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	Active
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I²C), CANbus, LINbus, SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	96KB (96K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.25V
Data Converters	A/D 25x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°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/c8051f586-imr

C8051F58x/F59x
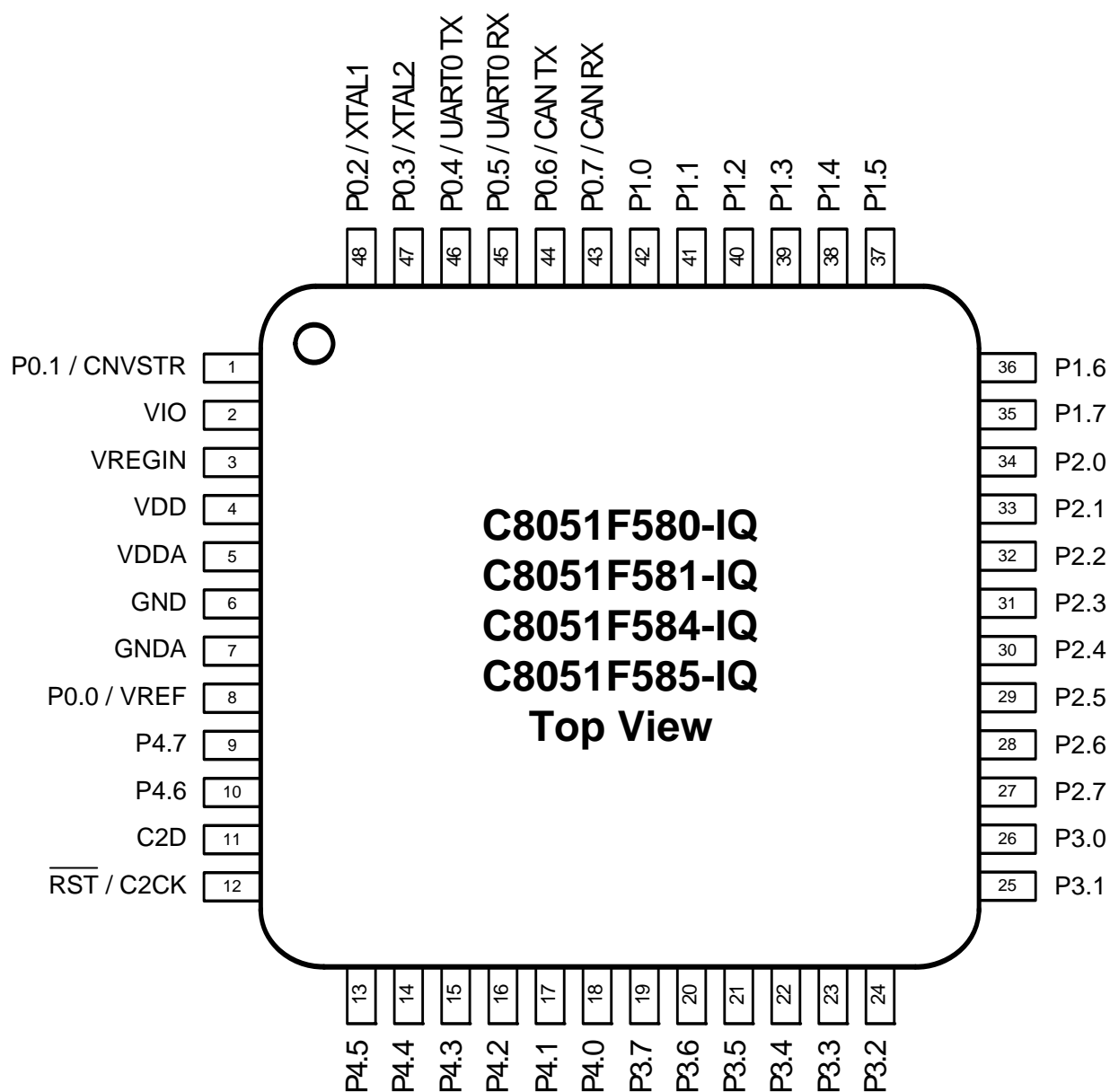
Figure 3.1. QFP-48 Pinout Diagram (Top View)

Rev. 1.3

SILICON LABS

5.2. Electrical Characteristics

Table 5.2. Global Electrical Characteristics

–40 to +125 °C, 24 MHz system clock unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Supply Input Voltage (V_{REGIN})		1.8	—	5.25	V
Digital Supply Voltage (V_{DD})	System Clock $\leq$ 25 MHz	V_{RST}[1]	—	2.75	V
	System Clock > 25 MHz	2		2.75	
Analog Supply Voltage (VDDA)	System Clock $\leq$ 25 MHz	V_{RST}[1]	—	2.75	V
(Must be connected to V_{DD})	System Clock > 25 MHz	2		2.75	
Digital Supply RAM Data Retention Voltage		—	1.5	—	
Port I/O Supply Voltage (V_{IO})	Normal Operation	1.8[2]	—	5.25	V
SYSCLK (System Clock)[3]		0	—	50	MHz
T_{SYSH} (SYSCLK High Time)		9	—	—	ns
T_{SYSL} (SYSCLK Low Time)		9	—	—	ns
Specified Operating Temperature Range		–40	—	+125	°C
Digital Supply Current—CPU Active (Normal Mode, fetching instructions from Flash)					
I_{DD}[4]	V_{DD} = 2.1 V, F = 200 kHz	—	150	—	µA
	V_{DD} = 2.1 V, F = 1.5 MHz	—	650	—	µA
	V_{DD} = 2.1 V, F = 25 MHz	—	8.5	11	mA
	V_{DD} = 2.1 V, F = 50 MHz	—	15	21	mA

