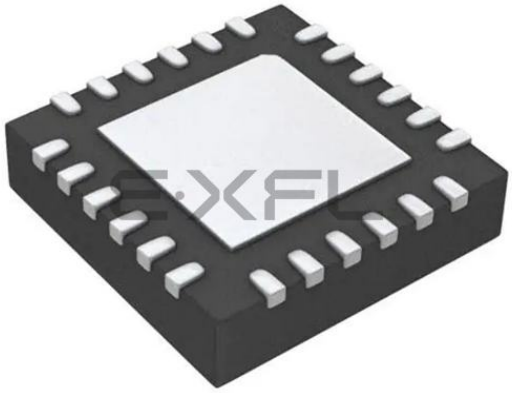

Welcome to E-XFL.COM

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	Active
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	21
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	24-WFQFN Exposed Pad
Supplier Device Package	24-QFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f395-a-gmr

5. QFN-20 Package Specifications

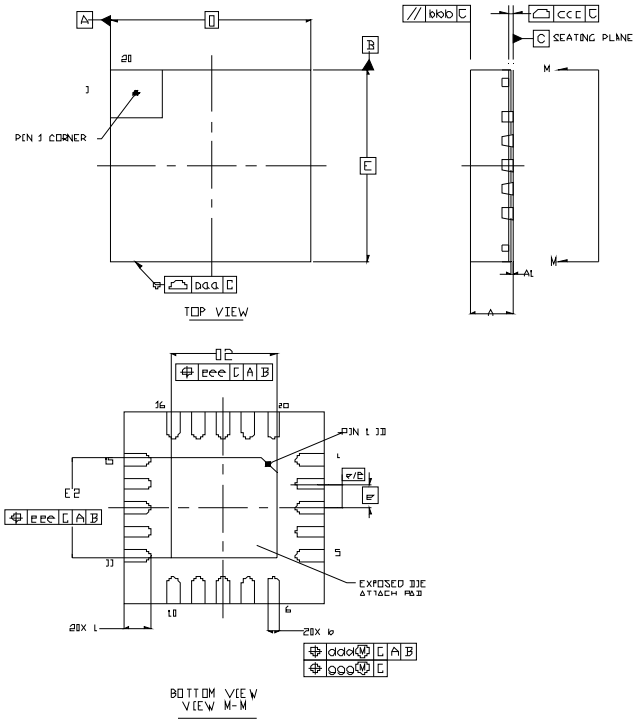

Figure 5.1. QFN-20 Package Drawing

Table 5.1. QFN-20 Package Dimensions

Dimension	Min	Typ	Max		Dimension	Min	Typ	Max
A	0.80	0.85	0.90		L	0.50	0.55	0.60
A1	0.00	0.035	0.05		aaa	—	—	0.10
b	0.20	0.25	0.30		bbb	—	—	0.10
D	4.00 BSC.				ccc	—	—	0.08
D2	2.00	2.10	2.20		ddd	—	—	0.10
e	0.50 BSC.				eee	—	—	0.10
E	4.00 BSC.				ggg			0.05
E2	2.00	2.10	2.20					

Notes:

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This drawing conforms to the JEDEC Solid State Outline MO-220, variation VGGD except for custom features D2, E2, Z, Y, and L which are toleranced per supplier designation.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

9.1. Output Code 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 AD0LJST bit (ADC0CN.0). When in Single-ended Mode, conversion codes are represented as 10-bit unsigned integers. Inputs are measured from 0 to VREF x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADC0H and ADC0L registers are set to 0.

Input Voltage (Single-Ended)	Right-Justified ADC0H:ADC0L (AD0LJST = 0)	Left-Justified ADC0H:ADC0L (AD0LJST = 1)
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 2s complement numbers. Inputs are measured from –VREF to VREF x 511/512. Example codes are shown below for both right-justified and left-justified data. 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.

