

Welcome to [E-XFL.COM](https://www.e-xfl.com)

### What is "[Embedded - Microcontrollers](#)"?

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

### Applications of "[Embedded - Microcontrollers](#)"

#### Details

Product Status	Not For New Designs
Core Processor	8051
Core Size	8-Bit
Speed	25MHz
Connectivity	SMBus (2-Wire/I <sup>2</sup> C), UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	8
Program Memory Size	2KB (2K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	10-VDFN Exposed Pad
Supplier Device Package	11-QFN (3x3)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/silicon-labs/c8051f305-gmr">https://www.e-xfl.com/product-detail/silicon-labs/c8051f305-gmr</a>

### 3. Global Electrical Characteristics

**Table 3.1. Global Electrical Characteristics**

–40 to +85 °C, 25 MHz system clock unless otherwise specified.

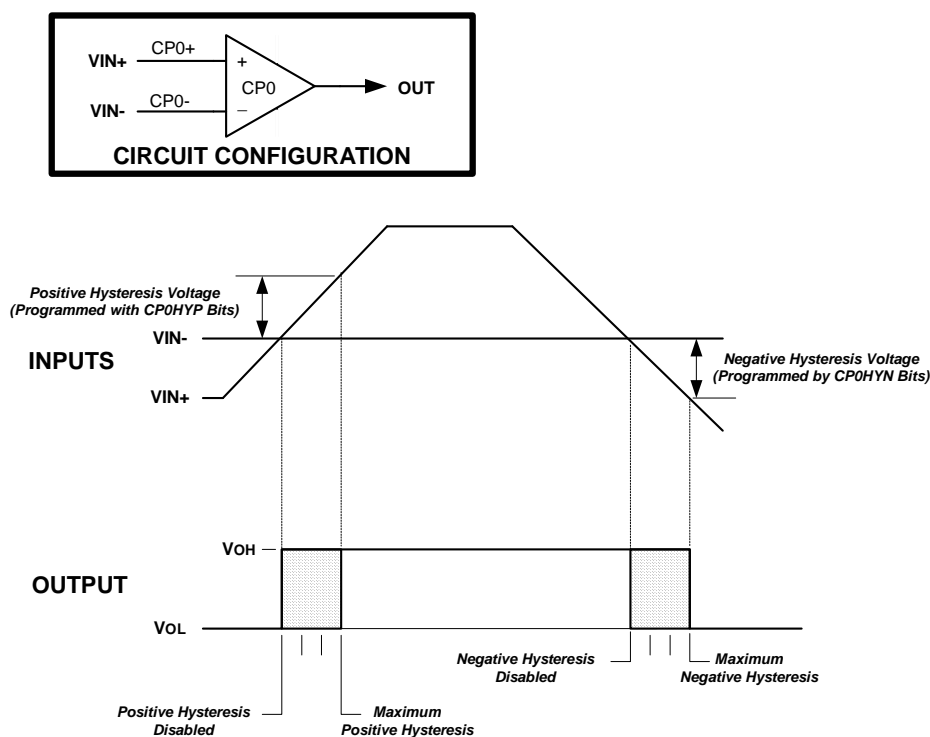
Parameter	Conditions	Min	Typ	Max	Units
Digital Supply Voltage		$V_{RST}^1$	3.0	3.6	V
Digital Supply RAM Data Retention Voltage		—	1.5	—	V
SYSCLK (System Clock) (Note 2)		0	—	25	MHz
$T_{SYSH}$ (SYSCLK High Time)		18	—	—	ns
$T_{SYSL}$ (SYSCLK Low Time)		18	—	—	ns
Specified Operating Temperature Range		–40	—	+85	°C
<b>Digital Supply Current—CPU Active (Normal Mode, fetching instructions from Flash)</b>					
$I_{DD}$ (Note 3)	$V_{DD} = 3.6\text{ V}$ , $F = 25\text{ MHz}$	—	9.4	10.2	mA
	$V_{DD} = 3.0\text{ V}$ , $F = 25\text{ MHz}$	—	6.6	7.2	mA
	$V_{DD} = 3.0\text{ V}$ , $F = 1\text{ MHz}$	—	0.45	—	mA
	$V_{DD} = 3.0\text{ V}$ , $F = 80\text{ kHz}$	—	36	—	μA
$I_{DD}$ Supply Sensitivity (Note 3)	$F = 25\text{ MHz}$	—	69	—	%/V
	$F = 1\text{ MHz}$	—	51	—	%/V
$I_{DD}$ Frequency Sensitivity (Note 3, Note 4)	$V_{DD} = 3.0\text{ V}$ , $F \leq 15\text{ MHz}$ , $T = 25\text{ °C}$	—	0.45	—	mA/MHz
	$V_{DD} = 3.0\text{ V}$ , $F > 15\text{ MHz}$ , $T = 25\text{ °C}$	—	0.16	—	mA/MHz
	$V_{DD} = 3.6\text{ V}$ , $F \leq 15\text{ MHz}$ , $T = 25\text{ °C}$	—	0.69	—	mA/MHz
	$V_{DD} = 3.6\text{ V}$ , $F > 15\text{ MHz}$ , $T = 25\text{ °C}$	—	0.20	—	mA/MHz
<b>Digital Supply Current—CPU Inactive (Idle Mode, not fetching instructions from Flash)</b>					
$I_{DD}$ (Note 3)	$V_{DD} = 3.6\text{ V}$ , $F = 25\text{ MHz}$	—	3.3	4.0	mA
	$V_{DD} = 3.0\text{ V}$ , $F = 25\text{ MHz}$	—	2.5	3.2	mA
	$V_{DD} = 3.0\text{ V}$ , $F = 1\text{ MHz}$	—	0.10	—	mA
	$V_{DD} = 3.0\text{ V}$ , $F = 80\text{ kHz}$	—	8	—	μA