Notes:
1. Given in Table 5.4 on page 48.
2. V_{IO} should not be lower than the V_{DD} voltage.
3. SYSCLK must be at least 32 kHz to enable debugging.
4. Based on device characterization data; Not production tested. Does not include oscillator supply current.
5. IDD can be estimated for frequencies $\leq$ 15 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate I_{DD} for >15 MHz, the estimate should be the current at 50 MHz minus the difference in current indicated by the frequency sensitivity number. For example: V_{DD} = 2.6 V; F = 20 MHz, I_{DD} = 21 mA - (50 MHz - 20 MHz) * 0.46 mA/MHz = 7.2 mA.
6. Idle IDD can be estimated for frequencies $\leq$ 1 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate Idle I_{DD} for >1 MHz, the estimate should be the current at 50 MHz minus the difference in current indicated by the frequency sensitivity number.
   For example: V_{DD} = 2.6 V; F = 5 MHz, Idle I_{DD} = 19 mA – (50 MHz – 5 MHz) x 0.38 mA/MHz = 1.9 mA.

SILICON LABS

SFR Definition 9.6. CPT2MD: Comparator2 Mode Selection

Bit	7	6	5	4	3	2	1	0
Name			CP2RIE	CP2FIE			CP2MD[1:0]	
Type	R	R	R/W	R/W	R	R	R/W	
Reset	0	0	0	0	0	0	1	0

SFR Address = 0x9B; SFR Page = 0x10

Bit	Name	Function
7:6	Unused	Read = 00b, Write = Don't Care.
5	CP2RIE	**Comparator2 Rising-Edge Interrupt Enable.** 0: Comparator2 Rising-edge interrupt disabled. 1: Comparator2 Rising-edge interrupt enabled.
4	CP2FIE	**Comparator2 Falling-Edge Interrupt Enable.** 0: Comparator2 Falling-edge interrupt disabled. 1: Comparator2 Falling-edge interrupt enabled.
3:2	Unused	Read = 00b, Write = don't care.
1:0	CP2MD[1:0]	**Comparator2 Mode Select.** These bits affect the response time and power consumption for Comparator2. 00: Mode 0 (Fastest Response Time, Highest Power Consumption) 01: Mode 1 10: Mode 2 11: Mode 3 (Slowest Response Time, Lowest Power Consumption)

C8051F58x/F59x

SFR Definition 9.7. CPT0MX: Comparator0 MUX Selection

Bit	7	6	5	4	3	2	1	0
Name		CMX0N[3:0]				CMX0P[3:0]		
Type		R/W				R/W		
Reset	0	1	1	1	0	1	1	1

SFR Address = 0x9C; SFR Page = 0x00

Bit	Name	Function
7:4	CMX0N[3:0]	**Comparator0 Negative Input MUX Selection.**
		0000: P0.1
		0001: P0.3
		0010: P0.5
		0011: P0.7
		0100: P1.1
		0101: P1.3
		0110: P1.5
		0111: P1.7
		1000: P2.1
		1001: P2.3
		1010: P2.5
		1011: P2.7
		1100–1111: None
3:0	CMX0P[3:0]	**Comparator0 Positive Input MUX Selection.**
		0000: P0.0
		0001: P0.2
		0010: P0.4
		0011: P0.6
		0100: P1.0
		0101: P1.2
		0110: P1.4
		0111: P1.6
		1000: P2.0
		1001: P2.2
		1010: P2.4
		1011: P2.6
		1100–1111: None

SFR Definition 14.5. EIE2: Extended Interrupt Enable 2

Bit	7	6	5	4	3	2	1	0
Name	ET5	ET4	ECP2	EPCA1	ES1	EMAT	ECAN0	EREG0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE7; SFR Page = All Pages

Bit	Name	Function
7	ET5	**Enable Timer 5 Interrupt.** This bit sets the masking of the Timer 5 interrupt. 0: Disable Timer 5 interrupts. 1: Enable interrupt requests generated by the TF5L or TF5H flags.
6	ET4	**Enable Timer 4 Interrupt.** This bit sets the masking of the Timer 4 interrupt. 0: Disable Timer 4 interrupts. 1: Enable interrupt requests generated by the TF4L or TF4H flags.
5	ECP2	**Enable Comparator2 (CP2) Interrupt.** This bit sets the masking of the CP2 interrupt. 0: Disable CP2 interrupts. 1: Enable interrupt requests generated by the CP2RIF or CP2FIF flags.
4	EPCA1	**Enable Programmable Counter Array (PCA1) Interrupt.** This bit sets the masking of the PCA1 interrupts. 0: Disable all PCA1 interrupts. 1: Enable interrupt requests generated by PCA1
3	ES1	**Enable UART1 Interrupt.** This bit sets the masking of the UART1 interrupt. 0: Disable UART1 interrupt. 1: Enable UART1 interrupt
2	EMAT	**Enable Port Match Interrupt.** This bit sets the masking of the Port Match interrupt. 0: Disable all Port Match interrupts. 1: Enable interrupt requests generated by a Port Match
1	ECAN0	**Enable CAN0 Interrupts.** This bit sets the masking of the CAN0 interrupt. 0: Disable all CAN0 interrupts. 1: Enable interrupt requests generated by CAN0.
0	EREG0	**Enable Voltage Regulator Dropout Interrupt.** This bit sets the masking of the Voltage Regulator Dropout interrupt. 0: Disable the Voltage Regulator Dropout interrupt. 1: Enable the Voltage Regulator Dropout interrupt.

