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	Obsolete
Core Processor	8051
Core Size	8-Bit
Speed	25MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	16
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.25K x 8
Voltage - Supply (Vcc/Vdd)	2.8V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-TQFP
Supplier Device Package	48-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f019r

Figure 4.1. TQFP-64 Pinout Diagram

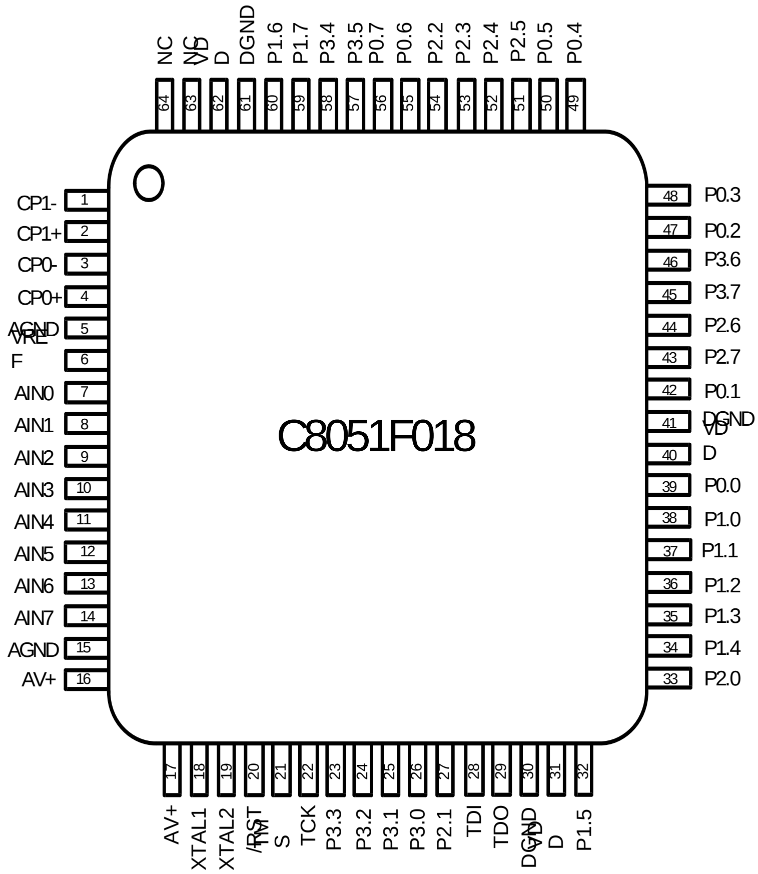
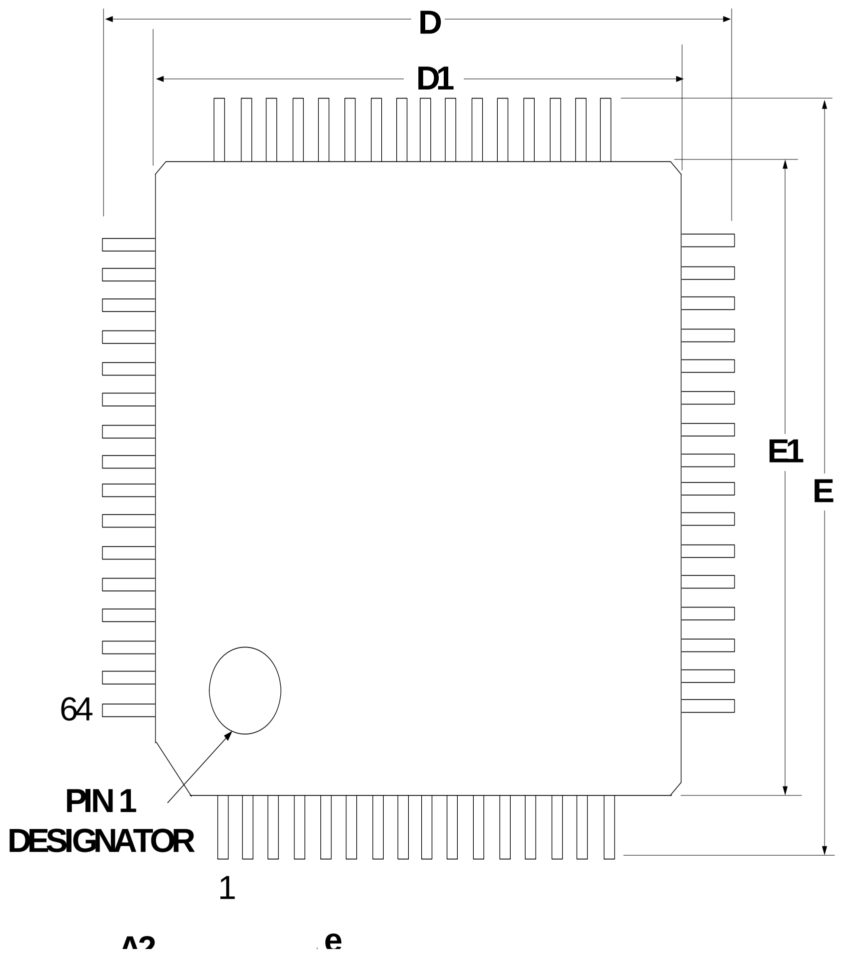


