency output mode.
- Capture on rising, falling or any edge.
- Compare function for arbitrary waveform generation.
- Software timer (internal compare) mode.
- Integrated watchdog timer.

14.3 Functional Description

14.3.1 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 14.1. 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)
100	System clock
101	External oscillator source divided by 8 [1]

Note:
1. Synchronized with the system clock.

14.3.2 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.

14.3.3 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-bit pulse width modulator, or 16-bit pulse width modulator. Table 14.2 PCA0CPM Bit Settings for PCA Capture/Compare Modules on page 150 summarizes the bit settings in the PCA0CPMn register used to select the PCA capture/compare module’s operating mode. Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt.

Table 14.2. PCA0CPM Bit Settings for PCA Capture/Compare Modules

Operational Mode	PCA0CPMn							
Bit Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Bit Number	7	6	5	4	3	2	1	0
Capture triggered by positive edge on CEXn	X	X	1	0	0	0	0	A

Operational Mode	PCA0CPMn							
Bit Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Bit Number	7	6	5	4	3	2	1	0
Capture triggered by negative edge on CEXn	X	X	0	1	0	0	0	A
Capture triggered by any transition on CEXn	X	X	1	1	0	0	0	A
Software Timer	X	B	0	0	1	0	0	A
High Speed Output	X	B	0	0	1	1	0	A
Frequency Output	X	B	0	0	0	1	1	A
8-Bit Pulse Width Modulator	0	B	0	0	C	0	1	A
16-Bit Pulse Width Modulator	1	B	0	0	C	0	1	A

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 = 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).
4. C = When set, a match event will cause the CCFn flag for the associated channel to be set.

14.3.4 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.

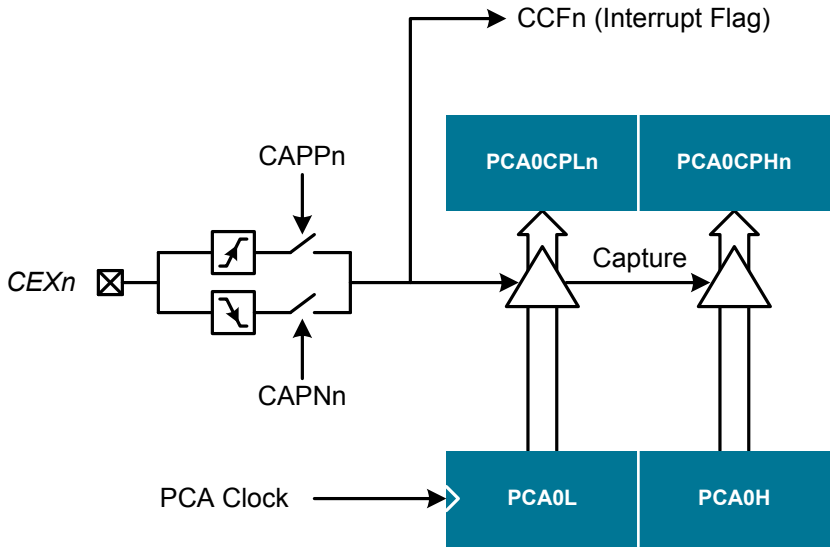


Figure 14.2. PCA Capture Mode Diagram

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

14.3.5 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.

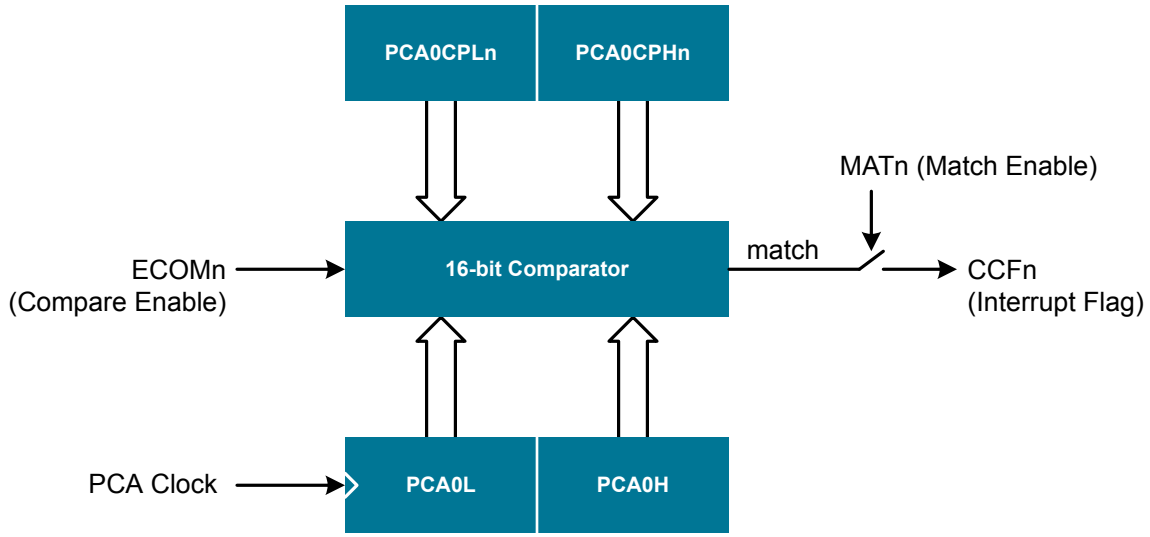


Figure 14.3. PCA Software Timer Mode Diagram

14.3.6 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.

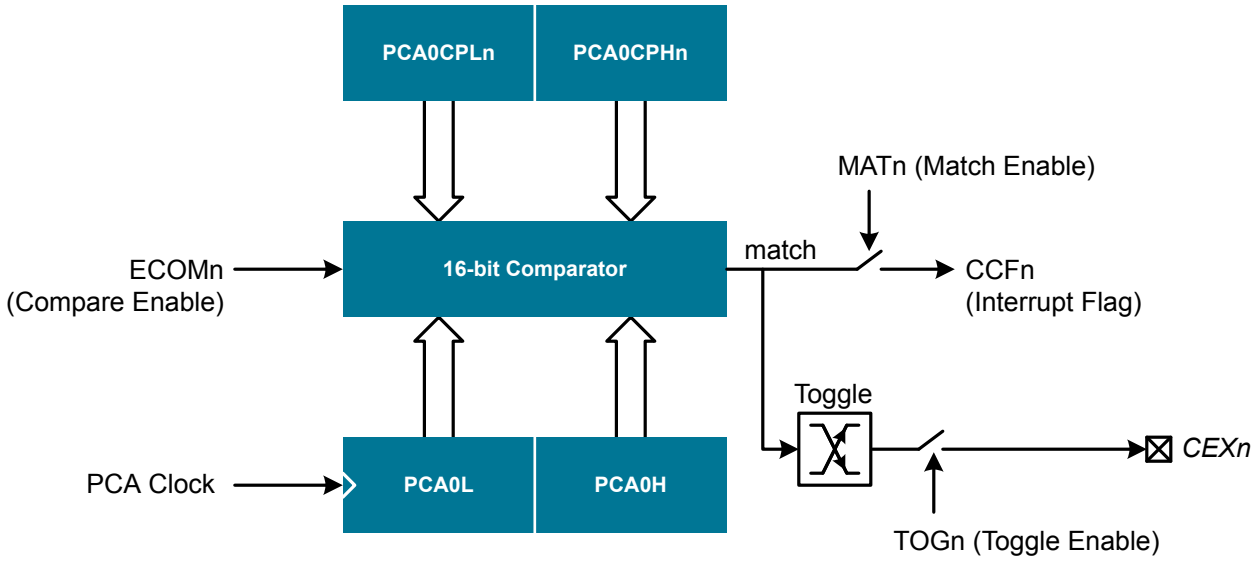


Figure 14.4. PCA High-Speed Output Mode Diagram

14.3.7 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.

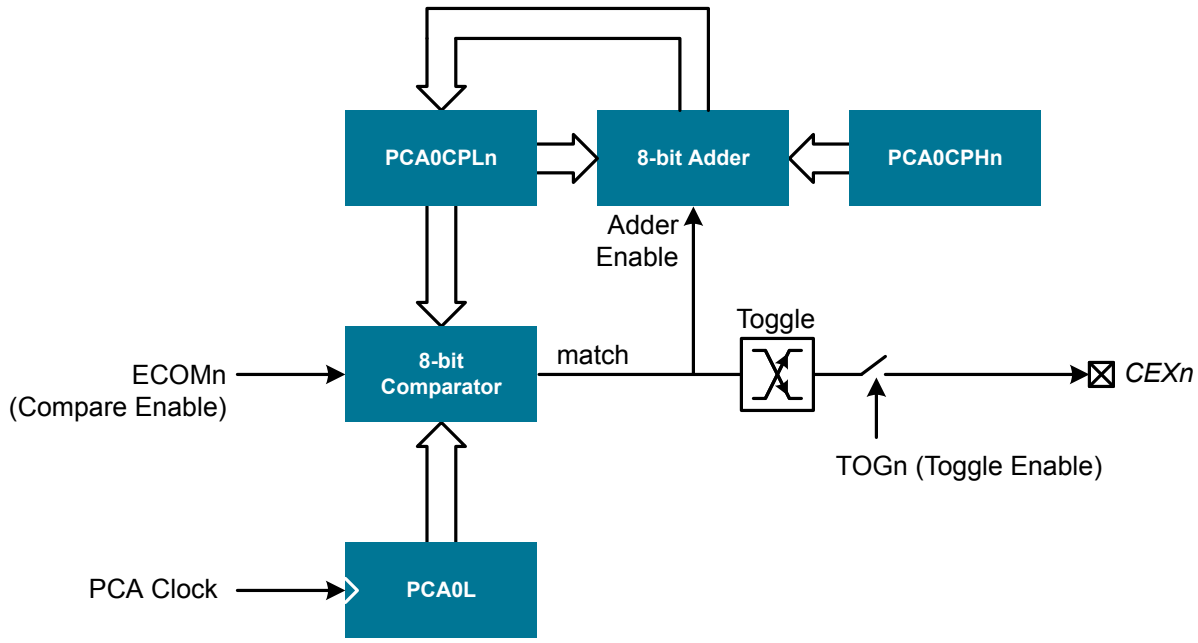


Figure 14.5. PCA Frequency Output Mode

14.3.8 PWM Waveform Generation

The PCA can generate edge-aligned PWM waveforms with resolutions of 8 or 16 bits. PWM resolution depends on the module setup, as specified within the individual module PCA0CPMn registers. Modules can be configured for 8-bit mode or for 16-bit mode individually using the PCA0CPMn registers.

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 8-bit PWM, the match edge occurs any time bits 7-0 of the PCA0CPn register match bits 7-0 of the PCA0 counter value.

The overflow edge occurs when an overflow of the PCA0 counter happens at the desired resolution. For example, with 8-bit PWM, the overflow edge occurs when bits 7-0 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.

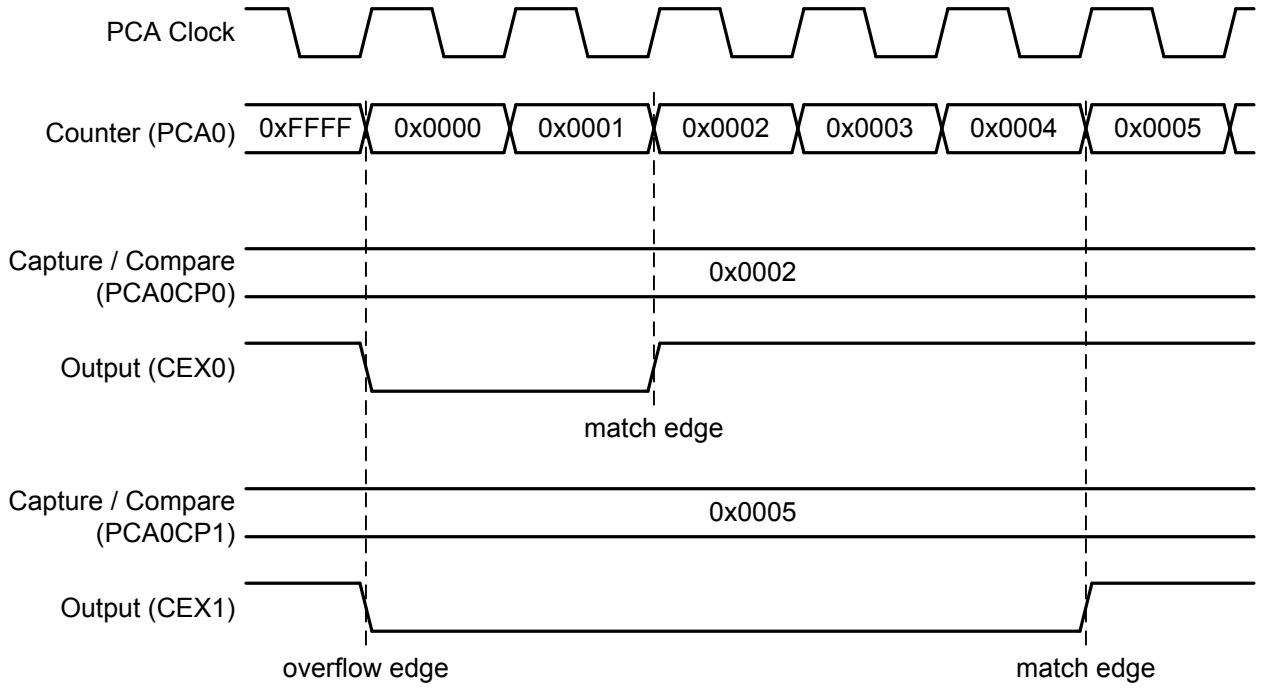


Figure 14.6. Edge-Aligned PWM Timing

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. A 0% duty cycle for the channel 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-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 14.7. N-bit Edge-Aligned PWM Duty Cycle (N = PWM resolution)

14.3.8.1 8-Bit PWM 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 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.

14.3.8.2 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.

14.3.9 Watchdog Timer Mode

A programmable watchdog timer (WDT) function is available through the last PCA module (module 4). The WDT is used to generate a reset if the time between writes to the WDT update register (PCA0CPH4) exceed a specified limit. The WDT can be configured and enabled/disabled as needed by software. With the WDTE bit set in the PCA0MD register, the last module operates as a watchdog timer (WDT). The module 4 high byte is compared to the PCA counter high byte; the module 4 low byte holds the offset to be used when WDT updates are performed. The Watchdog Timer is enabled on reset. Writes to some PCA registers are restricted while the Watchdog Timer is enabled. The WDT will generate a reset shortly after code begins execution. To avoid this reset, the WDT should be explicitly disabled (and optionally re-configured and re-enabled if it is used in the system).

Watchdog Timer Operation

While the WDT is enabled:

- PCA counter is forced on.
- Writes to PCA0L and PCA0H are not allowed.
- PCA clock source CPS field is frozen.
- PCA Idle control bit (CIDL) is frozen.
- Module 4 is forced into software timer mode.
- Writes to the Module 4 mode register (PCA0CPM4) are disabled.

While the WDT is enabled, writes to the CR bit will not change the PCA counter state; the counter will run until the WDT is disabled. The PCA counter run control bit (CR) will read zero if the WDT is enabled but user software has not enabled the PCA counter. If a match occurs between PCA0CPH4 and PCA0H while the WDT is enabled, a reset will be generated. To prevent a WDT reset, the WDT may be updated with a write of any value to PCA0CPH4. Upon a PCA0CPH4 write, PCA0H plus the offset held in PCA0CPL4 is loaded into PCA0CPH4.

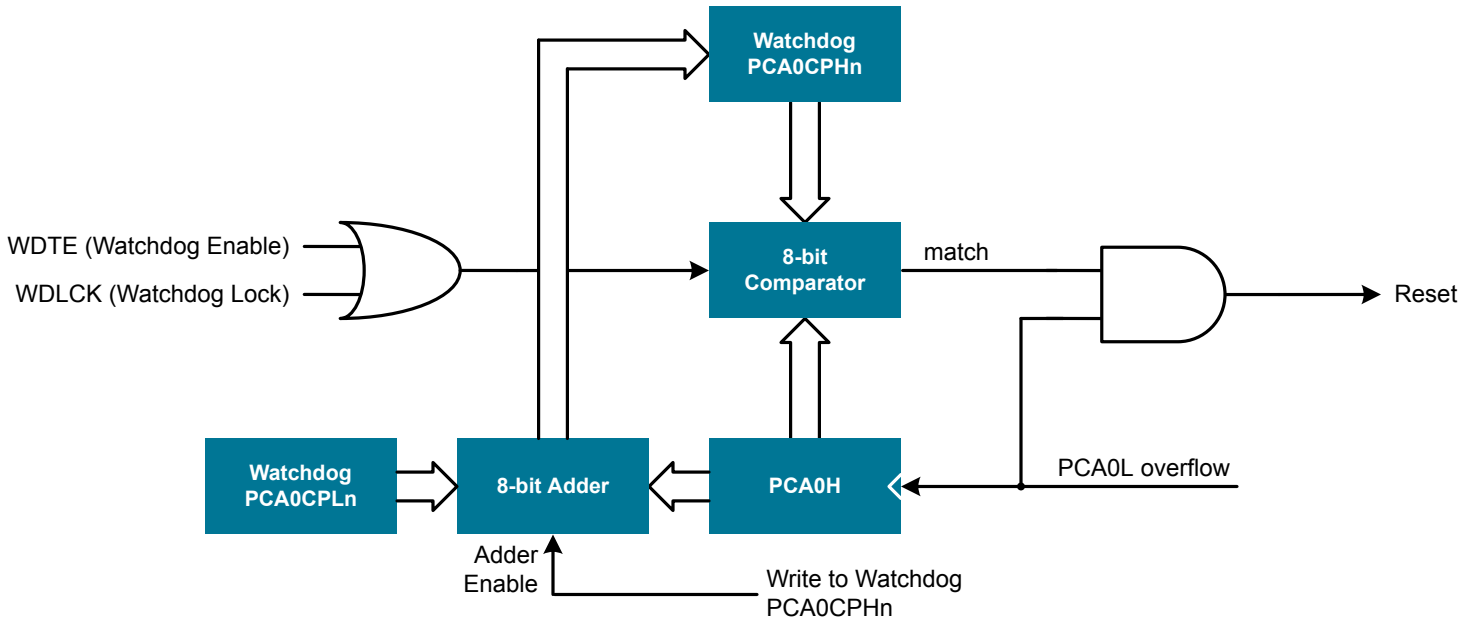


Figure 14.8. PCA Module 4 with Watchdog Timer Enabled

The 8-bit offset held in PCA0CPH4 is compared to the upper byte of the 16-bit PCA counter. This offset value is the number of PCA0L overflows before a reset. Up to 256 PCA clocks may pass before the first PCA0L overflow occurs, depending on the value of the PCA0L when the update is performed. The total offset is then given by the following equation in PCA clocks:

$$\text{Offset} = (256 \times PCA0CPL) + (256 - PCA0L)$$

Note: PCA0L is the value of the PCA0L register at the time of the update in this equation.

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH4 and PCA0H. Software may force a WDT reset by writing a 1 to the CCF4 flag in the PCA0CN0 register while the WDT is enabled.

Watchdog Timer Usage

To configure the WDT, perform the following tasks:

1. Disable the WDT by writing a 0 to the WDTE bit.
2. Select the desired PCA clock source (with the CPS field).
3. Load the WDT PCA0CPL with the desired WDT update offset value.
4. Configure the PCA Idle mode (set CIDL if the WDT should be suspended while the CPU is in Idle mode).
5. Enable the WDT by setting the WDTE bit to 1.
6. Reset the WDT timer by writing to PCA0CPH4.

The PCA clock source and Idle mode select cannot be changed while the WDT is enabled. The watchdog timer is enabled by setting the WDTE or WDLCK bits in the PCA0MD register. When WDLCK is set, the WDT cannot be disabled until the next system reset. If WDLCK is not set, the WDT is disabled by clearing the WDTE bit. The WDT is enabled following any reset. The PCA0 counter clock defaults to the system clock divided by 12, PCA0L defaults to 0x00, and PCA0CPL2 defaults to 0x00. This results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. lists some example timeout intervals for typical system clocks.

Table 14.3. Watchdog Timer Timeout Intervals

System Clock (Hz)	PCA0CPL4	Timeout Interval (ms)
24,500,000	255	32.1
24,500,000	128	16.2
24,500,000	32	4.1
3,062,500	255	257
3,062,500	128	129.5
3,062,500	32	33.1
32,000	255	24576
32,000	128	12384
32,000	32	3168
Note: The values in this table assume SYSCLK/12 as the PCA clock source and a PCA0L value of 0x00 at the update time.		

14.4 PCA0 Control Registers

14.4.1 PCA0CN0: PCA Control 0

Bit	7	6	5	4	3	2	1	0
Name	CF	CR	Reserved	CCF4	CCF3	CCF2	CCF1	CCF0
Access	RW	RW	R	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; 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	*Reserved*	*Must write reset value.*		
4	CCF4	0	RW	**PCA Module 4 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF4 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
3	CCF3	0	RW	**PCA Module 3 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF3 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
2	CCF2	0	RW	**PCA Module 2 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF2 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
1	CCF1	0	RW	**PCA Module 1 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF1 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.
0	CCF0	0	RW	**PCA Module 0 Capture/Compare Flag.** This bit is set by hardware when a match or capture occurs. When the CCF0 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by firmware.

14.4.2 PCA0MD: PCA Mode

Bit	7	6	5	4	3	2	1	0
Name	CIDL	WDTE	WDLCK	Reserved	CPS			ECF
Access	RW	RW	RW	R	RW			RW
Reset	0	1	0	0	0x0			0
SFR Page = ALL; 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	WDTE	1	RW	**Watchdog Timer Enable.**
	If this bit is set, PCA Module 4 is used as the watchdog timer.			
	Value	Name		Description
	0	DISABLED		Disable Watchdog Timer.
	1	ENABLED		Enable PCA Module 4 as the Watchdog Timer.
5	WDLCK	0	RW	**Watchdog Timer Lock.**
	This bit locks/unlocks the Watchdog Timer Enable. When WDLCK is set, the Watchdog Timer may not be disabled until the next system reset.			
	Value	Name		Description
	0	UNLOCKED		Watchdog Timer Enable unlocked.
	1	LOCKED		Watchdog Timer Enable locked.
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).

Bit	Name	Reset	Access	Description
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.

When the WDTE bit is set to 1, the other bits in the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled.

14.4.3 PCA0L: PCA Counter/Timer Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0L							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

When the WDTE bit is set to 1, the PCA0L register cannot be modified by firmware. To change the contents of the PCA0L register, the Watchdog Timer must first be disabled.

14.4.4 PCA0H: PCA Counter/Timer High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0H							
Access	RW							
Reset	0x00							
SFR Page = ALL; 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.

When the WDTE bit is set to 1, the PCA0H register cannot be modified by firmware. To change the contents of the PCA0H register, the Watchdog Timer must first be disabled.

14.4.5 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 = ALL; SFR Address: 0xDA								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 0 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 0 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 0 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 0 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 0 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF0 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 0 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX0 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 0 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX0 pin. 8-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 0 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF0) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF0 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF0 is set.

14.4.6 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 = ALL; 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.
A write to this register will clear the module's ECOM bit to a 0.				

14.4.7 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 = ALL; 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.
A write to this register will set the module's ECOM bit to a 1.				

14.4.8 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 = ALL; SFR Address: 0xDB								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 1 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 1 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 1 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 1 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 1 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF1 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 1 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX1 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 1 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX1 pin. 8-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 1 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF1) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF1 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF1 is set.

14.4.9 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 = ALL; 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.
A write to this register will clear the module's ECOM bit to a 0.				

14.4.10 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 = ALL; 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.
A write to this register will set the module's ECOM bit to a 1.				

14.4.11 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 = ALL; SFR Address: 0xDC								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 2 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 2 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 2 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 2 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 2 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF2 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 2 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX2 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 2 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX2 pin. 8-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 2 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF2) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF2 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF2 is set.

14.4.12 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 = ALL; 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.
A write to this register will clear the module's ECOM bit to a 0.				

14.4.13 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 = ALL; 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.
A write to this register will set the module's ECOM bit to a 1.				

14.4.14 PCA0CPM3: PCA Channel 3 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xDD								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 3 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 3 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 3 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 3 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 3 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF3 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 3 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX3 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 3 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX3 pin. 8-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 3 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF3) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF3 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF3 is set.

14.4.15 PCA0CPL3: PCA Channel 3 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL3							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xED								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL3	0x00	RW	**PCA Channel 3 Capture Module Low Byte.** The PCA0CPL3 register holds the low byte (LSB) of the 16-bit capture module.
A write to this register will clear the module's ECOM bit to a 0.				

14.4.16 PCA0CPH3: PCA Channel 3 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH3							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xEE								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH3	0x00	RW	**PCA Channel 3 Capture Module High Byte.** The PCA0CPH3 register holds the high byte (MSB) of the 16-bit capture module.
A write to this register will set the module's ECOM bit to a 1.				

14.4.17 PCA0CPM4: PCA Channel 4 Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xDE								

Bit	Name	Reset	Access	Description
7	PWM16	0	RW	**Channel 4 16-bit Pulse Width Modulation Enable.**
	This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.			
	Value	Name		Description
	0	8_BIT		8-bit PWM selected.
	1	16_BIT		16-bit PWM selected.
6	ECOM	0	RW	**Channel 4 Comparator Function Enable.**
	This bit enables the comparator function.			
5	CAPP	0	RW	**Channel 4 Capture Positive Function Enable.**
	This bit enables the positive edge capture capability.			
4	CAPN	0	RW	**Channel 4 Capture Negative Function Enable.**
	This bit enables the negative edge capture capability.			
3	MAT	0	RW	**Channel 4 Match Function Enable.**
	This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF4 bit in the PCA0MD register to be set to logic 1.			
2	TOG	0	RW	**Channel 4 Toggle Function Enable.**
	This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX4 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode.			
1	PWM	0	RW	**Channel 4 Pulse Width Modulation Mode Enable.**
	This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX4 pin. 8-bit PWM is used if PWM16 is cleared to 0; 16-bit mode is used if PWM16 is set to 1. If the TOG bit is also set, the module operates in Frequency Output Mode.			
0	ECCF	0	RW	**Channel 4 Capture/Compare Flag Interrupt Enable.**
	This bit sets the masking of the Capture/Compare Flag (CCF4) interrupt.			
	Value	Name		Description
	0	DISABLED		Disable CCF4 interrupts.
	1	ENABLED		Enable a Capture/Compare Flag interrupt request when CCF4 is set.