C8051F58x/F59x

18.5.3. Split Mode with Bank Select

When EMI0CF[3:2] are set to 10, the XRAM memory map is split into two areas, on-chip space and off-chip space.

- Effective addresses below the internal XRAM size boundary will access on-chip XRAM space.
- Effective addresses above the internal XRAM size boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is on-chip or off-chip. The upper 8-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 on-chip or off-chip, and the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

18.5.4. External Only

When EMI0CF[3:2] are set to 11, 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 internal XRAM size 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" described above). This allows the user 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.

18.6. 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, $\overline{RD}$ and $\overline{WR}$ strobe widths, and in multiplexed mode, the width of the ALE pulse are all programmable in units of SYSCLK periods through EMI0TC, shown in SFR Definition 18.3, and EMI0CF[1:0].

The timing for an off-chip MOVX instruction can be calculated by adding 4 SYSCLK cycles to the timing parameters defined by the EMI0TC register. Assuming non-multiplexed operation, the minimum execution time for an off-chip XRAM operation is 5 SYSCLK cycles (1 SYSCLK for $\overline{RD}$ or $\overline{WR}$ 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 for an off-chip XRAM operation in multiplexed mode is 7 SYSCLK cycles (2 for /ALE + 1 for $\overline{RD}$ or $\overline{WR}$ + 4). The programmable setup and hold times default to the maximum delay settings after a reset. Table 18.3 lists the ac parameters for the External Memory Interface, and Figure 18.4 through Figure 18.9 show the timing diagrams for the different External Memory Interface modes and MOVX operations.

18.6.2. Multiplexed Mode
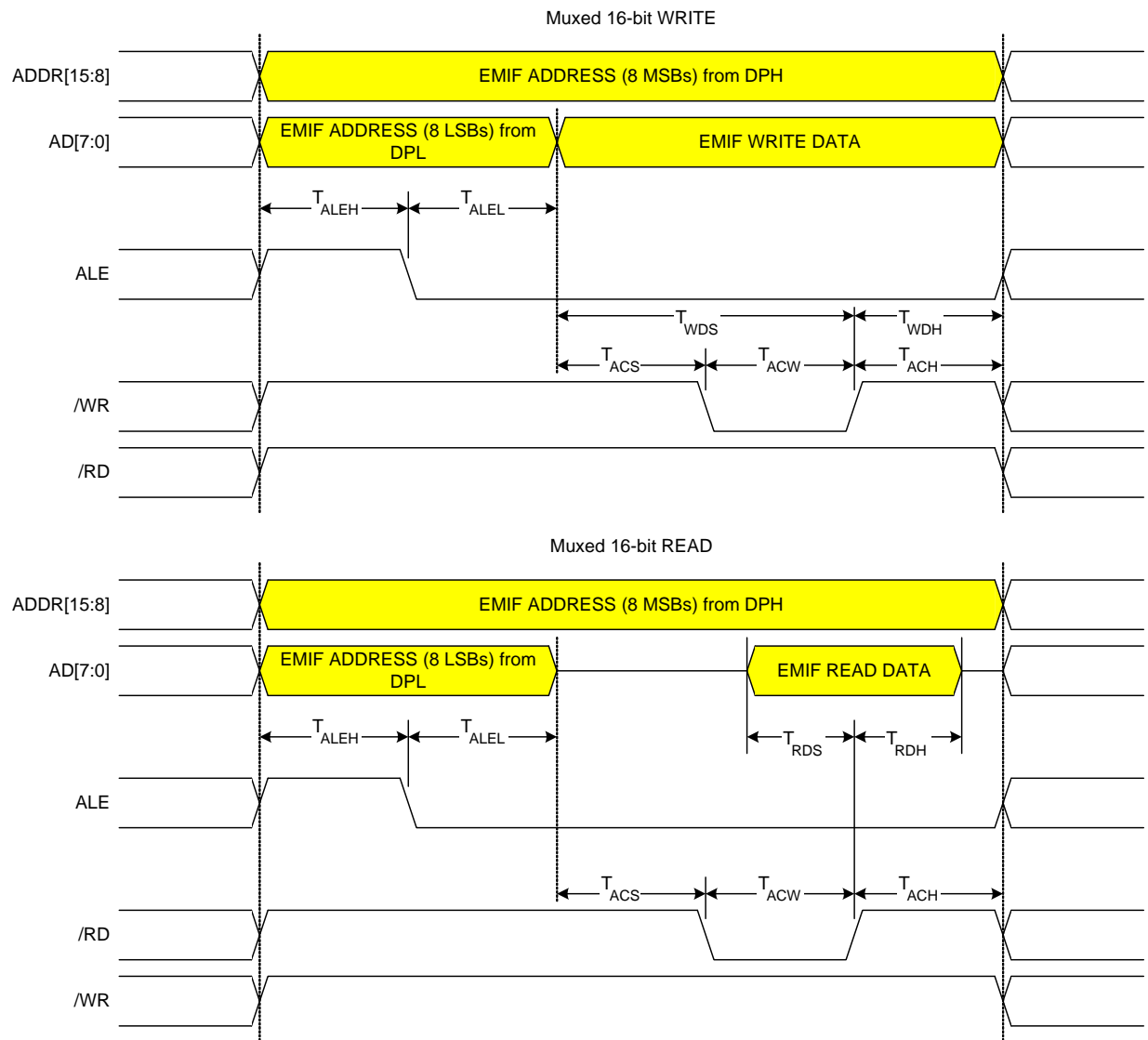### 18.6.2.1. 16-bit MOVX: EMI0CF[4:2] = 001, 010, or 011

