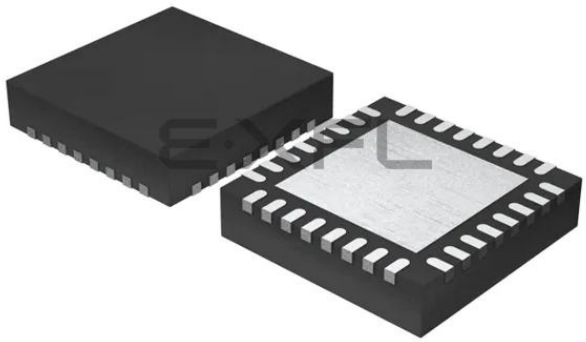


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Not For New Designs
Core Processor	8051
Core Size	8-Bit
Speed	48MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 21x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f342-gmr

5.3. Modes of Operation

ADC0 has a maximum conversion speed of 200 ksps. The ADC0 conversion clock is a divided version of the system clock, determined by the AD0SC bits in the ADC0CF register (system clock divided by (AD0SC + 1) for 0 ≤ AD0SC ≤ 31).

5.3.1. Starting a Conversion

A conversion can be initiated in one of five ways, depending on the programmed states of the ADC0 Start of Conversion Mode bits (AD0CM2–0) in register ADC0CN. Conversions may be initiated by one of the following:

1. Writing a ‘1’ to the AD0BUSY bit of register ADC0CN
2. A Timer 0 overflow (i.e., timed continuous conversions)
3. A Timer 2 overflow
4. A Timer 1 overflow
5. A rising edge on the CNVSTR input signal
6. A Timer 3 overflow

Writing a ‘1’ to AD0BUSY provides software control of ADC0 whereby conversions are performed "on-demand". During conversion, the AD0BUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the ADC0 interrupt flag (AD0INT). Note: When polling for ADC conversion completions, the ADC0 interrupt flag (AD0INT) should be used. Converted data is available in the ADC0 data registers, ADC0H:ADC0L, when bit AD0INT is logic 1. Note that when Timer 2 or Timer 3 overflows are used as the conversion source, Low Byte overflows are used if Timer 2/3 is in 8-bit mode; High byte overflows are used if Timer 2/3 is in 16-bit mode. See **Section “21. Timers” on page 235** for timer configuration.

Important Note About Using CNVSTR: The CNVSTR input pin also functions as a Port pin. When the CNVSTR input is used as the ADC0 conversion source, the associated Port pin should be skipped by the Digital Crossbar. To configure the Crossbar to skip a pin, set the corresponding bit in the PnSKIP register to ‘1’. See **Section “15. Port Input/Output” on page 142** for details on Port I/O configuration.

SFR Definition 6.1. REF0CN: Reference Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	-	-	REFSL	TEMPE	BIASE	REFBE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xD1

Bits7–3: UNUSED. Read = 00000b; Write = don't care.
Bit3: REFSL: Voltage Reference Select.
This bit selects the source for the internal voltage reference.
0: VREF pin used as voltage reference.
1: V_{DD} used as voltage reference.
Bit2: TEMPE: Temperature Sensor Enable Bit.
0: Internal Temperature Sensor off.
1: Internal Temperature Sensor on.
Bit1: BIASE: Internal Analog Bias Generator Enable Bit.
0: Internal Bias Generator off.
1: Internal Bias Generator on.
Bit0: REFBE: Internal Reference Buffer Enable Bit.
0: Internal Reference Buffer disabled.
1: Internal Reference Buffer enabled. Internal voltage reference driven on the VREF pin.

Table 6.1. Voltage Reference Electrical Characteristics

V_{DD} = 3.0 V; –40 to +85 °C Unless Otherwise Specified

Parameter	Conditions	Min	Typ	Max	Units
Internal Reference (REFBE = 1)					
Output Voltage	25 °C ambient	2.38	2.44	2.50	V
VREF Short-Circuit Current				10	mA
VREF Temperature Coefficient			15		ppm/°C
Load Regulation	Load = 0 to 200 µA to GND		1.5		ppm/µA
VREF Turn-on Time 1	4.7 µF tantalum, 0.1 µF ceramic bypass		2		ms
VREF Turn-on Time 2	0.1 µF ceramic bypass		20		µs
VREF Turn-on Time 3	no bypass cap		10		µs
Power Supply Rejection			140		ppm/V
External Reference (REFBE = 0)					
Input Voltage Range		0		V_{DD}	V
Input Current	Sample Rate = 200 ksps; VREF = 3.0 V		12		µA
Bias Generators					
ADC Bias Generator	BIASE = '1'		100		µA
Reference Bias Generator			40		µA

9. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; 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. Included are four 16-bit counter/timers (see description in **Section 21**), an enhanced full-duplex UART (see description in **Section 18**), an Enhanced SPI (see description in **Section 20**), 256 bytes of internal RAM, 128 byte Special Function Register (SFR) address space (**Section 9.2.6**), and 25 Port I/O (see description in **Section 15**). The CIP-51 also includes on-chip debug hardware (see description in **Section 23**), and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control-system solution in a single integrated circuit.