# C8051F300/1/2/3/4/5

The output of Comparator0 can be polled in software, used as an interrupt source, and/or routed to a Port pin. When routed to a Port pin, the Comparator0 output is available asynchronous or synchronous to the system clock; the asynchronous output is available even in STOP mode (with no system clock active). When disabled, the Comparator0 output (if assigned to a Port I/O pin via the Crossbar) defaults to the logic low state, and its supply current falls to less than 100 nA. See **Section “12.1. Priority Crossbar Decoder” on page 104** for details on configuring the Comparator0 output via the digital Crossbar. Comparator0 inputs can be externally driven from  $-0.25$  to  $(V_{DD}) + 0.25$  V without damage or upset. The complete electrical specifications for Comparator0 are given in Table 7.1.

The Comparator0 response time may be configured in software via the CP0MD1-0 bits in register CPT0MD (see SFR Definition 7.3). Selecting a longer response time reduces the amount of power consumed by Comparator0. See Table 7.1 for complete timing and power consumption specifications.



**Figure 7.2. Comparator Hysteresis Plot**

The hysteresis of Comparator0 is software-programmable via its Comparator0 Control register (CPT0CN). The user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage.

The Comparator0 hysteresis is programmed using Bits3–0 in the Comparator0 Control Register CPT0CN (shown in SFR Definition 7.1). The amount of negative hysteresis voltage is determined by the settings of the CP0HYN bits. As shown in Figure 7.2, settings of 20, 10 or 5 mV of negative hysteresis can be programmed, or negative hysteresis can be disabled. In a similar way, the amount of positive hysteresis is determined by the setting the CP0HYP bits.

# C8051F300/1/2/3/4/5

## SFR Definition 7.2. CPT0MX: Comparator0 MUX Selection

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
—	—	CMX0N1	CMX0N0	—	—	CMX0P1	CMX0P0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x9F

Bits7–6: UNUSED. Read = 00b, Write = don't care.

Bits6–4: CMX0N1–CMX0N0: Comparator0 Negative Input MUX Select.

These bits select which Port pin is used as the Comparator0 negative input.

CMX0N1	CMX0N0	Negative Input
0	0	P0.1
0	1	P0.3
1	0	P0.5
1	1	P0.7

Bits3–2: UNUSED. Read = 00b, Write = don't care.

Bits1–0: CMX0P1–CMX0P0: Comparator0 Positive Input MUX Select.

These bits select which Port pin is used as the Comparator0 positive input.