Figure 18.7. Multiplexed 16-bit MOVX Timing

C8051F58x/F59x

Table 18.3. AC Parameters for External Memory Interface

Parameter	Description	Min*	Max*	Units
T_{ACS}	Address/Control Setup Time	0	$3 \times T_{SYSCLK}$	ns
T_{ACW}	Address/Control Pulse Width	$1 \times T_{SYSCLK}$	$16 \times T_{SYSCLK}$	ns
T_{ACH}	Address/Control Hold Time	0	$3 \times T_{SYSCLK}$	ns
T_{ALEH}	Address Latch Enable High Time	$1 \times T_{SYSCLK}$	$4 \times T_{SYSCLK}$	ns
T_{ALEL}	Address Latch Enable Low Time	$1 \times T_{SYSCLK}$	$4 \times T_{SYSCLK}$	ns
T_{WDS}	Write Data Setup Time	$1 \times T_{SYSCLK}$	$19 \times T_{SYSCLK}$	ns
T_{WDH}	Write Data Hold Time	0	$3 \times T_{SYSCLK}$	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).				

SILICON LABS

19.3. Clock Multiplier

The Clock Multiplier generates an output clock which is 4 times the input clock frequency scaled by a programmable factor of 1, 2/3, 2/4 (or 1/2), 2/5, 2/6 (or 1/3), or 2/7. The Clock Multiplier's input can be selected from the external oscillator, or the internal or external oscillators divided by 2. This produces three possible base outputs which can be scaled by a programmable factor: Internal Oscillator x 2, External Oscillator x 2, or External Oscillator x 4. See Section 19.1 on page 176 for details on system clock selection.

The Clock Multiplier is configured via the CLKMUL register (SFR Definition 19.5). The procedure for configuring and enabling the Clock Multiplier is as follows:

1. Reset the Multiplier by writing 0x00 to register CLKMUL.
2. Select the Multiplier input source via the MULSEL bits.
3. Select the Multiplier output scaling factor via the MULDIV bits
4. Enable the Multiplier with the MULEN bit (CLKMUL | = 0x80).
5. Delay for >5 µs.
6. Initialize the Multiplier with the MULINIT bit (CLKMUL | = 0xC0).
7. Poll for MULRDY => 1.

Important Note: When using an external oscillator as the input to the Clock Multiplier, the external source must be enabled and stable before the Multiplier is initialized. See "19.4. External Oscillator Drive Circuit" on page 183 for details on selecting an external oscillator source.

The Clock Multiplier allows faster operation of the CIP-51 core and is intended to generate an output frequency between 25 and 50 MHz. The clock multiplier can also be used with slow input clocks. However, if the clock is below the minimum Clock Multiplier input frequency (FCM_{min}), the generated clock will consist of four fast pulses followed by a long delay until the next input clock rising edge. The average frequency of the output is equal to 4x the input, but the instantaneous frequency may be faster. See Figure 19.2 below for more information.
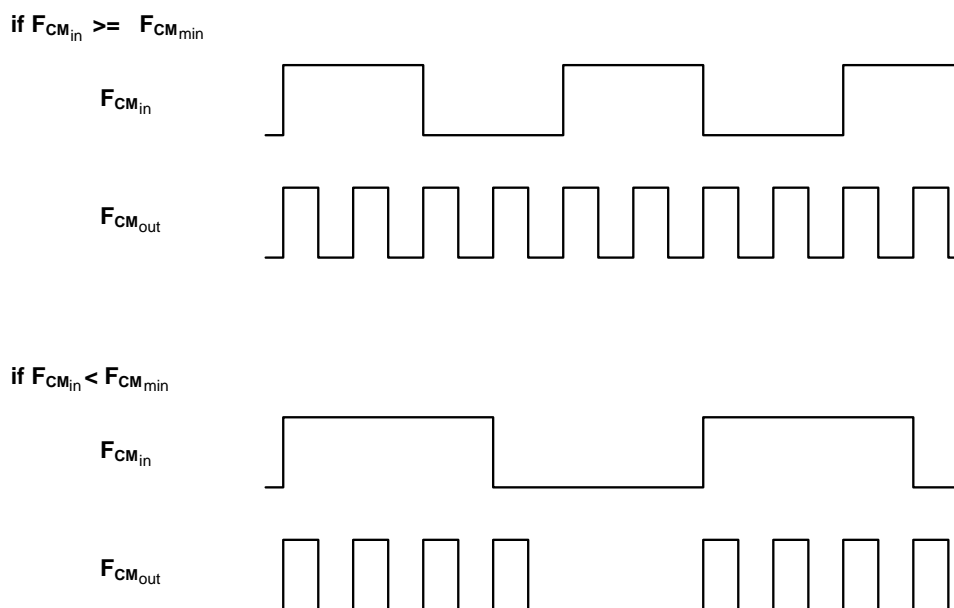

Figure 19.2. Example Clock Multiplier Output

SILICON LABS

SFR Definition 19.6. OSCXCN: External Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	XTLVLD	XOSCMD[2:0]				XFCN[2:0]		
Type	R	R/W			R	R/W		
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9F; SFR Page = 0x0F;