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability (see Figure 9.1 for a block diagram). The CIP-51 includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 0 to 48 MHz Clock Frequency
- 256 Bytes of Internal RAM
- 25 Port I/O
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

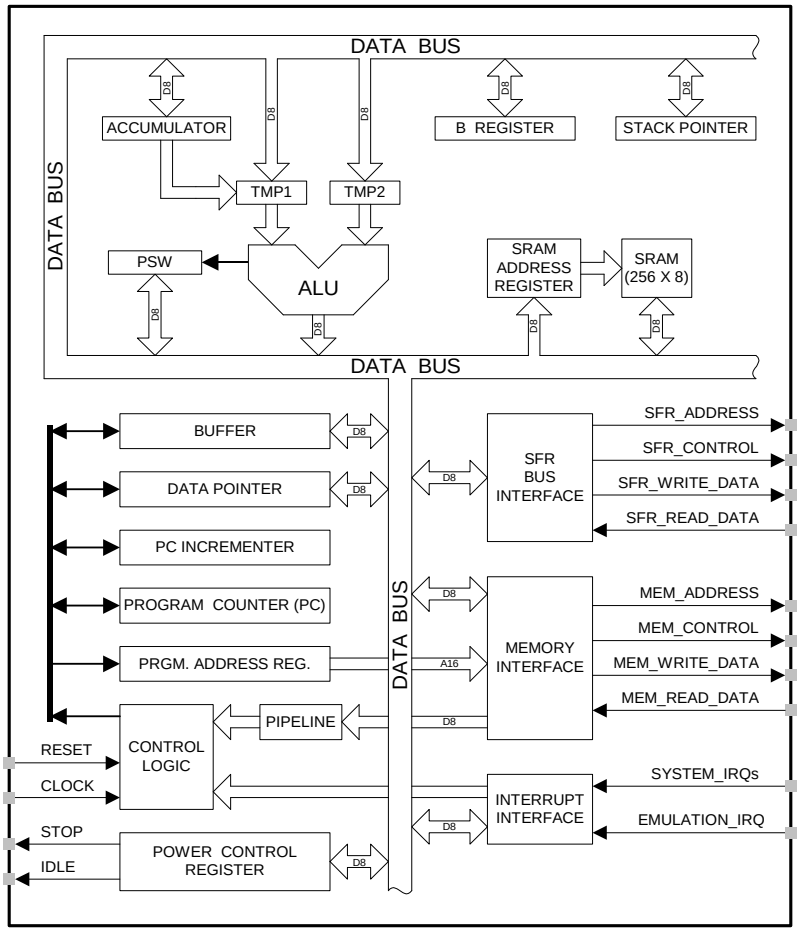


Figure 9.1. CIP-51 Block Diagram

Table 9.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
ORL A, #data	OR immediate to A	2	2
ORL direct, A	OR A to direct byte	2	2
ORL direct, #data	OR immediate to direct byte	3	3
XRL A, Rn	Exclusive-OR Register to A	1	1
XRL A, direct	Exclusive-OR direct byte to A	2	2
XRL A, @Ri	Exclusive-OR indirect RAM to A	1	2
XRL A, #data	Exclusive-OR immediate to A	2	2
XRL direct, A	Exclusive-OR A to direct byte	2	2
XRL direct, #data	Exclusive-OR immediate to direct byte	3	3
CLR A	Clear A	1	1
CPL A	Complement A	1	1
RL A	Rotate A left	1	1
RLC A	Rotate A left through Carry	1	1
RR A	Rotate A right	1	1
RRC A	Rotate A right through Carry	1	1
SWAP A	Swap nibbles of A	1	1
Data Transfer			
MOV A, Rn	Move Register to A	1	1
MOV A, direct	Move direct byte to A	2	2
MOV A, @Ri	Move indirect RAM to A	1	2
MOV A, #data	Move immediate to A	2	2
MOV Rn, A	Move A to Register	1	1
MOV Rn, direct	Move direct byte to Register	2	2
MOV Rn, #data	Move immediate to Register	2	2
MOV direct, A	Move A to direct byte	2	2
MOV direct, Rn	Move Register to direct byte	2	2
MOV direct, direct	Move direct byte to direct byte	3	3
MOV direct, @Ri	Move indirect RAM to direct byte	2	2
MOV direct, #data	Move immediate to direct byte	3	3
MOV @Ri, A	Move A to indirect RAM	1	2
MOV @Ri, direct	Move direct byte to indirect RAM	2	2
MOV @Ri, #data	Move immediate to indirect RAM	2	2
MOV DPTR, #data16	Load DPTR with 16-bit constant	3	3
MOVC A, @A+DPTR	Move code byte relative DPTR to A	1	3
MOVC A, @A+PC	Move code byte relative PC to A	1	3
MOVX A, @Ri	Move external data (8-bit address) to A	1	3
MOVX @Ri, A	Move A to external data (8-bit address)	1	3
MOVX A, @DPTR	Move external data (16-bit address) to A	1	3
MOVX @DPTR, A	Move A to external data (16-bit address)	1	3
PUSH direct	Push direct byte onto stack	2	2
POP direct	Pop direct byte from stack	2	2
XCH A, Rn	Exchange Register with A	1	1
XCH A, direct	Exchange direct byte with A	2	2
XCH A, @Ri	Exchange indirect RAM with A	1	2
XCHD A, @Ri	Exchange low nibble of indirect RAM with A	1	2