CMX0P1	CMX0P0	Positive Input
0	0	P0.0
0	1	P0.2
1	0	P0.4
1	1	P0.6

## SFR Definition 7.3. CPT0MD: Comparator0 Mode Selection

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
—	—	—	—	—	—	CP0MD1	CP0MD0	00000010
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x9D

Bits7–2: UNUSED. Read = 000000b, Write = don't care.

Bits1–0: CP0MD1–CP0MD0: Comparator0 Mode Select.

These bits select the response time for Comparator0.

Mode	CP0MD1	CP0MD0	CP0 Response Time (TYP)
0	0	0	Fastest Response Time
1	0	1	—
2	1	0	—
3	1	1	Lowest Power Consumption

**Table 7.1. Comparator0 Electrical Characteristics**

$V_{DD} = 3.0\text{ V}$ ,  $-40$  to  $+85\text{ }^{\circ}\text{C}$  unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Response Time: Mode 0, $V_{cm}^* = 1.5\text{ V}$	CP0+ – CP0– = 100 mV	—	100	—	ns
	CP0+ – CP0– = –100 mV	—	250	—	ns
Response Time: Mode 1, $V_{cm}^* = 1.5\text{ V}$	CP0+ – CP0– = 100 mV	—	175	—	ns
	CP0+ – CP0– = –100 mV	—	500	—	ns
Response Time: Mode 2, $V_{cm}^* = 1.5\text{ V}$	CP0+ – CP0– = 100 mV	—	320	—	ns
	CP0+ – CP0– = –100 mV	—	1100	—	ns
Response Time: Mode 3, $V_{cm}^* = 1.5\text{ V}$	CP0+ – CP0– = 100 mV	—	1050	—	ns
	CP0+ – CP0– = –100 mV	—	5200	—	ns
Common-Mode Rejection Ratio		—	1.5	4	mV/V
Positive Hysteresis 1	CP0HYP1–0 = 00	—	0	1	mV
Positive Hysteresis 2	CP0HYP1–0 = 01	3	5	7	mV
Positive Hysteresis 3	CP0HYP1–0 = 10	7	10	15	mV
Positive Hysteresis 4	CP0HYP1–0 = 11	15	20	25	mV
Negative Hysteresis 1	CP0HYN1–0 = 00	—	0	1	mV
Negative Hysteresis 2	CP0HYN1–0 = 01	3	5	7	mV
Negative Hysteresis 3	CP0HYN1–0 = 10	7	10	15	mV
Negative Hysteresis 4	CP0HYN1–0 = 11	15	20	25	mV
Inverting or Non-Inverting Input Voltage Range		–0.25	—	$V_{DD} + 0.25$	V
Input Capacitance		—	7	—	pF
Input Bias Current		–5	0.001	+5	nA
Input Offset Voltage		–5	—	+5	mV
<b>Power Supply</b>					
Power Supply Rejection		—	0.1	1	mV/V
Power-up Time		—	10	—	$\mu\text{s}$
Supply Current at DC	Mode 0	—	7.6	—	$\mu\text{A}$
	Mode 1	—	3.2	—	$\mu\text{A}$
	Mode 2	—	1.3	—	$\mu\text{A}$
	Mode 3	—	0.4	—	$\mu\text{A}$
*Note: $V_{cm}$ is the common-mode voltage on CP0+ and CP0–.					

# C8051F300/1/2/3/4/5

**Table 8.1. CIP-51 Instruction Set Summary (Continued)**

Mnemonic	Description	Bytes	Clock Cycles
JNZ rel	Jump if A does not equal zero	2	2/3
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	3/4
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3/4
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3/4
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4/5
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2/3
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3/4
NOP	No operation	1	1

## Notes on Registers, Operands and Addressing Modes:

**Rn** - Register R0-R7 of the currently selected register bank.

**@Ri** - Data RAM location addressed indirectly through R0 or R1.

**rel** - 8-bit, signed (two's complement) offset relative to the first byte of the following instruction. Used by SJMP and all conditional jumps.

**direct** - 8-bit internal data location's address. This could be a direct-access Data RAM location (0x00-0x7F) or an SFR (0x80-0xFF).