Figure 4.2. TQFP-64 Package Drawing

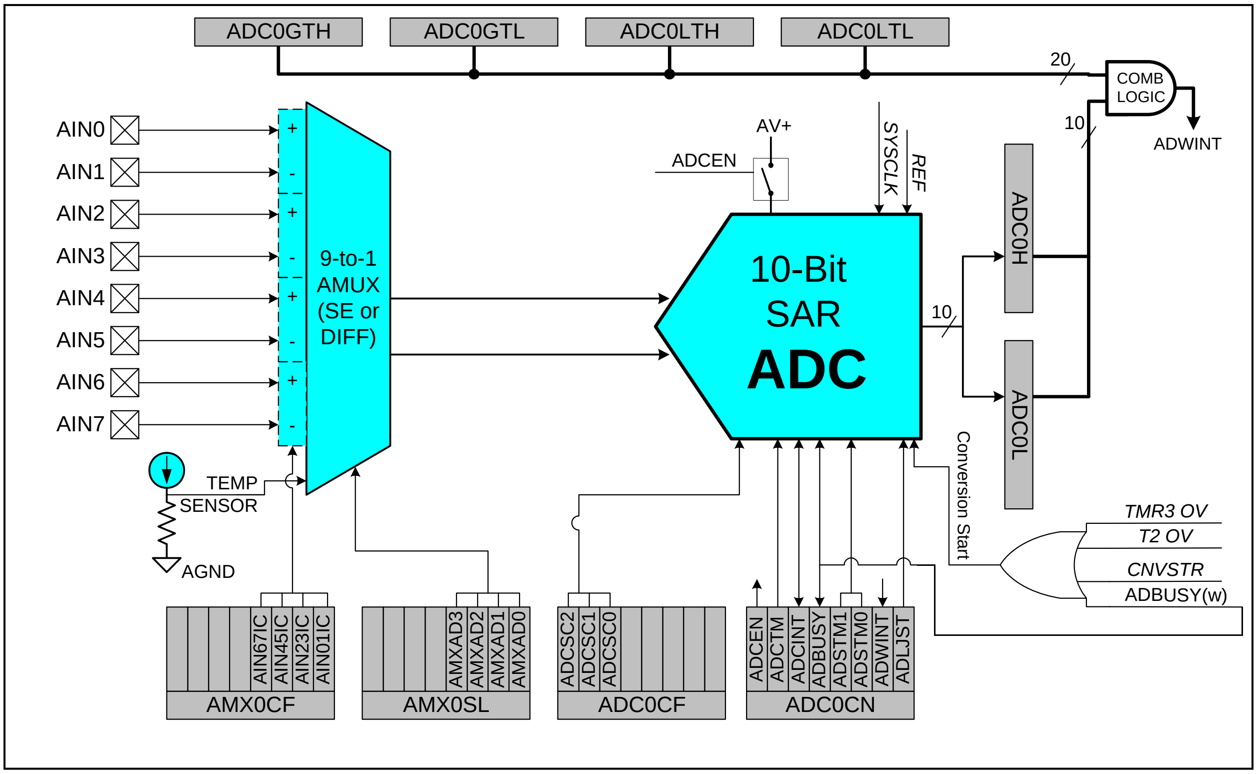


	MIN (mm)	NOM (mm)
A	-	-
A1	0.05	-
A2	0.95	-
b	0.17	0.22
D	-	12.00
D1	-	10.00
e	-	0.50

5. ADC

The ADC subsystem consists of a 9-channel configurable analog multiplexer (AMUX) and a 100ksps, 10-bit successive-approximation-register ADC with integrated track-and-hold and programmable window detector (see block diagram in Figure 5.1). The AMUX, PGA, Data Conversion Modes, and Window Detector are all configurable under software control via the Special Function Register's shown in Figure 5.1. The ADC subsystem (ADC, track-and-hold and PGA) is enabled only when the ADCEN bit in the ADC Control register (ADC0CN, Figure 5.7) is set to 1. The ADC subsystem is in low power shutdown when this bit is 0. The Bias Enable bit (BIASE) in the REF0CN register (see Figure 7.2) must be set to 1 in order to supply bias to the ADC.

Figure 5.1. 10-Bit ADC Functional Block Diagram



5.1. Analog Multiplexer

Eight of the AMUX channels are available for external measurements while the ninth channel is internally connected to an on-board temperature sensor (temperature transfer function is shown in

Figure 5.3). AMUX input pairs can be programmed to operate in either the differential or single-ended mode. This allows the user to select the best measurement technique for each input channel, and even accommodates mode changes "on-the-fly". The AMUX defaults to all single-ended inputs upon reset. There are two registers associated with the AMUX: the Channel Selection register AMX0SL (Figure 5.5), and the Configuration register AMX0CF (Figure 5.4). The table in Figure 5.5 shows AMUX functionality by channel for each possible configuration.

Figure 6.4. CPT1CN: Comparator 1 Control Register

R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CP1EN	CP1OUT	CP1RIF	CP1FIF	CP1HYP1	CP1HYP0	CP1HYN1	CP1HYN0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x9F

Bit7: CP1EN: Comparator 1 Enable Bit
- 0: Comparator 1 Disabled.
- 1: Comparator 1 Enabled.

Bit6: CP1OUT: Comparator 1 Output State Flag
- 0: Voltage on CP1+ < CP1-
- 1: Voltage on CP1+ > CP1-

Bit5: CP1RIF: Comparator 1 Rising-Edge Interrupt Flag
- 0: No Comparator 1 Rising-Edge Interrupt has occurred since this flag was cleared
- 1: Comparator 1 Rising-Edge Interrupt has occurred since this flag was cleared

Bit4: CP1FIF: Comparator 1 Falling-Edge Interrupt Flag
- 0: No Comparator 1 Falling-Edge Interrupt has occurred since this flag was cleared
- 1: Comparator 1 Falling-Edge Interrupt has occurred since this flag was cleared

Bit3-2: CP1HYP1-0: Comparator 1 Positive Hysteresis Control Bits
- 00: Positive Hysteresis Disabled
- 01: Positive Hysteresis = 2mV
- 10: Positive Hysteresis = 4mV
- 11: Positive Hysteresis = 10mV

Bit1-0: CP1HYN1-0: Comparator 1 Negative Hysteresis Control Bits
- 00: Negative Hysteresis Disabled
- 01: Negative Hysteresis = 2mV
- 10: Negative Hysteresis = 4mV
- 11: Negative Hysteresis = 10mV

Mnemonic	Description	Bytes	Clock Cycles
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	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 data pointer 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
BOOLEAN MANIPULATION			
CLR C	Clear carry	1	1
CLR bit	Clear direct bit	2	2
SETB C	Set carry	1	1
SETB bit	Set direct bit	2	2
CPL C	Complement carry	1	1
CPL bit	Complement direct bit	2	2
ANL C,bit	AND direct bit to carry	2	2
ANL C,/bit	AND complement of direct bit to carry	2	2
ORL C,bit	OR direct bit to carry	2	2
ORL C,/bit	OR complement of direct bit to carry	2	2
MOV C,bit	Move direct bit to carry	2	2
MOV bit,C	Move carry to direct bit	2	2
JC rel	Jump if carry is set	2	2/3
JNC rel	Jump if carry not set	2	2/3
JB bit,rel	Jump if direct bit is set	3	3/4
JNB bit,rel	Jump if direct bit is not set	3	3/4
JBC bit,rel	Jump if direct bit is set and clear bit	3	3/4

Address	Register	Description	Page No.
0xE2	XBR1	Port I/O Crossbar Configuration 2	90
0xE3	XBR2	Port I/O Crossbar Configuration 3	91
0x84-86, 0x96-97, 0x9C, 0xA1-A3, 0xA9-AC, 0xAE, 0xB3-B5, 0xB9, 0xBD, 0xC9, 0xCE, 0xDF, 0xE4-E5, 0xF1-F5		Reserved	

Figure 8.10. IP: Interrupt Priority

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	PT2	PS	PT1	PX1	PT0	PX0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0 (bit addressable)	SFR Address: 0xB8

Bits7-6: UNUSED. Read = 11b, Write = don't care.