Input Voltage (Differential)	Right-Justified ADC0H:ADC0L (AD0LJST = 0)	Left-Justified ADC0H:ADC0L (AD0LJST = 1)
VREF x 511/512	0x01FF	0x7FC0
VREF x 256/512	0x0100	0x4000
0	0x0000	0x0000
–VREF x 256/512	0xFF00	0xC000
–VREF	0xFE00	0x8000

16. Prefetch Engine

The C8051F39x/37x family of devices incorporate a 2-byte prefetch engine. Because the access time of the Flash memory is 40 ns, and the minimum instruction time is roughly 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.

Note: The prefetch engine should be disabled when the device is in suspend mode to save power.

SFR Definition 16.1. PFE0CN: Prefetch Engine Control

Bit	7	6	5	4	3	2	1	0
Name			PFEN					
Type	R	R	R/W	R	R	R	R	R
Reset	0	0	1	0	0	0	0	0

SFR Address = 0xB5; SFR Page = All Pages

Bit	Name	Function
7:6	Unused	Unused. Read = 00b, Write = don't care.
5	PFEN	**Prefetch Enable.** This bit enables the prefetch engine. 0: Prefetch engine is disabled. 1: Prefetch engine is enabled.
4:0	Unused	Unused. Read = 00000b. Write = don't care.

On exit from the PCA0 interrupt service routine, the CIP-51 will return to the SPI0 ISR. On execution of the RETI instruction, SFR page 0x00 used to access the PCA0 registers will be automatically popped off of the SFR page stack, and the contents at the SFRPGIDX = 001b location will be moved to the SFRPAGE register. Software in the SPI0 ISR can continue to access SFRs as it did prior to the PCA interrupt. Likewise, the contents at the SFRPGIDX = 010b location are moved to the SFRPGIDX = 001b location. Recall this was the SFR Page value 0x0F being used to access TS0CN before the SPI0 interrupt occurred. See Figure 19.5.