**#data** - 8-bit constant

**#data16** - 16-bit constant

**bit** - Direct-accessed bit in Data RAM or SFR

**addr11** - 11-bit destination address used by ACALL and AJMP. The destination must be within the same 2K-byte page of program memory as the first byte of the following instruction.

**addr16** - 16-bit destination address used by LCALL and LJMP. The destination may be anywhere within the 8K-byte program memory space.

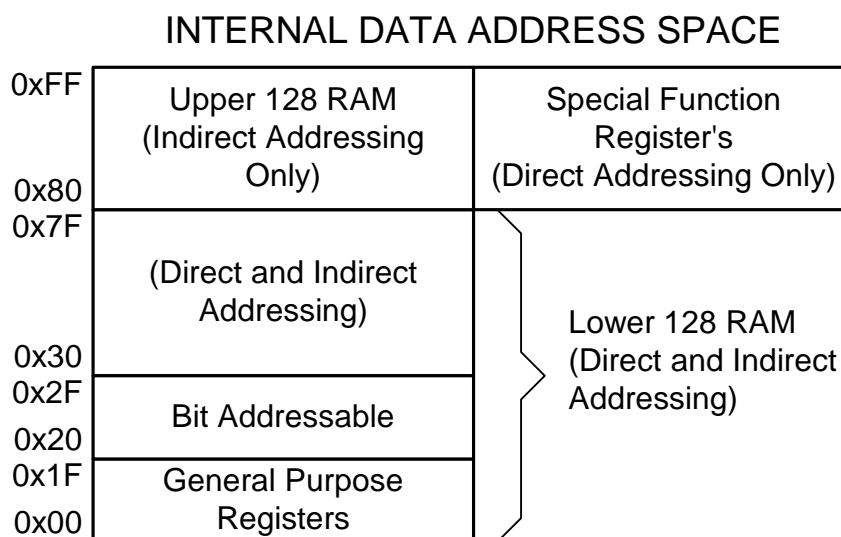
There is one unused opcode (0xA5) that performs the same function as NOP.  
All mnemonics copyrighted © Intel Corporation 1980.

# C8051F300/1/2/3/4/5

## 8.2.2. Data Memory

The CIP-51 includes 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight byte-wide registers. The next 16 bytes, locations 0x20 through 0x2F, may either be addressed as bytes or as 128 bit locations accessible with the direct addressing mode.

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFRs. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the upper 128 bytes of data memory. Figure 8.3 illustrates the data memory organization of the CIP-51.



**Figure 8.3. Data Memory Map**

## 8.2.3. General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in SFR Definition 8.4). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.



### 8.4.2. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the CIP-51 to enter Stop mode as soon as the instruction that sets the bit completes execution. In Stop mode the internal oscillator, CPU, and all digital peripherals are stopped; the state of the external oscillator circuit is not affected. Each analog peripheral (including the external oscillator circuit) may be shut down individually prior to entering Stop Mode. Stop mode can only be terminated by an internal or external reset. On reset, the CIP-51 performs the normal reset sequence and begins program execution at address 0x0000.

If enabled, the Missing Clock Detector will cause an internal reset and thereby terminate the Stop mode. The Missing Clock Detector should be disabled if the CPU is to be put in STOP mode for longer than the MCD timeout of 100  $\mu$ sec.

### SFR Definition 8.12. PCON: Power Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
GF5	GF4	GF3	GF2	GF1	GF0	STOP	IDLE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x87
<p>Bits7–2: GF5–GF0: General Purpose Flags 5-0. These are general purpose flags for use under software control.</p> <p>Bit1: STOP: Stop Mode Select. Setting this bit will place the CIP-51 in Stop mode. This bit will always be read as 0. 1: CPU goes into Stop mode (turns off internal oscillator).</p> <p>Bit0: IDLE: Idle Mode Select. Setting this bit will place the CIP-51 in Idle mode. This bit will always be read as 0. 1: CPU goes into Idle mode (shuts off clock to CPU, but clock to Timers, Interrupts, Serial Ports, and Analog Peripherals are still active).</p>								

## 10. Flash Memory

On-chip, reprogrammable 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 Table 10.1 for complete Flash memory electrical characteristics.

### 10.1. 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 “17. C2 Interface” on page 173**.

**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.**

#### 10.1.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function; Flash reads by user software are unrestricted. 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 10.2.