14.4.18 PCA0CPL4: PCA Channel 4 Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPL4							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xFD								

Bit	Name	Reset	Access	Description
7:0	PCA0CPL4	0x00	RW	**PCA Channel 4 Capture Module Low Byte.** The PCA0CPL4 register holds the low byte (LSB) of the 16-bit capture module.
A write to this register will clear the module's ECOM bit to a 0.				

14.4.19 PCA0CPH4: PCA Channel 4 Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPH4							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0xFE								

Bit	Name	Reset	Access	Description
7:0	PCA0CPH4	0x00	RW	**PCA Channel 4 Capture Module High Byte.** The PCA0CPH4 register holds the high byte (MSB) of the 16-bit capture module.
A write to this register will set the module's ECOM bit to a 1.				

15. External Memory Interface (EMIF0)

15.1 Introduction

The External Memory Interface (EMIF) enables access of off-chip memories and memory-mapped devices connected to the GPIO ports. The external memory space may be accessed using the external move instruction (MOVX) with the target address specified in either 8-bit or 16-bit formats.

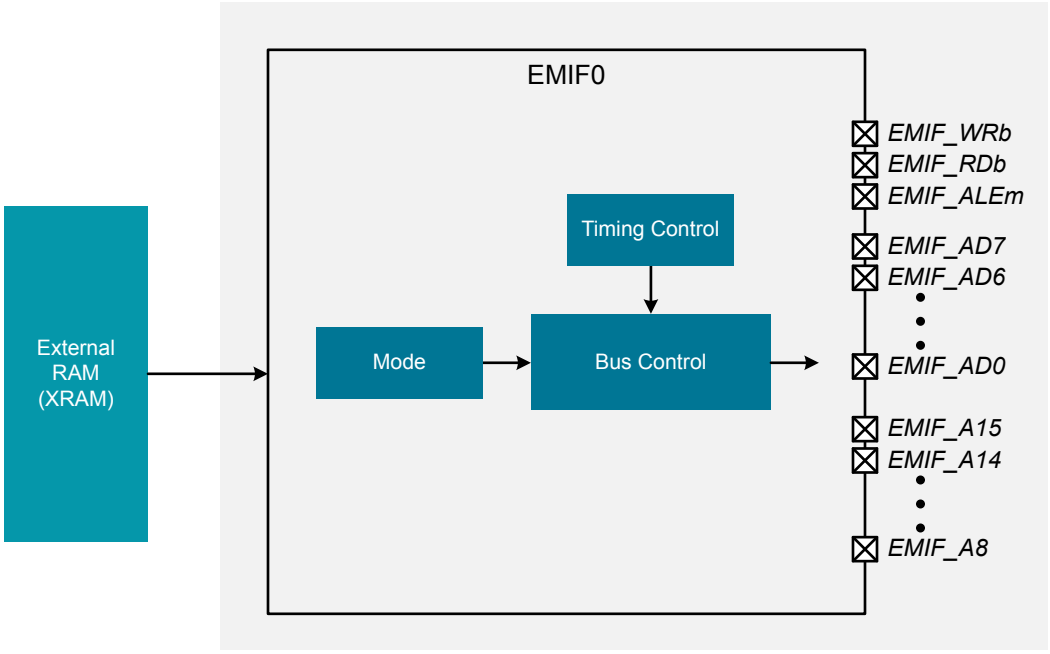


Figure 15.1. EMIF Block Diagram

15.2 Features

- Supports multiplexed and non-multiplexed memory access.
- Four external memory modes:
  - Internal only.
  - Split mode without bank select.
  - Split mode with bank select.
  - External only
- Configurable ALE (address latch enable) timing.
- Configurable address setup and hold times.
- Configurable write and read pulse widths.

15.3 Functional Description

15.3.1 Overview

The devices include RAM mapped into the external data memory space (XRAM). Devices with enough pins also have an External Memory Interface (EMIF0) which can be used to access off-chip memories and memory-mapped devices connected to the GPIO ports. The external memory space may be accessed using the external move instruction (MOVX) with the target address specified in either the data pointer (DPTR), or with the target address low byte in R0 or R1 and the target address high byte in the External Memory Interface Control Register (EMI0CN).

When using the MOVX instruction to access on-chip RAM, no additional initialization is required, and the MOVX instruction execution time is as specified in the core chapter. When using the MOVX instruction to access off-chip RAM or memory-mapped devices, both the Port I/O and EMIF should be configured for communication with external devices, and MOVX instruction timing is based on the value programmed in the Timing Control Register (EMI0TC).

Configuring the External Memory Interface for off-chip memory space access consists of four steps:

1. Configure the output modes of the associated port pins as either push-pull or open-drain (push-pull is most common) and skip the associated pins in the Crossbar (if necessary).
2. Configure port latches to “park” the EMIF pins in a dormant state (usually by setting them to logic 1).
3. Select the memory mode (on-chip only, split mode without bank select, split mode with bank select, or off-chip only).
4. Set up timing to interface with off-chip memory or peripherals.

15.3.2 Port I/O Configuration

When the External Memory Interface is used for off-chip access, the associated port pins are shared between the EMIF and the GPIO port latches. The Crossbar should be configured not to assign any signals to the associated port pins. In most configurations, the RDb, WRb, and ALEm pins need to be skipped in the Crossbar to ensure they are controlled by their port latches.

The External Memory Interface claims the associated port pins for memory operations only during the execution of an off-chip MOVX instruction. Once the MOVX instruction has completed, control of the Port pins reverts to the Port latches. The Port latches should be explicitly configured to “park” the External Memory Interface pins in a dormant state, most commonly by setting them to a logic 1.

During the execution of the MOVX instruction, the External Memory Interface will explicitly disable the drivers on all port pins that are acting as inputs (Data[7:0] during a Read operation, for example). For port pins acting as outputs (Data[7:0] during a Write operation, for example), the External Memory Interface will not automatically enable the output driver. The output mode (whether the pin is configured as open-drain or push-pull) of bi-directional and output only pins should be configured to the desired mode when the pin is being used as an output.

The output mode of the port pins while controlled by the GPIO latch is unaffected by the External Memory Interface operation and remains controlled by the PnMDOUT registers. In most cases, the output modes of all EMIF pins should be configured for push-pull mode.

15.3.2.1 EMIF Pin Mapping

Table 15.1. Multiplexed EMIF Pin Mapping

Multiplexed EMIF Signal Name	Description	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
WRb	Write Enable	P1.7	Not Available	Not Available
RDb	Read Enable	P1.6	Not Available	Not Available
ALEm	Address Latch Enable	P1.3	Not Available	Not Available
AD0m	Address/Data Bit 0	P4.0	Not Available	Not Available
AD1m	Address/Data Bit 1	P4.1	Not Available	Not Available
AD2m	Address/Data Bit 2	P4.2	Not Available	Not Available
AD3m	Address/Data Bit 3	P4.3	Not Available	Not Available
AD4m	Address/Data Bit 4	P4.4	Not Available	Not Available

Multiplexed EMIF Signal Name	Description	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
AD5m	Address/Data Bit 5	P4.5	Not Available	Not Available
AD6m	Address/Data Bit 6	P4.6	Not Available	Not Available
AD7m	Address/Data Bit 7	P4.7	Not Available	Not Available
A8m	Address Bit 8	P3.0	Not Available	Not Available
A9m	Address Bit 9	P3.1	Not Available	Not Available
A10m	Address Bit 10	P3.2	Not Available	Not Available
A11m	Address Bit 11	P3.3	Not Available	Not Available
A12m	Address Bit 12	P3.4	Not Available	Not Available
A13m	Address Bit 13	P3.5	Not Available	Not Available
A14m	Address Bit 14	P3.6	Not Available	Not Available
A15m	Address Bit 15	P3.7	Not Available	Not Available

Table 15.2. Non-Multiplexed EMIF Pin Mapping

Non-Multiplexed EMIF Signal Name	Description	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
WRb	Write Enable	P1.7	Not Available	Not Available
RDb	Read Enable	P1.6	Not Available	Not Available
D0	Data Bit 0	P4.0	Not Available	Not Available
D1	Data Bit 1	P4.1	Not Available	Not Available
D2	Data Bit 2	P4.2	Not Available	Not Available
D3	Data Bit 3	P4.3	Not Available	Not Available
D4	Data Bit 4	P4.4	Not Available	Not Available
D5	Data Bit 5	P4.5	Not Available	Not Available
D6	Data Bit 6	P4.6	Not Available	Not Available
D7	Data Bit 7	P4.7	Not Available	Not Available
A0	Address Bit 0	P3.0	Not Available	Not Available
A1	Address Bit 1	P3.1	Not Available	Not Available
A2	Address Bit 2	P3.2	Not Available	Not Available
A3	Address Bit 3	P3.3	Not Available	Not Available
A4	Address Bit 4	P3.4	Not Available	Not Available
A5	Address Bit 5	P3.5	Not Available	Not Available
A6	Address Bit 6	P3.6	Not Available	Not Available
A7	Address Bit 7	P3.7	Not Available	Not Available
A8	Address Bit 8	P2.0	Not Available	Not Available
A9	Address Bit 9	P2.1	Not Available	Not Available
A10	Address Bit 10	P2.2	Not Available	Not Available
A11	Address Bit 11	P2.3	Not Available	Not Available

Non-Multiplexed EMIF Signal Name	Description	QFP48 Pin Name	QFP32 Pin Name	QFN32 Pin Name
A12	Address Bit 12	P2.4	Not Available	Not Available
A13	Address Bit 13	P2.5	Not Available	Not Available
A14	Address Bit 14	P2.6	Not Available	Not Available
A15	Address Bit 15	P2.7	Not Available	Not Available

15.3.3 Multiplexed External Memory Interface

For a Multiplexed external memory interface, the Data Bus and the lower 8-bits of the Address Bus share the same Port pins: AD[7:0]m. For most devices with an 8-bit interface, the upper address bits are not used and can be used as GPIO if the external memory interface is used in 8-bit non-banked mode. If the external memory interface is used in 8-bit banked mode or 16-bit mode, then the address pins will be driven with the upper address bits and cannot be used as GPIO.

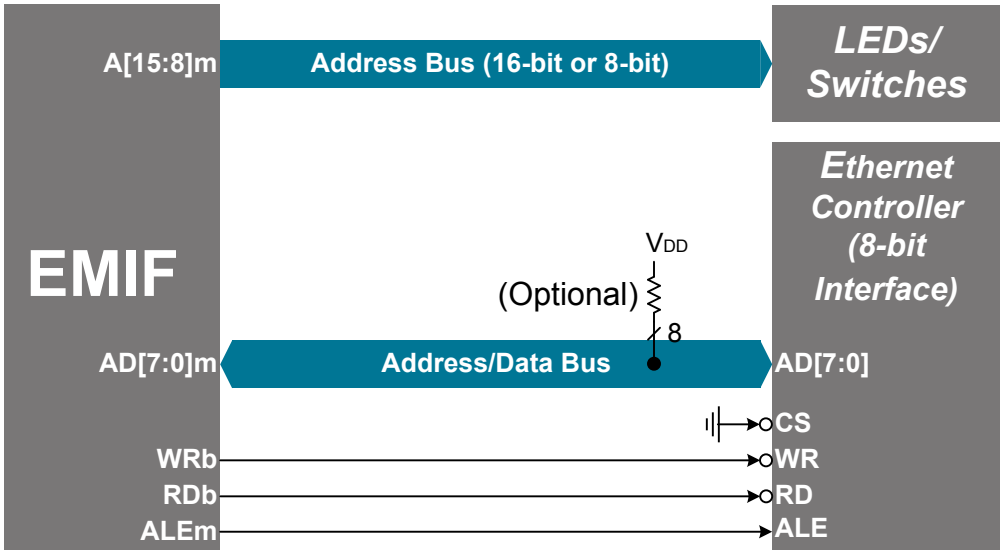


Figure 15.2. Multiplexed Configuration Example

Many devices with a slave parallel memory interface, such as SRAM chips, only support a non-multiplexed memory bus. When interfacing to such a device, an external latch (74HC373 or equivalent logic gate) can be used to hold the lower 8-bits of the RAM address during the second half of the memory cycle when the address/data bus contains data. The external latch, controlled by the ALEm (Address Latch Enable) signal, is automatically driven by the External Memory Interface logic. An example SRAM interface showing multiplexed to non-multiplexed conversion is shown in below.

This example is showing that the external MOVX operation can be broken into two phases delineated by the state of the ALEm signal. During the first phase, ALEm is high and the lower 8-bits of the Address Bus are presented to AD[7:0]m. During this phase, the address latch is configured such that the Q outputs reflect the states of the D inputs. When ALEm falls, signaling the beginning of the second phase, the address latch outputs remain fixed and are no longer dependent on the latch inputs. Later in the second phase, the Data Bus controls the state of the AD[7:0]m port at the time RDb or WRb is asserted.

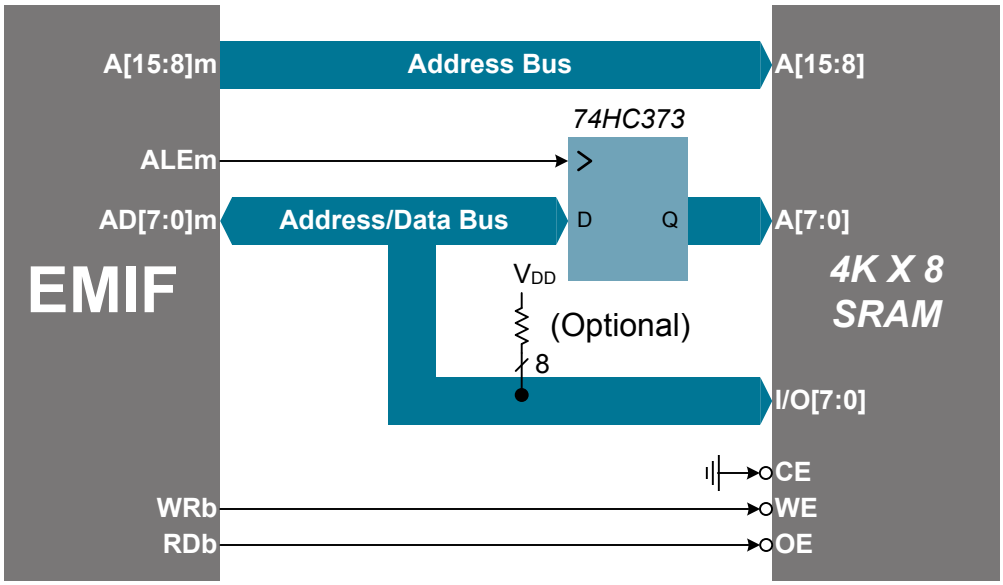


Figure 15.3. Multiplexed to Non-Multiplexed Configuration Example

15.3.4 Non-Multiplexed External Memory Interface

In Non-multiplexed mode, the Data Bus and the Address Bus pins are not shared. An example of a Non-multiplexed Configuration is shown in Figure 15.4 Non-Multiplexed Configuration Example on page 177.

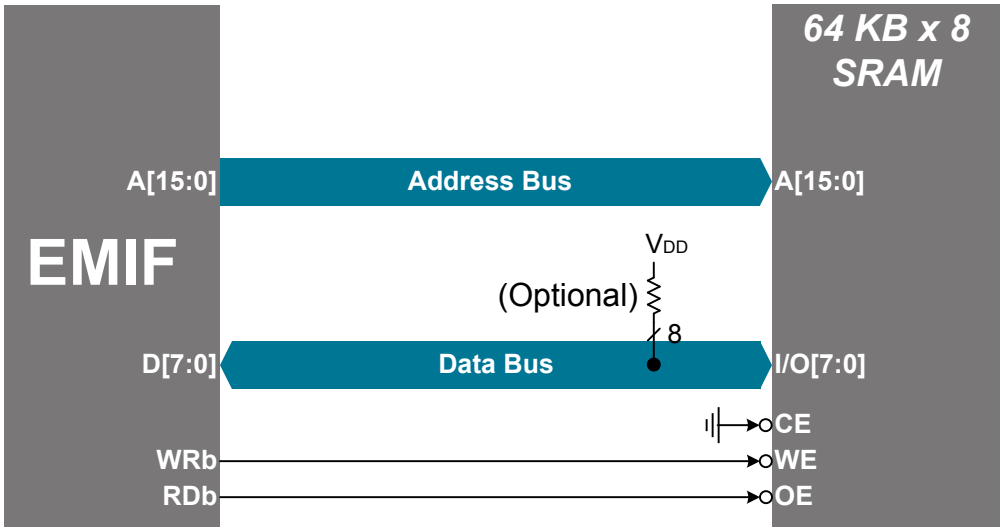


Figure 15.4. Non-Multiplexed Configuration Example

15.3.5 Operating Modes

The external data memory space can be configured in one of four operating modes based on the EMIF Mode bits in the EMI0CF register. These modes are as follows:

- Internal Only
- Split Mode without Bank Select
- Split Mode with Bank Select
- External Only

Timing diagrams for the different modes can be found in the Multiplexed Mode Section.

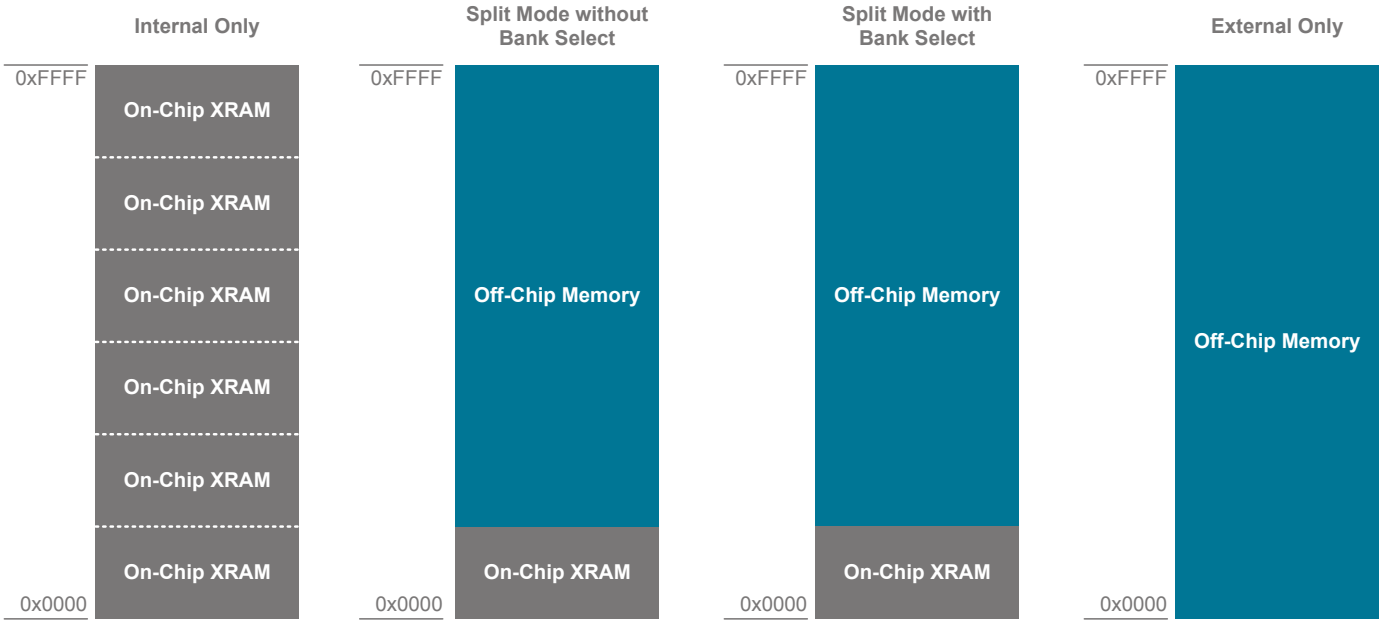


Figure 15.5. EMIF Operating Modes

Internal Only

In Internal Only mode, all MOVX instructions will target the internal XRAM space on the device. Memory accesses to addresses beyond the populated space will wrap and will always target on-chip XRAM. As an example, if the entire address space is consecutively written and the data pointer is incremented after each write, the write pointer will always point to the first byte of on-chip XRAM after the last byte of on-chip XRAM has been written.

- 8-bit MOVX operations use the contents of EMI0CN to determine the high-byte of the effective address and R0 or R1 to determine the low-byte of the effective address.
- 16-bit MOVX operations use the contents of the 16-bit DPTR to determine the effective address.

Split Mode without Bank Select

In Split Mode without Bank Select, the XRAM memory map is split into two areas: on-chip space and off-chip space.

- Effective addresses below the on-chip XRAM boundary will access on-chip XRAM space.
- Effective addresses above the on-chip XRAM boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is onchip or off-chip. However, in the No Bank Select mode, an 8-bit MOVX operation will not drive the upper bits A[15:8] of the Address Bus during an off-chip access. This allows firmware to manipulate the upper address bits at will by setting the port state directly via the port latches. This behavior is in contrast with Split Mode with Bank Select. The lower 8-bits of the Address Bus A[7:0] are driven, determined by R0 or R1.
- 16-bit MOVX operations use the contents of DPTR to determine whether the memory access is onchip or off-chip, and unlike 8-bit MOVX operations, the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

Split Mode with Bank Select

In Split Mode with Bank Select, the XRAM memory map is split into two areas: on-chip space and off-chip space.

- Effective addresses below the on-chip XRAM boundary will access on-chip XRAM space.
- Effective addresses above the on-chip XRAM boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is onchip or off-chip. The upper bits of the Address Bus A[15:8] are determined by EMI0CN, and the lower 8-bits of the Address Bus A[7:0] are determined by R0 or R1. All 16-bits of the Address Bus A[15:0] are driven in Bank Select mode.
- 16-bit MOVX operations use the contents of DPTR to determine whether the memory access is onchip or off-chip, and the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transactions.

External Only

In External Only mode, all MOVX operations are directed to off-chip space. On-chip XRAM is not visible to the CPU. This mode is useful for accessing off-chip memory located between 0x0000 and the on-chip XRAM boundary.

- 8-bit MOVX operations ignore the contents of EMI0CN. The upper Address bits A[15:8] are not driven (identical behavior to an off-chip access in Split Mode without Bank Select). This allows firmware to manipulate the upper address bits at will by setting the port state directly. The lower 8-bits of the effective address A[7:0] are determined by the contents of R0 or R1.
- 16-bit MOVX operations use the contents of DPTR to determine the effective address A[15:0]. The full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

15.3.6 Timing

The timing parameters of the External Memory Interface can be configured to enable connection to devices having different setup and hold time requirements. The Address Setup time, Address Hold time, RDb and WRb strobe widths, and in multiplexed mode, the width of the ALE pulse are all programmable in units of SYSCLK periods.

The timing for an off-chip MOVX instruction can be calculated by adding 4 SYSCLK cycles to the timing parameters defined by the EMIF registers. Assuming non-multiplexed operation, the minimum execution time for an off-chip XRAM operation is 5 SYSCLK cycles (1 SYSCLK for RDb or WRb pulse + 4 SYSCLKs). For multiplexed operations, the Address Latch Enable signal will require a minimum of 2 additional SYSCLK cycles. Therefore, the minimum execution time of an off-chip XRAM operation in multiplexed mode is 7 SYSCLK cycles (2 SYSCLKs for ALEm, 1 for RDb or WRb + 4 SYSCLKs). The programmable setup and hold times default to the maximum delay settings after a reset.

Table 15.3. External Memory Interface Timing

Parameter	Description	Min	Max	Units
T_{ACS}	Address/Control Setup Time	0	3 x T_{SYSCLK}	ns
T_{ACW}	Address/Control Pulse Width	1 x T_{SYSCLK}	16 x T_{SYSCLK}	ns
T_{ACH}	Address/Control Hold Time	0	3 x T_{SYSCLK}	ns
T_{ALEH}	Address Latch Enable High Time	1 x T_{SYSCLK}	4 x T_{SYSCLK}	ns
T_{ALEL}	Address Latch Enable Low Time	1 x T_{SYSCLK}	4 x T_{SYSCLK}	ns
T_{WDS}	Write Data Setup Time	1 x T_{SYSCLK}	19 x T_{SYSCLK}	ns
T_{WDH}	Write Data Hold Time	0	3 x T_{SYSCLK}	ns
T_{RDS}	Read Data Setup Time	20	—	ns
T_{RDH}	Read Data Hold Time	0	—	ns

Note: T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).

15.3.6.1 Multiplexed Mode

Figure 15.6 Multiplexed 16-bit MOVX Timing on page 180 through Figure 15.8 Multiplexed 8-bit MOVX with Bank Select Timing on page 182 show the timing diagrams for the different External Memory Interface multiplexed modes and MOVX operations.