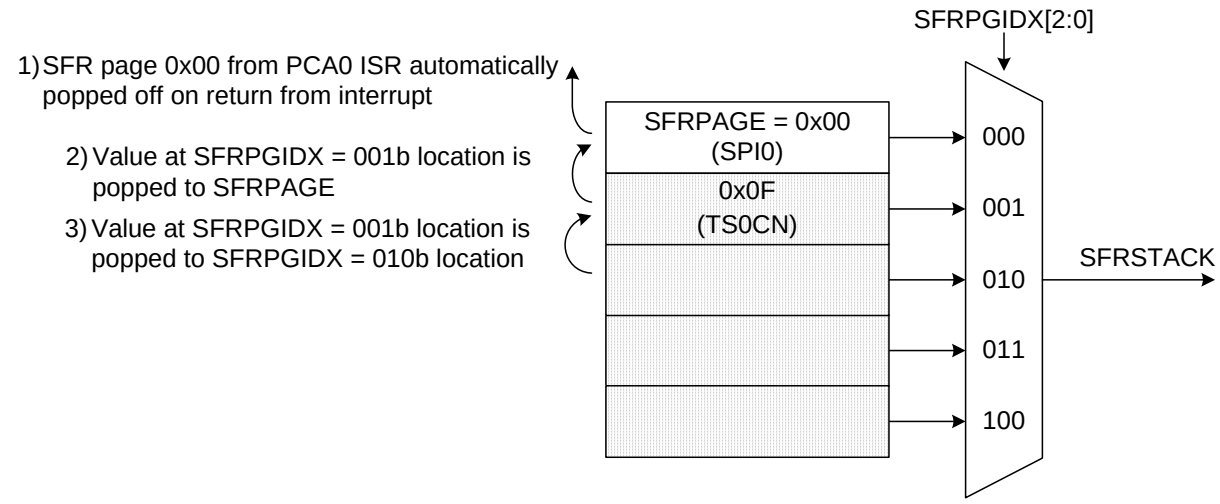

Figure 19.5. SFR Page Stack Upon Return from PCA0 Interrupt

SFR Definition 19.2. SFRPGCN: SFR Page Control

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

SFR Address = 0xCF; SFR Page = All Pages

Bit	Name	Function
7	Reserved	Must Write 0b
6:4	SFRPGIDX[2:0]	**SFR Page Stack Index.** This field can be used to access the SFRPAGE values stored in the SFR page stack. It selects which level of the stack is accessible when reading the SFRSTACK register. 000: SFRSTACK contains the value of SFRPAGE, the first/top byte of the SFR page stack 001: SFRSTACK contains the value of the second byte of the SFR page stack 010: SFRSTACK contains the value of the third byte of the SFR page stack 011: SFRSTACK contains the value of the forth byte of the SFR page stack 100: SFRSTACK contains the value of the fifth/bottom byte of the SFR page stack 101: Invalid index 11x: Invalid index
3:1	Reserved	Must Write 000b
0	SFRPGEN	**SFR Automatic Page Control Enable.** This bit is used to enable automatic page switching on ISR entry/exit. When set to 1, the current SFRPAGE value will be pushed onto the SFR page stack, and SFRPAGE will be set to the page corresponding to the flag which generated the interrupt; upon ISR exit, hardware will pop the value from the SFR page stack and restore SFRPAGE. 0: Disable automatic SFR paging. 1: Enable automatic SFR paging.

SFR Definition 20.6. EIP1H: Extended Interrupt Priority 1 High

Bit	7	6	5	4	3	2	1	0
Name	PHT3		PHCP0	PHPCA0	PHADC0	PHWADC0	PHMAT	PHSMB0
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 = 0x85; SFR Page = All Pages

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

21. Flash Memory

On-chip, re-programmable Flash memory is included for program code and non-volatile data storage. The Flash memory can be programmed in-system, a single byte at a time, through the C2 interface or by software using the MOVX instruction. Once cleared to logic 0, a Flash bit must be erased to set it back to logic 1. Flash bytes would typically be erased (set to 0xFF) before being reprogrammed. 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. Refer to Section “7. Electrical Characteristics” on page 32 for complete Flash memory electrical characteristics.

21.1. Programming The Flash Memory

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. This is the only means for programming a non-initialized device. For details on the C2 commands to program Flash memory, see Section “33. C2 Interface” on page 297.

To ensure the integrity of Flash contents, it is strongly recommended that the on-chip V_{DD} Monitor be enabled in any system that includes code that writes and/or erases Flash memory from software. See Section 21.4 for more details.

21.1.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The Flash Lock and Key Register (FLKEY) must be written with the correct key codes, in sequence, before Flash operations may be performed. The key codes are: 0xA5, 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 a following Flash operation can be performed. The FLKEY register is detailed in SFR Definition 21.2.

21.1.2. Flash Erase Procedure

The Flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before writing to Flash memory using MOVX, Flash write operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1 (this directs the MOVX writes to target Flash memory); and (2) Writing the Flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software.

A write to Flash memory can clear bits to logic 0 but cannot set them; only an erase operation can set bits to logic 1 in Flash. **A byte location to be programmed should be erased before a new value is written.** The Flash memory is organized in 512-byte pages. The erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire 512-byte page, perform the following steps:

1. Disable interrupts (recommended).
2. Set thePSEE bit (register PSCTL).
3. Set the PSWE bit (register PSCTL).
4. Write the first key code to FLKEY: 0xA5.
5. Write the second key code to FLKEY: 0xF1.
6. Using the MOVX instruction, write a data byte to any location within the 512-byte page to be erased.
7. Clear the PSWE and PSEE bits.

22.1.1. Slave Address Byte

The master begins a transmission by sending a START condition followed by the slave address byte (SAB).