13.3. Configuring the External Memory Interface

Configuring the External Memory Interface consists of five 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.
2. Configure Port latches to “park” the EMIF pins in a dormant state (usually by setting them to logic ‘1’).
3. Select Multiplexed mode or Non-multiplexed mode.
4. Select the memory mode (on-chip only, split mode without bank select, split mode with bank select, or off-chip only).
5. Set up timing to interface with off-chip memory or peripherals.

Each of these five steps is explained in detail in the following sections. The Port selection, Multiplexed mode selection, and Mode bits are located in the EMI0CF register shown in SFR Definition 13.2.

13.4. Port Configuration

The External Memory Interface appears on Ports 4, 3, 2, and 1 when it is used for off-chip memory access. When the EMIF is used, the Crossbar should be configured to skip over the control lines P1.7 ($\overline{WR}$), P1.6 ($\overline{RD}$), and if multiplexed mode is selected P1.3 (ALE) using the P1SKIP register. For more information about configuring the Crossbar, see **Section “Figure 15.1. Port I/O Functional Block Diagram (Port 0 through Port 3)” on page 142**.

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 or to the Crossbar settings for those pins. See **Section “15. Port Input/ Output” on page 142** for more information about the Crossbar and Port operation and configuration. **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). The Output mode of the Port pins (whether the pin is configured as Open-Drain or Push-Pull) 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.

SFR Definition 13.2. EMI0CF: External Memory Configuration

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	USBFAE	-	EMD2	EMD1	EMD0	EALE1	EALE0	00000011
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0x85

Bit7: Unused. Read = 0b. Write = don't care.

Bit6: USBFAE: USB FIFO Access Enable.
0: USB FIFO RAM not available through MOVX instructions.
1: USB FIFO RAM available using MOVX instructions. The 1k 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 $\geq$ 2 x SYSCLK) to access this area with MOVX instructions.**

Bit5: Unused. Read = 0b. Write = don't care.

Bit4: EMD2: EMIF Multiplex Mode Select.
0: EMIF operates in multiplexed address/data mode.
1: EMIF operates in non-multiplexed mode (separate address and data pins).

Bits3–2: EMD1–0: EMIF Operating Mode Select.
These bits control the operating mode of the External Memory Interface.
00: Internal Only: MOVX accesses on-chip XRAM only. All effective addresses alias to on-chip memory space.
01: Split Mode without Bank Select: Accesses below the on-chip XRAM boundary are directed on-chip. Accesses above the on-chip XRAM boundary are directed off-chip. 8-bit off-chip MOVX operations use the current contents of the Address High port latches to resolve upper address byte. Note that in order to access off-chip space, EMI0CN must be set to a page that is not contained in the on-chip address space.
10: Split Mode with Bank Select: Accesses below the on-chip XRAM boundary are directed on-chip. Accesses above the on-chip XRAM boundary are directed off-chip. 8-bit off-chip MOVX operations use the contents of EMI0CN to determine the high-byte of the address.
11: External Only: MOVX accesses off-chip XRAM only. On-chip XRAM is not visible to the CPU.

Bits1–0: EALE1–0: ALE Pulse-Width Select Bits (only has effect when EMD2 = 0).
00: ALE high and ALE low pulse width = 1 SYSCLK cycle.
01: ALE high and ALE low pulse width = 2 SYSCLK cycles.
10: ALE high and ALE low pulse width = 3 SYSCLK cycles.
11: ALE high and ALE low pulse width = 4 SYSCLK cycles.

SFR Definition 13.3. EMI0TC: External Memory Timing Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
EAS1	EAS0	EWR3	EWR2	EWR1	EWR0	EAH1	EAH0	11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0x84

Bits7–6: EAS1–0: EMIF Address Setup Time Bits.
00: Address setup time = 0 SYSCLK cycles.
01: Address setup time = 1 SYSCLK cycle.
10: Address setup time = 2 SYSCLK cycles.
11: Address setup time = 3 SYSCLK cycles.