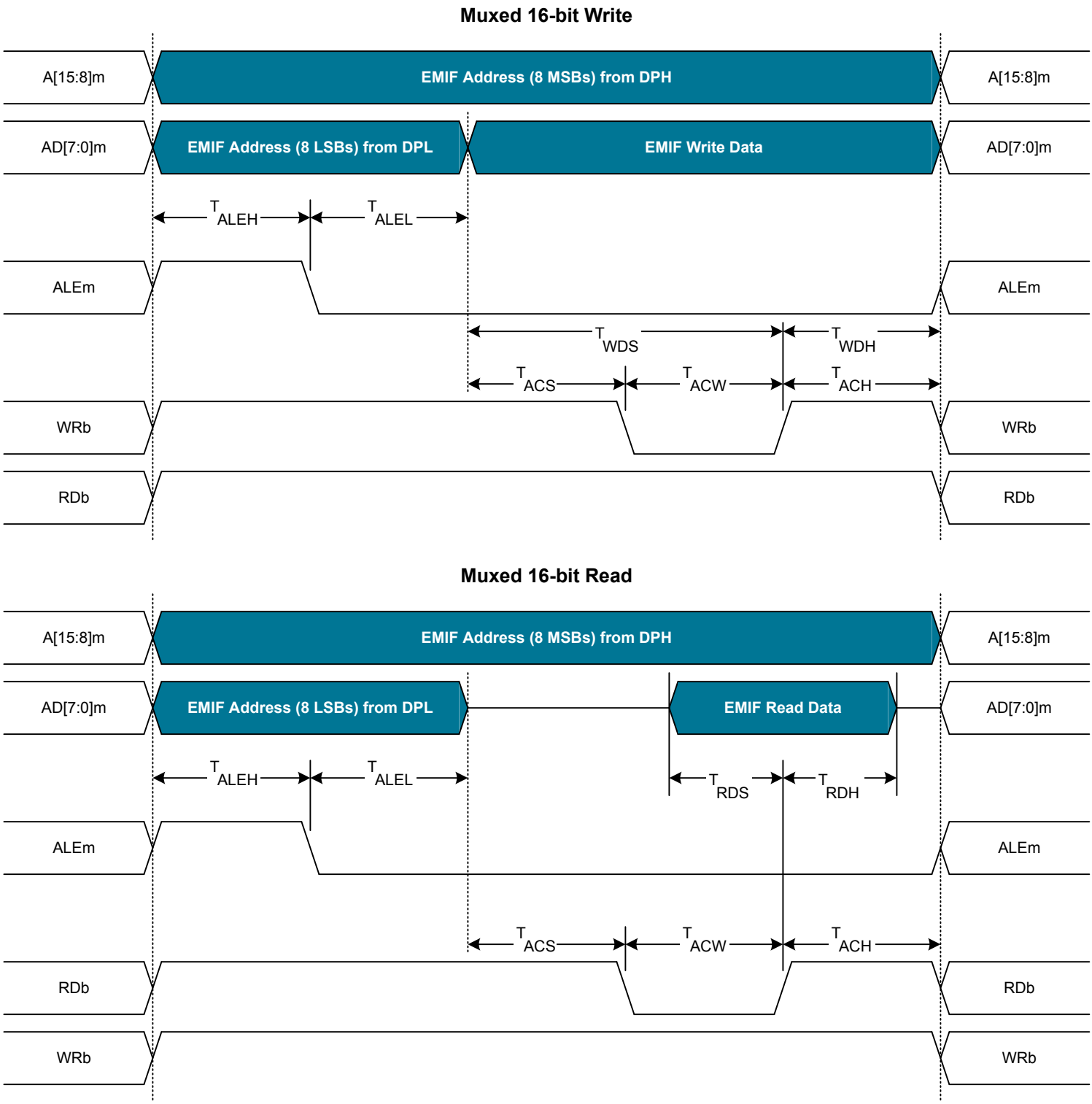
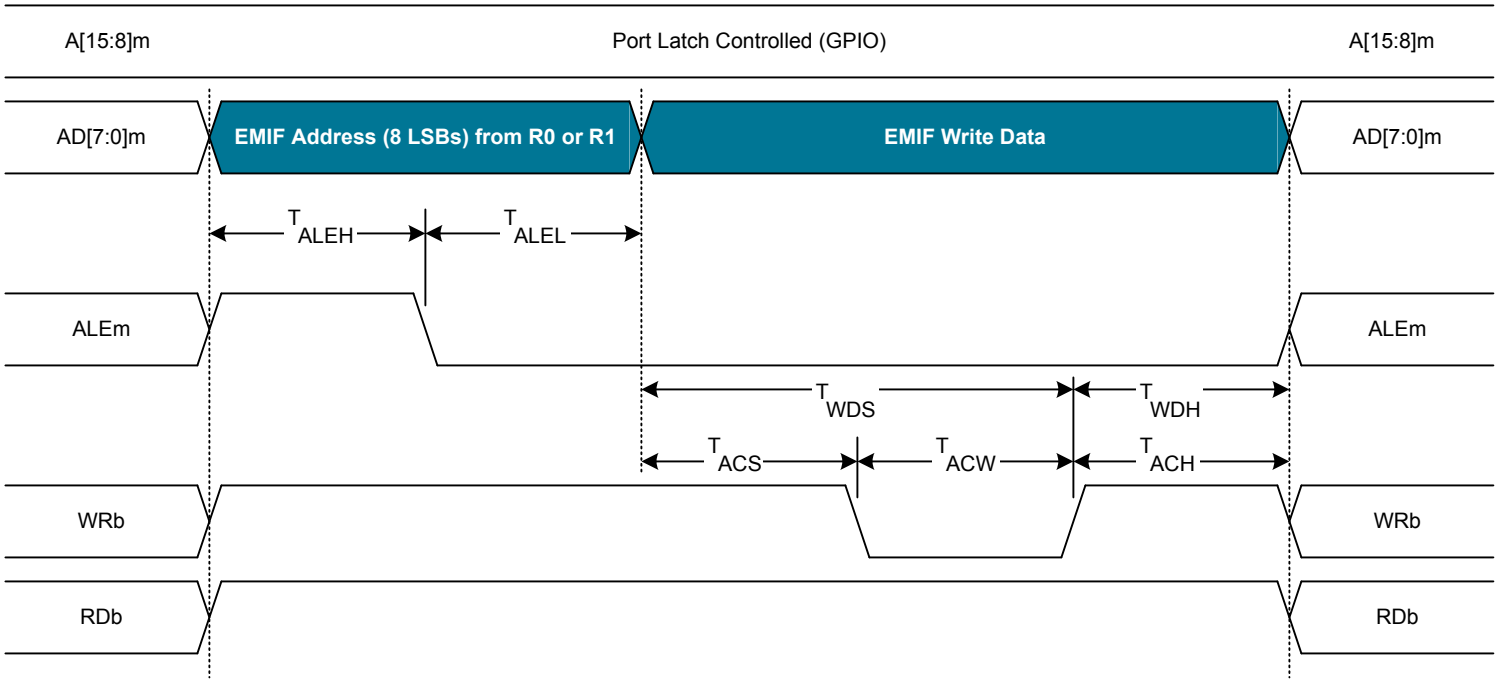


Figure 15.6. Multiplexed 16-bit MOVX Timing

Muxed 8-bit Write Without Bank Select



Muxed 8-bit Read Without Bank Select

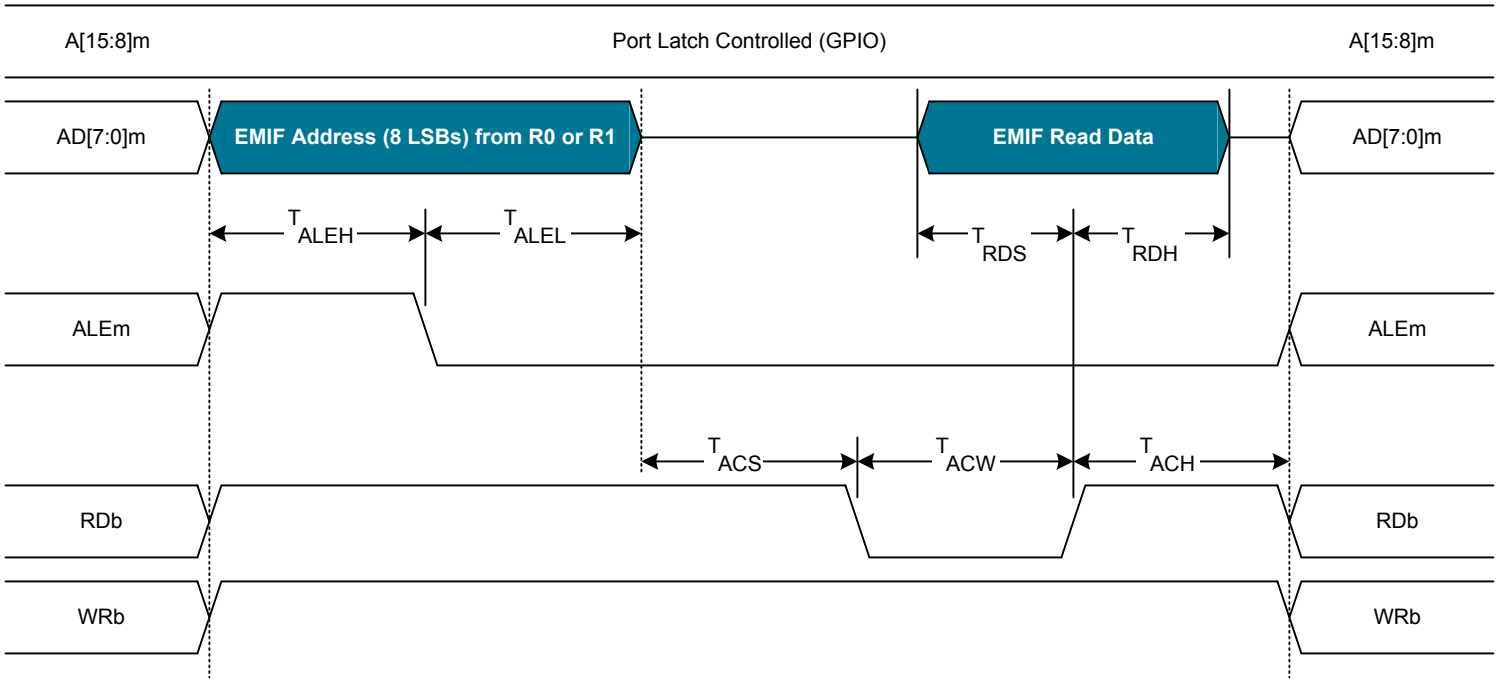


Figure 15.7. Multiplexed 8-bit MOVX without Bank Select Timing

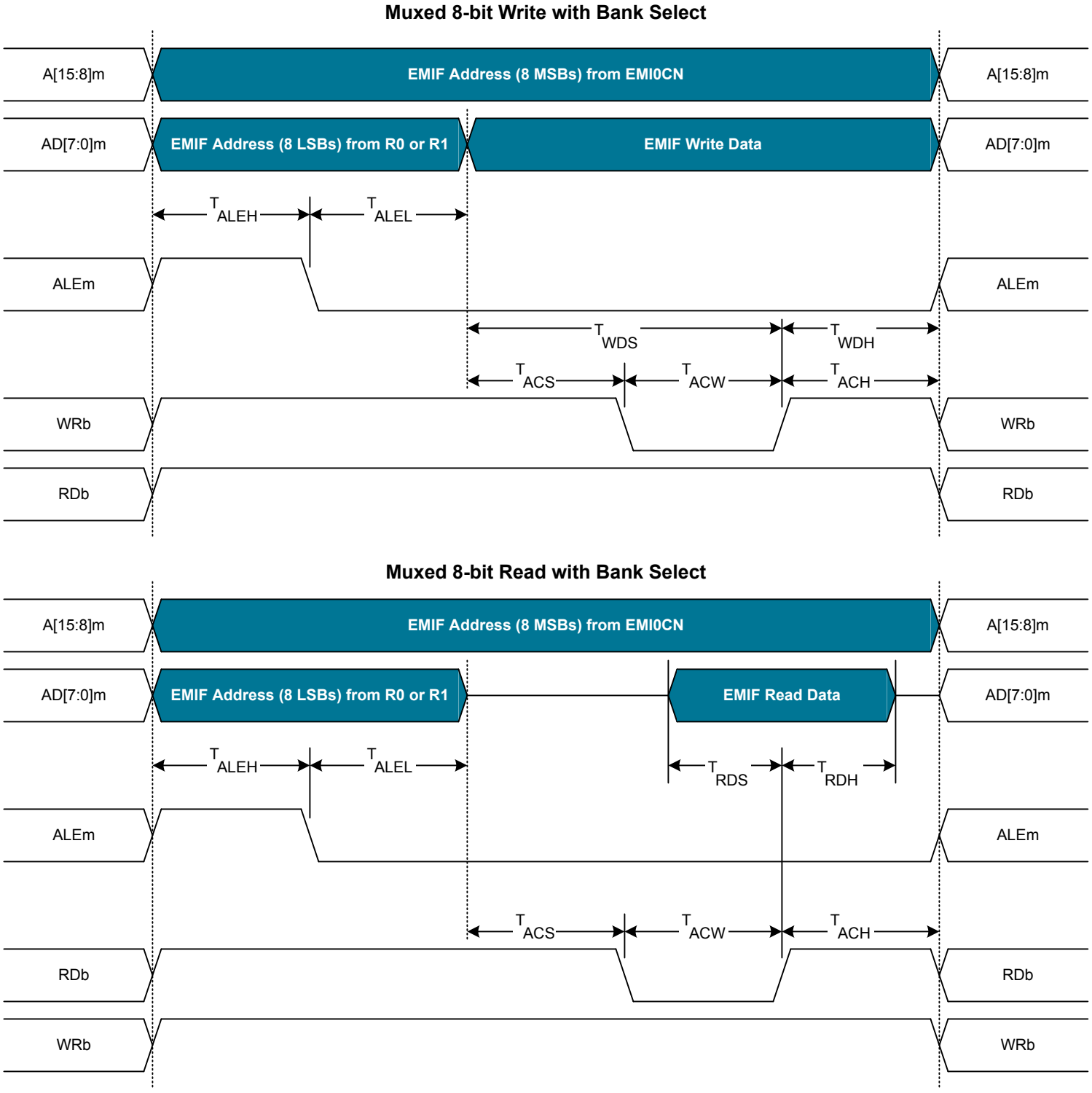


Figure 15.8. Multiplexed 8-bit MOVX with Bank Select Timing

15.3.6.2 Non-Multiplexed Mode

Figure 15.9 Non-Multiplexed 16-bit MOVX Timing on page 183 through Figure 15.11 Non-Multiplexed 8-bit MOVX with Bank Select Timing on page 185 show the timing diagrams for the different External Memory Interface non-multiplexed modes and MOVX operations.

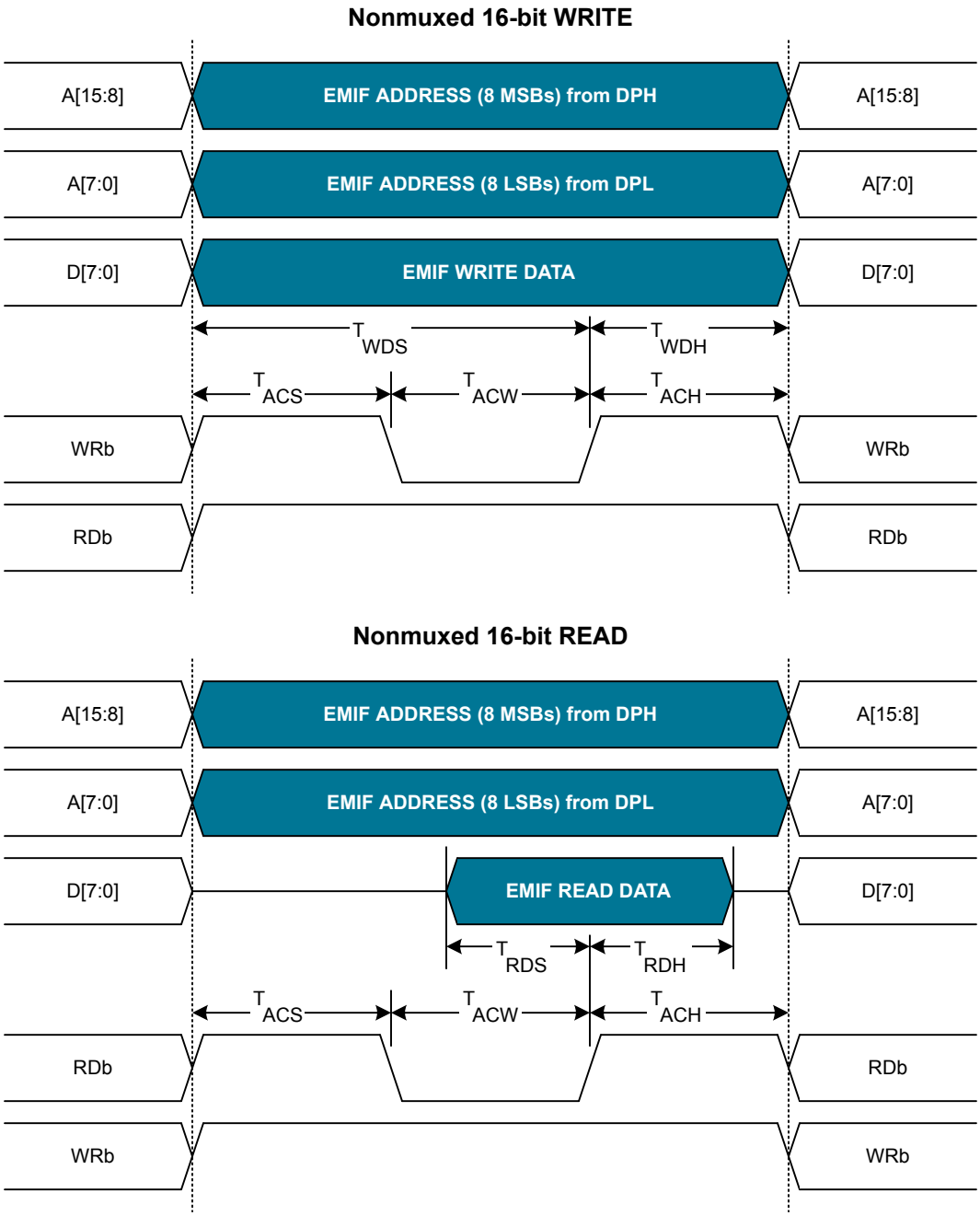
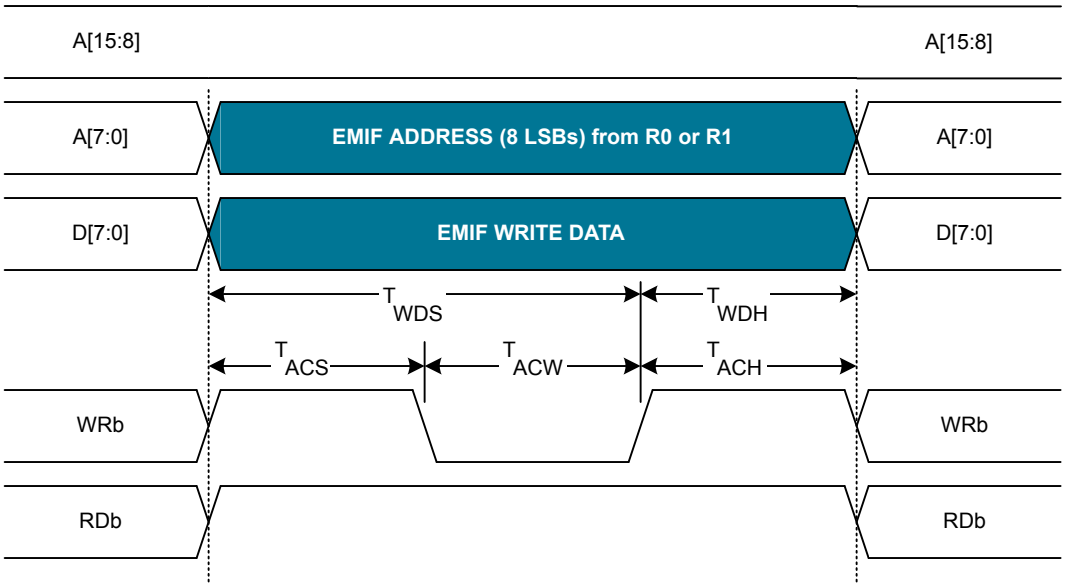


Figure 15.9. Non-Multiplexed 16-bit MOVX Timing

Nonmuxed 8-bit WRITE without Bank Select



Nonmuxed 8-bit READ without Bank Select

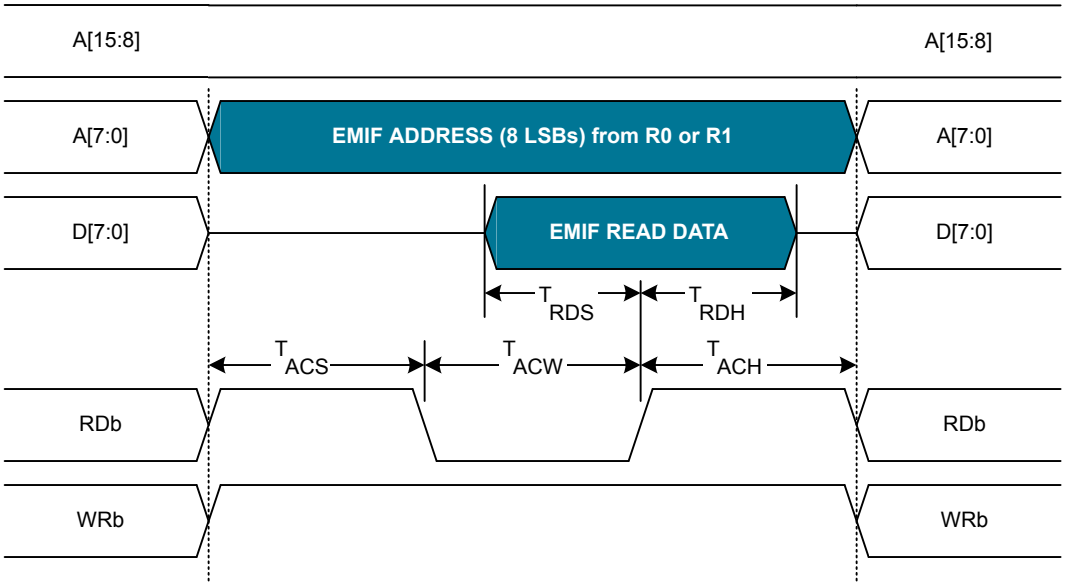


Figure 15.10. Non-Multiplexed 8-bit MOVX without Bank Select Timing

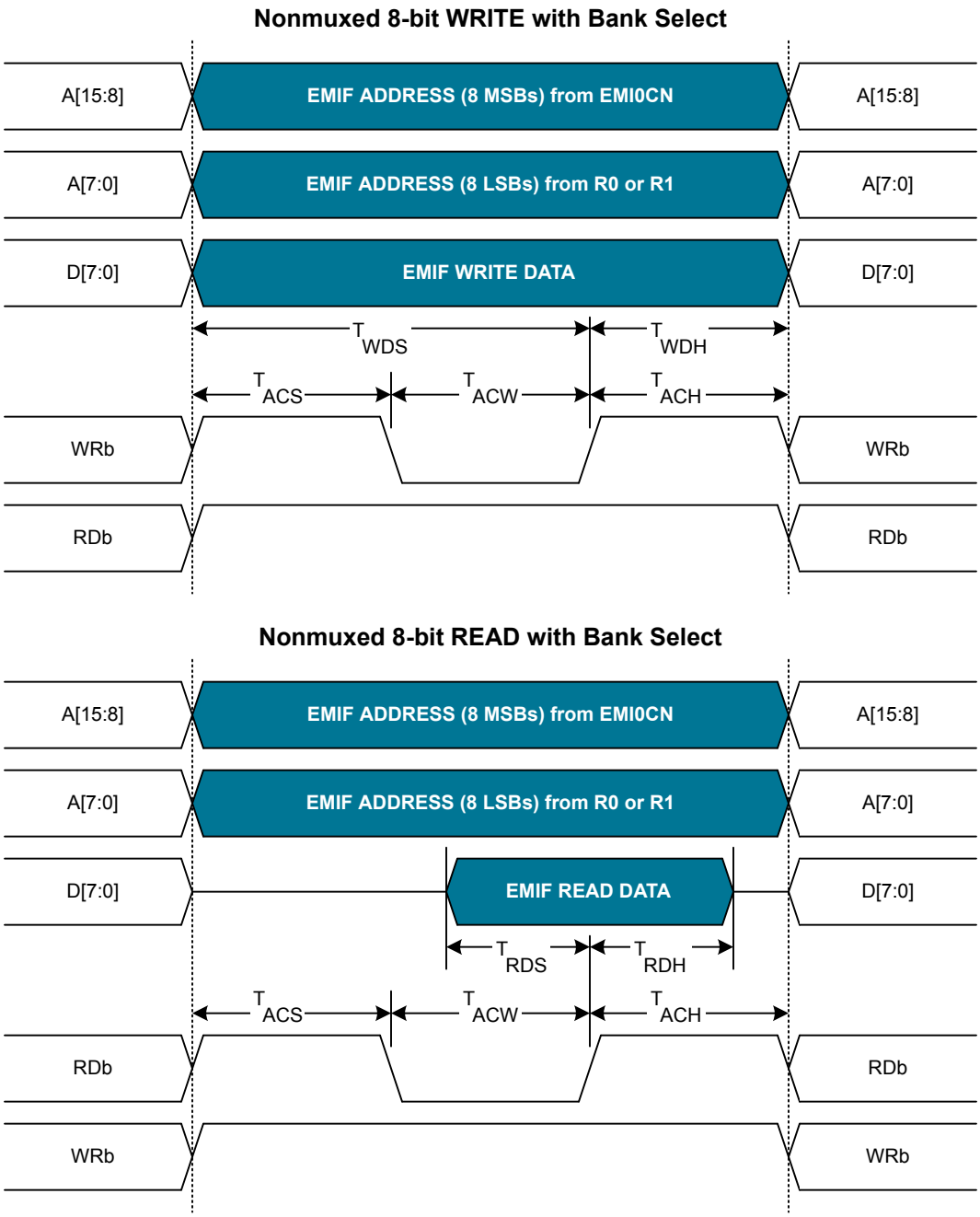


Figure 15.11. Non-Multiplexed 8-bit MOVX with Bank Select Timing

15.4 EMIF0 Control Registers

15.4.1 EMI0CN: External Memory Interface Control

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

Bit	Name	Reset	Access	Description
7:0	PGSEL	0x00	RW	**XRAM Page Select.** The XRAM Page Select field provides the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. 0x00: 0x0000 to 0x00FF 0x01: 0x0100 to 0x01FF ... 0xFE: 0xFE00 to 0xFEFF 0xFF: 0xFF00 to 0xFFFF

15.4.2 EMI0CF: External Memory Configuration

Bit	7	6	5	4	3	2	1	0
Name	Reserved	USBFAE	Reserved	MUXMD	EMD		EALE	
Access	RW	RW	RW	RW	RW		RW	
Reset	0	0	0	0	0x0		0x3	

SFR Page = ALL; SFR Address: 0x85

Bit	Name	Reset	Access	Description
7	*Reserved*	*Must write reset value.*		
6	USBFAE	0	RW	**USB FIFO Access Enable.**
	Value	Name		Description
	0	FIFO_ACCESS_DISABLED		USB FIFO RAM not available through MOVX instructions.
	1	FIFO_ACCESS_ENABLED		USB FIFO RAM available using MOVX instructions. The 1 KB of USB RAM will be mapped in XRAM space at addresses 0x0400 to 0x07FF. The USB clock must be active and greater than or equal to twice the SYSCLK (USBCLK > 2 x SYSCLK) to access this area with MOVX instructions.
5	*Reserved*	*Must write reset value.*		
4	MUXMD	0	RW	**EMIF Multiplex Mode Select.**
	Value	Name		Description
	0	MULTIPLEXED		EMIF operates in multiplexed address/data mode.
	1	NON_MULTIPLEXED		EMIF operates in non-multiplexed mode (separate address and data pins).
3:2	EMD	0x0	RW	**EMIF Operating Mode Select.**
	Value	Name		Description
	0x0	INTERNAL_ONLY		Internal Only: MOVX accesses on-chip XRAM only. All effective addresses alias to on-chip memory space.
	0x1	SPLIT_WITHOUT_BANK_SELECT		Split Mode without Bank Select: Accesses below the internal XRAM boundary are directed on-chip. Accesses above the internal XRAM boundary are directed off-chip. 8-bit off-chip MOVX operations use the current contents of the Address high port latches to resolve the upper address byte. To access off chip space, EMI0CN must be set to a page that is not contained in the on-chip address space.
	0x2	SPLIT_WITH_BANK_SELECT		Split Mode with Bank Select: Accesses below the internal XRAM boundary are directed on-chip. Accesses above the internal XRAM boundary are directed off-chip. 8-bit off-chip MOVX operations uses the contents of EMI0CN to determine the high-byte of the address.
	0x3	EXTERNAL_ONLY		External Only: MOVX accesses off-chip XRAM only. On-chip XRAM is not visible to the core.
1:0	EALE	0x3	RW	**ALE Pulse-Width Select.** These bits only have an effect when the EMIF is in multiplexed mode (MUXMD = 0).