#### 10.1.2. Flash Erase Procedure

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: (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 but cannot set them; only an erase operation can set bits in Flash. **A byte location to be programmed should be erased before a new value is written.** The 8k byte 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:

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

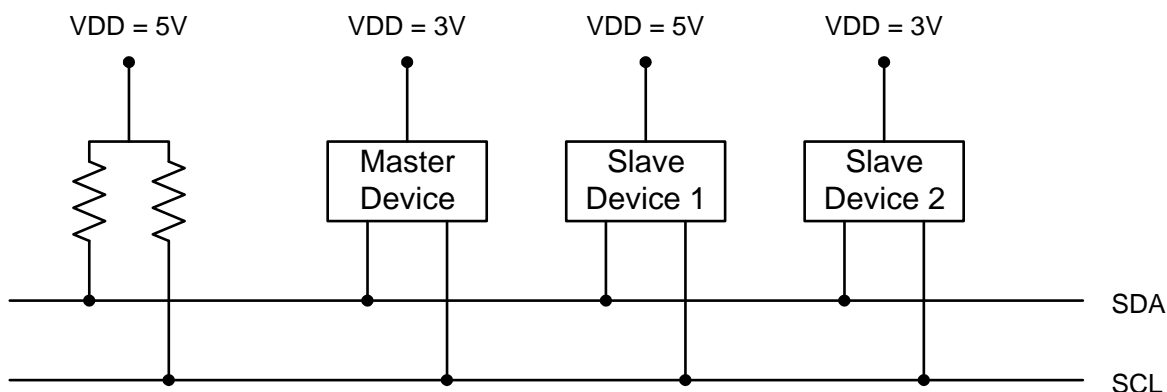
## 13.1. Supporting Documents

It is assumed the reader is familiar with or has access to the following supporting documents:

1. The I<sup>2</sup>C-Bus and How to Use It (including specifications), Philips Semiconductor.
2. The I<sup>2</sup>C-Bus Specification – Version 2.0, Philips Semiconductor.
3. System Management Bus Specification – Version 1.1, SBS Implementers Forum.

## 13.2. SMBus Configuration

Figure 13.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 and 5.0 V; different devices on the bus may operate at different voltage levels. The bidirectional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pull-up resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.



**Figure 13.2. Typical SMBus Configuration**

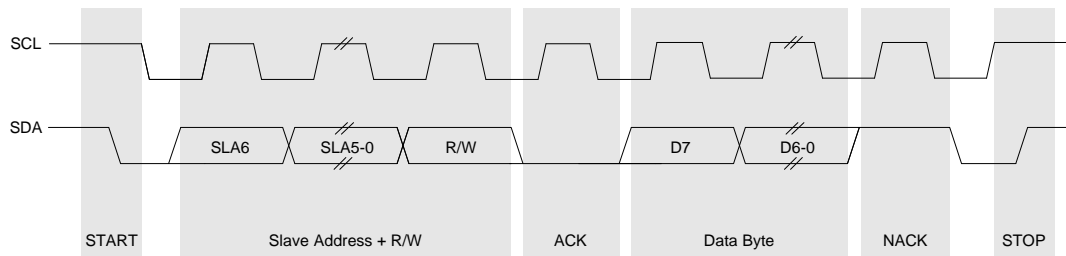
## 13.3. SMBus Operation

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. Note that it is not necessary to specify one device as the Master in a system; any device that transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Each byte that is received (by a master or slave) must be acknowledged (ACK) with a low SDA during a high SCL (see Figure 13.3). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

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

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 13.3 illustrates a typical SMBus transaction.