Slave Address Byte (SAB) Definition

Bit	7	6	5	4	3	2	1	0
Name	SLA						ADDR MSB	R/W
Value	1	0	1	0	0	0	Varies	Varies

Bit	Name	Function
7:2	SLA	**Slave Address of EEPROM.** Always 101000b.
1	ADDR MSB	**Most Significant Addressing Bit.** This bit is concatenated to the 8-bit address counter to create a 9-bit address used by EEPROM read and write operations. 0: Address locations 0x000 to 0x0FF are targeted by the EEPROM operations. 1: Address locations 0x100 to 0x1FF are targeted by the EEPROM operations.
0	R/W	**EEPROM Read/Write Direction Bit.** Instructs the EEPROM to perform a read or write operation 0: Perform an EEPROM write operation 1: Perform an EEPROM read operation

Figure 22.1. Slave Address Byte Definition

22.1.2. Acknowledgement (ACK)

During an acknowledgement (ACK), the master or EEPROM forces the EESDA line to a logic low when EESCL is logic high.

22.1.3. Not-Acknowledgement (NACK)

During a not-acknowledgement (NACK), the master or EEPROM allows the EESDA line to be pulled up to a logic high when EESCL is logic high.

22.1.4. Reset

The EEPROM can be reset in case the SMBus communication is accidentally interrupted (e.g. power loss) or needs to be terminated mid-stream. The reset is initialized when the master device creates a START condition. To do this, it may be necessary for the master device to monitor EESDA up to nine times while cycling the EESCL signal. During this process, the master checks for a logic high on EESDA for each rising edge of EESCL.

22.3. Read Operation

There are two operations to read the EEPROM: current address read and selective address read. Both read operations can read up to 256 bytes within a single read operation.

22.3.1. Current Address Read

A current address read accesses the data at the EEPROM internal address counter's current location.

The address counter in the EEPROM maintains the address of the last byte accessed, incremented by one. For example, if the previous operation was a read or write operation addressed to address location n, the internal address counter automatically increments to address n+1.

To perform a current address read operation:

1. The master sends the START condition and the slave address byte with the R/W bit set to 1.
2. The EEPROM generates an ACK and transmits the byte of data (D[7:0]) stored at the address specified by the address counter. This address will be the address from the last read or write operation incremented by one.
3. The EEPROM increments the internal address counter by one.
4. (Optional) To read additional bytes:
    a. The master generates an ACK.
    b. The EEPROM transmits the byte of data stored at the address specified by the address counter.
    c. The EEPROM increments the internal address counter by one.
    d. Repeat Step 4a through 4c until the master is done reading bytes.
5. The master generates a NACK.
6. The master generates a STOP condition.
7. The EEPROM terminates the transmission.

Note: If the previous operation targeted the last byte of the EEPROM, the EEPROM will transmit the data from address location 0x00 for a current address read operation.

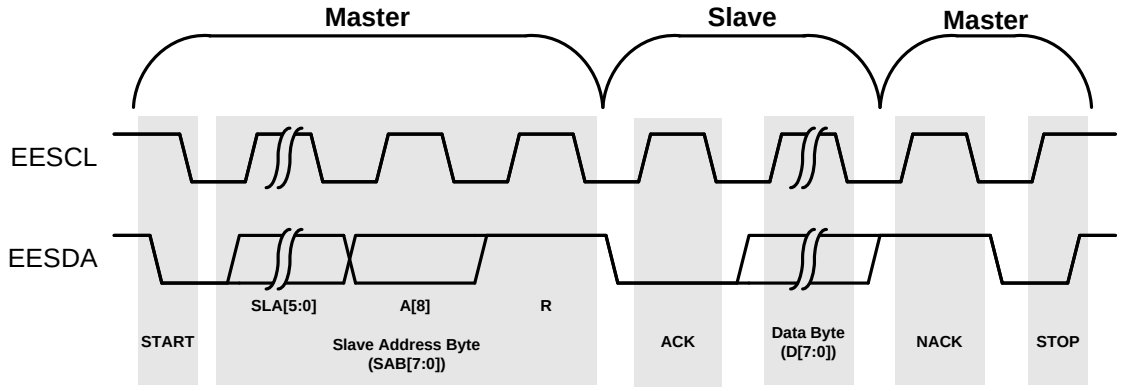

Figure 22.4. Current Address Read Operation (Single Byte)