Bit	Name	Reset	Access	Description
	Value	Name		Description
	0x0	1_CLOCK		ALE high and ALE low pulse width = 1 SYSCLK cycle.
	0x1	2_CLOCKS		ALE high and ALE low pulse width = 2 SYSCLK cycles.
	0x2	3_CLOCKS		ALE high and ALE low pulse width = 3 SYSCLK cycles.
	0x3	4_CLOCKS		ALE high and ALE low pulse width = 4 SYSCLK cycles.

15.4.3 EMI0TC: External Memory Timing Control

Bit	7	6	5	4	3	2	1	0
Name	ASETUP		PWIDTH				AHOLD	
Access	RW		RW				RW	
Reset	0x3		0xF				0x3	
SFR Page = ALL; SFR Address: 0x84								

Bit	Name	Reset	Access	Description
7:6	ASETUP	0x3	RW	**EMIF Address Setup Time.**
	Value	Name		Description
	0x0	0_CLOCKS		Address setup time = 0 SYSCLK cycles.
	0x1	1_CLOCK		Address setup time = 1 SYSCLK cycle.
	0x2	2_CLOCKS		Address setup time = 2 SYSCLK cycles.
	0x3	3_CLOCKS		Address setup time = 3 SYSCLK cycles.
5:2	PWIDTH	0xF	RW	**EMIF <overline>WR</overline> and <overline>RD</overline> Pulse-Width Control.**
	Value	Name		Description
	0x0	1_CLOCK		/WR and /RD pulse width is 1 SYSCLK cycle.
	0x1	2_CLOCKS		/WR and /RD pulse width is 2 SYSCLK cycles.
	0x2	3_CLOCKS		/WR and /RD pulse width is 3 SYSCLK cycles.
	0x3	4_CLOCKS		/WR and /RD pulse width is 4 SYSCLK cycles.
	0x4	5_CLOCKS		/WR and /RD pulse width is 5 SYSCLK cycles.
	0x5	6_CLOCKS		/WR and /RD pulse width is 6 SYSCLK cycles.
	0x6	7_CLOCKS		/WR and /RD pulse width is 7 SYSCLK cycles.
	0x7	8_CLOCKS		/WR and /RD pulse width is 8 SYSCLK cycles.
	0x8	9_CLOCKS		/WR and /RD pulse width is 9 SYSCLK cycles.
	0x9	10_CLOCKS		/WR and /RD pulse width is 10 SYSCLK cycles.
	0xA	11_CLOCKS		/WR and /RD pulse width is 11 SYSCLK cycles.
	0xB	12_CLOCKS		/WR and /RD pulse width is 12 SYSCLK cycles.
	0xC	13_CLOCKS		/WR and /RD pulse width is 13 SYSCLK cycles.
	0xD	14_CLOCKS		/WR and /RD pulse width is 14 SYSCLK cycles.
	0xE	15_CLOCKS		/WR and /RD pulse width is 15 SYSCLK cycles.
	0xF	16_CLOCKS		/WR and /RD pulse width is 16 SYSCLK cycles.
1:0	AHOLD	0x3	RW	**EMIF Address Hold Time.**
	Value	Name		Description
	0x0	0_CLOCKS		Address hold time = 0 SYSCLK cycles.
	0x1	1_CLOCK		Address hold time = 1 SYSCLK cycle.

Bit	Name	Reset	Access	Description
	0x2	2_CLOCKS		Address hold time = 2 SYSCLK cycles.
	0x3	3_CLOCKS		Address hold time = 3 SYSCLK cycles.

16. Universal Serial Bus (USB0)

16.1 Introduction

The USB0 module provides Full/Low Speed function for USB peripheral implementations. The USB function controller (USB0) consists of a Serial Interface Engine (SIE), USB transceiver (including matching resistors and configurable pull-up resistors), 1 KB FIFO block, and clock recovery mechanism for crystal-less operation. No external components are required. The USB0 module is Universal Serial Bus Specification 2.0 compliant.

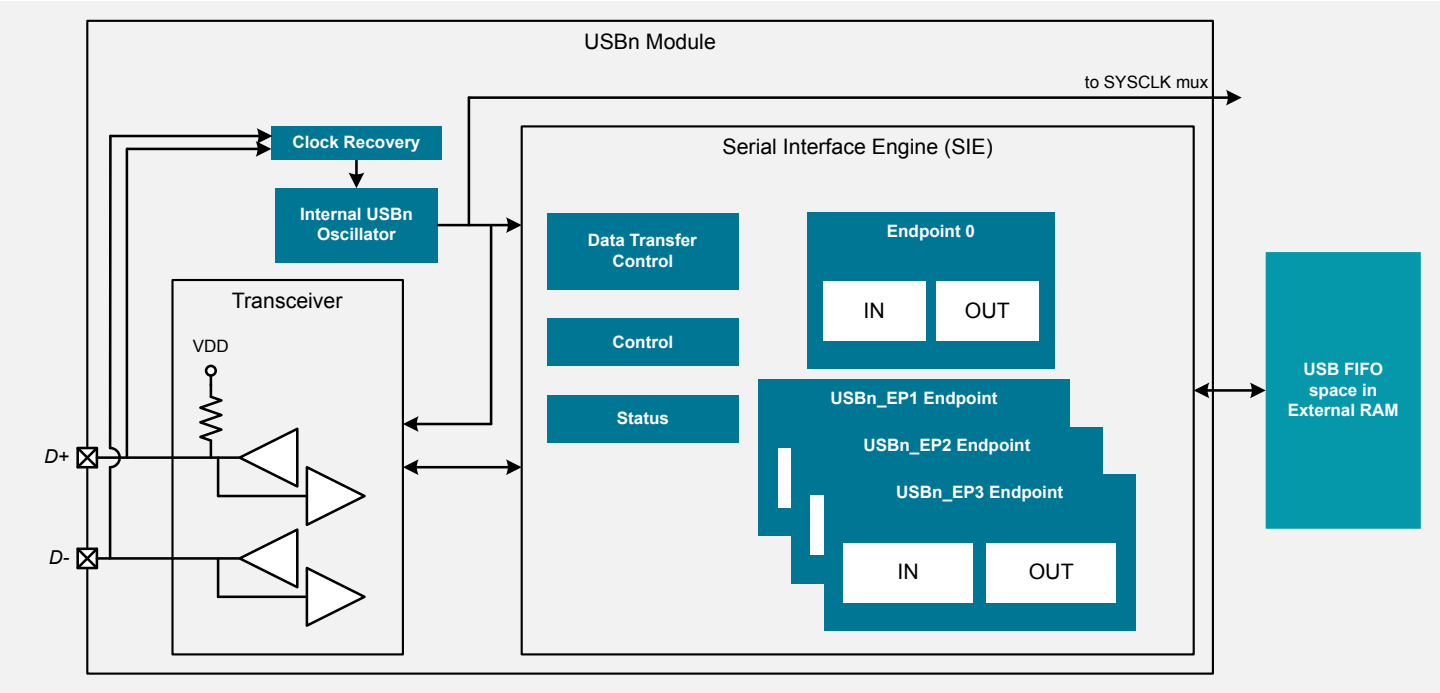


Figure 16.1. USB Block Diagram

16.2 Features

The USB0 module includes the following features:

- Full and Low Speed functionality.
- Implements 4 bidirectional endpoints.
- 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.

16.3 Functional Description

16.3.1 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 16.1. 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

16.3.2 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.

16.3.3 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 CLKSEL register. A typical full speed application would configure the USB clock to run directly from the HFOSC0 oscillator, while a typical low speed application would configure the clock for HFOSC0/8. 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 HFOSC0 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.

16.3.4 VBUS Control

The VBUS signal is a dedicated input pin on the device which is used to indicateswhen a host device has been connected to or disconnected from the USB peripheral.

The USB VBUS line must be connected to the VBUS pin when using the device in a USB network. The VBUS signal should only be connected to the VREGIN pin when operating the device as a bus-powered function.

The VBUS pin may also generate interrupts on rising or falling edges, if enabled. The VBUS interrupt is edge-sensitive and has no associated interrupt pending flag. Firmware may read the VBSTAT bit in register REG01CN at any time to determine the current logic level of the VBUS signal.

If USB is selected as a reset source, a system reset will be generated by activity on the VBUS pin.

16.3.5 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.

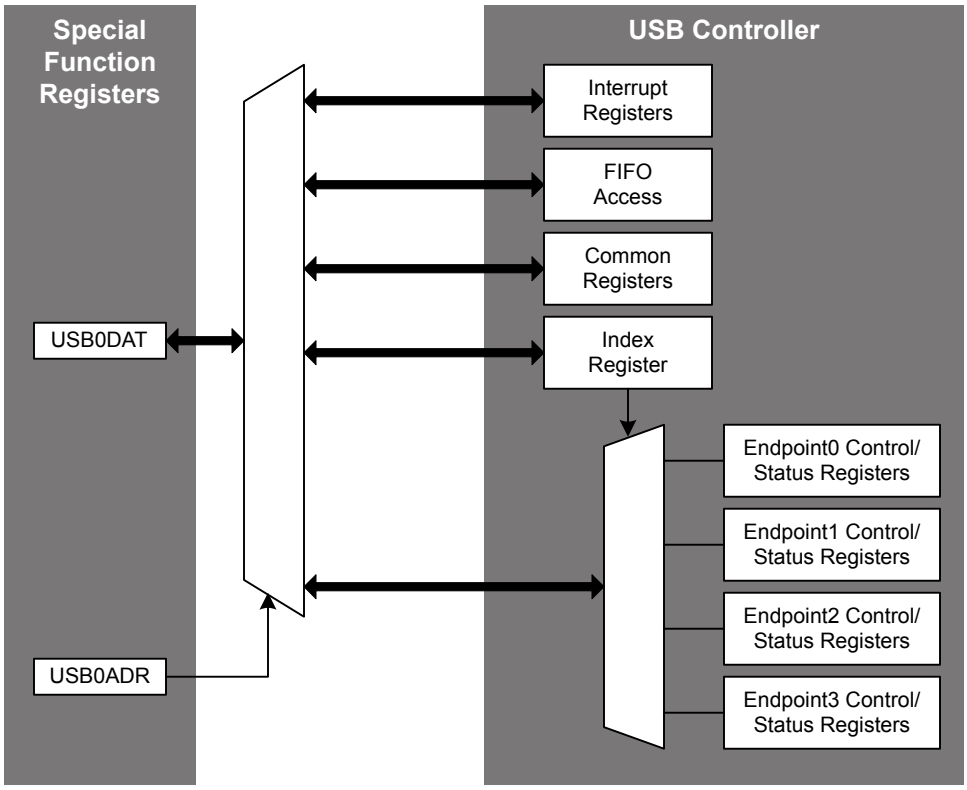


Figure 16.2. USB Indirect Register Access

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

Table 16.2. 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
FRAMEL	0x0C	Frame Number Low Byte

USB Register Name	USB Register Address	Description
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

16.3.6 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).

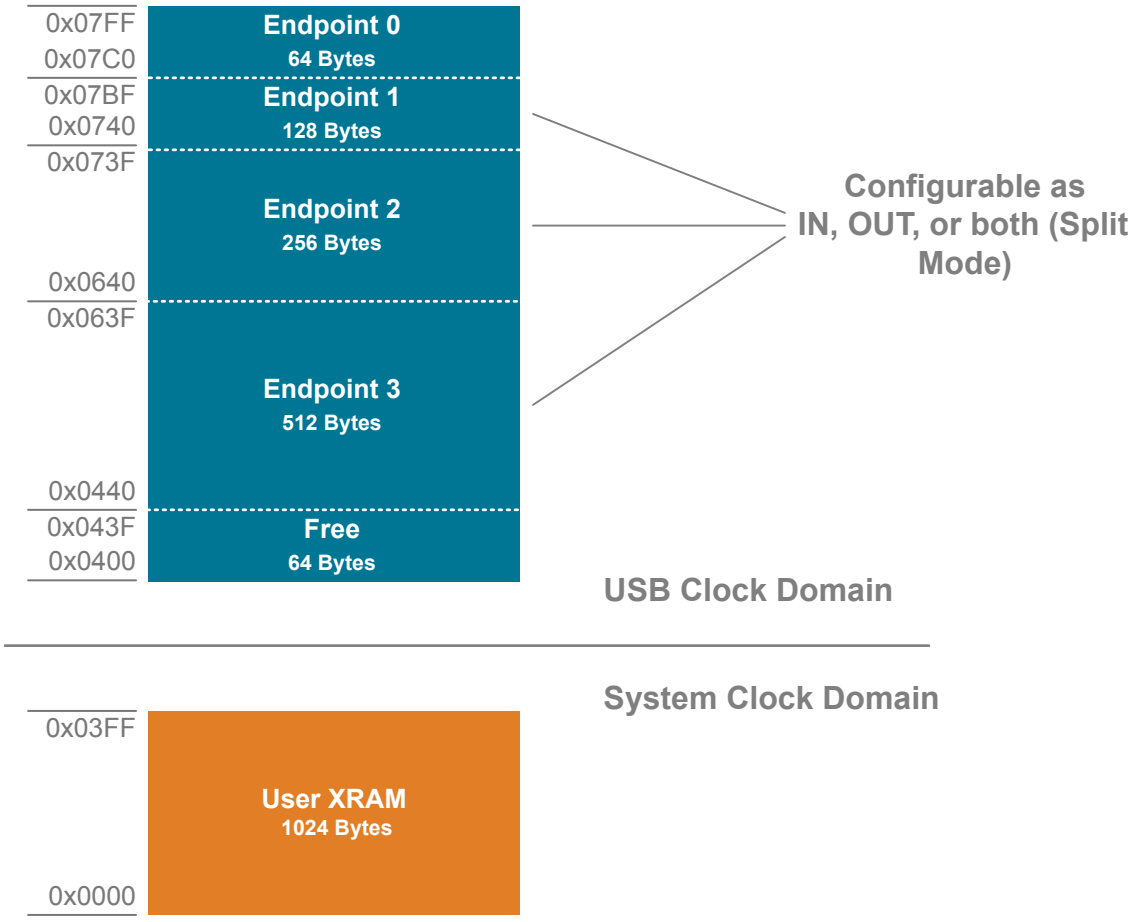


Figure 16.3. FIFO Memory Map

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 16.3. 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. To access the FIFO RAM directly using MOVX instructions, the following conditions must be met:

1. The USBFAE bit in register EMI0CF must be set to 1.
2. The USB clock must be greater than or equal to twice the SYSCLK:

   USBCLK > 2 × SYSCLK

When the USBFAE bit is set, the USB FIFO space is mapped into XRAM space at addresses 0x0400 to 0x07FF. The normal XRAM (on-chip or external) at the same addresses cannot be accessed when the USBFAE bit is set to 1.

Note: The USB clock must be active when accessing FIFO space.

16.3.7 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.

16.3.8 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.

16.3.9 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.

16.3.10 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.

16.3.11 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.

16.3.12 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.

16.4 USB0 Control Registers

16.4.1 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 = ALL; SFR Address: 0xD7								

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).
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).
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.
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).
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.

Bit	Name	Reset	Access	Description
	1	DIFFERENTIAL_ONE		Differential 1 signalling on the bus.
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.
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.

16.4.2 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: 0x96								

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).
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.			

16.4.3 USB0DAT: USB0 Data

Bit	7	6	5	4	3	2	1	0
Name	USB0DAT							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0x97								

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.

16.4.4 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.

16.4.5 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.				

16.4.6 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.				

16.4.7 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.				

16.4.8 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.				

16.4.9 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.				

16.4.10 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.

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.				

16.4.11 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.
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.
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.				

16.4.12 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.				

16.4.13 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.				

16.4.14 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.				

16.4.15 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	Name		Description
	0	NOT_SET		OUT Endpoint 3 interrupt inactive.
	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	Name		Description
	0	NOT_SET		OUT Endpoint 2 interrupt inactive.
	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	Name		Description
	0	NOT_SET		OUT Endpoint 1 interrupt inactive.
	1	SET		OUT Endpoint 1 interrupt active.
0	*Reserved*	*Must write reset value.*		
This register is accessed indirectly using the USB0ADR and USB0DAT registers.				

16.4.16 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.

16.4.17 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.				

16.4.18 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.				

16.4.19 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.				

16.4.20 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.				

16.4.21 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.				

16.4.22 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.				

16.4.23 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.

Bit	Name	Reset	Access	Description
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.

16.4.24 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.				

16.4.25 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.

16.4.26 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.				

16.4.27 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.				

16.4.28 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.				

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.

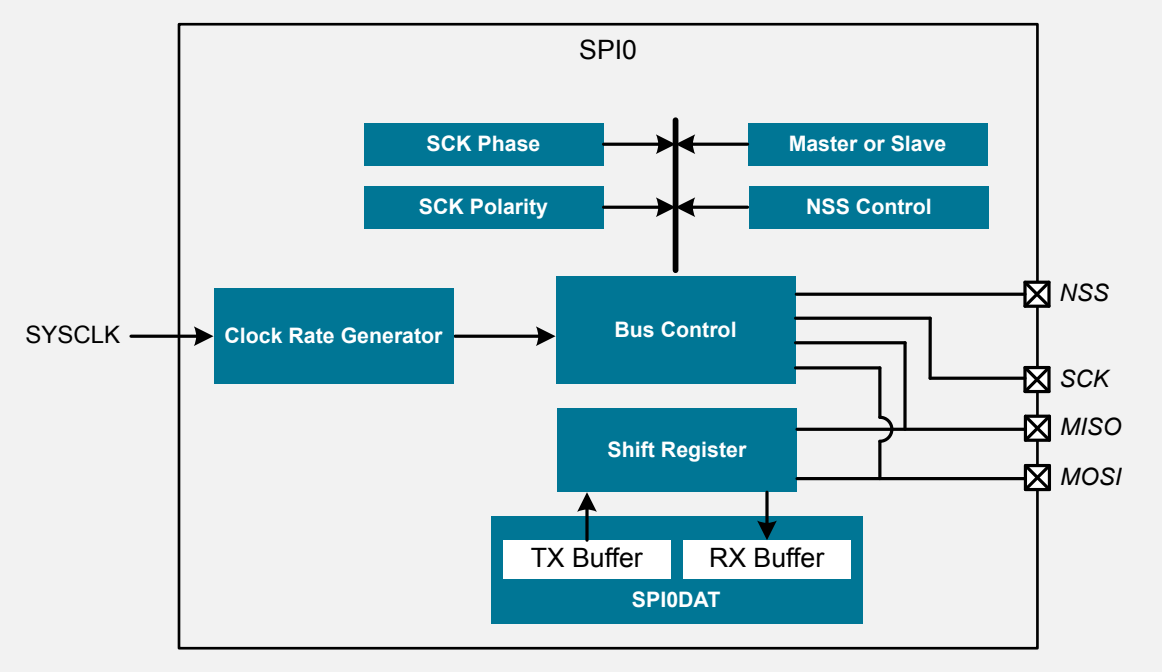


Figure 17.1. SPI Block Diagram

17.2 Features

The SPI module includes the following features:

- Supports 3- or 4-wire operation in master or slave modes.
- Supports external clock frequencies up to SYSCLK / 2 in master mode and SYSCLK / 10 in slave mode.
- Support for four clock phase and polarity options.
- 8-bit dedicated clock clock rate generator.
- 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.

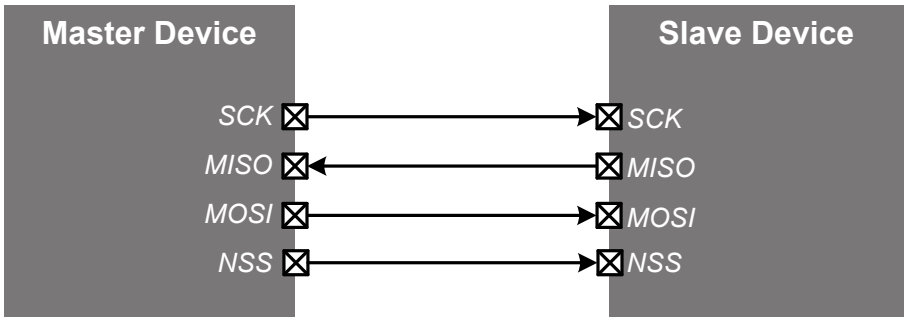


Figure 17.2. 4-Wire Connection Diagram

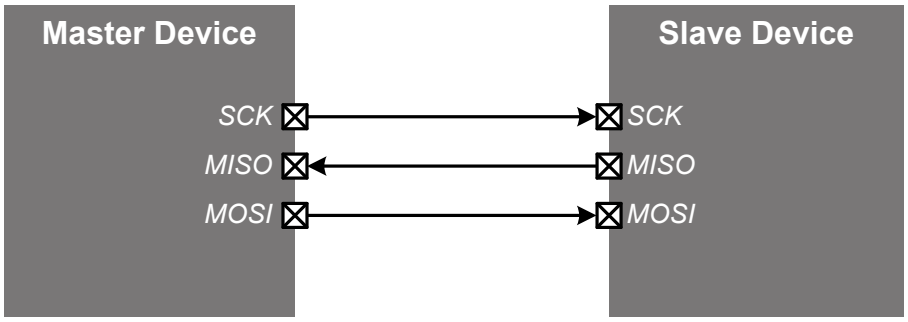


Figure 17.3. 3-Wire Connection Diagram

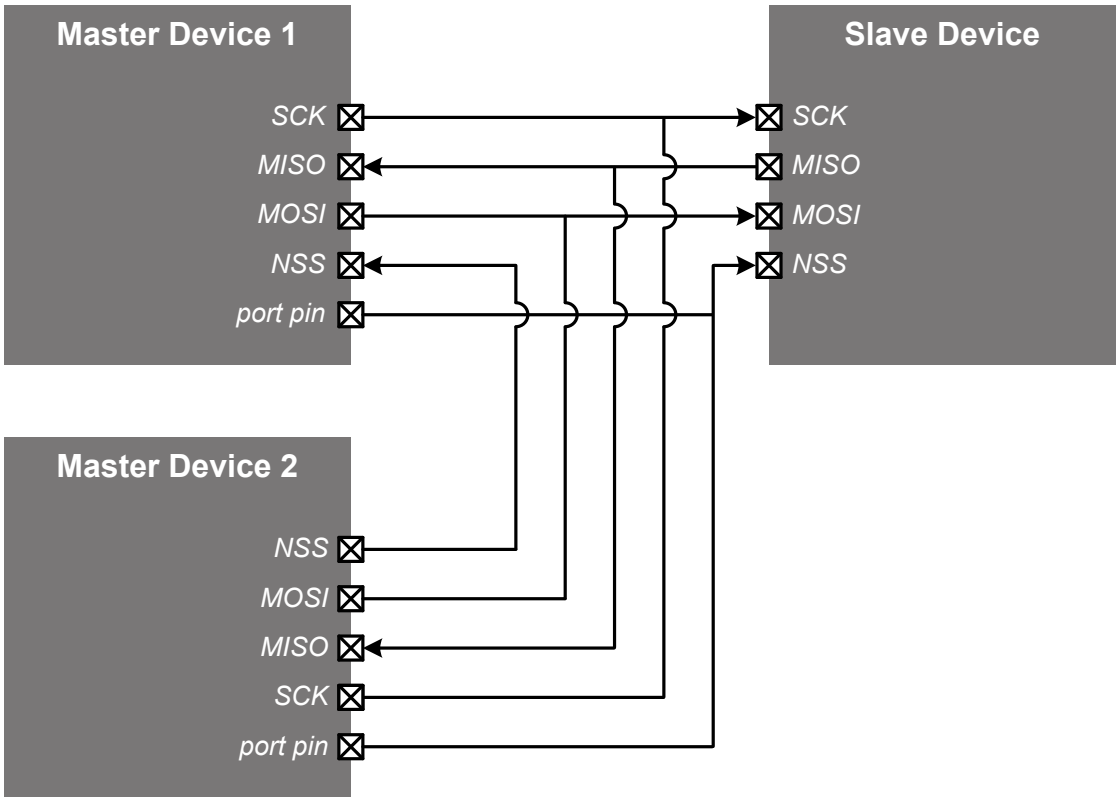


Figure 17.4. Multi-Master Connection Diagram

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.