Bit	Name	Function
7	XTLVLD	**Crystal Oscillator Valid Flag.** (Read only when XOSCMD = 11x.) 0: Crystal Oscillator is unused or not yet stable. 1: Crystal Oscillator is running and stable.
6:4	XOSCMD[2:0]	**External Oscillator Mode Select.** 00x: External Oscillator circuit off. 010: External CMOS Clock Mode. 011: External CMOS Clock Mode with divide by 2 stage. 100: RC Oscillator Mode. 101: Capacitor Oscillator Mode. 110: Crystal Oscillator Mode. 111: Crystal Oscillator Mode with divide by 2 stage.
3	Unused	Read = 0b; Write =0b
2:0	XFCN[2:0]	**External Oscillator Frequency Control Bits.** Set according to the desired frequency for Crystal or RC mode. Set according to the desired K Factor for C mode. {table below}

XFCN	Crystal Mode	RC Mode	C Mode
000	f $\leq$ 32 kHz	f $\leq$ 25 kHz	K Factor = 0.87
001	32 kHz $<$ f $\leq$ 84 kHz	25 kHz $<$ f $\leq$ 50 kHz	K Factor = 2.6
010	84 kHz $<$ f $\leq$ 225 kHz	50 kHz $<$ f $\leq$ 100 kHz	K Factor = 7.7
011	225 kHz $<$ f $\leq$ 590 kHz	100 kHz $<$ f $\leq$ 200 kHz	K Factor = 22
100	590 kHz $<$ f $\leq$ 1.5 MHz	200 kHz $<$ f $\leq$ 400 kHz	K Factor = 65
101	1.5 MHz $<$ f $\leq$ 4 MHz	400 kHz $<$ f $\leq$ 800 kHz	K Factor = 180
110	4 MHz $<$ f $\leq$ 10 MHz	800 kHz $<$ f $\leq$ 1.6 MHz	K Factor = 664
111	10 MHz $<$ f $\leq$ 30 MHz	1.6 MHz $<$ f $\leq$ 3.2 MHz	K Factor = 1590

SILICON LABS

SFR Definition 20.3. XBR2: Port I/O Crossbar Register 2

Bit	7	6	5	4	3	2	1	0
Name	WEAKPUD	XBARE	Reserved		CP2AE	CP2E	URT1E	LIN0E
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC7; SFR Page = 0x0F

Bit	Name	Function
7	WEAKPUD	**Port I/O Weak Pullup Disable.** 0: Weak Pullups enabled (except for Ports whose I/O are configured for analog mode). 1: Weak Pullups disabled.
6	XBARE	**Crossbar Enable.** 0: Crossbar disabled. 1: Crossbar enabled.
5:4	Reserved	Always Write to 00b.
3	CP2AE	**Comparator2 Asynchronous Output Enable.** 0: Asynchronous CP2 unavailable at Port pin. 1: Asynchronous CP2 routed to Port pin.
2	CP2E	**Comparator2 Output Enable.** 0: CP2 unavailable at Port pin. 1: CP2 routed to Port pin.
1	URT1E	**UART1 I/O Output Enable.** 0: UART1 I/O unavailable at Port pin. 1: UART1 TX0, RX0 routed to Port pins.
0	LIN0E	**LIN I/O Output Enable.** 0: LIN I/O unavailable at Port pin. 1: LIN_TX, LIN_RX routed to Port pins.

22.2. CAN Registers

CAN registers are classified as follows:

1. **CAN Controller Protocol Registers**: CAN control, interrupt, error control, bus status, test modes.

2. **Message Object Interface Registers**: Used to configure 32 Message Objects, send and receive data to and from Message Objects. The CIP-51 MCU accesses the CAN message RAM via the Message Object Interface Registers. Upon writing a message object number to an IF1 or IF2 Command Request Register, the contents of the associated Interface Registers (IF1 or IF2) will be transferred to or from the message object in CAN RAM.

3. **Message Handler Registers**: These read only registers are used to provide information to the CIP-51 MCU about the message objects (MSGVLD flags, Transmission Request Pending, New Data Flags) and Interrupts Pending (which Message Objects have caused an interrupt or status interrupt condition).

For the registers other than CAN0CFG, refer to the Bosch CAN User's Guide for information on the function and use of the CAN Control Protocol Registers.

22.2.1. CAN Controller Protocol Registers

The CAN Control Protocol Registers are used to configure the CAN controller, process interrupts, monitor bus status, and place the controller in test modes.

The registers are: CAN Control Register (CAN0CN), CAN Clock Configuration (CAN0CFG), CAN Status Register (CAN0STA), CAN Test Register (CAN0TST), Error Counter Register, Bit Timing Register, and the Baud Rate Prescaler (BRP) Extension Register.

22.2.2. Message Object Interface Registers

There are two sets of Message Object Interface Registers used to configure the 32 Message Objects that transmit and receive data to and from the CAN bus. Message objects can be configured for transmit or receive, and are assigned arbitration message identifiers for acceptance filtering by all CAN nodes.

Message Objects are stored in Message RAM, and are accessed and configured using the Message Object Interface Registers.

22.2.3. Message Handler Registers

The Message Handler Registers are read only registers. The message handler registers provide interrupt, error, transmit/receive requests, and new data information.

23.4.3. 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, as the interface may be in the process of shifting a byte of data into or out of the register.

Data in SMB0DAT is always shifted out MSB first. After a byte has been received, the first bit of received data is located at the MSB of SMB0DAT. While data is being shifted out, data on the bus is simultaneously being shifted in. SMB0DAT always contains the last data byte present on the bus. In the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data or address in SMB0DAT.