SFR Definition 23.2. CRC0IN: CRC0 Data Input

Bit	7	6	5	4	3	2	1	0
Name	CRC0IN[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9C; SFR Page = All Pages

Bit	Name	Function
7:0	CRC0IN[7:0]	**CRC0 Data Input.** Each write to CRC0IN results in the written data being computed into the existing CRC result according to the CRC algorithm described in Section 23.1

SFR Definition 23.3. CRC0DAT: CRC0 Data Output

Bit	7	6	5	4	3	2	1	0
Name	CRC0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9E; SFR Page = All Pages

Bit	Name	Function
7:0	CRC0DAT[7:0]	**CRC0 Data Output.** Each read or write performed on CRC0DAT targets the CRC result bits pointed to by the CRC0 Result Pointer (CRC0PNT bits in CRC0CN).

SFR Definition 23.5. CRC0CNT: CRC0 Automatic Flash Sector Count

Bit	7	6	5	4	3	2	1	0
Name	CRCDONE			CRC0CNT[4:0]				
Type	R	R/W		R/W				
Reset	1	0	0	0	0	0	0	0

SFR Address = 0xDE; SFR Page = All Pages

Bit	Name	Function
7	CRCDONE	**CRCDONE Automatic CRC Calculation Complete.** Set to 0 when a CRC calculation is in progress. Code execution is stopped during a CRC calculation; therefore, reads from firmware will always return 1.
6:5	Reserved	Must write 00b.
4:0	CRC0CNT[4:0]	**Automatic CRC Calculation Block Count.** These bits specify the number of Flash blocks to include in an automatic CRC calculation. The last address of the last Flash block included in the automatic CRC calculation is (CRC0ST+CRC0CNT) x Block Size - 1. **Notes:** **1.** The block size is 256 bytes. **2.** The maximum number of blocks that may be computed in a single operation is 31. To compute a CRC on all 32 blocks, perform one operation on 31 blocks, then perform a second operation on 1 block without clearing the CRC result.

27.2. Assigning Port I/O Pins to Analog and Digital Functions

Port I/O pins P0.0 - P2.3 can be assigned to various analog, digital, and external interrupt functions. The Port pins assigned to analog functions should be configured for analog I/O, and Port pins assigned to digital or external interrupt functions should be configured for digital I/O.

27.2.1. Assigning Port I/O Pins to Analog Functions

Table 27.1 shows all available analog functions that require Port I/O assignments. **Port pins selected for these analog functions should have their corresponding bit in PnSKIP set to '1'.** This reserves the pin for use by the analog function and does not allow it to be claimed by the Crossbar. Table 27.1 shows the potential mapping of Port I/O to each analog function.

Table 27.1. Port I/O Assignment for Analog Functions

Analog Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
ADC Input	P0.0 - P2.3	AMX0P, AMX0N, PnSKIP, PnMDIN
Comparator0 Input	P0.0 - P2.3	CPT0MX, PnSKIP, PnMDIN
Voltage Reference (VREF0)	P0.0	REF0CN, PnSKIP, PnMDIN
Current DAC Output (IDA0)	P0.1	IDA0CN, PnSKIP, PnMDIN
Current DAC Output (IDA1)	P1.0 (20-pin devices) P1.2 (24-pin devices)	IDA1CN, PnSKIP, PnMDIN
External Oscillator in Crystal Mode (XTAL1)	P0.2	OSCXCN, PnSKIP, PnMDIN
External Oscillator in RC, C, or Crystal Mode (XTAL2)	P0.3	OSCXCN, PnSKIP, PnMDIN

SFR Definition 27.13. P1MDOUT: Port 1 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P1MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA5; SFR Page = All Pages

Bit	Name	Function
7:0	P1MDOUT[7:0]	**Output Configuration Bits for P1.7–P1.0 (respectively).** These bits are ignored if the corresponding bit in register P1MDIN is logic 0. 0: Corresponding P1.n Output is open-drain. 1: Corresponding P1.n Output is push-pull.