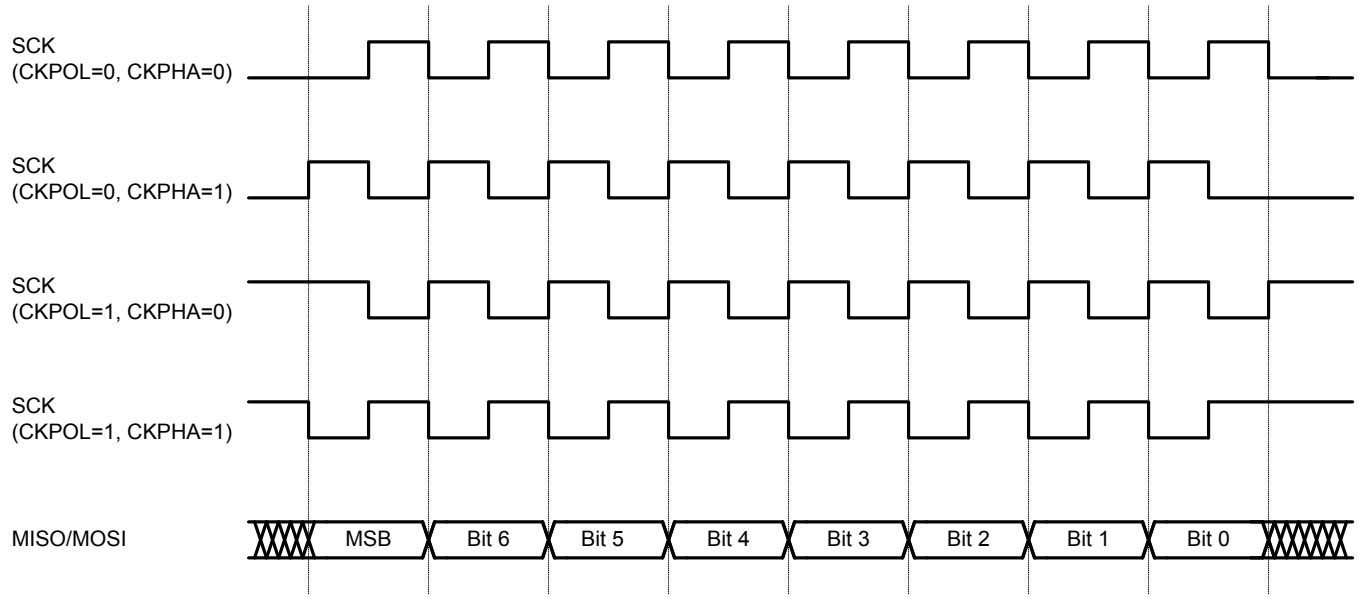


Figure 17.5. Master Mode Data/Clock Timing

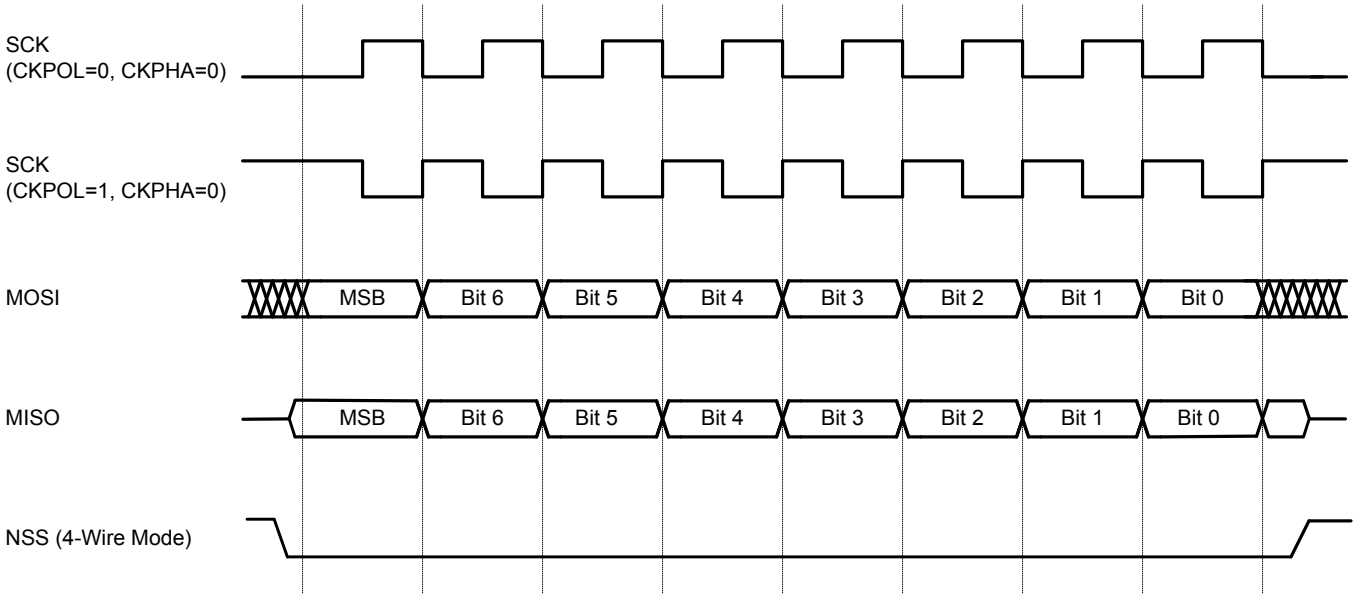


Figure 17.6. Slave Mode Data/Clock Timing (CKPHA = 0)

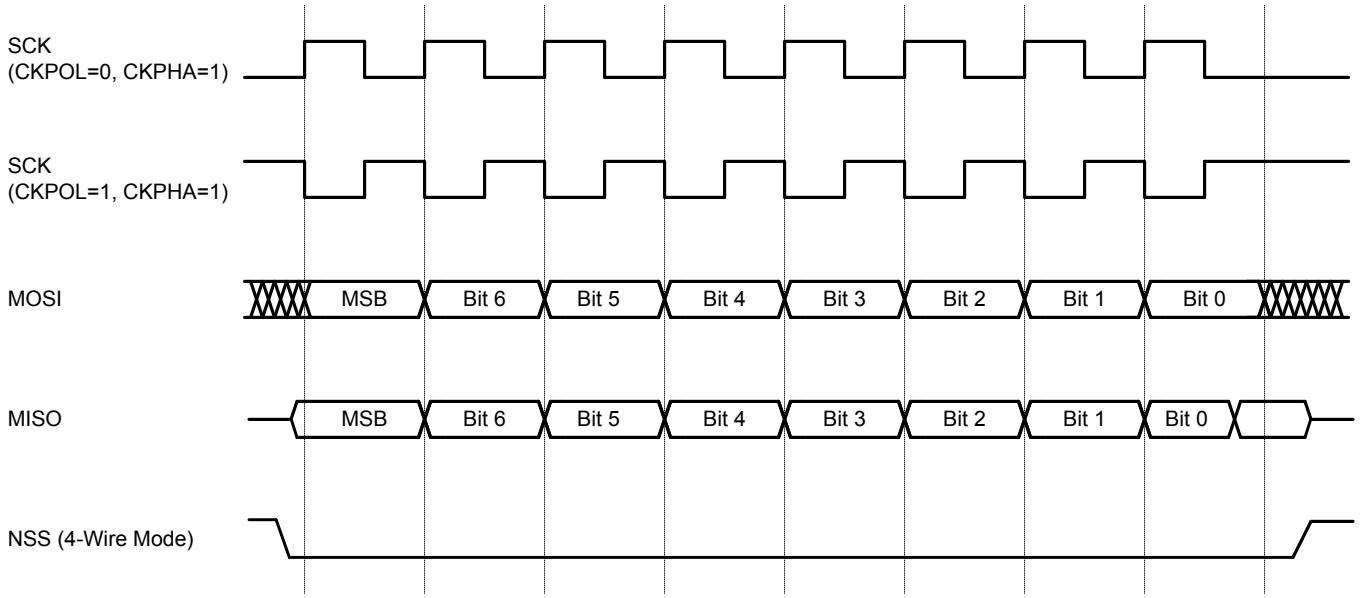


Figure 17.7. Slave Mode Data/Clock Timing (CKPHA = 1)

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:

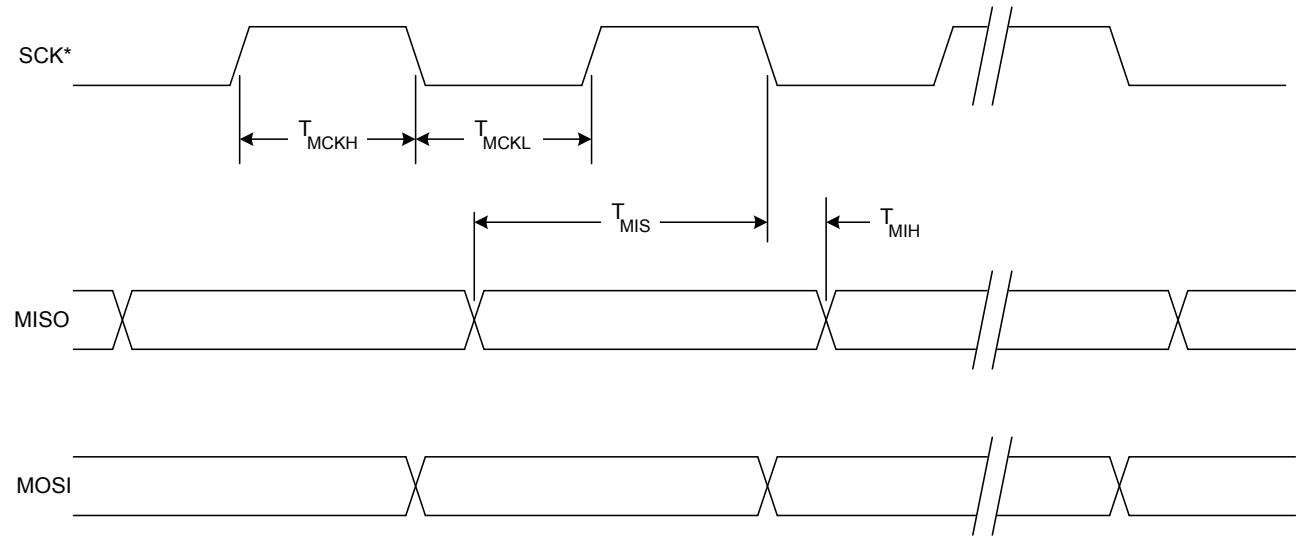
1. Write the data to be sent to SPInDAT. The transfer will begin on the bus at this time.
2. Wait for the SPIF flag to generate an interrupt, or poll SPIF until it is set to 1.
3. Read the received data from SPInDAT.
4. Clear the SPIF flag to 0.
5. Repeat the sequence for any additional transfers.

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.

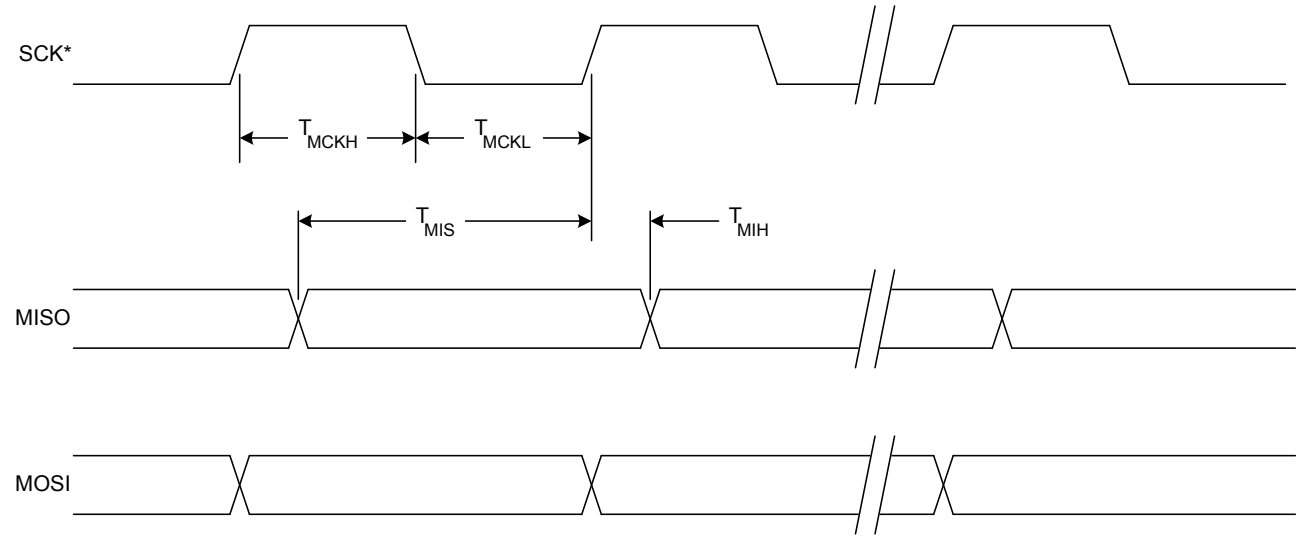
1. Write any data to be sent to SPInDAT. The transfer will not begin until the external master device initiates it.
2. Wait for the SPIF flag to generate an interrupt, or poll SPIF until it is set to 1.
3. Read the received data from SPInDAT.
4. Clear the SPIF flag to 0.
5. Repeat the sequence for any additional transfers.

17.3.6 SPI Timing Diagrams



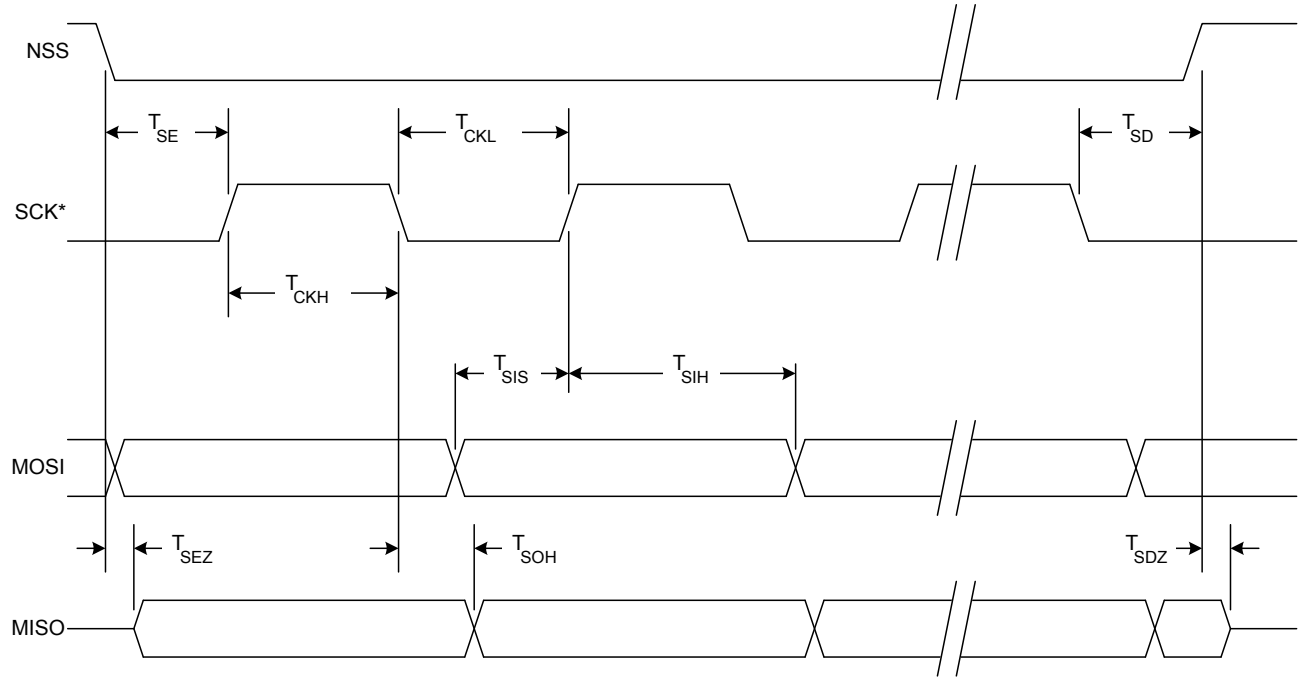
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 17.8. SPI Master Timing (CKPHA = 0)



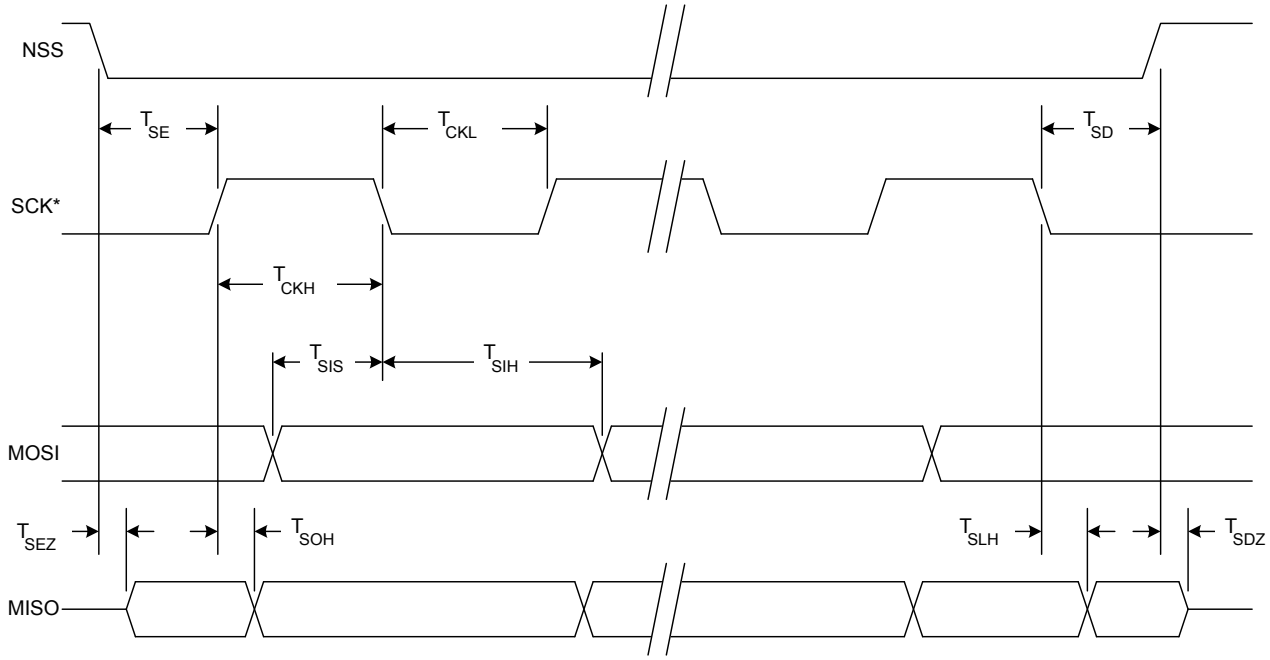
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

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 Shift Edge	1 x T_{SYSCLK} + 20	—	ns
T_{MIH}	SCK Shift Edge to MISO Change	0	—	ns
Slave Mode Timing				
T_{SE}	NSS Falling to First SCK Edge	2 x T_{SYSCLK}	—	ns
T_{SD}	Last SCK Edge to NSS Rising	2 x T_{SYSCLK}	—	ns
T_{SEZ}	NSS Falling to MISO Valid	—	4 x T_{SYSCLK}	ns
T_{SDZ}	NSS Rising to MISO High-Z	—	4 x T_{SYSCLK}	ns
T_{CKH}	SCK High Time	5 x T_{SYSCLK}	—	ns
T_{CKL}	SCK Low Time	5 x T_{SYSCLK}	—	ns
T_{SIS}	MOSI Valid to SCK Sample Edge	2 x T_{SYSCLK}	—	ns
T_{SIH}	SCK Sample Edge to MOSI Change	2 x T_{SYSCLK}	—	ns
T_{SOH}	SCK Shift Edge to MISO Change	—	4 x T_{SYSCLK}	ns
T_{SLH}	Last SCK Edge to MISO Change (CKPHA = 1 ONLY)	6 x T_{SYSCLK}	8 x T_{SYSCLK}	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	RXBMT
Access	R	RW	RW	RW	R	R	R	R
Reset	0	0	0	0	0	1	1	1
SFR Page = ALL; 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_CENTERED_FIRST		Data centered on first edge of SCK period.
	1	DATA_CENTERED_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.

Bit	Name	Reset	Access	Description
0	RXBMT	1	R	**Receive Buffer Empty.** This bit is valid in slave mode only and will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode.
In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device.				

17.4.2 SPI0CN0: SPI0 Control

Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSMD		TXBMT	SPIEN
Access	RW	RW	RW	RW	RW		R	RW
Reset	0	0	0	0	0x1		1	0
SFR Page = ALL; 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 SPI interrupts are enabled, 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 TXBMT 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 valid for slave mode only and is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by firmware.
3:2	NSSMD	0x1	RW	**Slave Select Mode.** Selects between the following NSS operation modes:

Value	Name	Description
0x0	3_WIRE	3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin.
0x1	4_WIRE_SLAVE	4-Wire Slave or Multi-Master Mode. NSS is an input to the device.
0x2	4_WIRE_MASTER_NSS_LOW	4-Wire Single-Master Mode. NSS is an output and logic low.
0x3	4_WIRE_MASTER_NSS_HIGH	4-Wire Single-Master Mode. NSS is an output and logic high.

1	TXBMT	1	R	**Transmit Buffer Empty.** This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer.
0	SPIEN	0	RW	**SPI0 Enable.**

Value	Name	Description
0	DISABLED	Disable the SPI module.
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 = ALL; 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 = ALL; 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.

18. System Management Bus / I2C (SMB0 and SMB1)

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.

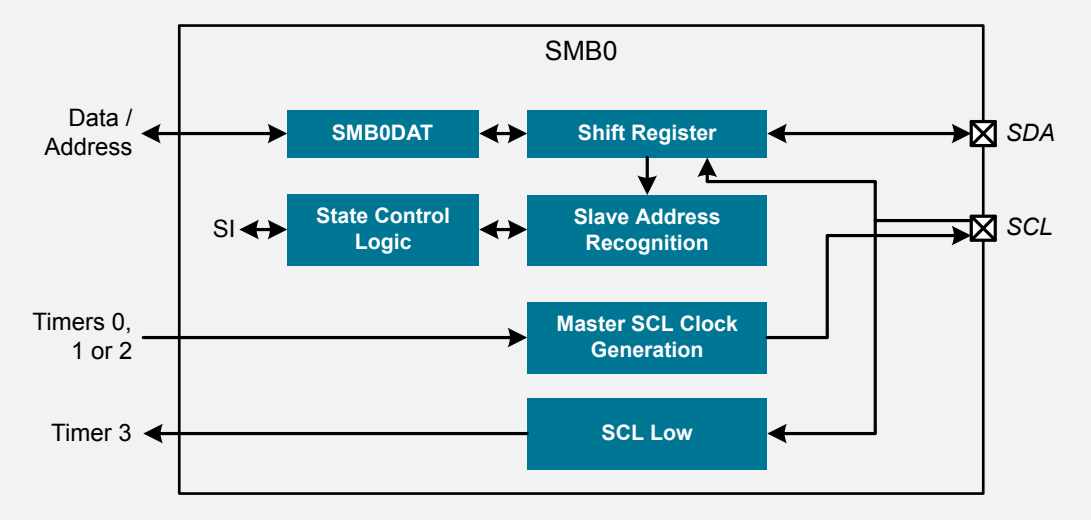


Figure 18.1. SMBus 0 Block Diagram

18.2 Features

The SMBus modules include 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.

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.

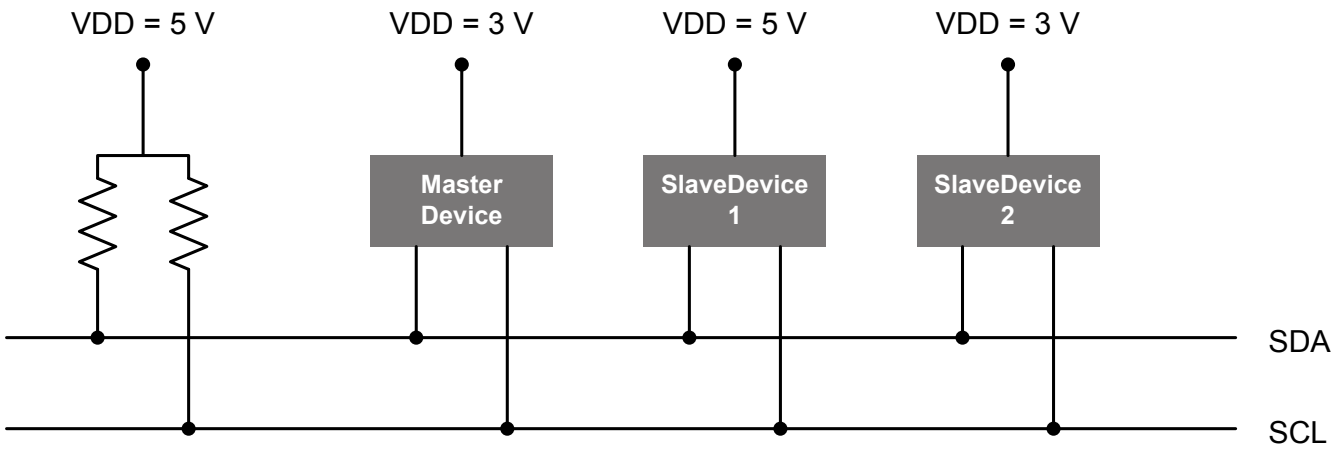


Figure 18.2. Typical SMBus System Connection

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 237). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

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

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 18.3 SMBus Transaction on page 237 illustrates a typical SMBus transaction.

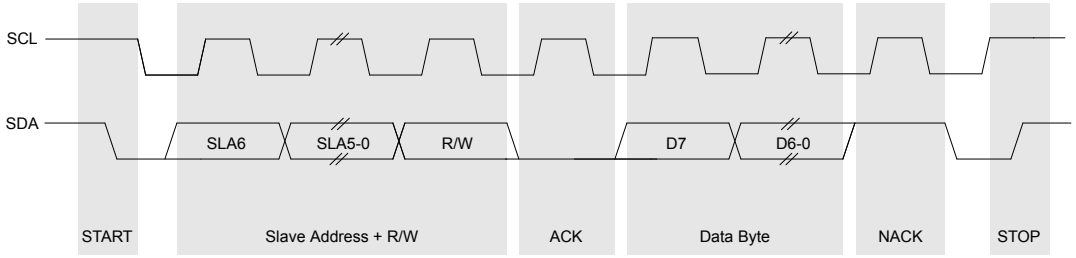


Figure 18.3. SMBus Transaction

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 237). 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.