Bits5–2: EWR3–0: EMIF $\overline{WR}$ and $\overline{RD}$ Pulse-Width Control Bits.
0000: $\overline{WR}$ and $\overline{RD}$ pulse width = 1 SYSCLK cycle.
0001: $\overline{WR}$ and $\overline{RD}$ pulse width = 2 SYSCLK cycles.
0010: $\overline{WR}$ and $\overline{RD}$ pulse width = 3 SYSCLK cycles.
0011: $\overline{WR}$ and $\overline{RD}$ pulse width = 4 SYSCLK cycles.
0100: $\overline{WR}$ and $\overline{RD}$ pulse width = 5 SYSCLK cycles.
0101: $\overline{WR}$ and $\overline{RD}$ pulse width = 6 SYSCLK cycles.
0110: $\overline{WR}$ and $\overline{RD}$ pulse width = 7 SYSCLK cycles.
0111: $\overline{WR}$ and $\overline{RD}$ pulse width = 8 SYSCLK cycles.
1000: $\overline{WR}$ and $\overline{RD}$ pulse width = 9 SYSCLK cycles.
1001: $\overline{WR}$ and $\overline{RD}$ pulse width = 10 SYSCLK cycles.
1010: $\overline{WR}$ and $\overline{RD}$ pulse width = 11 SYSCLK cycles.
1011: $\overline{WR}$ and $\overline{RD}$ pulse width = 12 SYSCLK cycles.
1100: $\overline{WR}$ and $\overline{RD}$ pulse width = 13 SYSCLK cycles.
1101: $\overline{WR}$ and $\overline{RD}$ pulse width = 14 SYSCLK cycles.
1110: $\overline{WR}$ and $\overline{RD}$ pulse width = 15 SYSCLK cycles.
1111:$\overline{WR}$ and $\overline{RD}$ pulse width = 16 SYSCLK cycles.

Bits1–0: EAH1–0: EMIF Address Hold Time Bits.
00: Address hold time = 0 SYSCLK cycles.
01: Address hold time = 1 SYSCLK cycle.
10: Address hold time = 2 SYSCLK cycles.
11: Address hold time = 3 SYSCLK cycles.

SFR Definition 15.21. P4MDIN: Port4 Input Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xF5

Bits7–0: Analog Input Configuration Bits for P4.7–P4.0 (respectively).
Port pins configured as analog inputs have their weak pull-up, digital driver, and digital receiver disabled.
0: Corresponding P4.n pin is configured as an analog input.
1: Corresponding P4.n pin is not configured as an analog input.

Note: P4 is only available on 48-pin devices.

SFR Definition 15.22. P4MDOUT: Port4 Output Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xAE

Bits7–0: Output Configuration Bits for P4.7–P4.0 (respectively); ignored if corresponding bit in register P4MDIN is logic 0.
0: Corresponding P4.n Output is open-drain.
1: Corresponding P4.n Output is push-pull.

Note: P4 is only available on 48-pin devices.

mode feature of the internal oscillator.

USB0 exits Suspend mode when any of the following occur: (1) Resume signaling is detected or generated, (2) Reset signaling is detected, or (3) a device or USB reset occurs. If suspended, the internal oscillator will exit Suspend mode upon any of the above listed events.

Resume Signaling: USB0 will exit 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 (POWER.7), the ISO Update function is enabled. With ISO Update enabled, new packets written to an ISO IN endpoint will not be transmitted until a new Start-Of-Frame (SOF) is received. If the ISO IN endpoint receives an IN token before a SOF, USB0 will transmit a zero-length packet. When ISOUP = ‘1’, ISO Update is enabled for all ISO endpoints.

USB Enable: USB0 is disabled following a Power-On-Reset (POR). USB0 is enabled by clearing the USBINH bit (POWER.4). Once written to ‘0’, the USBINH can only be set to ‘1’ by one of the following: (1) a Power-On-Reset (POR), or (2) an asynchronous USB0 reset generated by writing ‘1’ to the USBRST bit (POWER.3).

Software should perform all USB0 configuration before enabling USB0. The configuration sequence should be performed as follows:

Step 1. Select and enable the USB clock source.
Step 2. Reset USB0 by writing USBRST= ‘1’.
Step 3. Configure and enable the USB Transceiver.
Step 4. Perform any USB0 function configuration (interrupts, Suspend detect).
Step 5. Enable USB0 by writing USBINH = ‘0’.

USB Register Definition 16.17. E0CSR: USB0 Endpoint0 Control

R/W	R/W	R/W	R	R/W	R/W	R/W	R	Reset Value
SSUEND	SOPRDY	SDSTL	SUEND	DATAEND	STSTL	INPRDY	OPRDY	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	USB Address: 0x11

Bit7: SSUEND: Serviced Setup End
Write: Software should set this bit to ‘1’ after servicing a Setup End (bit SUEND) event. Hardware clears the SUEND bit when software writes ‘1’ to SSUEND.
Read: This bit always reads ‘0’.

Bit6: SOPRDY: Serviced OPRDY
Write: Software should write ‘1’ to this bit after servicing a received Endpoint0 packet. The OPRDY bit will be cleared by a write of ‘1’ to SOPRDY.
Read: This bit always reads ‘0’.

Bit5: SDSTL: Send Stall
Software 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.