Bit5: PT2 Timer 2 Interrupt Priority Control.
This bit sets the priority of the Timer 2 interrupts.
0: Timer 2 interrupts set to low priority level.
1: Timer 2 interrupts set to high priority level.

Bit4: PS: Serial Port (UART) Interrupt Priority Control.
This bit sets the priority of the Serial Port (UART) interrupts.
0: UART interrupts set to low priority level.
1: UART interrupts set to high priority level.

Bit3: PT1: Timer 1 Interrupt Priority Control.
This bit sets the priority of the Timer 1 interrupts.
0: Timer 1 interrupts set to low priority level.
1: Timer 1 interrupts set to high priority level.

Bit2: PX1: External Interrupt 1 Priority Control.
This bit sets the priority of the External Interrupt 1 interrupts.
0: External Interrupt 1 set to low priority level.
1: External Interrupt 1 set to high priority level.

Bit1: PT0: Timer 0 Interrupt Priority Control.
This bit sets the priority of the Timer 0 interrupts.
0: Timer 0 interrupt set to low priority level.
1: Timer 0 interrupt set to high priority level.

Bit0: PX0: External Interrupt 0 Priority Control.
This bit sets the priority of the External Interrupt 0 interrupts.
0: External Interrupt 0 set to low priority level.
1: External Interrupt 0 set to high priority level.

9. FLASH MEMORY

These devices include 16k + 128 bytes of on-chip, reprogrammable Flash memory for program code and non-volatile data storage. The Flash memory can be programmed in-system, a single byte at a time, through the JTAG interface or by software using the MOVX instruction. Once cleared to 0, a Flash bit must be erased to set it back to 1. The 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. Refer to Table 9.1 for the electrical characteristics of the Flash memory.

9.1. Programming The Flash Memory

The simplest means of programming the Flash memory is through the JTAG 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 JTAG commands to program Flash memory, see Section 19.2.

The Flash memory can be programmed by software using the MOVX 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 setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1. Writing to Flash remains enabled until the PSWE bit is cleared by software.

Writes to Flash memory can clear bits but cannot set them. Only an erase operation can set bits in Flash. Therefore, the byte location to be programmed must be erased before a new value can be written. The 16kbyte Flash memory is organized in 512-byte sectors. The erase operation applies to an entire sector (setting all bytes in the sector to 0xFF). Setting the PSEE Program Store Erase Enable bit (PSCTL.1) and PSWE (PSCTL.0) bit to logic 1 and then using the MOVX command to write a data byte to any byte location within the sector will erase an entire 512-byte sector. The data byte written can be of any value because it is not actually written to the Flash. Flash erasure remains enabled until the PSEE bit is cleared by software. The following sequence illustrates the algorithm for programming the Flash memory by software:

1. Enable Flash Memory write/erase in FLSCL Register using FLASCL bits.
2. Set PSEE (PSCTL.1) to enable Flash sector erase.
3. Set PSWE (PSCTL.0) to enable Flash writes.
4. Use MOVX to write a data byte to any location within the 512-byte sector to be erased.
5. Clear PSEE to disable Flash sector erase.
6. Use MOVX to write a data byte to the desired byte location within the erased 512-byte sector. Repeat until finished. (Any number of bytes can be written from a single byte to and entire sector.)
7. Clear the PSWE bit to disable Flash writes.

Write/Erase timing is automatically controlled by hardware based on the prescaler value held in the Flash Memory Timing Prescaler register (FLSCL). The 4-bit prescaler value FLASCL determines the time interval for write/erase operations. The FLASCL value required for a given system clock is shown in Figure 9.4, along with the formula used to derive the FLASCL values. When FLASCL is set to 1111b, the write/erase operations are disabled. Note that code execution in the 8051 is stalled while the Flash is being programmed or erased.

Table 9.1. FLASH Memory Electrical Characteristics

VDD = 2.8 to 3.6V, -40°C to +85°C unless otherwise specified.

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Endurance		20k	100k		Erase/Wr
Erase Cycle Time		10			ms
Write Cycle Time		40			µs

10. EXTERNAL RAM

The C8051F018/9 includes 1024 bytes of RAM mapped into the external data memory space. All of these address locations may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using MOVX indirect addressing mode. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN as shown in Figure 10.1). **Note: the MOVX instruction is also used for writes to the Flash memory. See Section 9 for details. The MOVX instruction accesses XRAM by default (i.e. PSTCL.0 = 0).**

For any of the addressing modes the upper 5-bits of the 16-bit external data memory address word are “don’t cares”. As a result, the 1024-byte RAM is mapped modulo style over the entire 64k external data memory address range. For example, the XRAM byte at address 0x0000 is also at address 0x0400, 0x0800, 0x0C00, 0x1000, etc. This is a useful feature when doing a linear memory fill, as the address pointer doesn’t have to be reset when reaching the RAM block boundary.

Figure 10.1. EMI0CN: External Memory Interface Control

R	R	R	R	R	R/W	R/W	R/W	Reset Value
-	-	-	-	-	-	PGSEL1	PGSEL0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xAF

Bits 7-2: Not Used – reads 000000b

Bits 1-0: PGSEL[1:0]: XRAM Page Select Bits

The XRAM Page Select Bits provide the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. The upper 6-bits are “don’t cares”, so the 1k address blocks are repeated modulo over the entire 64k external data memory address space.