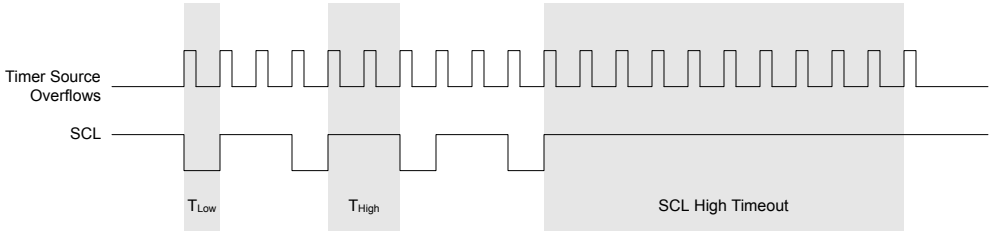


Figure 18.4. Typical SMBus SCL Generation

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 Timing Control

The SDD field in the SMBus Timing Control register is used to delay the recognition of the falling edge of the SDA signal. This feature should be applied in cases where a data bit transition occurs close to the SCL falling edge that may cause a false START detection when there is a significant mismatch between the impedance or capacitance on the SDA and SCL lines. This feature should also be applied to improve the recognition of the repeated START bit when the SCL bus capacitance is very high. These kinds of events are not expected in a standard SMBus- or I2C-compliant system.

Note: In most systems this parameter should not be adjusted, and it is recommended that it be left at its default value.

The SDD field can be used to delay the recognition of the SDA falling edge by the SMBus hardware by 2, 4, or 8 SYSCLKs.

SMBus Control Register

SMB0CN0 is used to control the interface and to provide status information. The higher four bits of SMB0CN0 (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

STA and STO indicate that a START and/or STOP has been detected or generated since the last SMBus interrupt. STA and STO are also used to generate START and STOP conditions when operating as a master. Writing a 1 to STA will cause the SMBus interface to enter Master Mode and generate a START when the bus becomes free (STA is not cleared by hardware after the START is generated). Writing a 1 to STO while in Master Mode will cause the interface to generate a STOP and end the current transfer after the next ACK cycle. If STO and STA are both set (while in Master Mode), a STOP followed by a START will be generated.

The ARBLOST bit indicates that the interface has lost an arbitration. This may occur anytime the interface is transmitting (master or slave). A lost arbitration while operating as a slave indicates a bus error condition. ARBLOST is cleared by hardware each time SI is cleared.

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost.

Note: The SMBus interface is stalled while SI is set; if SCL is held low at this time, the bus is stalled until software clears SI.

Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 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 244 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 245 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.

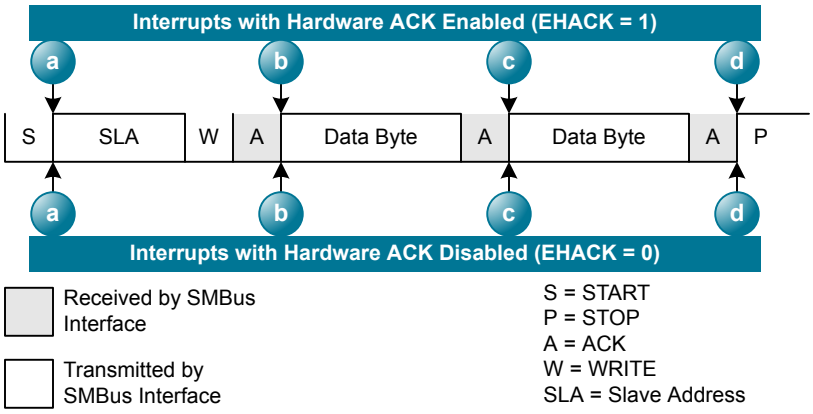


Figure 18.5. Typical Master Write Sequence

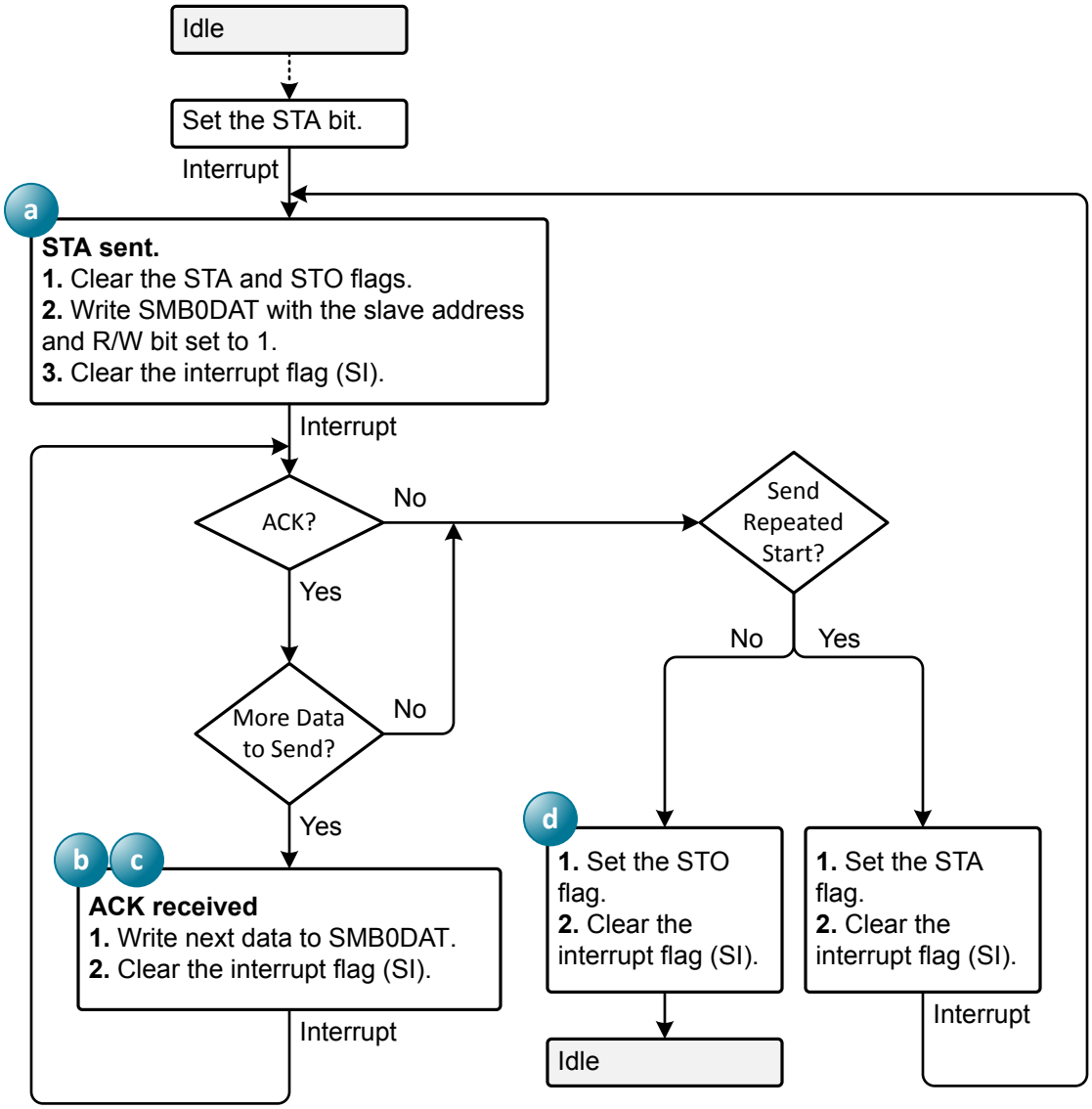


Figure 18.6. Master Write Sequence State Diagram (EHACK = 1)

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 246 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 247 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.

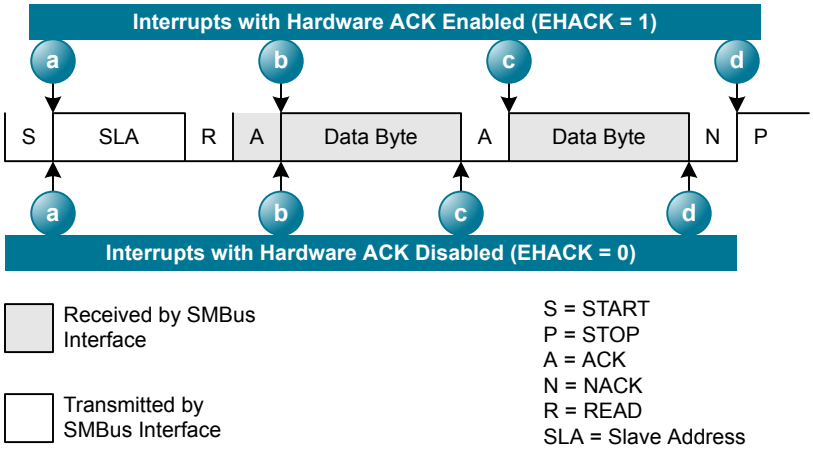


Figure 18.7. Typical Master Read Sequence

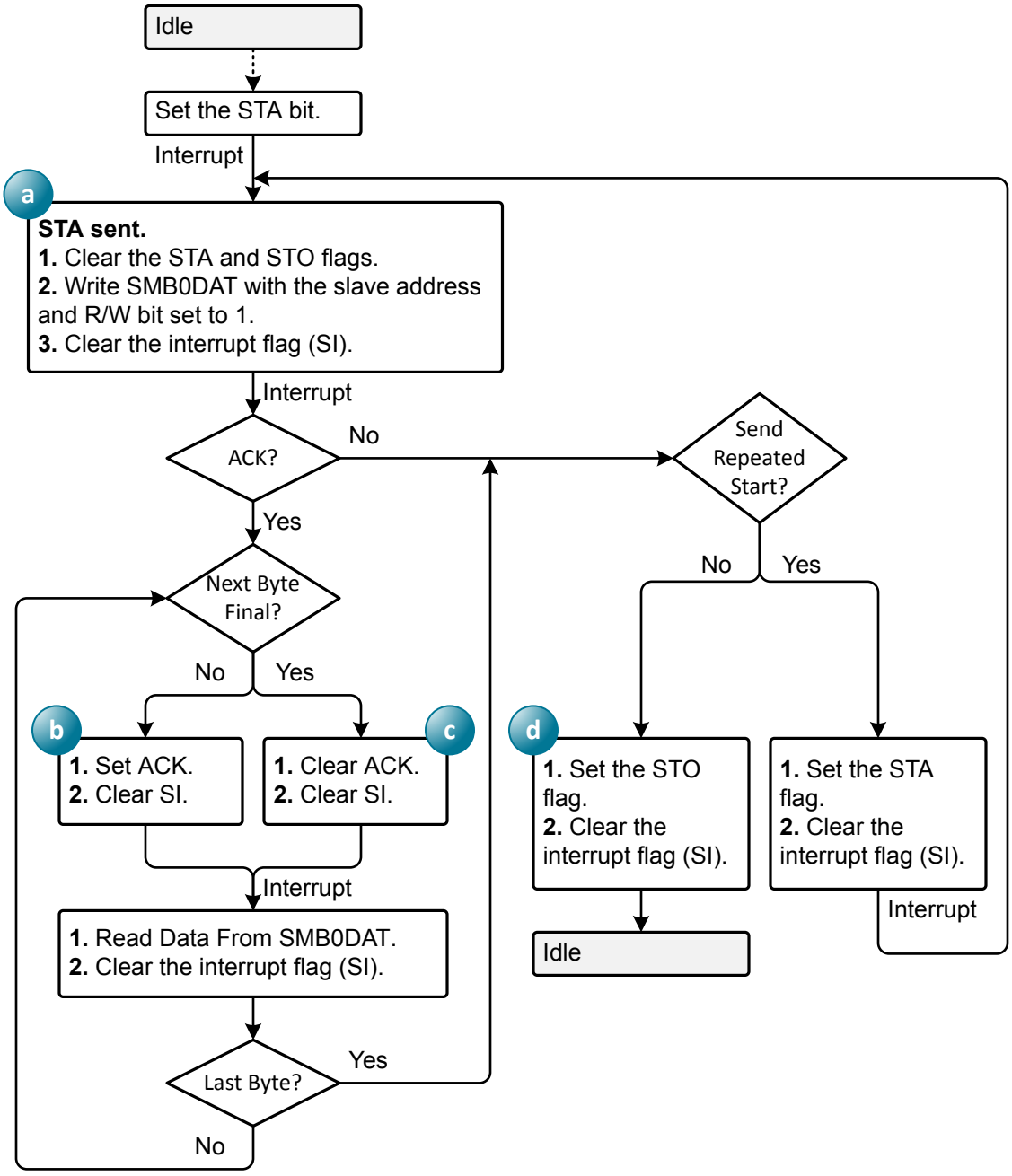


Figure 18.8. Master Read Sequence State Diagram (EHACK = 1)

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 248 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 249. 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.

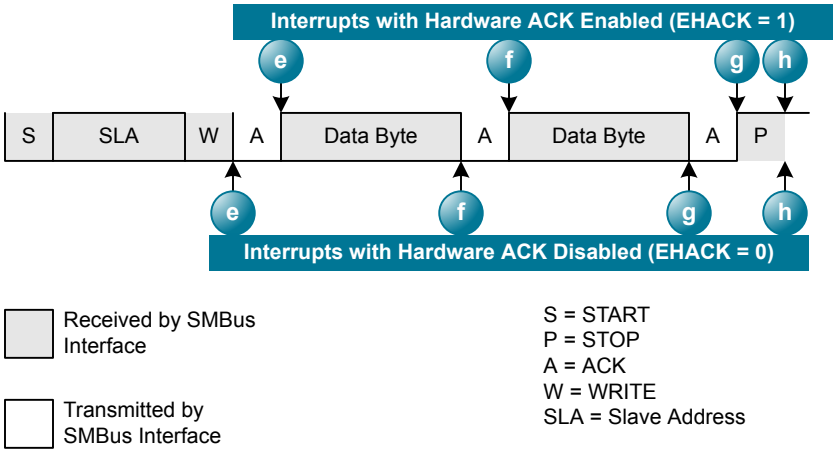


Figure 18.9. Typical Slave Write Sequence

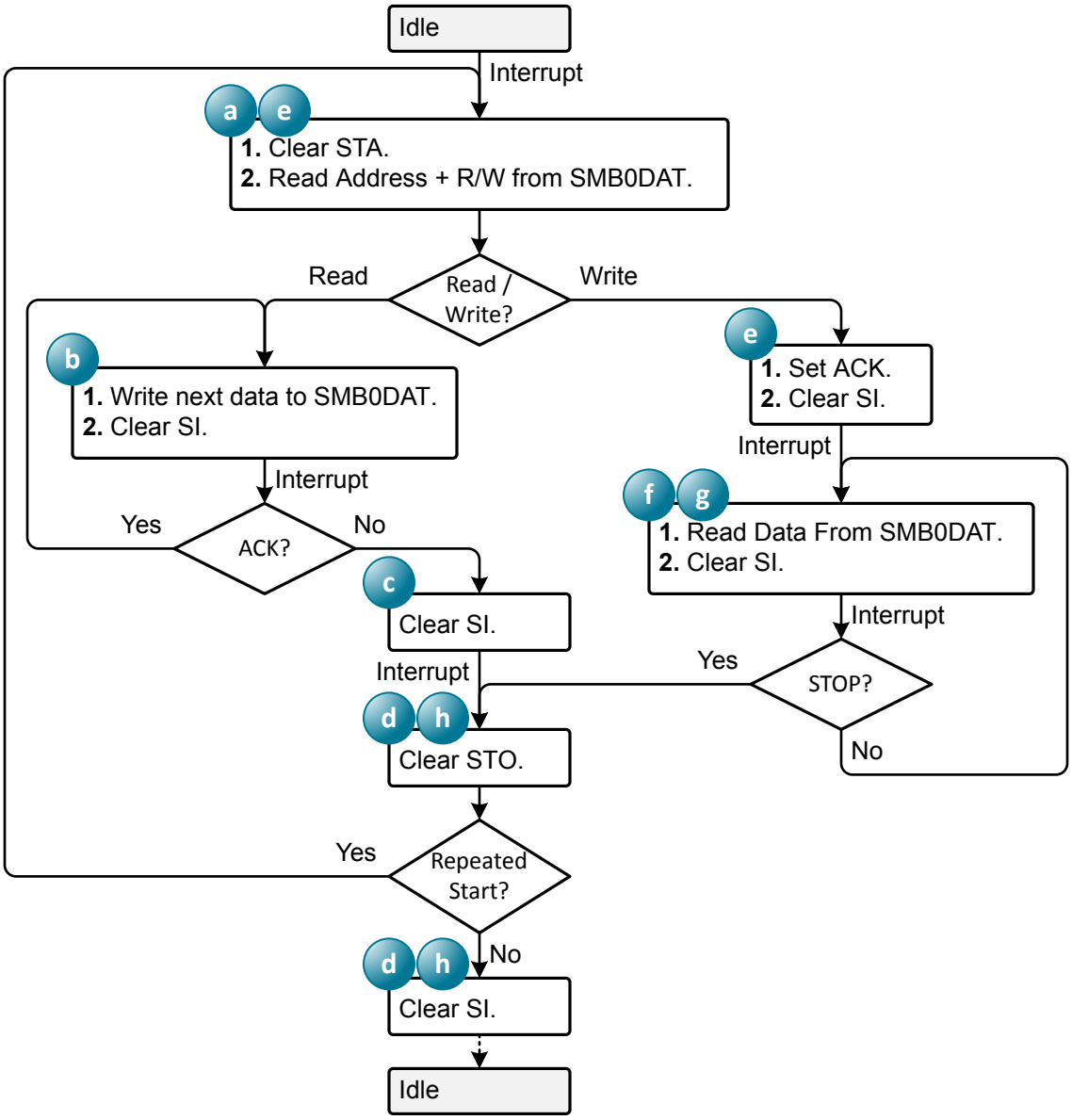


Figure 18.10. Slave State Diagram (EHACK = 1)

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 250 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 249. 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.

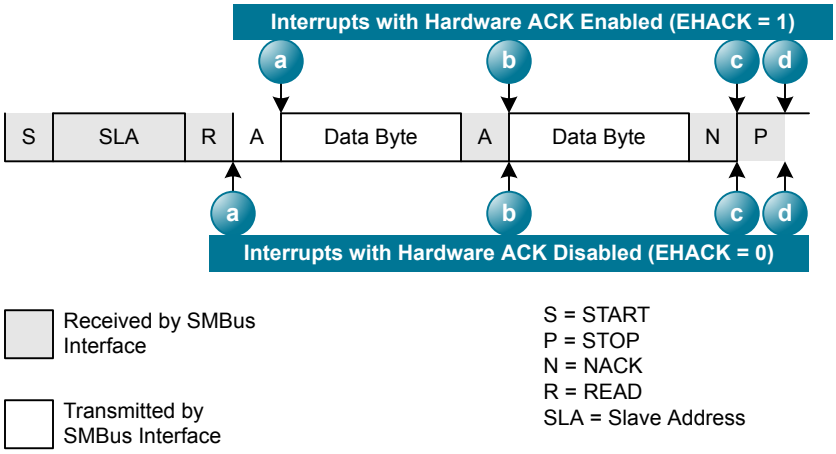


Figure 18.11. Typical Slave Read Sequence

18.4 SMBus Global Setup Registers

18.4.1 SMBTC: SMBus Timing and Pin Control

Bit	7	6	5	4	3	2	1	0
Name	Reserved				SMB1SDD		SMB0SDD	
Access	RW				RW		RW	
Reset	0x0				0x0		0x0	
SFR Page = 0xF; SFR Address: 0xB9								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3:2	SMB1SDD	0x0	RW	**SMBus 1 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 requirement (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.

Bit	Name	Reset	Access	Description
1:0	SMB0SDD	0x0	RW	**SMBus 0 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.

18.5 SMB0 Control Registers

18.5.1 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; 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.

18.5.2 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; SFR Address: 0xC0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	MASTER	0	R	**SMBus Master/Slave Indicator.** This read-only bit indicates when the SMBus is operating as a master.
	Value	Name		Description
	0	SLAVE		SMBus operating in slave mode.
	1	MASTER		SMBus operating in master mode.
6	TXMODE	0	R	**SMBus Transmit Mode Indicator.** This read-only bit indicates when the SMBus is operating as a transmitter.
	Value	Name		Description
	0	RECEIVER		SMBus in Receiver Mode.
	1	TRANSMITTER		SMBus in Transmitter Mode.
5	STA	0	RW	**SMBus Start Flag.** When reading STA, a '1' indicates that a start or repeated start condition was detected on the bus. Writing a '1' to the STA bit initiates a start or repeated start on the bus.
4	STO	0	RW	**SMBus Stop Flag.** When reading STO, a '1' indicates that a stop condition was detected on the bus (in slave mode) or is pending (in master mode). When acting as a master, writing a '1' to the STO bit initiates a stop condition on the bus. This bit is cleared by hardware.
3	ACKRQ	0	R	**SMBus Acknowledge Request.**
	Value	Name		Description
	0	NOT_SET		No ACK requested.
	1	REQUESTED		ACK requested.
2	ARBLOST	0	R	**SMBus Arbitration Lost Indicator.**
	Value	Name		Description
	0	NOT_SET		No arbitration error.
	1	ERROR		Arbitration error occurred.

Bit	Name	Reset	Access	Description
1	ACK	0	RW	**SMBus Acknowledge.** When read as a master, the ACK bit indicates whether an ACK (1) or NACK (0) is received during the most recent byte transfer. As a slave, this bit should be written to send an ACK (1) or NACK (0) to a master request. Note that the logic level of the ACK bit on the SMBus interface is inverted from the logic of the register ACK bit.
0	SI	0	RW	**SMBus Interrupt Flag.** This bit is set by hardware to indicate that the current SMBus state machine operation (such as writing a data or address byte) is complete, and the hardware needs additional control from the firmware to proceed. While SI is set, SCL is held low and SMBus is stalled. SI must be cleared by firmware. Clearing SI initiates the next SMBus state machine operation.

18.5.3 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; SFR Address: 0xCF								

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.

18.5.4 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; SFR Address: 0xCE								

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.

18.5.5 SMB0DAT: SMBus 0 Data

Bit	7	6	5	4	3	2	1	0
Name	SMB0DAT							
Access	RW							
Reset	0x00							
SFR Page = 0x0; SFR Address: 0xC2								

Bit	Name	Reset	Access	Description
7:0	SMB0DAT	0x00	RW	**SMBus 0 Data.** The SMB0DAT register contains a byte of data to be transmitted on the SMBus serial interface or a byte that has just been received on the SMBus serial interface. The CPU can safely read from or write to this register whenever the SI serial interrupt flag is set to logic 1. The serial data in the register remains stable as long as the SI flag is set. When the SI flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register.

18.6 SMB1 Control Registers

18.6.1 SMB1CF: SMBus 1 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 = 0xF; 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	TIMER5_HIGH	Timer 5 High Byte Overflow.
0x2	TIMER2_HIGH	Timer 2 High Byte Overflow.
0x3	TIMER2_LOW	Timer 2 Low Byte Overflow.

18.6.2 SMB1CN0: SMBus 1 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 = 0xF; SFR Address: 0xC0 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	MASTER	0	R	**SMBus Master/Slave Indicator.**
	This read-only bit indicates when the SMBus is operating as a master.			
	Value	Name		Description
	0	SLAVE		SMBus operating in slave mode.
	1	MASTER		SMBus operating in master mode.
6	TXMODE	0	R	**SMBus Transmit Mode Indicator.**
	This read-only bit indicates when the SMBus is operating as a transmitter.			
	Value	Name		Description
	0	RECEIVER		SMBus in Receiver Mode.
	1	TRANSMITTER		SMBus in Transmitter Mode.
5	STA	0	RW	**SMBus Start Flag.**
	When reading STA, a '1' indicates that a start or repeated start condition was detected on the bus.			
	Writing a '1' to the STA bit initiates a start or repeated start on the bus.			
4	STO	0	RW	**SMBus Stop Flag.**
	When reading STO, a '1' indicates that a stop condition was detected on the bus (in slave mode) or is pending (in master mode).			
	When acting as a master, writing a '1' to the STO bit initiates a stop condition on the bus. This bit is cleared by hardware.			
3	ACKRQ	0	R	**SMBus Acknowledge Request.**
	Value	Name		Description
	0	NOT_SET		No ACK requested.
	1	REQUESTED		ACK requested.
2	ARBLOST	0	R	**SMBus Arbitration Lost Indicator.**
	Value	Name		Description
	0	NOT_SET		No arbitration error.
	1	ERROR		Arbitration error occurred.

Bit	Name	Reset	Access	Description
1	ACK	0	RW	**SMBus Acknowledge.** When read as a master, the ACK bit indicates whether an ACK (1) or NACK (0) is received during the most recent byte transfer. As a slave, this bit should be written to send an ACK (1) or NACK (0) to a master request. Note that the logic level of the ACK bit on the SMBus interface is inverted from the logic of the register ACK bit.
0	SI	0	RW	**SMBus Interrupt Flag.** This bit is set by hardware to indicate that the current SMBus state machine operation (such as writing a data or address byte) is complete, and the hardware needs additional control from the firmware to proceed. While SI is set, SCL is held low and SMBus is stalled. SI must be cleared by firmware. Clearing SI initiates the next SMBus state machine operation.

18.6.3 SMB1ADR: SMBus 1 Slave Address

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

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.

18.6.4 SMB1ADM: SMBus 1 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM							EHACK
Access	RW							RW
Reset	0x7F							0
SFR Page = 0xF; SFR Address: 0xCE								

Bit	Name	Reset	Access	Description
7:1	SLVM	0x7F	RW	**SMBus Slave Address Mask.** Defines which bits of register SMB1ADR 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_AUTOMATIC	Automatic slave address recognition and hardware acknowledge is enabled.