SFR Definition 27.14. P1SKIP: Port 1 Skip

Bit	7	6	5	4	3	2	1	0
Name	P1SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD5; SFR Page = All Pages

Bit	Name	Function
7:0	P1SKIP[7:0]	**Port 1 Crossbar Skip Enable Bits.** These bits select Port 1 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P1.n pin is not skipped by the Crossbar. 1: Corresponding P1.n pin is skipped by the Crossbar.

Table 28.6. SMBus Status Decoding: Hardware ACK Enabled (EHACK = 1) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Receiver	1000	0	0	1	A master data byte was received; ACK sent.	Set ACK for next data byte; Read SMB0DAT.	0	0	1	1000
						Set NACK to indicate next data byte as the last data byte; Read SMB0DAT.	0	0	0	1000
						Initiate repeated START.	1	0	0	1110
						Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	X	1100
		0	0	0	A master data byte was received; NACK sent (last byte).	Read SMB0DAT; send STOP.	0	1	0	—
						Read SMB0DAT; Send STOP followed by START.	1	1	0	1110
						Initiate repeated START.	1	0	0	1110
						Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	X	1100
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X	0001
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X	0100
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X	0001
	0101	0	X	X	An illegal STOP or bus error was detected while a Slave Transmission was in progress.	Clear STO.	0	0	X	—

29.2. Operational Modes

UART0 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S0MODE bit (SCON0.7). Typical UART connection options are shown in Figure 29.3.

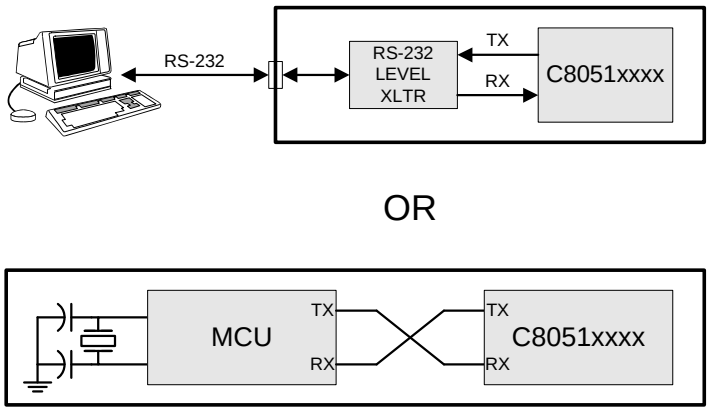

Figure 29.3. UART Interconnect Diagram

29.2.1. 8-Bit UART

8-Bit UART mode uses a total of 10 bits per data byte: one start bit, eight data bits (LSB first), and one stop bit. Data are transmitted LSB first from the TX0 pin and received at the RX0 pin. On receive, the eight data bits are stored in SBUF0 and the stop bit goes into RB80 (SCON0.2).

Data transmission begins when software writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to logic 1. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: RI0 must be logic 0, and if MCE0 is logic 1, the stop bit must be logic 1. In the event of a receive data overrun, the first received 8 bits are latched into the SBUF0 receive register and the following overrun data bits are lost.

If these conditions are met, the eight bits of data is stored in SBUF0, the stop bit is stored in RB80 and the RI0 flag is set. If these conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set. An interrupt will occur if enabled when either TI0 or RI0 is set.

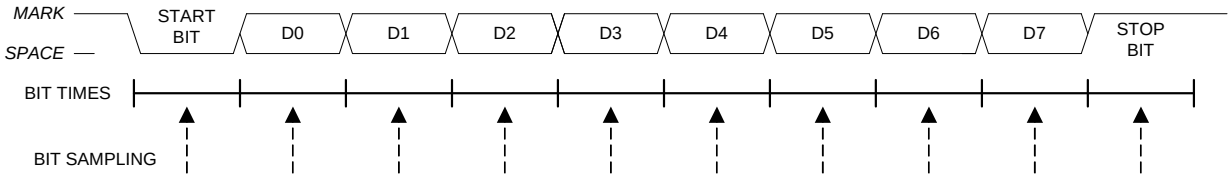

Figure 29.4. 8-Bit UART Timing Diagram

31.2.3. Low-Frequency Oscillator (LFO) Capture Mode

The Low-Frequency Oscillator Capture Mode allows the LFO clock to be measured against the system clock or an external oscillator source. Timer 2 can be clocked from the system clock, the system clock divided by 12, or the external oscillator divided by 8, depending on the T2ML (CKCON.4), and T2XCLK settings.