Bit4: SUEND: Setup End
Hardware sets this read-only bit to ‘1’ when a control transaction ends before software has written ‘1’ to the DATAEND bit. Hardware clears this bit when software writes ‘1’ to SSUEND.

Bit3: DATAEND: Data End
Software 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.

Bit2: STSTL: Sent Stall
Hardware sets this bit to ‘1’ after transmitting a STALL handshake signal. This flag must be cleared by software.

Bit1: INPRDY: IN Packet Ready
Software should write ‘1’ to this bit after loading a data packet into the Endpoint0 FIFO for transmit. Hardware clears this bit and generates an interrupt under either 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.

Bit0: OPRDY: 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 software writes ‘1’ to the SOPRDY bit.

A Bulk or Interrupt pipe can be shut down (or Halted) by writing ‘1’ to the SDSTL bit (EOUTCSRL.5). 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 (EOUTCSRL.6) 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.

16.13.2.Endpoints1-3 OUT Isochronous Mode

When the ISO bit (EOUTCSRH.6) 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 (EOUTCSRL.0) 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 (EOUTCSRL.2) 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 (EOUTCSRL.3) will be set to ‘1’. Software should check the DATAERR bit each time a data packet is unloaded from an ISO OUT endpoint FIFO.

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

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 17.3 illustrates a typical SMBus transaction.

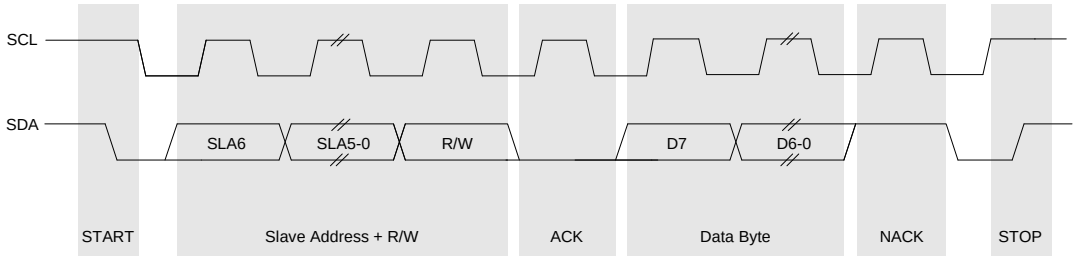


Figure 17.3. SMBus Transaction

17.3.1. 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 **Section “17.3.4. SCL High (SMBus Free) Timeout” on page 191**). 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.

19.3. Configuration and Operation

UART1 provides standard asynchronous, full duplex communication. It can operate in a point-to-point serial communications application, or as a node on a multi-processor serial interface. To operate in a point-to-point application, where there are only two devices on the serial bus, the MCE1 bit in SMOD1 should be cleared to '0'. For operation as part of a multi-processor communications bus, the MCE1 and XBE1 bits should both be set to '1'. In both types of applications, data is transmitted from the microcontroller on the TX1 pin, and received on the RX1 pin. The TX1 and RX1 pins are configured using the crossbar and the Port I/O registers, as detailed in **Section "15. Port Input/Output" on page 142**.

In typical UART communications, The transmit (TX) output of one device is connected to the receive (RX) input of the other device, either directly or through a bus transceiver, as shown in Figure 19.5.

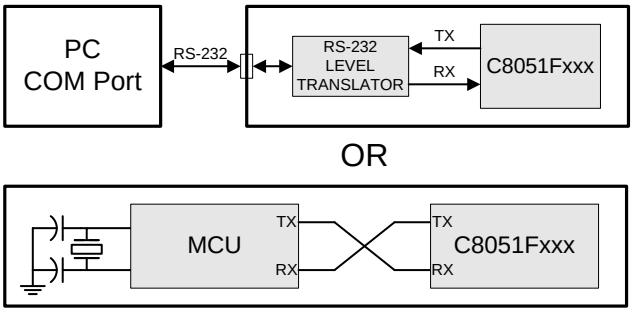


Figure 19.5. Typical UART Interconnect Diagram

19.3.1. Data Transmission

Data transmission is double-buffered, and begins when software writes a data byte to the SBUF1 register. Writing to SBUF1 places data in the Transmit Holding Register, and the Transmit Holding Register Empty flag (THRE1) will be cleared to '0'. If the UARTs shift register is empty (i.e., no transmission is in progress) the data will be placed in the shift register, and the THRE1 bit will be set to '1'. If a transmission is in progress, the data will remain in the Transmit Holding Register until the current transmission is complete. The TI1 Transmit Interrupt Flag (SCON1.1) will be set at the end of any transmission (the beginning of the stop-bit time). If enabled, an interrupt will occur when TI1 is set.

If the extra bit function is enabled (XBE1 = '1') and the parity function is disabled (PE1 = '0'), the value of the TBX1 (SCON1.3) bit will be sent in the extra bit position. When the parity function is enabled (PE1 = '1'), hardware will generate the parity bit according to the selected parity type (selected with S1PT[1:0]), and append it to the data field. Note: when parity is enabled, the extra bit function is not available.