18.6.5 SMB1DAT: SMBus 1 Data

Bit	7	6	5	4	3	2	1	0
Name	SMB1DAT							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0xC2								

Bit	Name	Reset	Access	Description
7:0	SMB1DAT	0x00	RW	**SMBus 1 Data.** The SMB1DAT register contains a byte of data to be transmitted on the SMBus serial interface or a byte that has just been received on the SMBus serial interface. The CPU can safely read from or write to this register whenever the SI serial interrupt flag is set to logic 1. The serial data in the register remains stable as long as the SI flag is set. When the SI flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register.

19. Timers (Timer0, Timer1, Timer2, Timer3, Timer4, and Timer5)

19.1 Introduction

Six counter/timers ar included in the device: two are 16-bit counter/timers compatible with those found in the standard 8051, and four are 16-bit auto-reload timers for timing peripherals or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2, Timer 3, Timer 4, and Timer 5 are also identical and offer both 16-bit and split 8-bit timer functionality with auto-reload capabilities. Timer 2 and Timer 3 are capable of performing a capture function on the low-frequency oscillator output. Timer 4 and Timer 5 do not support a capture function.

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. Timer 2–5 may be clocked by the system clock, system clock divided by 12, or external oscillator divided by 8.

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 and Timer 3 Modes	Timer 4 and Timer 5 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	LFOSC0 edge or USB SOF capture	
Two 8-bit counter/timers (Timer 0 only)		

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

- Clock sources include SYSCLK, SYSCLK divided by 12, or the External Clock divided by 8.
- 16-bit auto-reload timer mode
- Dual 8-bit auto-reload timer mode
- USB start-of-frame or falling edge of LFOSC0 capture (Timer 2 and Timer 3)

19.3 Functional Description

19.3.1 System Connections

All four 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 and Timer 3 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	T3 High Overflow	T3 Low Overflow	T4 High Overflow	T4 Low Overflow	T5 High Overflow	T5 Low Overflow
UART0 Baud Rate		Yes								
SMBus 0 Clock Rate (Master)	Yes	Yes	Yes	Yes						
SMBus 0 SCL Low Timeout					Yes					
SMBus 1 Clock Rate (Master)	Yes		Yes	Yes					Yes	
SMBus 1 SCL Low Timeout							Yes			
PCA0 Clock	Yes									
ADC0 Conversion Start	Yes		Yes[1]	Yes[1]	Yes[1]	Yes[1]	Yes[1]	Yes[1]	Yes[1]	Yes[1]

Notes:
1. The high-side overflow is used when the timer is in 16-bit mode. The low-side overflow is used in 8-bit mode.

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:

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

The CT0 bit in the TMOD register selects the counter/timer's clock source. When CT0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register. Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it must be held at a given level for at least two full system clock cycles to ensure the level is properly sampled. Clearing CT selects the clock defined by the T0M bit in register CKCON0. When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON0.

Setting the TR0 bit enables the timer when either GATE0 in the TMOD register is logic 0 or based on the input signal INT0. The IN0PL bit setting in IT01CF changes which state of INT0 input starts the timer counting. Setting GATE0 to 1 allows the timer to be controlled by the external input signal INT0, facilitating pulse width measurements.

Table 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.

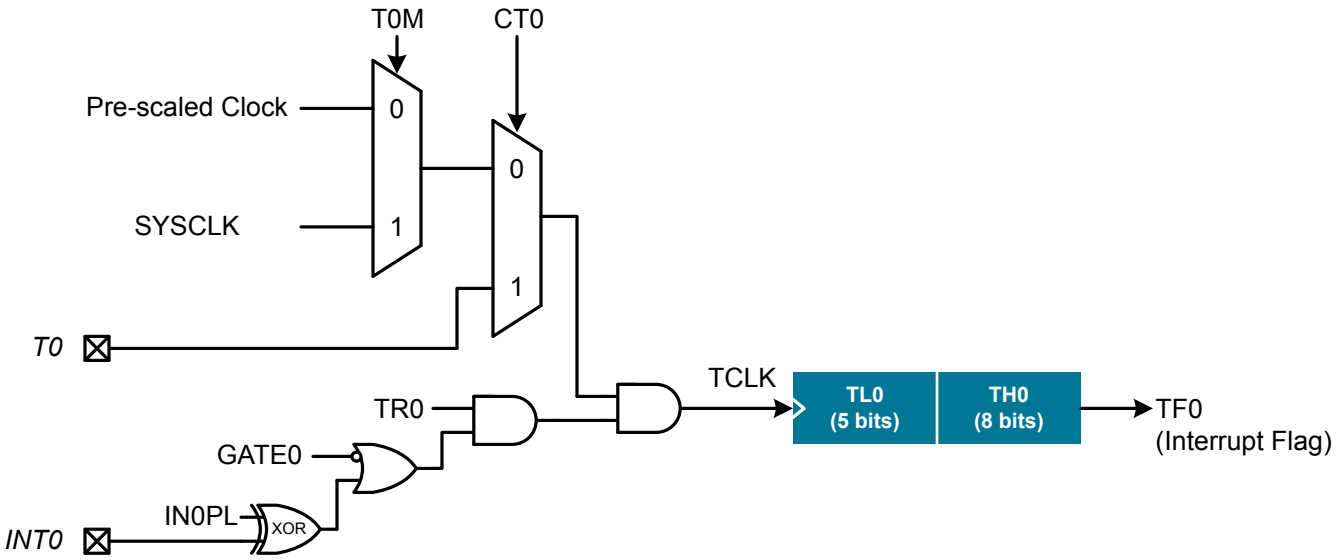


Figure 19.1. T0 Mode 0 Block Diagram

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.

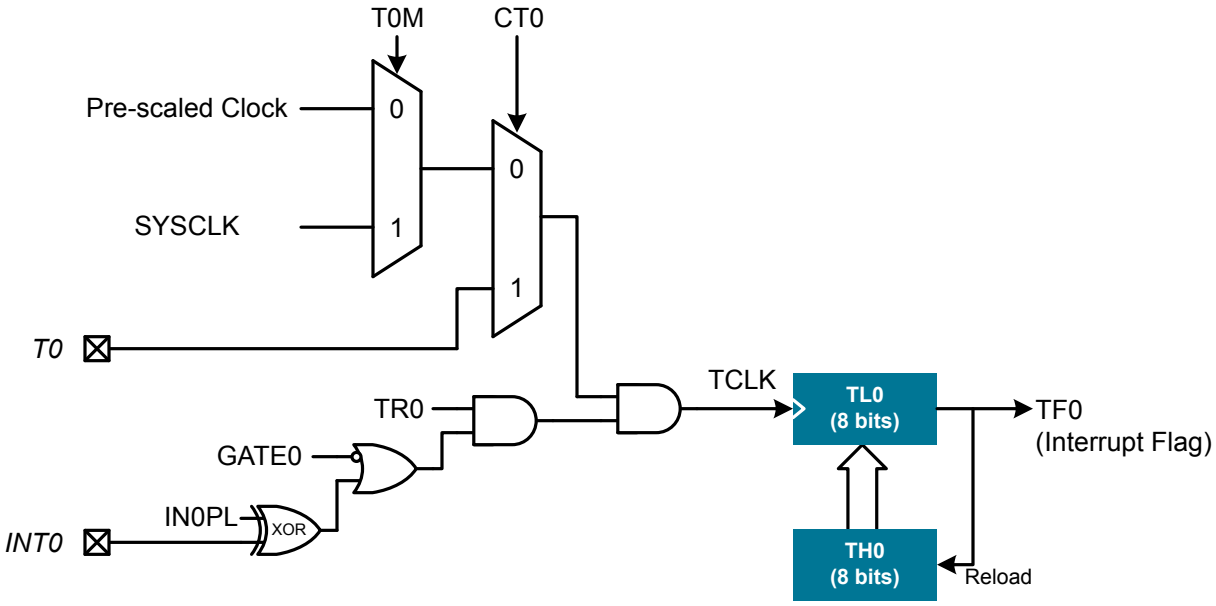


Figure 19.2. T0 Mode 2 Block Diagram

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.

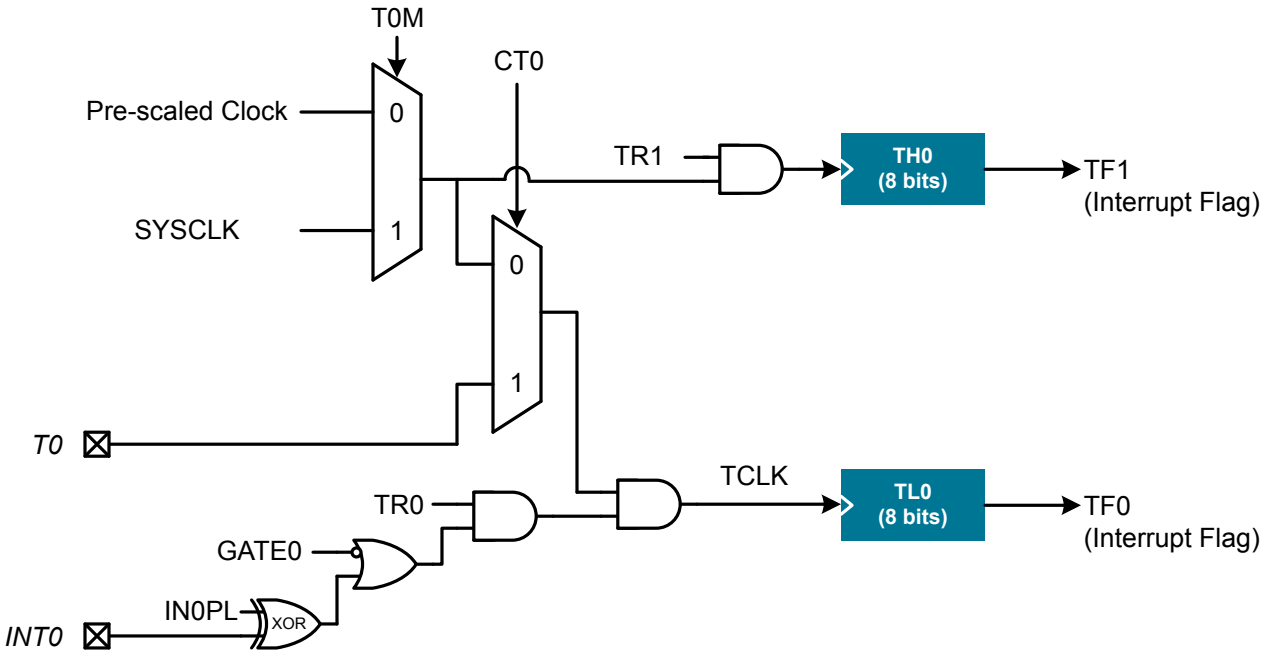


Figure 19.3. T0 Mode 3 Block Diagram

19.3.3 Timer 2, Timer 3, Timer 4, and Timer 5

Timer 2, Timer 3, Timer 4, and Timer 5 are functionally equivalent, with the only differences being the top-level connections to other parts of the system.

The timers are 16 bits wide, formed by two 8-bit SFRs: TMRnL (low byte) and TMRnH (high byte). Each timer may operate in 16-bit auto-reload mode, dual 8-bit auto-reload (split) mode, or capture mode.

Clock Selection

Clocking for each timer is configured using the TnXCLK bit field and the TnML and TnMH bits. Timer 2–5 may be clocked by the system clock, system clock divided by 12, or external oscillator divided by 8 (synchronized with SYSCLK). The maximum frequency for the external oscillator is:

$$F_{SYSCLK} > F_{EXTOSC} \times \frac{6}{7}$$

When operating in one of the 16-bit modes, the low-side timer clock is used to clock the entire 16-bit timer.

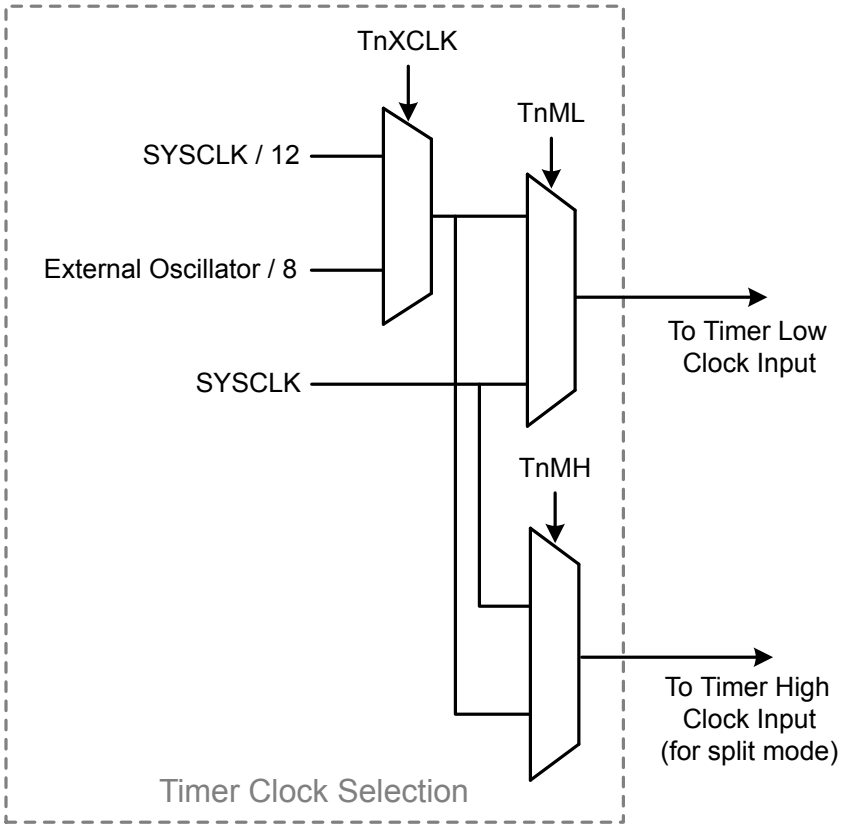


Figure 19.4. Timer 2 and 3 Clock Source Selection

Capture Sources

Capture mode allows an input to be measured against the selected clock source. Timer 2 and Timer 3 are capable of performing a capture function on the low-frequency oscillator output. Timer 4 and Timer 5 do not support a capture function.

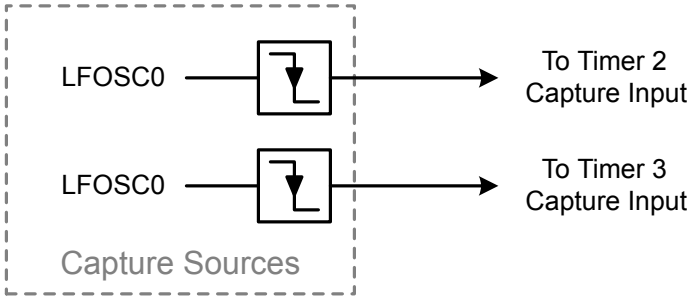


Figure 19.5. Timer 2 and 3 Capture Sources

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:

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

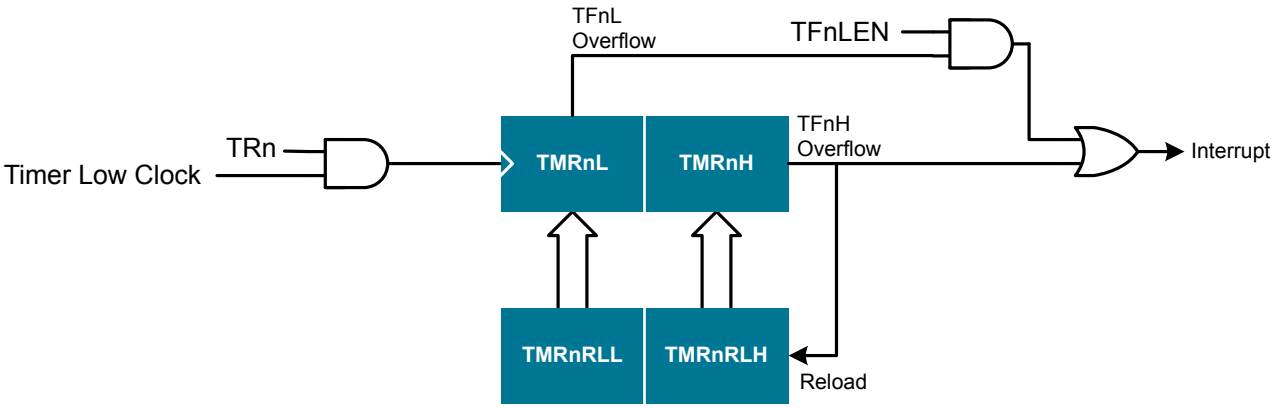


Figure 19.6. 16-Bit Mode Block Diagram

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.

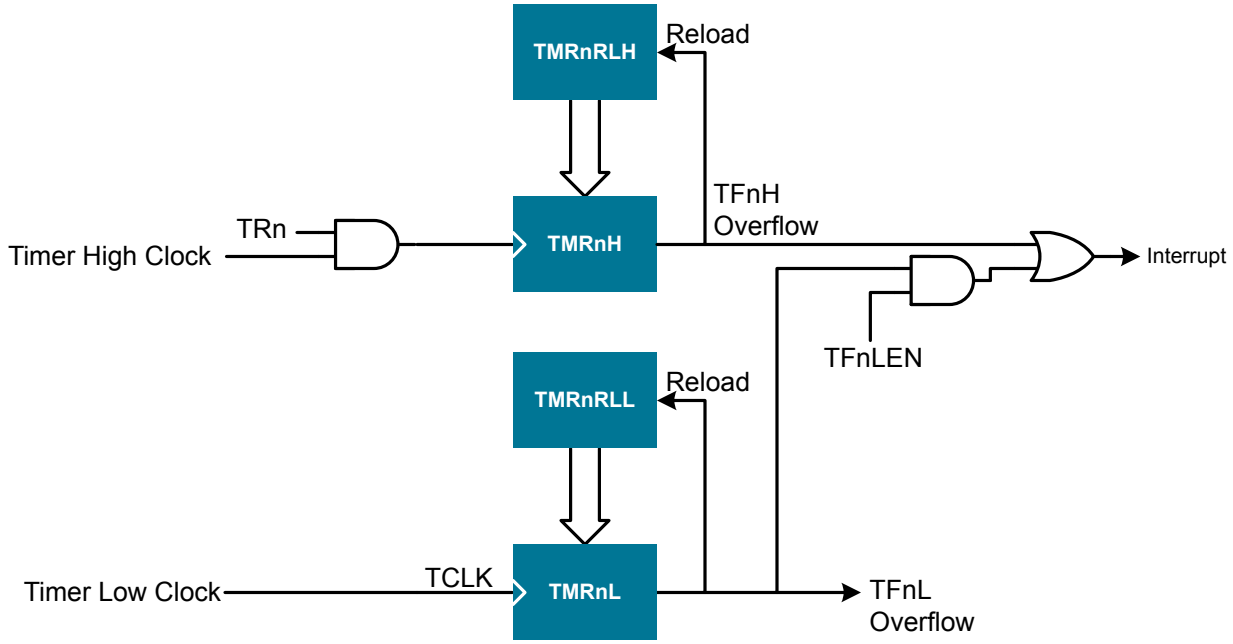


Figure 19.7. 8-Bit Split Mode Block Diagram

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.

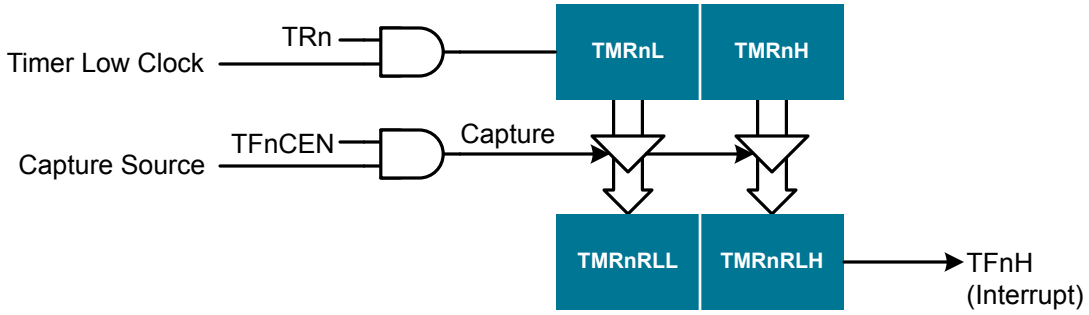


Figure 19.8. Capture Mode Block Diagram

19.4 Timer 0, 1, 2, 3, 4, and 5 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.

Bit	Name	Reset	Access	Description
1:0	SCA	0x0	RW	**Timer 0/1 Prescale.**

These bits control the Timer 0/1 Clock Prescaler:

Value	Name	Description
0x0	SYSCLK_DIV_12	System clock divided by 12.
0x1	SYSCLK_DIV_4	System clock divided by 4.
0x2	SYSCLK_DIV_48	System clock divided by 48.
0x3	EXTOSC_DIV_8	External oscillator divided by 8 (synchronized with the system clock).

19.4.2 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.

19.4.3 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

19.4.4 CKCON1: Clock Control 1

Bit	7	6	5	4	3	2	1	0
Name	Reserved				T5MH	T5ML	T4MH	T4ML
Access	R				RW	RW	RW	RW
Reset	0x0				0	0	0	0
SFR Page = 0xF; SFR Address: 0xE4								

Bit	Name	Reset	Access	Description
7:4	*Reserved*	*Must write reset value.*		
3	T5MH	0	RW	**Timer 5 High Byte Clock Select.**

Selects the clock supplied to the Timer 5 high byte (split 8-bit timer mode only).

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 5 high byte uses the clock defined by T5XCLK in TMR5CN.
1	SYSCLK	Timer 5 high byte uses the system clock.

Bit	Name	Reset	Access	Description
2	T5ML	0	RW	**Timer 5 Low Byte Clock Select.**

Selects the clock supplied to Timer 5. If Timer 5 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 5 low byte uses the clock defined by T5XCLK in TMR5CN.
1	SYSCLK	Timer 5 low byte uses the system clock.

Bit	Name	Reset	Access	Description
1	T4MH	0	RW	**Timer 4 High Byte Clock Select.**

Selects the clock supplied to the Timer 4 high byte (split 8-bit timer mode only).

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 4 high byte uses the clock defined by T4XCLK in TMR4CN0.
1	SYSCLK	Timer 4 high byte uses the system clock.

Bit	Name	Reset	Access	Description
0	T4ML	0	RW	**Timer 4 Low Byte Clock Select.**

Selects the clock supplied to Timer 4. If Timer 4 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.

Value	Name	Description
0	EXTERNAL_CLOCK	Timer 4 low byte uses the clock defined by T4XCLK in TMR4CN0.
1	SYSCLK	Timer 4 low byte uses the system clock.

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	T2CSS	T2XCLK
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0; 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 based on the selected input capture source, 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	T2CSS	0	RW	**Timer 2 Capture Source Select.** This bit selects the source of a capture event when bit TF2CEN is set to 1.

Value	Name	Description
0	USB_SOF_CAPTURE	Capture source is USB SOF event.
1	LFOSC_CAPTURE	Capture source is falling edge of Low-Frequency Oscillator.

Bit	Name	Reset	Access	Description
0	T2XCLK	0	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
0	SYSCLK_DIV_12	Timer 2 clock is the system clock divided by 12.

Bit	Name	Reset	Access	Description
	1	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; 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; 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; 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; 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 TMR3CN0: Timer 3 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF3H	TF3L	TF3LEN	TF3CEN	T3SPLIT	TR3	T3CSS	T3XCLK
Access	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0x0; 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 based on the selected input capture source, 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	T3CSS	0	RW	**Timer 3 Capture Source Select.** This bit selects the source of a capture event when bit TF3CEN is set to 1.

Value	Name	Description
0	USB_SOF_CAPTURE	Capture source is USB SOF event.
1	LFOSC_CAPTURE	Capture source is falling edge of Low-Frequency Oscillator.

Bit	Name	Reset	Access	Description
0	T3XCLK	0	RW	**Timer 3 External Clock Select.** T3XCLK selects the external clock source for Timer 3. If Timer 3 is in 8-bit mode, T3XCLK 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
0	SYSCLK_DIV_12	Timer 3 clock is the system clock divided by 12.

Bit	Name	Reset	Access	Description
	1	EXTOSC_DIV_8		Timer 3 clock is the external oscillator divided by 8 (synchronized with SYSCLK).

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; 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; 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; 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; 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 TMR4CN0: Timer 4 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF4H	TF4L	TF4LEN	Reserved	T4SPLIT	TR4	Reserved	T4XCLK
Access	RW	RW	RW	R	RW	RW	R	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0xF; SFR Address: 0x91								

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	*Reserved*	*Must write reset value.*		
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.

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	*Reserved*	*Must write reset value.*		
0	T4XCLK	0	RW	**Timer 4 External Clock Select.** T4XCLK selects the external clock source for Timer 4. If Timer 4 is in 8-bit mode, T4XCLK 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
0	SYSCLK_DIV_12	Timer 4 clock is the system clock divided by 12.
1	EXTOSC_DIV_8	Timer 4 clock is the external oscillator divided by 8 (synchronized with SYSCLK).

19.4.20 TMR4RLL: Timer 4 Reload Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLL							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0x92								

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.

19.4.21 TMR4RLH: Timer 4 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLH							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0x93								

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.

19.4.22 TMR4L: Timer 4 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4L							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0x94								

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.

19.4.23 TMR4H: Timer 4 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4H							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0x95								

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.

19.4.24 TMR5CN0: Timer 5 Control 0

Bit	7	6	5	4	3	2	1	0
Name	TF5H	TF5L	TF5LEN	Reserved	T5SPLIT	TR5	Reserved	T5XCLK
Access	RW	RW	RW	R	RW	RW	R	RW
Reset	0	0	0	0	0	0	0	0
SFR Page = 0xF; SFR Address: 0xC8 (bit-addressable)								

Bit	Name	Reset	Access	Description
7	TF5H	0	RW	**Timer 5 High Byte Overflow Flag.** Set by hardware when the Timer 5 high byte overflows from 0xFF to 0x00. In 16-bit mode, this will occur when Timer 5 overflows from 0xFFFF to 0x0000. When the Timer 5 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 5 interrupt service routine. This bit must be cleared by firmware.
6	TF5L	0	RW	**Timer 5 Low Byte Overflow Flag.** Set by hardware when the Timer 5 low byte overflows from 0xFF to 0x00. TF5L will be set when the low byte overflows regardless of the Timer 5 mode. This bit must be cleared by firmware.
5	TF5LEN	0	RW	**Timer 5 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 5 Low Byte interrupts. If Timer 5 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 5 overflows.
4	*Reserved*	*Must write reset value.*		
3	T5SPLIT	0	RW	**Timer 5 Split Mode Enable.** When this bit is set, Timer 5 operates as two 8-bit timers with auto-reload.