Setting TF2CEN to 1 enables the LFO Capture Mode for Timer 2. In this mode, T2SPLIT should be set to 0, as the full 16-bit timer is used. Upon a falling edge of the low-frequency oscillator, the contents of Timer 2 (TMR2H:TMR2L) are loaded into the Timer 2 reload registers (TMR2RLH:TMR2RLL) and the TF2H flag is set. By recording the difference between two successive timer capture values, the LFO clock frequency can be determined with respect to the Timer 2 clock. The Timer 2 clock should be much faster than the LFO to achieve an accurate reading.

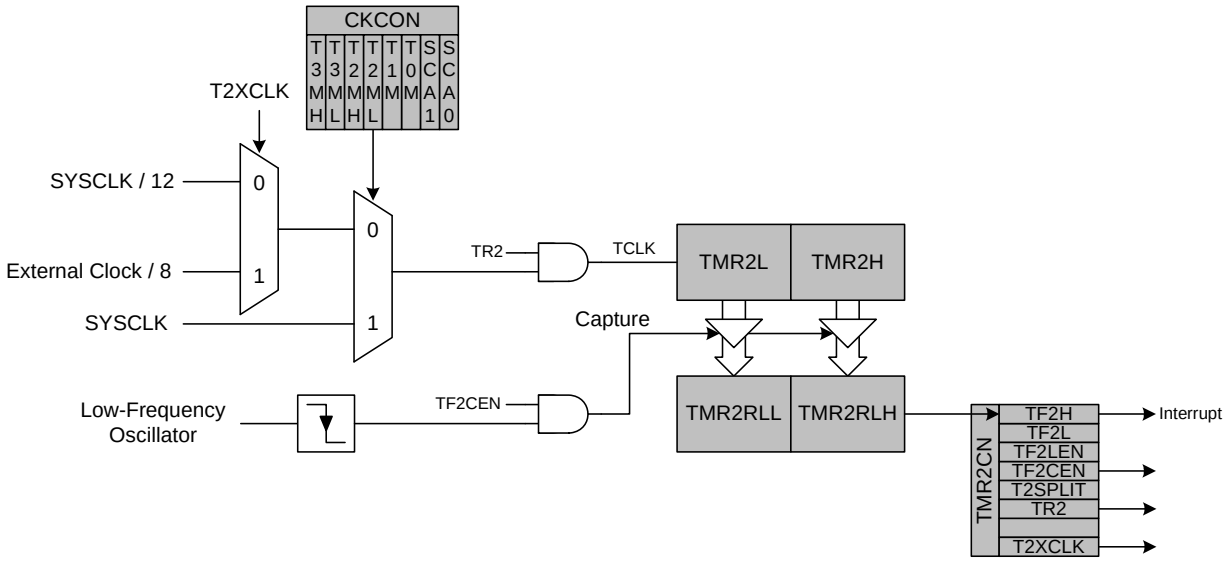

Figure 31.6. Timer 2 Low-Frequency Oscillation Capture Mode Block Diagram

SFR Definition 31.19. TMR4CN: Timer 4 Control

Bit	7	6	5	4	3	2	1	0
Name	TF4H	TF4L	TF4LEN		T4SPLIT	TR4		T4XCLK
Type	R/W	R/W	R/W	R	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x91; SFR Page = F