19.3.2. Data Reception

Data reception can begin any time after the REN1 Receive Enable bit (SCON1.4) is set to logic 1. After the stop bit is received, the data byte will be stored in the receive FIFO if the following conditions are met: the receive FIFO (3 bytes deep) must not be full, and the stop bit(s) must be logic 1. In the event that the receive FIFO is full, the incoming byte will be lost, and a Receive FIFO Overrun Error will be generated (OVR1 in register SCON1 will be set to logic 1). If the stop bit(s) were logic 0, the incoming data will not be stored in the receive FIFO. If the reception conditions are met, the data is stored in the receive FIFO, and the RI1 flag will be set. Note: when MCE1 = '1', RI1 will only be set if the extra bit was equal to '1'. Data can be read from the receive FIFO by reading the SBUF1 register. The SBUF1 register represents the oldest byte in the FIFO. After SBUF1 is read, the next byte in the FIFO is immediately loaded into SBUF1, and

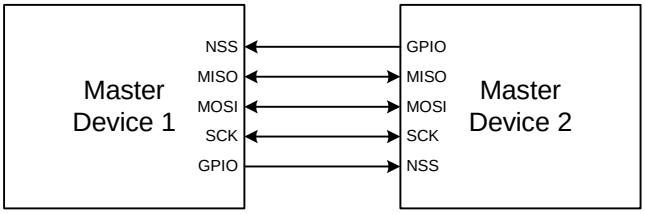


Figure 20.2. Multiple-Master Mode Connection Diagram

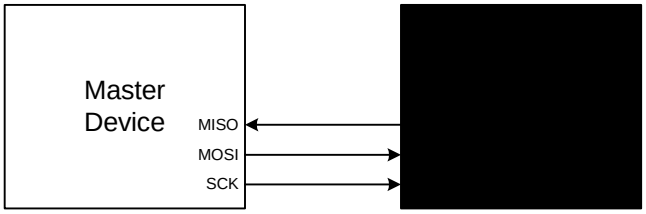


Figure 20.3. 3-Wire Single Master and Slave Mode Connection Diagram

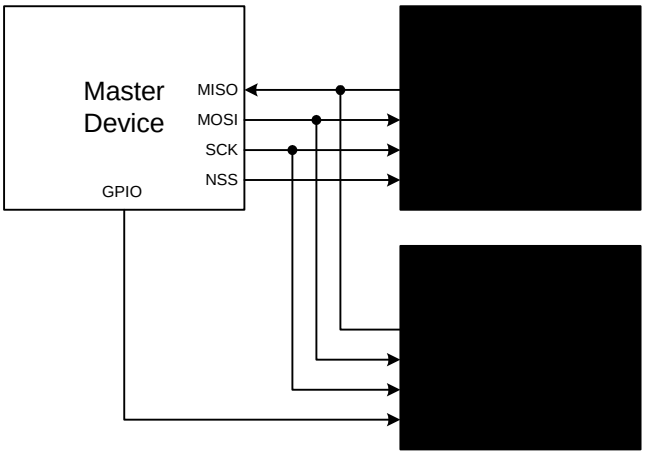
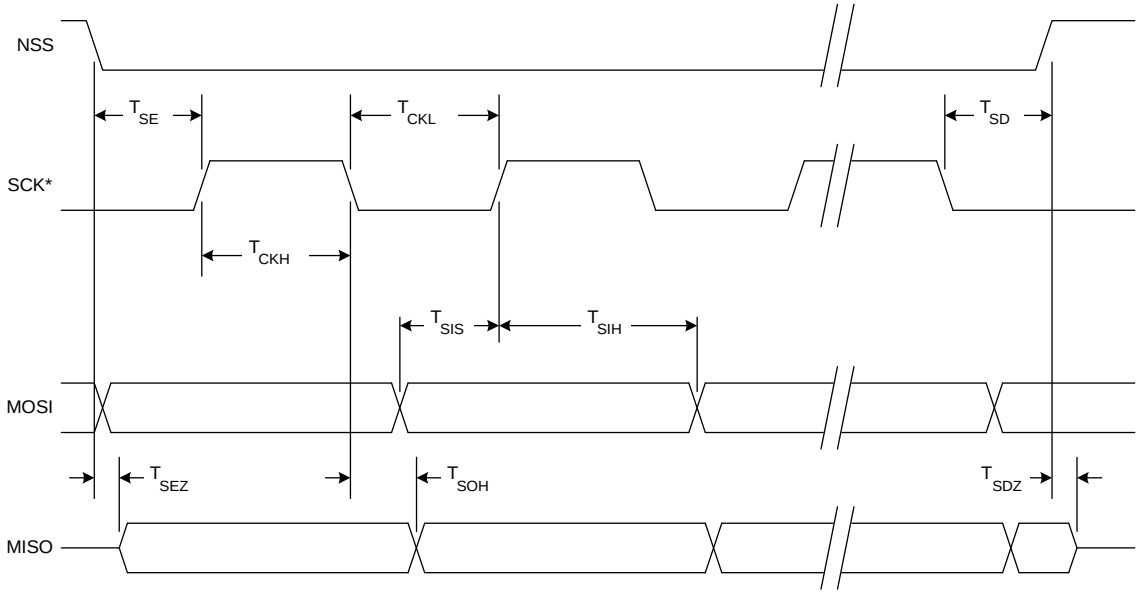
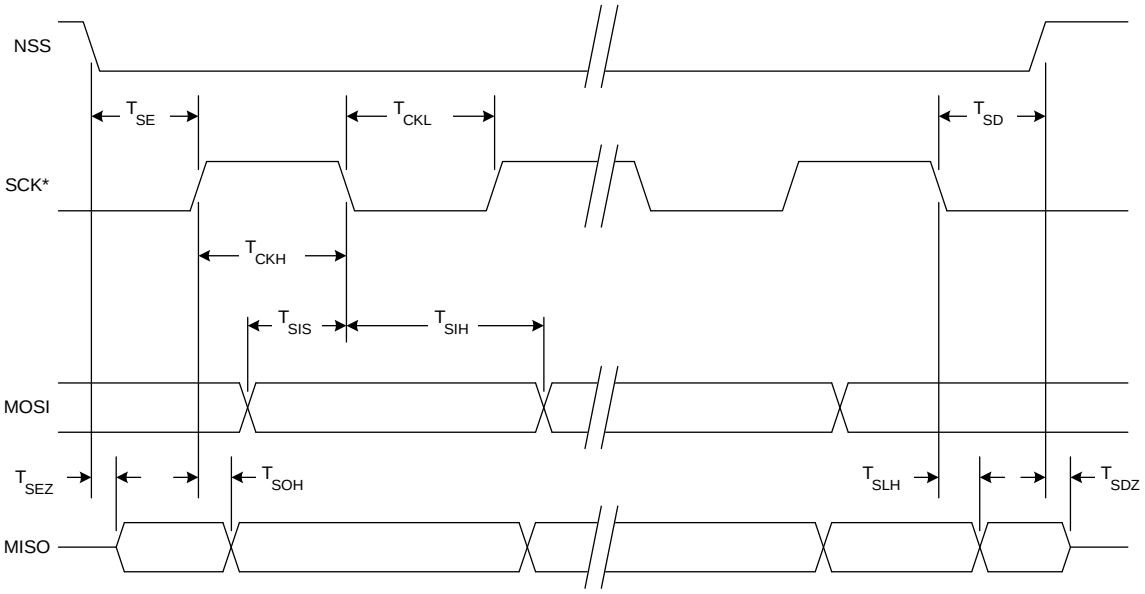