Value	Name	Description
0	16_BIT_RELOAD	Timer 5 operates in 16-bit auto-reload mode.
1	8_BIT_RELOAD	Timer 5 operates as two 8-bit auto-reload timers.

Bit	Name	Reset	Access	Description
2	TR5	0	RW	**Timer 5 Run Control.** Timer 5 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR5H only; TMR5L is always enabled in split mode.
1	*Reserved*	*Must write reset value.*		
0	T5XCLK	0	RW	**Timer 5 External Clock Select.** T5XCLK selects the external clock source for Timer 5. If Timer 5 is in 8-bit mode, T5XCLK selects the external oscillator clock source for both timer bytes. However, the Timer 5 Clock Select bits (T5MH and T5ML) may still be used to select between the external clock and the system clock for either timer.

Value	Name	Description
0	SYSCLK_DIV_12	Timer 5 clock is the system clock divided by 12.
1	EXTOSC_DIV_8	Timer 5 clock is the external oscillator divided by 8 (synchronized with SYSCLK).

19.4.25 TMR5RLL: Timer 5 Reload Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLL							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0xCA								

Bit	Name	Reset	Access	Description
7:0	TMR5RLL	0x00	RW	**Timer 5 Reload Low Byte.** When operating in one of the auto-reload modes, TMR5RLL holds the reload value for the low byte of Timer 5 (TMR5L). When operating in capture mode, TMR5RLL is the captured value of TMR5L.

19.4.26 TMR5RLH: Timer 5 Reload High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLH							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0xCB								

Bit	Name	Reset	Access	Description
7:0	TMR5RLH	0x00	RW	**Timer 5 Reload High Byte.** When operating in one of the auto-reload modes, TMR5RLH holds the reload value for the high byte of Timer 5 (TMR5H). When operating in capture mode, TMR5RLH is the captured value of TMR5H.

19.4.27 TMR5L: Timer 5 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5L							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0xCC								

Bit	Name	Reset	Access	Description
7:0	TMR5L	0x00	RW	**Timer 5 Low Byte.** In 16-bit mode, the TMR5L register contains the low byte of the 16-bit Timer 5. In 8-bit mode, TMR5L contains the 8-bit low byte timer value.

19.4.28 TMR5H: Timer 5 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5H							
Access	RW							
Reset	0x00							
SFR Page = 0xF; SFR Address: 0xCD								

Bit	Name	Reset	Access	Description
7:0	TMR5H	0x00	RW	**Timer 5 High Byte.** In 16-bit mode, the TMR5H register contains the high byte of the 16-bit Timer 5. In 8-bit mode, TMR5H contains the 8-bit high byte timer value.

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).

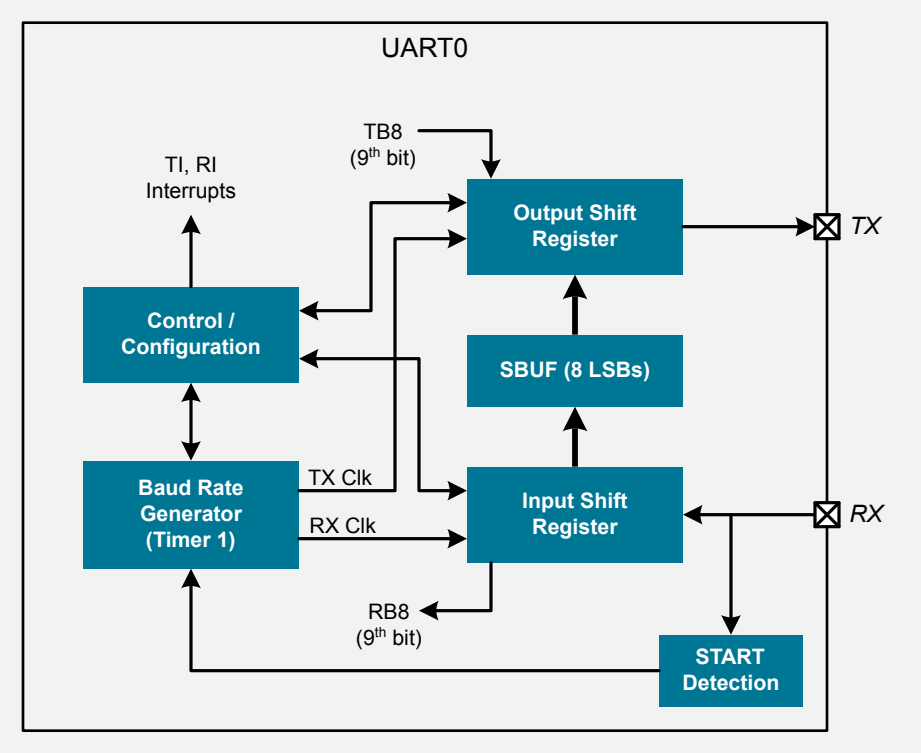


Figure 20.1. UART0 Block Diagram

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.
- Single-byte FIFO on transmit and receive.

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.

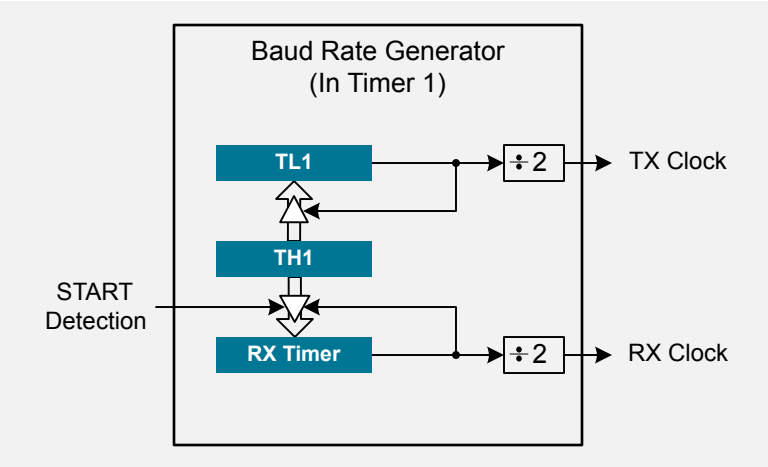


Figure 20.2. UART0 Baud Rate Logic Block Diagram

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.

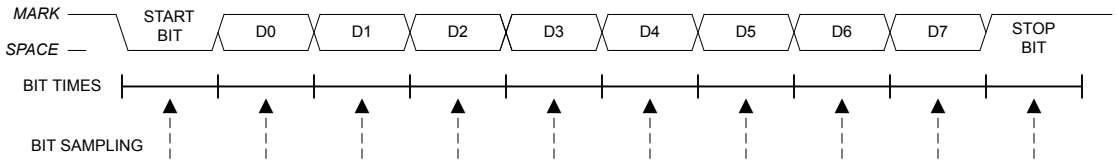


Figure 20.3. 8-Bit Data Transfer

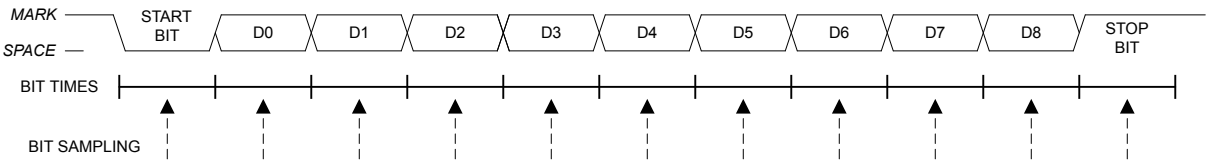


Figure 20.4. 9-Bit Data Transfer

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).

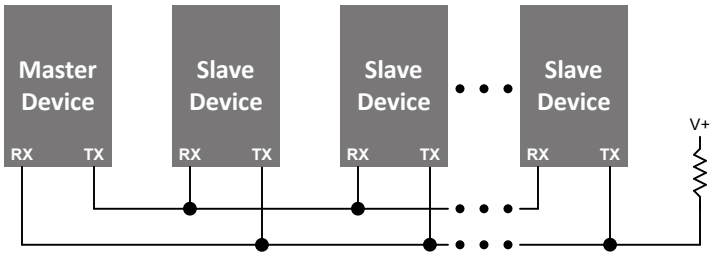


Figure 20.5. Multi-Processor Mode Interconnect Diagram

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	RW	RW	RW
Reset	0	1	0	0	0	0	0	0

SFR Page = ALL; SFR Address: 0x98 (bit-addressable)

Bit	Name	Reset	Access	Description
7	SMODE	0	RW	**Serial Port 0 Operation Mode.**

Selects the UART0 Operation Mode.

Value	Name	Description
0	8_BIT	8-bit UART with Variable Baud Rate (Mode 0).
1	9_BIT	9-bit UART with Variable Baud Rate (Mode 1).

6	*Reserved*	*Must write reset value.*		
5	MCE	0	RW	**Multiprocessor Communication Enable.**

This bit enables checking of the stop bit or the 9th bit in multi-drop communication buses. The function of this bit is dependent on the UART0 operation mode selected by the SMODE bit. In Mode 0 (8-bits), the peripheral will check that the stop bit is logic 1. In Mode 1 (9-bits) the peripheral will check for a logic 1 on the 9th bit.

Value	Name	Description
0	MULTI_DISABLED	Ignore level of 9th bit / Stop bit.
1	MULTI_ENABLED	RI is set and an interrupt is generated only when the stop bit is logic 1 (Mode 0) or when the 9th bit is logic 1 (Mode 1).

4	REN	0	RW	**Receive Enable.**

Value	Name	Description
0	RECEIVE_DISABLED	UART0 reception disabled.
1	RECEIVE_ENABLED	UART0 reception enabled.

3	TB8	0	RW	**Ninth Transmission Bit.**

The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0).

2	RB8	0	RW	**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 by hardware when a byte of data has been transmitted by UART0 (after the 8th bit in 8-bit UART Mode, or at the beginning of the STOP bit in 9-bit UART Mode). When the UART0 interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by firmware.

0	RI	0	RW	**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). When the UART0 interrupt is enabled, setting this bit to 1 causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by firmware.

20.4.2 SBUF0: UART0 Serial Port Data Buffer

Bit	7	6	5	4	3	2	1	0
Name	SBUF0							
Access	RW							
Reset	0x00							
SFR Page = ALL; SFR Address: 0x99								

Bit	Name	Reset	Access	Description
7:0	SBUF0	0x00	RW	**Serial Data Buffer.** This SFR accesses two registers: a transmit shift register and a receive latch register. When data is written to SBUF0, it goes to the transmit shift register and is held for serial transmission. Writing a byte to SBUF0 initiates the transmission. A read of SBUF0 returns the contents of the receive latch.

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.

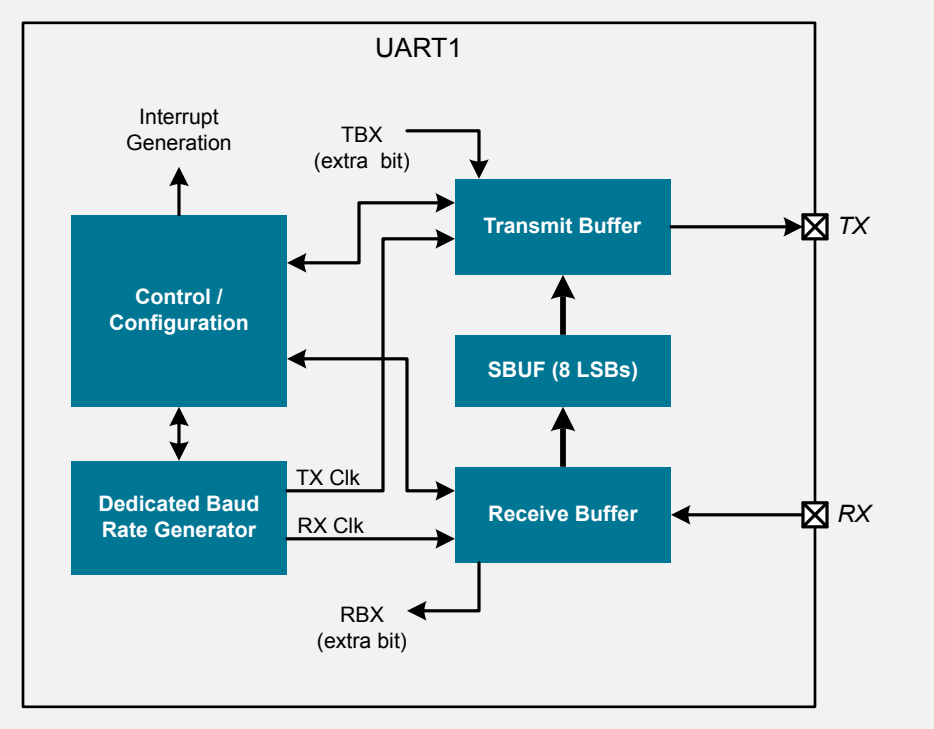


Figure 21.1. UART 1 Block Diagram

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.
- Three byte FIFO on receive.

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.

$$\text{Baud Rate} = \frac{\text{SYSCLK}}{(65536 - (\text{SBRL1})) \times 2 \times \text{Prescaler}}$$

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.

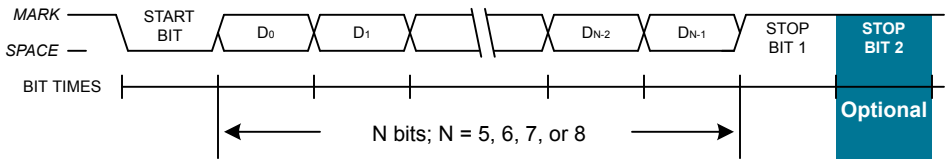


Figure 21.2. UART1 Timing Without Parity or Extra Bit

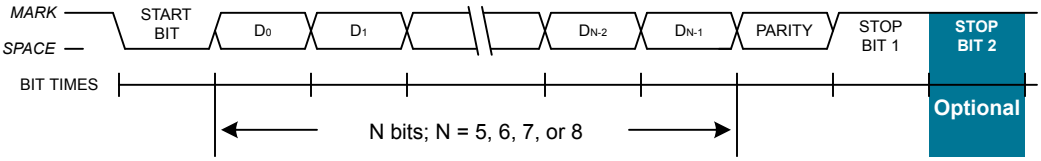


Figure 21.3. UART1 Timing With Parity

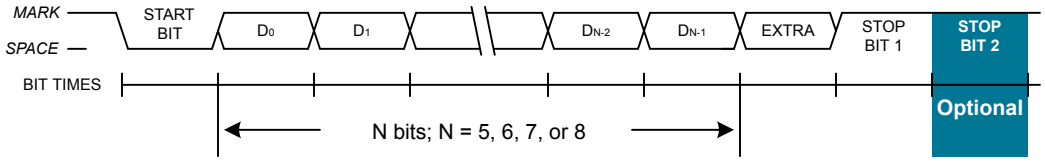


Figure 21.4. UART1 Timing With Extra Bit

21.3.3 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.

21.3.4 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).

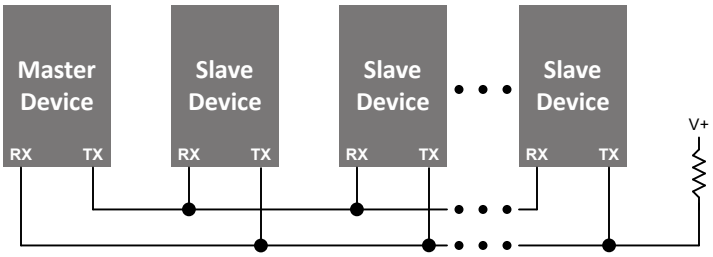


Figure 21.5. Multi-Processor Mode Interconnect Diagram

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	Varies	0	0	0	0	0
SFR Page = ALL; SFR Address: 0xD2								

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	0	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 = ALL; SFR Address: 0xE5

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.
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.
4	PE	0	RW	**Parity Enable.**
	This bit activates hardware parity generation and checking. The parity type is selected by the SPT field when parity is enabled.			
	Value	Name		Description
	0	PARITY_DISABLED		Disable hardware parity.
	1	PARITY_ENABLED		Enable hardware parity.
3:2	SDL	0x3	RW	**Data Length.**
	Value	Name		Description
	0x0	5_BITS		5 bits.
	0x1	6_BITS		6 bits.
	0x2	7_BITS		7 bits.
	0x3	8_BITS		8 bits.
1	XBE	0	RW	**Extra Bit Enable.**
	When enabled, the value of TBX in the SCON1 register will be appended to the data field.			
	Value	Name		Description
	0	DISABLED		Disable the extra bit.

Bit	Name	Reset	Access	Description
	1	ENABLED		Enable the extra bit.
0	SBL	0	RW	**Stop Bit Length.**
	Value	Name		Description
	0	SHORT		Short: Stop bit is active for one bit time.
	1	LONG		Long: Stop bit is active for two bit times (data length = 6, 7, or 8 bits) or 1.5 bit times (data length = 5 bits).

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 = ALL; SFR Address: 0xD3								

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, 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 will overwrite 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 = ALL; SFR Address: 0xAC								

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.
5:2	*Reserved*	*Must write reset value.*		
1:0	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.

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 = ALL; SFR Address: 0xB5								

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 = ALL; SFR Address: 0xB4								

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))

22. C2 Debug and Programming Interface

22.1 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.

22.2 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.

22.3 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.

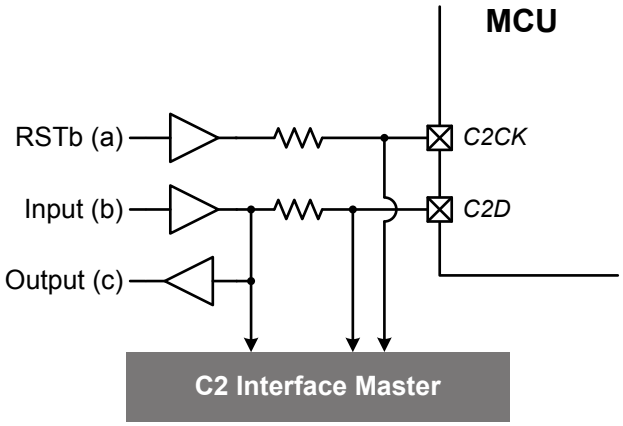


Figure 22.1. Typical C2 Pin Sharing

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.

22.4 C2 Interface Registers

22.4.1 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

22.4.2 C2DEVID: C2 Device ID

Bit	7	6	5	4	3	2	1	0
Name	C2DEVID							
Access	R							
Reset	0x28							
C2 Address: 0x00								

Bit	Name	Reset	Access	Description
7:0	C2DEVID	0x28	R	**Device ID.** This read-only register returns the 8-bit device ID: 0x28 (EFM8UB2).

22.4.3 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.

22.4.4 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.

22.4.5 C2FPDAT: C2 Flash Programming Data

Bit	7	6	5	4	3	2	1	0
Name	C2FPDAT							
Access	RW							
Reset	0x00							
C2 Address: 0xAD								

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

23. Revision History

Revision 0.3

March, 2019

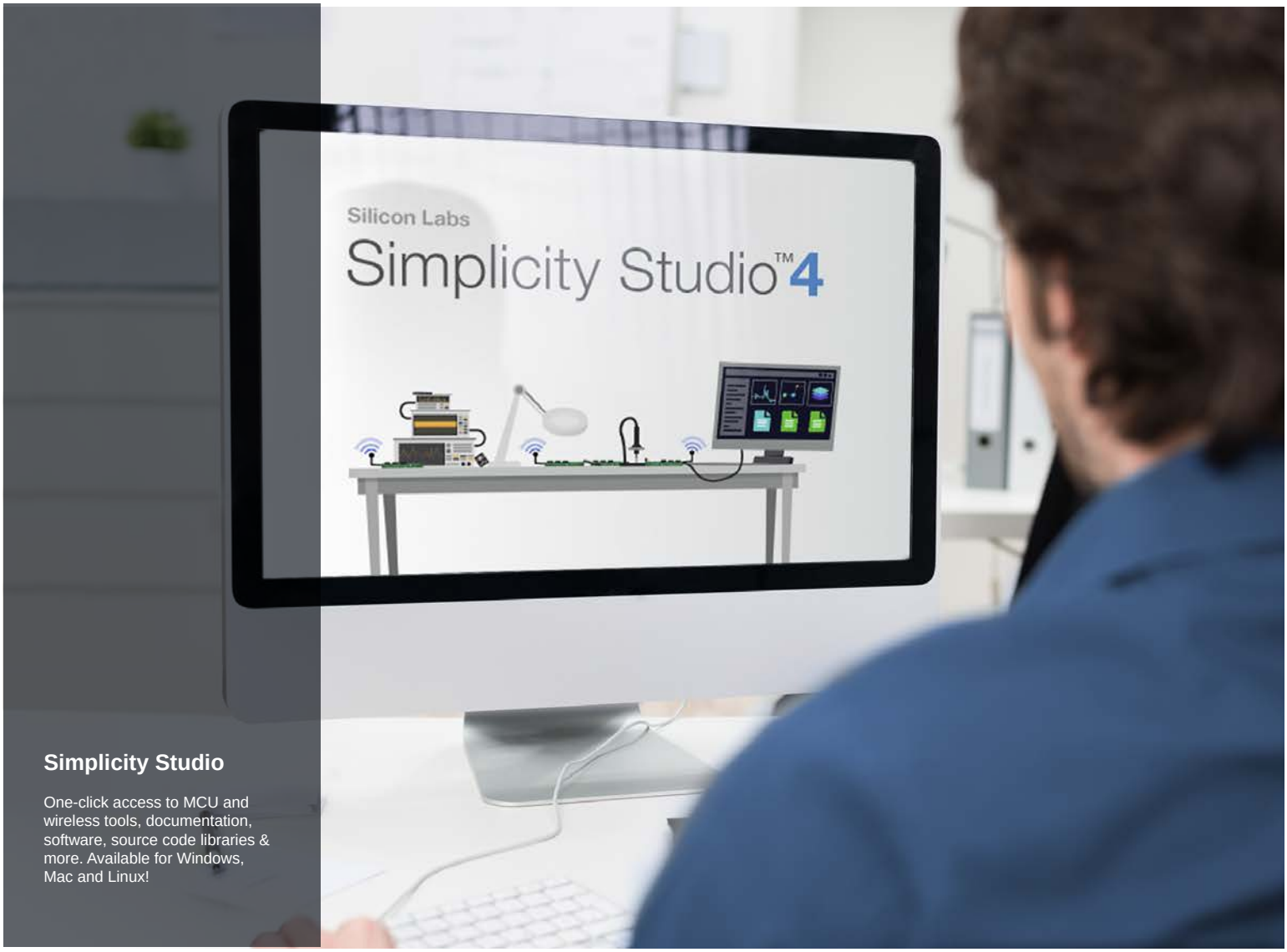
- In 1.10 Bootloader, expanded this section and added a recommendation for code security.
- In Figure 1.1 Detailed EFM8UB2 Block Diagram on page 11, updated RAM size in the core to not double count USB RAM.
- In 1.2 Power and 7.1 Introduction, added information regarding stop mode.
- In 1.6 Communications and Other Digital Peripherals and 20.2 Features, added information regarding UART0 FIFO.
- In 2.4 Memory Map and Flash Memory, updated figures with bootloader information.
- In 4.3.1 Security Options, added a recommendation for code security.
- In Table 4.2 Flash Security Summary—Firmware Permissions on page 34, added read-only area and removed user page.
- In Table 4.3 Flash Security Summary—C2 Permissions on page 35, added read-only area.
- In 4.3.3 Flash Write and Erase Precautions, under System Clock, replaced references to crystal sources with external sources.
- In 5.1 Unique Identifier, added information about randomness, sequence and uniqueness.
- In 6.2.3 Interrupt Summary, updated pending flag registers for SMBus0 and Timer3 overflow.
- In 8.3.2 HFOSC0 48 MHz Internal Oscillator, added note regarding changing calibration register from the default value.
- In 8.3.3 LFOSC0 80 kHz Internal Oscillator, updated edge used for capture.
- In 8.3.7 Clock Configuration, replaced references to HFOSC1 with HFOSC0.
- In 8.4.1 CLKSEL: Clock Select, added prefetch requirement to footnote.
- In 9.3.2 Power-On Reset, replaced references to V_{RST} with V_{POR} and removed typical power-on reset time (see EFM8UB2 Data Sheet).
- In 10.1 Introduction, updated max clock frequency and MIPS.
- In 10.4.7 PFE0CN: Prefetch Engine Control, updated the reset value.
- In 11.2 Features, removed references to two drive strengths and port match and added pin availability for INT0 and INT1.
- In 11.3.3 Priority Crossbar Decoder and 11.3.4 INT0 and INT1, removed references to port match.
- In 12.3.5 Input Tracking, updated the number of SAR clocks for ADTM = 1.
- In Figure 12.3 Track and Conversion Example Timing (Normal, Non-Burst Operation) on page 128, updated the number of SAR clocks for ADTM = 1 and number of SAR clocks for conversion.
- In Figure 13.1 Comparator Block Diagram on page 139, removed LDO and GND from input muxes.
- In 13.1 Introduction and 13.3.3 Input Selection, removed references to internal signals.
- In 14.3.1 Counter / Timer, updated PCA timebase input options.
- In 14.3.3 Capture/Compare Modules, 14.3.8 PWM Waveform Generation, and 14.3.8.1 8-Bit PWM Mode, removed references to PCA0PWM register.
- In 18.3.3 Configuring the SMBus Module, removed references to pin swap and updated the usage of the SDD field for SMBus timing.
- In 18.6.1 SMB1CF: SMBus 1 Configuration, updated the name and description for value 0x1 of the SMBCS field.
- In 21.4.3 SBUF1: UART1 Serial Port Data Buffer, removed references to TXNF.

Revision 0.2

Updated 11.3.3.1 Crossbar Functional Map to properly print the full crossbar map.

Revision 0.1

Initial release.



Simplicity Studio

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

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS

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

http://www.silabs.com