Bit	Name	Function
7	TF4H	**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 is not automatically cleared by hardware.
6	TF4L	**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 is not automatically cleared by hardware.
5	TF4LEN	**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	Unused	Read = 0b; Write = don't care.
3	T4SPLIT	**Timer 4 Split Mode Enable.** When this bit is set, Timer 4 operates as two 8-bit timers with auto-reload. 0: Timer 4 operates in 16-bit auto-reload mode. 1: Timer 4 operates as two 8-bit auto-reload timers.
2	TR4	**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	Unused	Read = 0b; Write = don't care.
0	T4XCLK	**Timer 4 External Clock Select.** This bit selects the external clock source for Timer 4. However, the Timer 4 Clock Select bits (T4MH and T4ML in register CKCON1) may still be used to select between the external clock and the system clock for either timer. 0: Timer 4 clock is the system clock divided by 12. 1: Timer 4 clock is the external clock divided by 8 (synchronized with SYSCLK).

SFR Definition 31.23. TMR4H Timer 4 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x95; SFR Page = F

Bit	Name	Function
7:0	TMR4H[7:0]	**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.

32.3.3. 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 PCA0CN 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. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

Important Note About Capture/Compare Registers: 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.

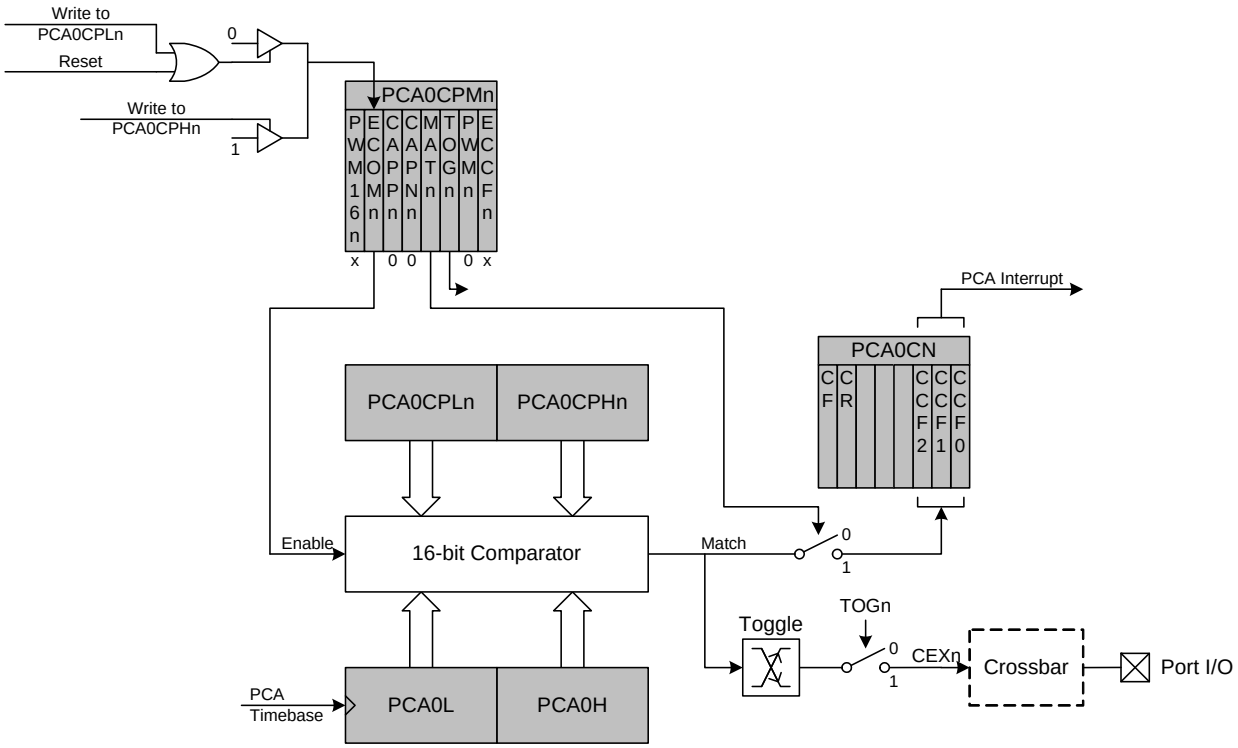

Figure 32.6. PCA High-Speed Output Mode Diagram