00: xxxxxx00b
01: xxxxxx01b
10: xxxxxx10b
11: xxxxxx11b

11.4. External Reset

The external /RST pin provides a means for external circuitry to force the MCU into a reset state. Asserting an active-low signal on the /RST pin will cause the MCU to enter the reset state. Although there is a weak internal pullup, it may be desirable to provide an external pull-up and/or decoupling of the /RST pin to avoid erroneous noise-induced resets. The MCU will remain in reset until at least 12 clock cycles after the active-low /RST signal is removed. The PINRSF flag (RSTSRC.0) is set on exit from an external reset. The /RST pin is also 5V tolerant.

11.5. Missing Clock Detector Reset

The Missing Clock Detector is essentially a one-shot circuit that is triggered by the MCU system clock. If the system clock goes away for more than 100µs, the one-shot will time out and generate a reset. After a Missing Clock Detector reset, the MCDRSF flag (RSTSRC.2) will be set, signifying the MSD as the reset source; otherwise, this bit reads 0. The state of the /RST pin is unaffected by this reset. Setting the MSCLKE bit in the OSCICN register (see Figure 12.2) enables the Missing Clock Detector.

11.6. Comparator 0 Reset

Comparator 0 can be configured as an active-low reset input by writing a 1 to the C0RSEF flag (RSTSRC.5). Comparator 0 should be enabled using CPT0CN.7 (see Figure 6.3) at least 20µs prior to writing to C0RSEF to prevent any turn-on chatter on the output from generating an unwanted reset. When configured as a reset, if the non-inverting input voltage (on CP0+) is less than the inverting input voltage (on CP0-), the MCU is put into the reset state. After a Comparator 0 Reset, the C0RSEF flag (RSTSRC.5) will read 1 signifying Comparator 0 as the reset source; otherwise, this bit reads 0. The state of the /RST pin is unaffected by this reset. Also, Comparator 0 can generate a reset with or without the system clock.

11.7. External CNVSTR Pin Reset

The external CNVSTR signal can be configured as an active-low reset input by writing a 1 to the CNVRSEF flag (RSTSRC.6). The CNVSTR signal can appear on any of the P0, P1, or P2 I/O pins as described in Section 13.1. (Note that the Crossbar must be configured for the CNVSTR signal to be routed to the appropriate Port I/O.) The Crossbar should be configured and enabled before the CNVRSEF is set to configure CNVSTR as a reset source. When configured as a reset, CNVSTR is active-low and level sensitive. After a CNVSTR reset, the CNVRSEF flag (RSTSRC.6) will read 1 signifying CNVSTR as the reset source; otherwise, this bit reads 0. The state of the /RST pin is unaffected by this reset.

11.8. Watchdog Timer Reset

The MCU includes a programmable Watchdog Timer (WDT) running off the system clock. The WDT will force the MCU into the reset state when the watchdog timer overflows. To prevent the reset, the WDT must be restarted by application software before the overflow occurs. If the system experiences a software/hardware malfunction preventing the software from restarting the WDT, the WDT will overflow and cause a reset. This should prevent the system from running out of control.

The WDT is automatically enabled and started with the default maximum time interval on exit from all resets. If desired the WDT can be disabled by system software or locked on to prevent accidental disabling. Once locked, the WDT cannot be disabled until the next system reset. The state of the /RST pin is unaffected by this reset.

Table 11.1. Reset Electrical Characteristics

-40°C to +85°C unless otherwise specified.

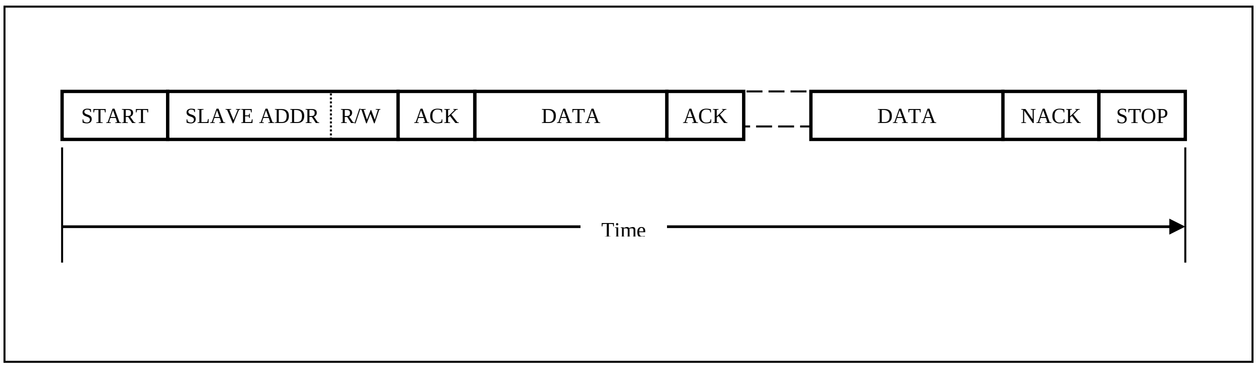
PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
/RST Output Low Voltage	I_{OL} = 8.5mA, VDD = 2.8 to 3.6V			0.6	V
/RST Input High Voltage		0.7 x VDD			V
/RST Input Low Voltage				0.3 x VDD	V
/RST Input Leakage Current	/RST = 0.0V		20		µA
VDD for /RST Output Valid		1.0			V
AV+ for /RST Output Valid		1.0			V
VDD POR Threshold (V_{RST})		2.40	2.55	2.80	V
Reset Time Delay	/RST rising edge after crossing reset threshold	80	100	120	ms
Missing Clock Detector Timeout	Time from last system clock to reset generation	100	220	500	µs

14.2. Operation

A typical SMBus transaction consists of a START condition, followed by an address byte, one or more bytes of data, and a STOP condition. The address byte and each of the data bytes are followed by an ACKNOWLEDGE bit from the receiver. The address byte consists of a 7-bit address plus a direction bit. The direction bit (R/W) occupies the least-significant bit position of the address. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation. A general call address (0x00 +R/W) is recognized by all slave devices allowing a master to address multiple slave devices simultaneously.

All transactions are initiated by the master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the 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 ACKNOWLEDGE from the slave at the end of each byte. If it is a READ operation, the slave transmits the data waiting for an ACKNOWLEDGE 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 14.3 illustrates a typical SMBus transaction.