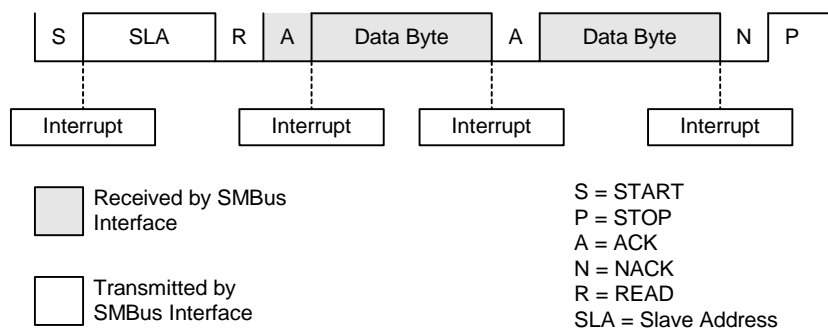
**Figure 13.3. SMBus Transaction**

### 13.3.1. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see **Section "13.3.4. SCL High (SMBus Free) Timeout" on page 114**). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

## 13.5.2. Master Receiver Mode

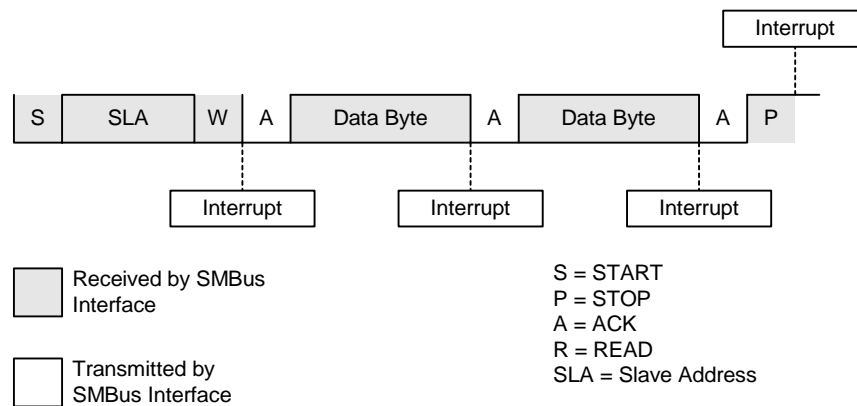
Serial data is received on SDA while the serial clock is output on SCL. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data. After each byte is received, ACKRQ is set to '1' and an interrupt is generated. Software must write the ACK bit (SMB0CN.1) to define the outgoing acknowledge value (Note: writing a '1' to the ACK bit generates an ACK; writing a '0' generates a NACK). Software should write a '0' to the ACK bit after the last byte is received, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. Note that the interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 13.6 shows a typical Master Receiver sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the 'data byte transferred' interrupts occur **before** the ACK cycle in this mode.



**Figure 13.6. Typical Master Receiver Sequence**

### 13.5.3. Slave Receiver Mode

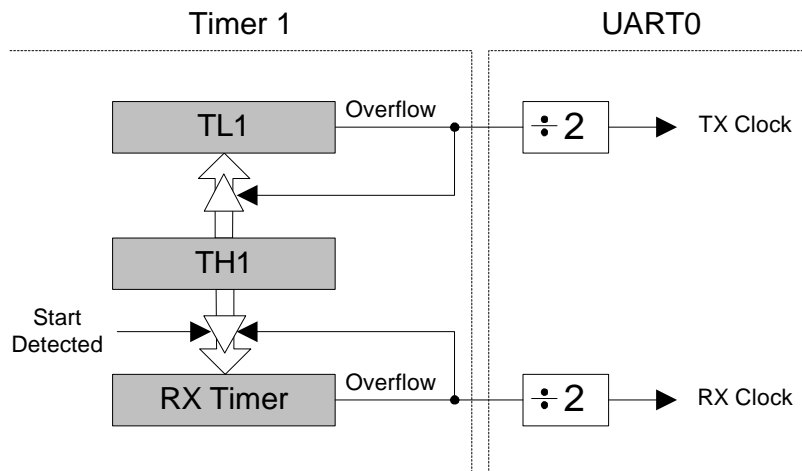
Serial data is received on SDA and the clock is received on SCL. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. Upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. Software responds to the received slave address with an ACK, or ignores the received slave address with a NACK. If the received slave address is ignored, slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received. Software must write the ACK bit after each received byte to ACK or NACK the received byte. The interface exits Slave Receiver Mode after receiving a STOP. Note that the interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver; see **Section 13.5.4** for details on this procedure. Figure 13.7 shows a typical Slave Receiver sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the 'data byte transferred' interrupts occur **before** the ACK cycle in this mode.