SFR Definition 23.3. SMB0DAT: SMBus Data

Bit	7	6	5	4	3	2	1	0
Name	SMB0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC2; SMB0DAT = 0x00

Bit	Name	Function
7:0	SMB0DAT[7:0]	**SMBus 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 read from or write to this register whenever the SI serial interrupt flag (SMB0CN.0) 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.

23.5. SMBus Transfer 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. As a receiver, the interrupt for an ACK occurs **before** the ACK. As a transmitter, interrupts occur **after** the ACK.

C8051F58x/F59x

$$\text{Baud Rate} = \frac{\text{SYSCLK}}{(65536 - (\text{SBRLH0:SBRLL0}))} \times \frac{1}{2} \times \frac{1}{\text{Prescaler}}$$

Equation 24.1. UART0 Baud Rate

A quick reference for typical baud rates and clock frequencies is given in Table 24.1.

Table 24.1. Baud Rate Generator Settings for Standard Baud Rates

	Target Baud Rate (bps)	Actual Baud Rate (bps)	Baud Rate Error	Oscillator Divide Factor	SB0PS[1:0] (Prescaler Bits)	Reload Value in SBRLH0:SBRLL0
SYSCLK = 48	230400	230769	0.16%	208	11	0xFF98
	115200	115385	0.16%	416	11	0xFF30
	57600	57554	0.08%	834	11	0xFE5F
	28800	28812	0.04%	1666	11	0xFCBF
	14400	14397	0.02%	3334	11	0xF97D
	9600	9600	0.00%	5000	11	0xF63C
	2400	2400	0.00%	20000	11	0xD8F0
	1200	1200	0.00%	40000	11	0xB1E0
SYSCLK = 24	230400	230769	0.16%	104	11	0xFFCC
	115200	115385	0.16%	208	11	0xFF98
	57600	57692	0.16%	416	11	0xFF30
	28800	28777	0.08%	834	11	0xFE5F
	14400	14406	0.04%	1666	11	0xFCBF
	9600	9600	0.00%	2500	11	0xFB1E
	2400	2400	0.00%	10000	11	0xEC78
	1200	1200	0.00%	20000	11	0xD8F0
SYSCLK = 12	230400	230769	0.16%	52	11	0xFFE6
	115200	115385	0.16%	104	11	0xFFCC
	57600	57692	0.16%	208	11	0xFF98
	28800	28846	0.16%	416	11	0xFF30
	14400	14388	0.08%	834	11	0xFE5F
	9600	9600	0.00%	1250	11	0xFD8F
	2400	2400	0.00%	5000	11	0xF63C
	1200	1200	0.00%	10000	11	0xEC78

SILICON LABS

26.3. SPI0 Slave Mode Operation

When SPI0 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 a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted through the shift register, the SPIF flag is set to logic 1, and the byte is copied into the receive buffer. Data is read from the receive buffer by reading SPI0DAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the shift register by writing to SPI0DAT. Writes to SPI0DAT are double-buffered, and are placed in the transmit buffer first. If the shift register is empty, the contents of the transmit buffer will immediately be transferred into the shift register. When the shift register already contains data, the SPI will load the shift register with the transmit buffer's contents after the last SCK edge of the next (or current) SPI transfer.

When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. The default, 4-wire slave mode, is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In 4-wire mode, the NSS signal is routed to a port pin and configured as a digital input. SPI0 is enabled when NSS is logic 0, and disabled when NSS is logic 1. The bit counter is reset on a falling edge of NSS. Note that the NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 26.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and 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, SPI0 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 SPI0 with the SPIEN bit. Figure 26.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

26.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

All of the following bits must be cleared by software.

1. The SPI Interrupt Flag, SPIF (SPI0CN.7) is set to logic 1 at the end of each byte transfer. This flag can occur in all SPI0 modes.
2. The Write Collision Flag, WCOL (SPI0CN.6) is set to logic 1 if a write to SPI0DAT is attempted when the transmit buffer has not been emptied to the SPI shift register. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written.This flag can occur in all SPI0 modes.
3. The Mode Fault Flag MODF (SPI0CN.5) is set to logic 1 when SPI0 is configured as a master, and for multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTEN and SPIEN bits in SPI0CN are set to logic 0 to disable SPI0 and allow another master device to access the bus.
4. The Receive Overrun Flag RXOVRN (SPI0CN.4) is set to logic 1 when configured as a slave, and a transfer is completed and the receive buffer still holds an unread byte from a previous transfer. The new byte is not transferred to the receive buffer, allowing the previously received data byte to be read. The data byte which caused the overrun is lost.

SFR Definition 26.2. SPI0CN: SPI0 Control

Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSMD[1:0]		TXBMT	SPIEN
Type	R/W	R/W	R/W	R/W	R/W		R	R/W
Reset	0	0	0	0	0	1	1	0

SFR Address = 0xF8; Bit-Addressable; SFR Page = 0x00

Bit	Name	Function
7	SPIF	**SPI0 Interrupt Flag.** This bit is set to logic 1 by hardware at the end of a data transfer. If interrupts are enabled, setting this bit causes the CPU to vector to the SPI0 interrupt service routine. This bit is not automatically cleared by hardware. It must be cleared by software.
6	WCOL	**Write Collision Flag.** This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) to indicate a write to the SPI0 data register was attempted while a data transfer was in progress. It must be cleared by software.
5	MODF	**Mode Fault Flag.** This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). This bit is not automatically cleared by hardware. It must be cleared by software.
4	RXOVRN	**Receive Overrun Flag (valid in slave mode only).** This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) 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. This bit is not automatically cleared by hardware. It must be cleared by software.
3:2	NSSMD[1:0]	**Slave Select Mode.** Selects between the following NSS operation modes: (See Section 26.2 and Section 26.3). 00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin. 01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device. 1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0.
1	TXBMT	**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	**SPI0 Enable.** 0: SPI disabled. 1: SPI enabled.