Figure 14.3. SMBus Transaction



The SMBus interface may be configured to operate as either a master or a slave. At any particular time, it will be operating in one of the following four modes:

14.2.1. Master Transmitter Mode

Serial data is transmitted on SDA while the serial clock is output on SCL. The first byte transmitted contains the address of the target slave device and the data direction bit. In this case the data direction bit (R/W) will be logic 0 to indicate a "WRITE" operation. The master then transmits one or more bytes of serial data. After each byte is transmitted, an acknowledge bit is generated by the slave. To indicate the beginning and the end of the serial transfer, the master device outputs START and STOP conditions.

14.2.2. Master Receiver Mode

Serial data is received on SDA while the serial clock is output on SCL. The first byte is transmitted by the master and contains the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 to indicate a "READ" operation. Serial data is then received from the slave on SDA while the master outputs the serial clock. The slave transmits one or more bytes of serial data. After each byte is received, an acknowledge bit is transmitted by the master. The master outputs START and STOP conditions to indicate the beginning and end of the serial transfer.

14.2.3. Slave Transmitter Mode

Serial data is transmitted on SDA while the serial clock is received on SCL. First, a byte is received that contains an address and data direction bit. In this case the data direction bit (R/W) will be logic 1 to indicate a "READ" operation. If the received address matches the slave's assigned address (or a general call address is received) one or more bytes of serial data are transmitted to the master. After each byte is received, an acknowledge bit is transmitted by the master. The master outputs START and STOP conditions to indicate the beginning and end of the serial transfer.

14.6.5. Status Register

The SMB0STA Status register holds an 8-bit status code indicating the current state of the SMBus. There are 28 possible SMBus states, each with a corresponding unique status code. The five most significant bits of the status code vary while the three least-significant bits of a valid status code are fixed at zero when SI = 1. Therefore, all possible status codes are multiples of eight. This facilitates the use of status codes in software as an index used to branch to appropriate service routines (allowing 8 bytes of code to service the state or jump to a more extensive service routine).

For the purposes of user software, the contents of the SMB0STA register is only defined when the SI flag is logic 1. Software should never write to the SMB0STA register. Doing so will yield indeterminate results. The 28 SMBus states, along with their corresponding status codes, are given in Table 14.1.

Figure 14.8. SMB0STA: SMBus Status Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
STA7	STA6	STA5	STA4	STA3	STA2	STA1	STA0	11111000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xC1

Bits7-3: STA7-STA3: SMBus Status Code.
These bits contain the SMBus Status Code. There are 28 possible status codes. Each status code corresponds to a single SMBus state. A valid status code is present in SMB0STA when the SI flag (SMB0CN.3) is set. The content of SMB0STA is not defined when the SI flag is logic 0. Writing to the SMB0STA register at any time will yield indeterminate results.

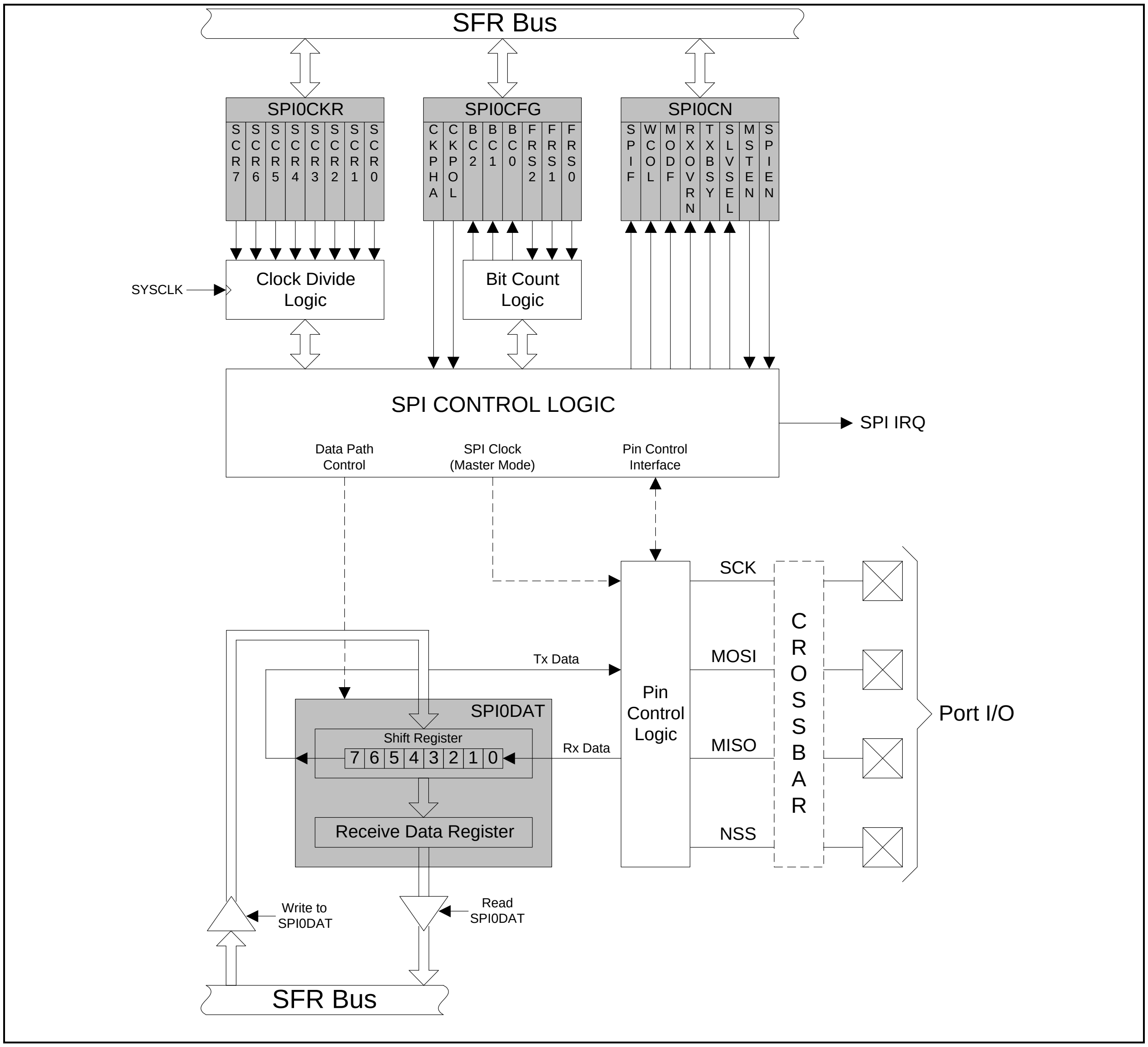
Bits2-0: STA2-STA0: The three least significant bits of SMB0STA are always read as logic 0 when the SI flag is logic 1.