**Figure 13.7. Typical Slave Receiver Sequence**

## 14.1. Enhanced Baud Rate Generation

The UART0 baud rate is generated by Timer 1 in 8-bit auto-reload mode. The TX clock is generated by TL1; the RX clock is generated by a copy of TL1 (shown as RX Timer in Figure 14.2), which is not user accessible. Both TX and RX Timer overflows are divided by two to generate the TX and RX baud rates. The RX Timer runs when Timer 1 is enabled, and uses the same reload value (TH1). However, an RX Timer reload is forced when a START condition is detected on the RX pin. This allows a receive to begin any time a START is detected, independent of the TX Timer state.



**Figure 14.2. UART0 Baud Rate Logic**

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see **Section “15.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload” on page 145**). The Timer 1 reload value should be set so that overflows will occur at two times the desired UART baud rate frequency. Note that Timer 1 may be clocked by one of five sources: SYSCLK, SYSCLK / 4, SYSCLK / 12, SYSCLK / 48, or the external oscillator clock / 8. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 14.1.

$$UartBaudRate = \frac{T1_{CLK}}{(256 - T1H)} \times \frac{1}{2}$$

### Equation 14.1. UART0 Baud Rate

Where  $T1_{CLK}$  is the frequency of the clock supplied to Timer 1, and  $T1H$  is the high byte of Timer 1 (reload value). Timer 1 clock frequency is selected as described in **Section “15.2. Timer 2” on page 151**. A quick reference for typical baud rates and system clock frequencies is given in Tables 14.1 through 14.6. Note that the internal oscillator may still generate the system clock when the external oscillator is driving Timer 1 (see **Section “15.1. Timer 0 and Timer 1” on page 143** for more details).

## SFR Definition 15.1. TCON: Timer Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
						(bit addressable)		0x88
Bit7:	<p>TF1: Timer 1 Overflow Flag. Set by hardware when Timer 1 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine. 0: No Timer 1 overflow detected. 1: Timer 1 has overflowed.</p>							
Bit6:	<p>TR1: Timer 1 Run Control. 0: Timer 1 disabled. 1: Timer 1 enabled.</p>							
Bit5:	<p>TF0: Timer 0 Overflow Flag. Set by hardware when Timer 0 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine. 0: No Timer 0 overflow detected. 1: Timer 0 has overflowed.</p>							
Bit4:	<p>TR0: Timer 0 Run Control. 0: Timer 0 disabled. 1: Timer 0 enabled.</p>							
Bit3:	<p>IE1: External Interrupt 1. This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine if IT1 = 1. When IT1 = 0, this flag is set to '1' when /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 8.11).</p>							
Bit2:	<p>IT1: Interrupt 1 Type Select. This bit selects whether the configured /INT1 interrupt will be edge or level sensitive. /INT1 is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 8.11). 0: /INT1 is level triggered. 1: /INT1 is edge triggered.</p>							
Bit1:	<p>IE0: External Interrupt 0. This flag is set by hardware when an edge/level of type defined by IT0 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine if IT0 = 1. When IT0 = 0, this flag is set to '1' when /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 8.11).</p>							
Bit0:	<p>IT0: Interrupt 0 Type Select. This bit selects whether the configured /INT0 interrupt will be edge or level sensitive. /INT0 is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 8.11). 0: /INT0 is level triggered. 1: /INT0 is edge triggered.</p>							



## 15.2.2. 8-bit Timers with Auto-Reload

When T2SPLIT is set, Timer 2 operates as two 8-bit timers (TMR2H and TMR2L). Both 8-bit timers operate in auto-reload mode as shown in Figure 15.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. The Timer 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bit (T2XCLK in TMR2CN), as follows:

T2MH	T2XCLK	TMR2H Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

T2ML	T2XCLK	TMR2L Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

Note: External clock divided by 8 is synchronized with the system clock, and the external clock must be less than or equal to the system clock to operate in this mode.