Figure 20.4. 4-Wire Single Master Mode and Slave Mode Connection Diagram



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

Figure 20.10. SPI Slave Timing (CKPHA = 0)



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

Figure 20.11. SPI Slave Timing (CKPHA = 1)

21. Timers

Each MCU includes four counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and two are 16-bit auto-reload timer for use with the ADC, SMBus, USB (frame measurements), Low-Frequency Oscillator (period measurements), 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 and Timer 3 offer 16-bit and split 8-bit timer functionality with auto-reload.

Timer 0 and Timer 1 Modes:	Timer 2 Modes:	Timer 3 Modes:
13-bit counter/timer	16-bit timer with auto-reload	16-bit timer with auto-reload
16-bit counter/timer		
8-bit counter/timer with auto-reload	Two 8-bit timers with auto-reload	Two 8-bit timers with auto-reload
Two 8-bit counter/timers (Timer 0 only)		

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 (See SFR Definition 21.3 for pre-scaled clock selection).

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 and Timer 3 may be clocked by the system clock, the system clock divided by 12, or the external oscillator clock source 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's frequency can be counted. The input signal need not be periodic, but it should be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

21.1. Timer 0 and Timer 1

Each timer is 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 (**Section “9.3.5. Interrupt Register Descriptions” on page 90**); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (**Section 9.3.5**). 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. Each operating mode is described below.

21.1.1. 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 (TCON.5) is set and an interrupt will occur if Timer 0 interrupts are enabled.

21.1.4. 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, C/T0, 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.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