SFR Definition 27.11. TMR2L: Timer 2 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCC; SFR Page = 0x00

Bit	Name	Function
7:0	TMR2L[7:0]	**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.

SFR Definition 27.12. TMR2H Timer 2 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCD; SFR Page = 0x00

Bit	Name	Function
7:0	TMR2H[7:0]	**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.

SILICON LABS

28.2. PCA0 Interrupt Sources

Figure 28.3 shows a diagram of the PCA0 interrupt tree. There are five independent event flags that can be used to generate a PCA0 interrupt. They are as follows: the main PCA0 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, 9th, 10th, or 11th bit of the PCA0 counter, and the individual flags for each PCA0 channel (CCF0, CCF1, CCF2, CCF3, CCF4 and CCF5), 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.
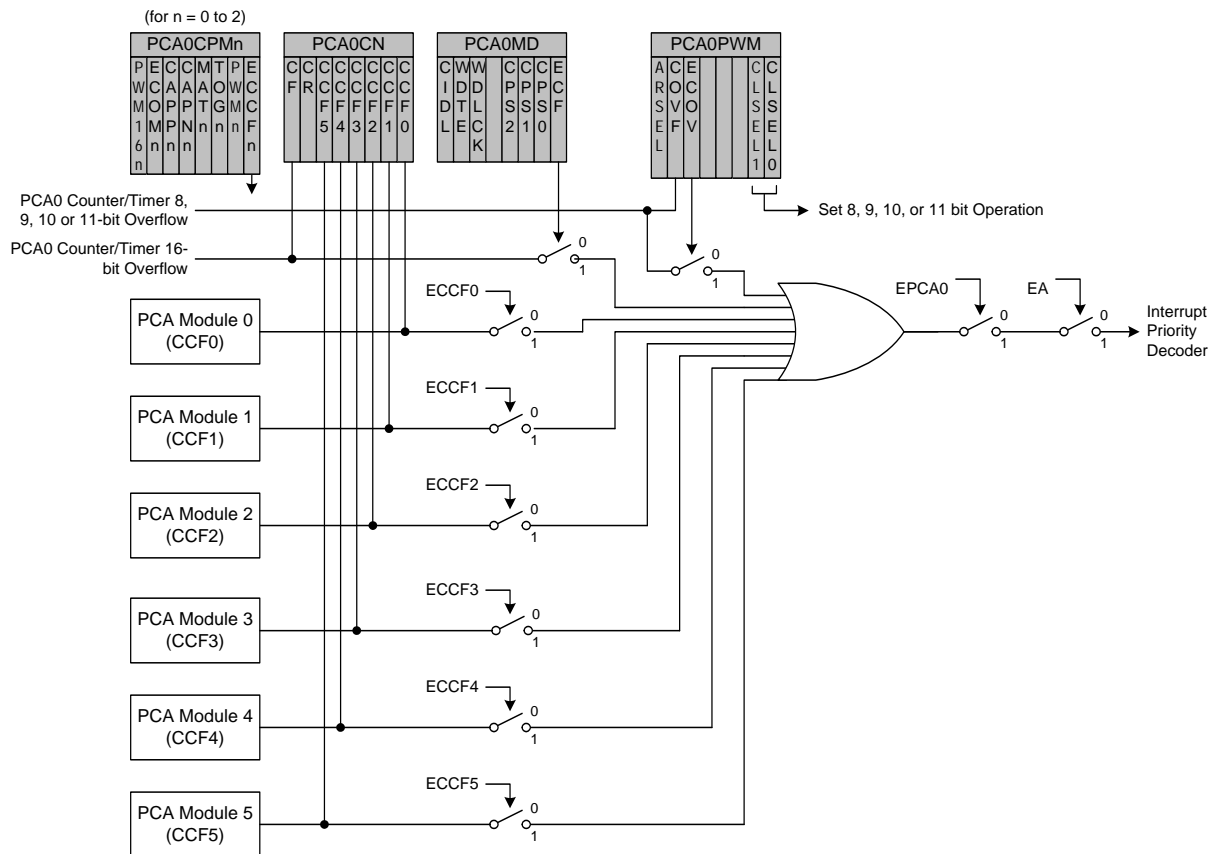

Figure 28.3. PCA0 Interrupt Block Diagram

29. Programmable Counter Array 1 (PCA1)

The Programmable Counter Array (PCA1) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. PCA1 consists of a dedicated 16-bit counter/timer and six 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between eight sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0, 4, or 5 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section "29.3. Capture/Compare Modules" on page 336). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing PCA1 to be clocked by a precision external oscillator while the internal oscillator drives the system clock. PCA1 is configured and controlled through the system controller's Special Function Registers. The PCA1 block diagram is shown in Figure 29.1