15. SERIAL PERIPHERAL INTERFACE BUS

The Serial Peripheral Interface (SPI) provides access to a four-wire, full-duplex, serial bus. SPI supports the connection of multiple slave devices to a master device on the same bus. A separate slave-select signal (NSS) is used to select a slave device and enable a data transfer between the master and the selected slave. Multiple masters on the same bus are also supported. Collision detection is provided when two or more masters attempt a data transfer at the same time. The SPI can operate as either a master or a slave. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency.

When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS, and the serial input data synchronously with the system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less that 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of ¼ the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the system clock.

Figure 15.1. SPI Block Diagram



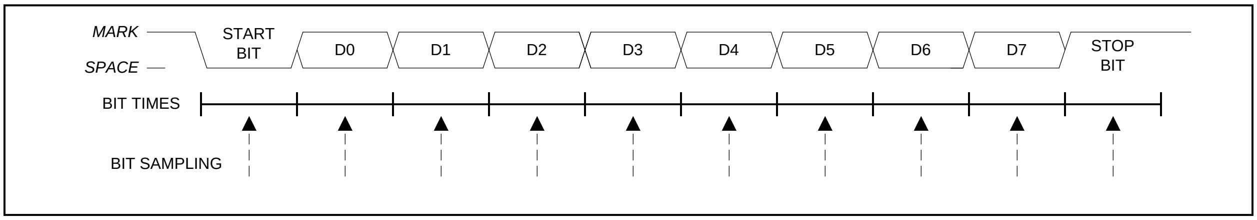
16.1.2. Mode 1: 8-Bit UART, Variable Baud Rate

Mode 1 provides standard asynchronous, full duplex communication using a total of 10 bits per data byte: one start bit, eight data bits (LSB first), and one stop bit (see the timing diagram in Figure 16.4). Data are transmitted from the TX pin and received at the RX pin (see the interconnection diagram in Figure 16.5). On receive, the eight data bits are stored in SBUF and the stop bit goes into RB8 (SCON.2).

Data transmission begins when an instruction writes a data byte to the SBUF register. The TI Transmit Interrupt Flag (SCON.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN Receive Enable bit (SCON.4) is set to logic 1. After the stop bit is received, the data byte will be loaded into the SBUF receive register if the following conditions are met: RI must be logic 0, and if SM2 is logic 1, the stop bit must be logic 1.

If these conditions are met, the eight bits of data are stored in SBUF, the stop bit is stored in RB8 and the RI flag is set. If these conditions are not met, SBUF and RB8 will not be loaded and the RI flag will not be set. An interrupt will occur if enabled when either TI or RI is set.

Figure 16.4. UART Mode 1 Timing Diagram



The baud rate generated in Mode 1 is a function of timer overflow. The UART can use Timer 1 operating in *8-bit Counter/Timer with Auto-Reload Mode*, or Timer 2 operating in *Baud Rate Generator Mode* to generate the baud rate (note that the TX and RX clock sources are selected separately). On each timer overflow event (a rollover from all ones (0xFF for Timer 1, 0xFFFF for Timer 2) to zero), a clock is sent to the baud rate logic.

When Timer 1 is selected as a baud rate source, the SMOD bit (PCON.7) selects whether or not to divide the Timer 1 overflow rate by two. On reset, the SMOD bit is logic 0, thus selecting the lower speed baud rate by default. The SMOD bit affects the baud rate generated by Timer 1 as follows:

*Mode 1 Baud Rate = (1 / 32) * T1_OVERFLOWRATE (when the SMOD bit is set to logic 0).*
*Mode 1 Baud Rate = (1 / 16) * T1_OVERFLOWRATE (when the SMOD bit is set to logic 1).*

When Timer 2 is selected as a baud rate source, the baud rate generated by Timer 2 is as follows:

*Mode 1 Baud Rate = (1 / 16) * T2_OVERFLOWRATE.*

The Timer 1 overflow rate is determined by the Timer 1 clock source (T1CLK) and reload value (TH1). The frequency of T1CLK can be selected as SYSCLK, SYSCLK/12, or an external clock source. The Timer 1 overflow rate can be calculated as follows:

T1_OVERFLOWRATE = T1CLK / (256 – TH1).

For example, assume TMOD = 0x20.
If T1M (CKCON.4) is logic 1, then the above equation becomes:

T1_OVERFLOWRATE = (SYSCLK) / (256 – TH1).

If T1M (CKCON.4) is logic 0, then the above equation becomes:

T1_OVERFLOWRATE = (SYSCLK/12) / (256 – TH1).

16.2. Multiprocessor Communications

Modes 2 and 3 support multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its ninth bit is logic 1; in a data byte, the ninth bit is always set to logic 0.

Setting the SM2 bit (SCON.5) of a slave processor configures its UART such that when a stop bit is received, the UART will generate an interrupt only if the ninth bit is logic one (RB8 = 1) signifying an address byte has been received. In the UART's interrupt handler, software will compare the received address with the slave's own assigned 8-bit address. If the addresses match, the slave will clear its SM2 bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their SM2 bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its SM2 bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