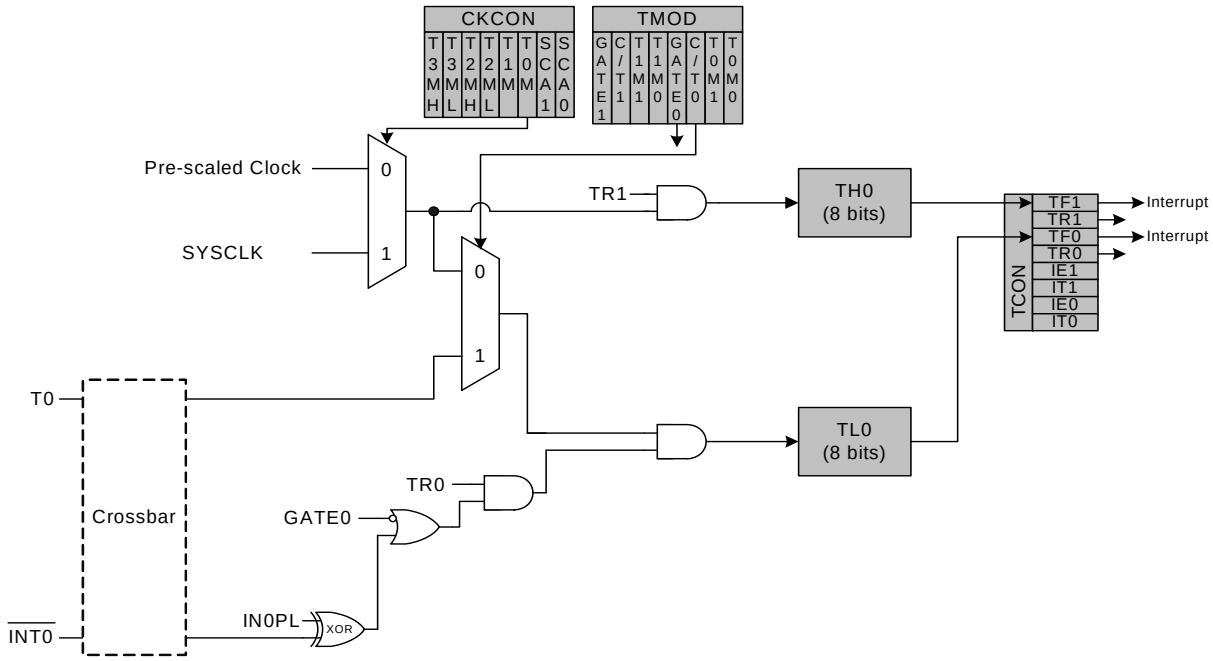


Figure 21.3. T0 Mode 3 Block Diagram

SFR Definition 21.4. TL0: Timer 0 Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0x8A

Bits 7–0: TL0: Timer 0 Low Byte.
The TL0 register is the low byte of the 16-bit Timer 0.

SFR Definition 21.5. TL1: Timer 1 Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0x8B

Bits 7–0: TL1: Timer 1 Low Byte.
The TL1 register is the low byte of the 16-bit Timer 1.

SFR Definition 21.6. TH0: Timer 0 High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0x8C

Bits 7–0: TH0: Timer 0 High Byte.
The TH0 register is the high byte of the 16-bit Timer 0.

SFR Definition 21.7. TH1: Timer 1 High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0x8D

Bits 7–0: TH1: Timer 1 High Byte.
The TH1 register is the high byte of the 16-bit Timer 1.

22.2.1. 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 PCA0CN is set to logic 1 and an interrupt request is generated if CCF interrupts are 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.

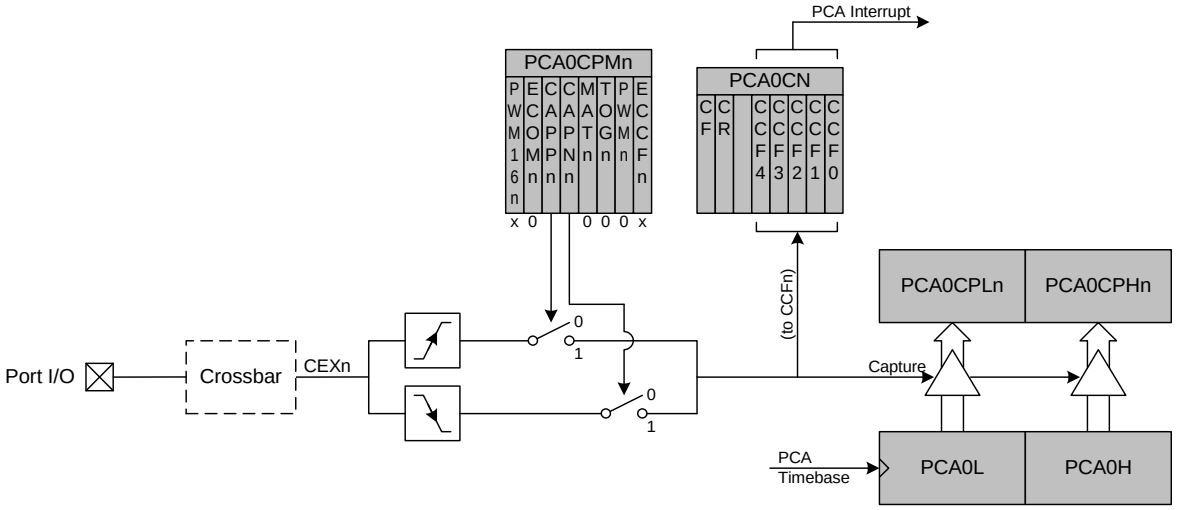


Figure 22.4. PCA Capture Mode Diagram

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