Note: PCA0 and PCA1 are fully independent peripherals. PCA0 offers channels CEX0 - CEX5, and PCA1 offers channels CEX6-CEX11. PCA0 and PCA1 are identical, except that PCA0 Module 5 may be used as a watchdog timer.
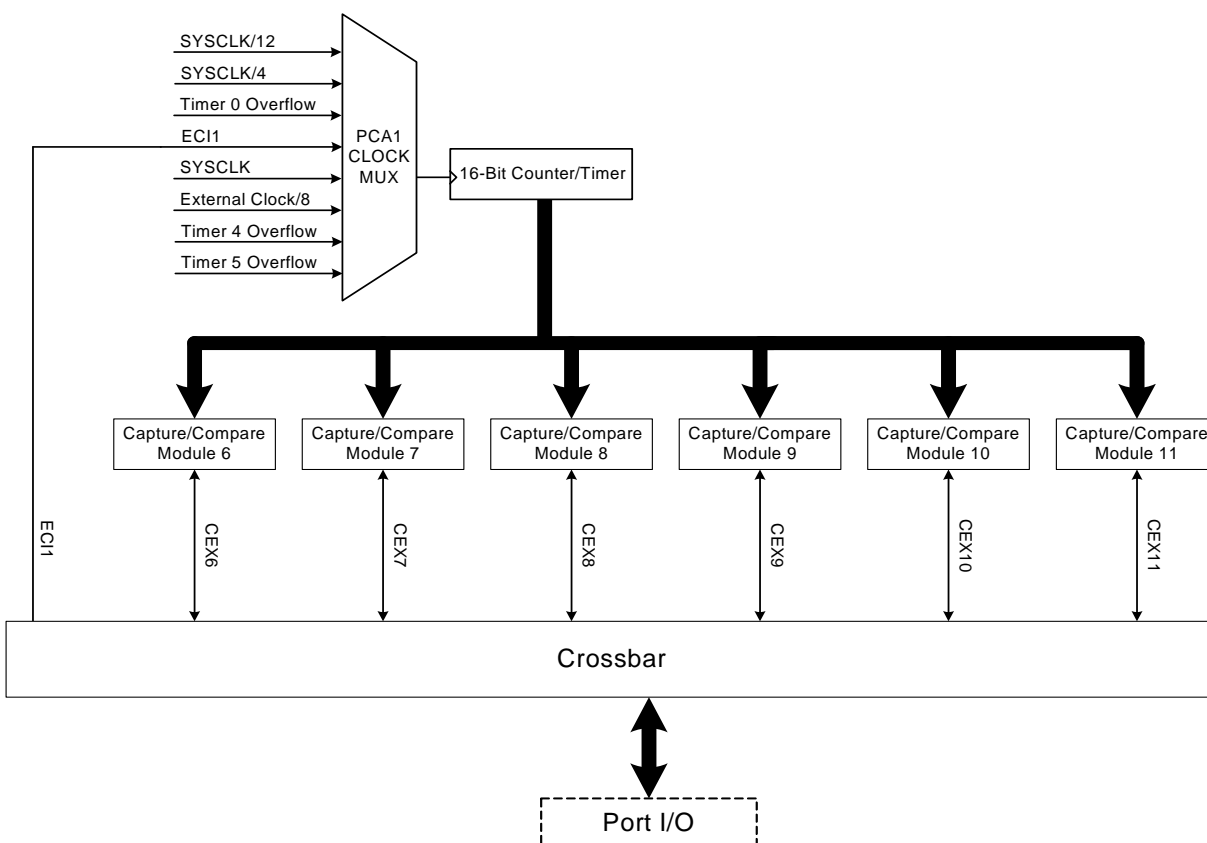

Figure 29.1. PCA1 Block Diagram

29.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes PCA1 to capture the value of the PCA1 counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA1CPLn and PCA1CPHn). The CAPP1n and CAPN1n bits in the PCA1CPMn 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 PCA1CN 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 CAPP1n and CAPN1n 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.
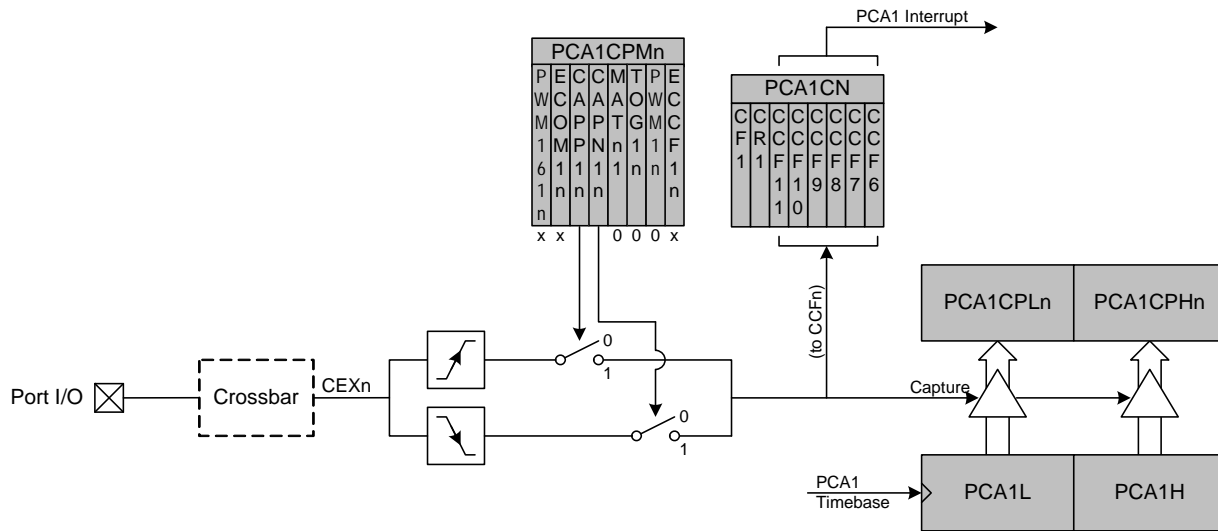

Figure 29.4. PCA1 Capture Mode Diagram

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