Figure 16.7. UART Multi-Processor Mode Interconnect Diagram

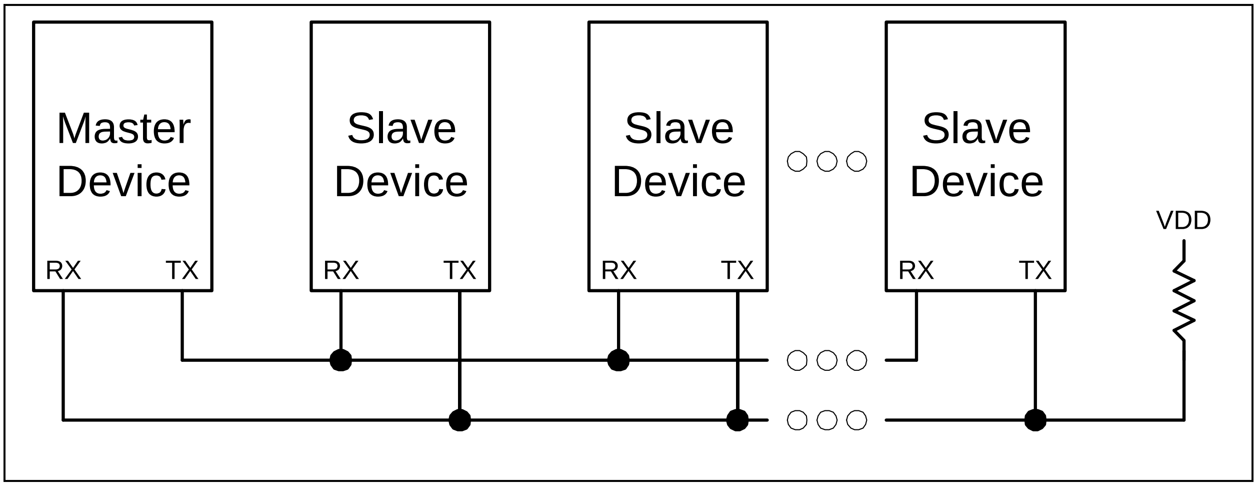


Figure 17.5. TMOD: Timer Mode Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
GATE1	C/T1	T1M1	T1M0	GATE0	C/T0	T0M1	T0M0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x89

Bit7: GATE1: Timer 1 Gate Control.
0: Timer 1 enabled when TR1 = 1 irrespective of /INT1 logic level.
1: Timer 1 enabled only when TR1 = 1 AND /INT1 = logic level one.

Bit6: C/T1: Counter/Timer 1 Select.
0: Timer Function: Timer 1 incremented by clock defined by T1M bit (CKCON.4).
1: Counter Function: Timer 1 incremented by high-to-low transitions on external input pin (T1).

Bits5-4: T1M1-T1M0: Timer 1 Mode Select.
These bits select the Timer 1 operation mode.

T1M1	T1M0	Mode
0	0	Mode 0: 13-bit counter/timer
0	1	Mode 1: 16-bit counter/timer
1	0	Mode 2: 8-bit counter/timer with auto-reload
1	1	Mode 3: Timer 1 Inactive/stopped

Bit3: GATE0: Timer 0 Gate Control.
0: Timer 0 enabled when TR0 = 1 irrespective of /INT0 logic level.
1: Timer 0 enabled only when TR0 = 1 AND /INT0 = logic level one.

Bit2: C/T0: Counter/Timer Select.
0: Timer Function: Timer 0 incremented by clock defined by T0M bit (CKCON.3).
1: Counter Function: Timer 0 incremented by high-to-low transitions on external input pin (T0).

Bits1-0: T0M1-T0M0: Timer 0 Mode Select.
These bits select the Timer 0 operation mode.

T0M1	T0M0	Mode
0	0	Mode 0: 13-bit counter/timer
0	1	Mode 1: 16-bit counter/timer
1	0	Mode 2: 8-bit counter/timer with auto-reload
1	1	Mode 3: Two 8-bit counter/timers

17.2.3. Mode 2: Baud Rate Generator

Timer 2 can be used as a baud rate generator for the serial port (UART) when the UART is operated in modes 1 or 3 (refer to Section 16.1 for more information on UART operational modes). In Baud Rate Generator mode, Timer 2 works similarly to the auto-reload mode. On overflow, the 16-bit value held in the two capture registers (RCAP2H, RCAP2L) is automatically loaded into the counter/timer register. However, the TF2 overflow flag is not set and no interrupt is generated. Instead, the overflow event is used as the input to the UART's shift clock. Timer 2 overflows can be used to generate baud rates for transmit and/or receive independently.

The Baud Rate Generator mode is selected by setting RCLK (T2CON.5) and/or TCLK (T2CON.4) to logic one. When RCLK or TCLK is set to logic 1, Timer 2 operates in the auto-reload mode regardless of the state of the CP/RL2 bit. The baud rate for the UART, when operating in mode 1 or 3, is determined by the Timer 2 overflow rate:

Baud Rate = Timer 2 Overflow Rate / 16.

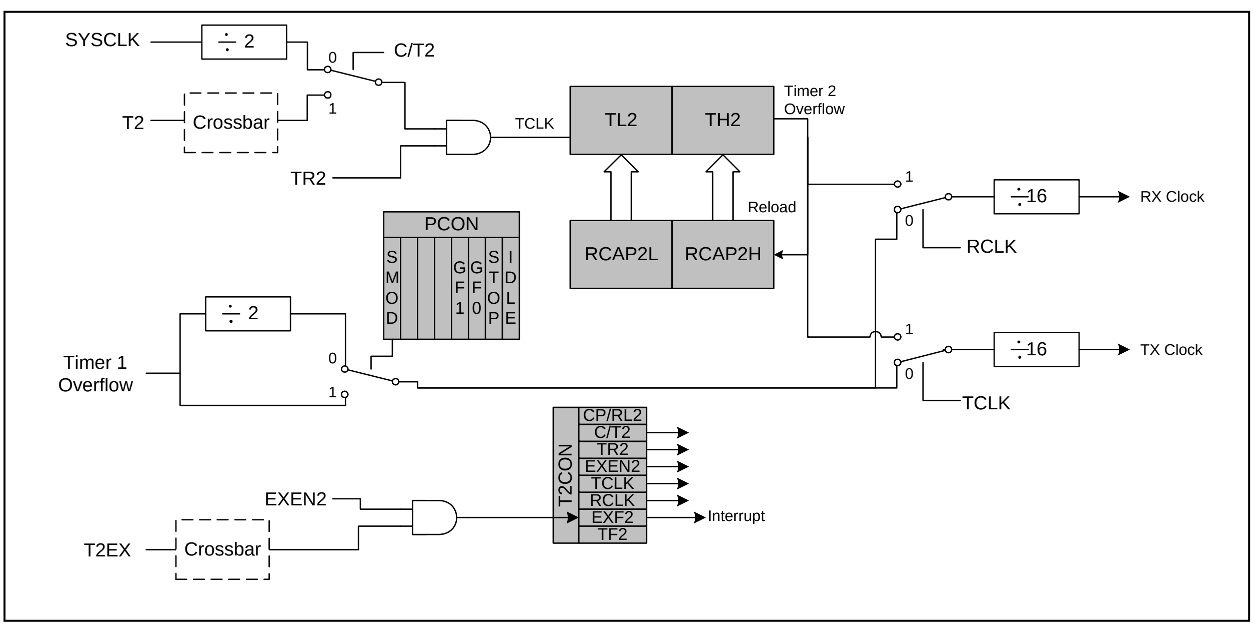
Note, in all other modes, the timebase for the timer is the system clock divided by one or twelve as selected by the T2M bit in CKCON. However, in Baud Rate Generator mode, the timebase is the system clock divided by two. No other divisor selection is possible. If a different time base is required, setting the C/T2 bit to logic 1 will allow the timebase to be derived from the external input pin T2. In this case, the baud rate for the UART is calculated as:

*Baud Rate = FCLK / [32 * (65536 – [RCAP2H:RCAP2L])]*

Where FCLK is the frequency of the signal supplied to T2 and [RCAP2H:RCAP2L] is the 16-bit value held in the capture registers.

As explained above, in Baud Rate Generator mode, Timer 2 does not set the TF2 overflow flag and therefore cannot generate an interrupt. However, if EXEN2 is set to logic 1, a high-to-low transition on the T2EX input pin will set the EXF2 flag and a Timer 2 interrupt will occur if enabled. Therefore, the T2EX input may be used as an additional external interrupt source.

Figure 17.13. T2 Mode 2 Block Diagram



18.1. Capture/Compare Modules

Each module can be configured to operate independently in one of four operation modes: Edge-triggered Capture, Software Timer, High Speed Output, or Pulse Width Modulator. Each module has Special Function Registers (SFRs) associated with it in the CIP-51 system controller. These registers are used to exchange data with a module and configure the module's mode of operation.

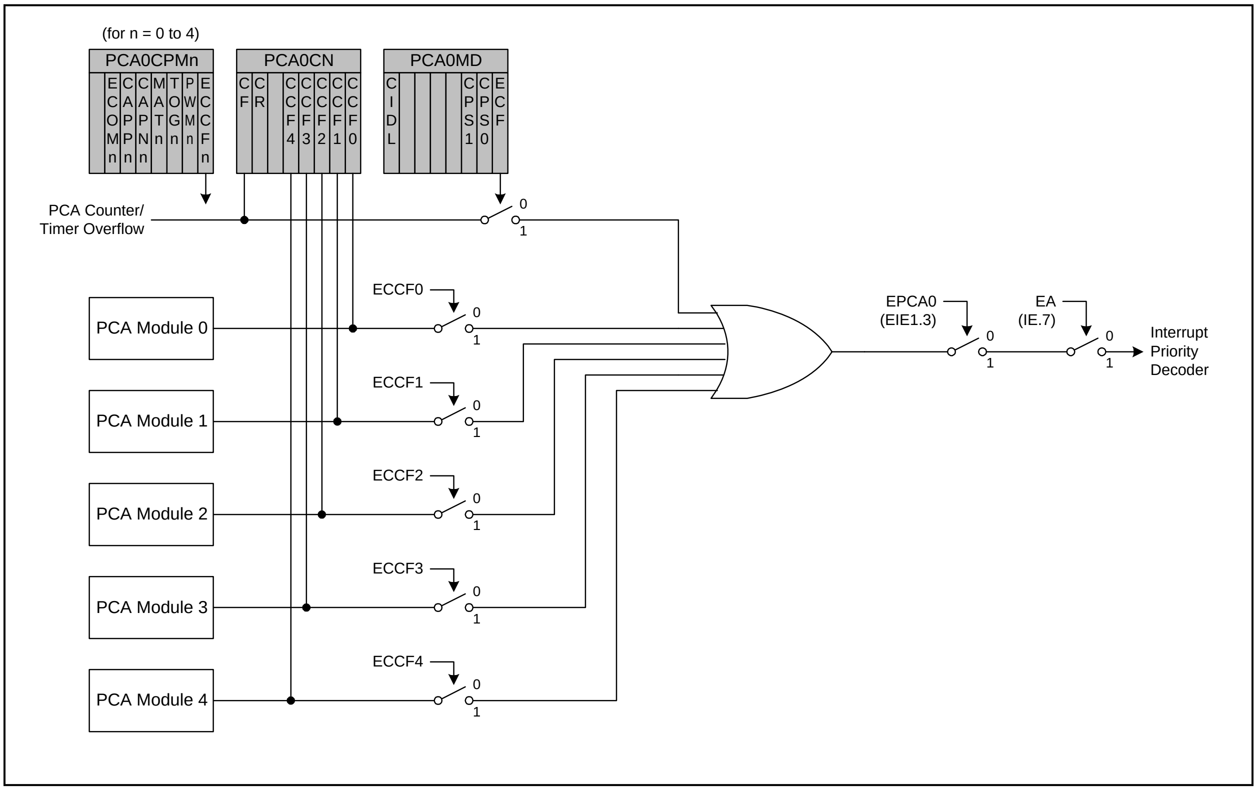
Table 18.1 summarizes the bit settings in the PCA0CPMn registers used to place the PCA capture/compare modules into different operating modes. Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt. Note: PCA0 interrupts must be globally enabled before individual CCFn interrupts are recognized. PCA0 interrupts are globally enabled by setting the EA bit (IE.7) and the EPCA0 bit (EIE1.3) to logic 1. See Figure 18.2 for details on the PCA interrupt configuration.

Table 18.1. PCA0CPM Register Settings for PCA Capture/Compare Modules

ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	Operation Mode
X	1	0	0	0	0	X	Capture triggered by positive edge on CEXn
X	0	1	0	0	0	X	Capture triggered by negative edge on CEXn
X	1	1	0	0	0	X	Capture triggered by transition on CEXn
1	0	0	1	0	0	X	Software Timer
1	0	0	1	1	0	X	High Speed Output
1	0	0	X	0	1	X	Pulse Width Modulator

X = Don't Care

Figure 18.2. PCA Interrupt Block Diagram



18.1.1. 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.

Figure 18.3. PCA Capture Mode Diagram