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LEN (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LEN is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

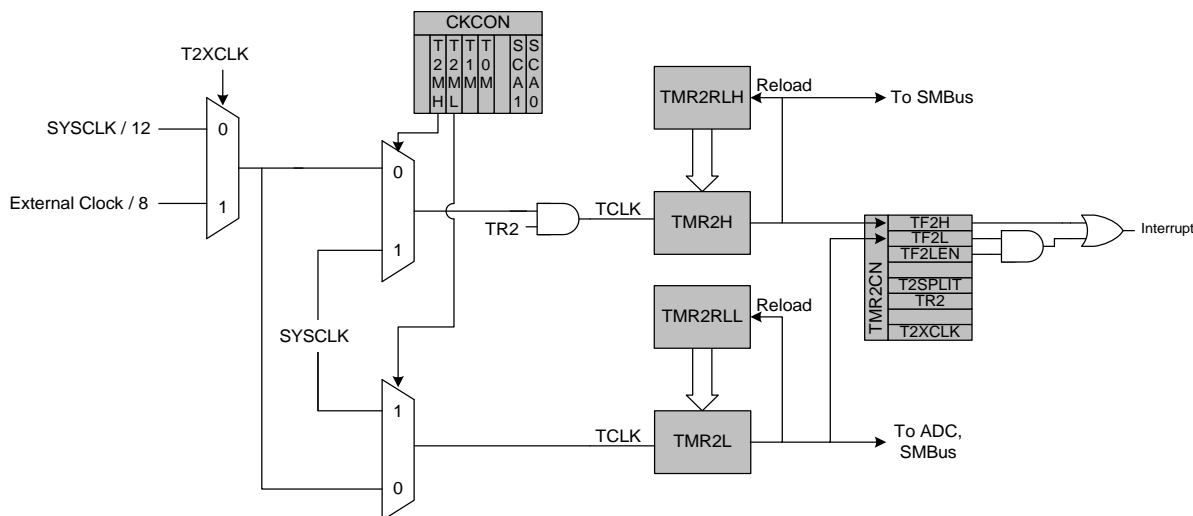
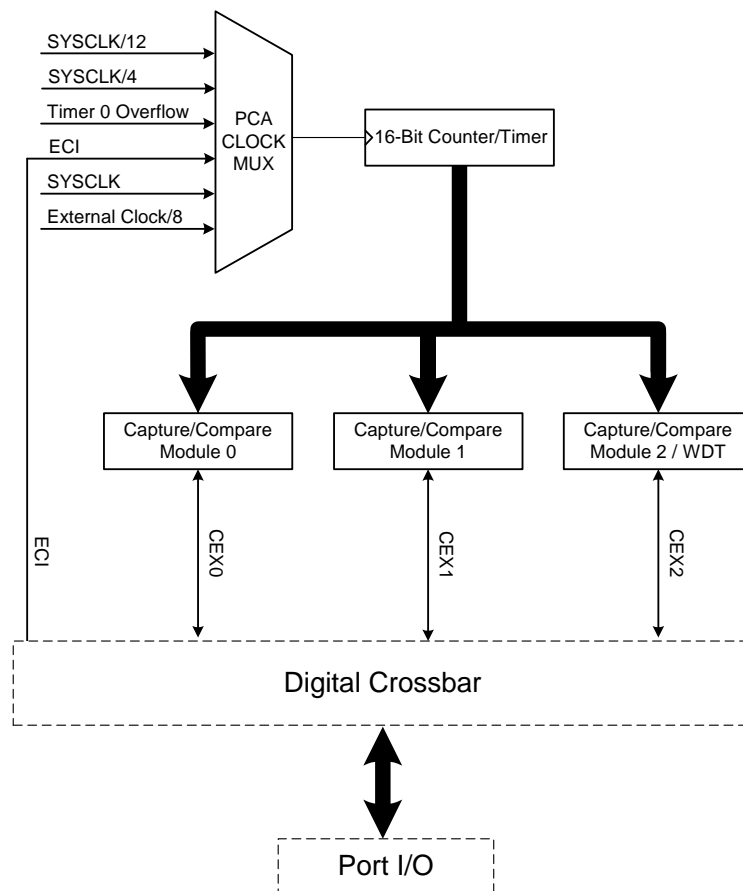


Figure 15.5. Timer 2 8-Bit Mode Block Diagram

## 16. Programmable Counter Array

The Programmable Counter Array (PCA0) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. The PCA consists of a dedicated 16-bit counter/timer and three 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 (See **Section “12.1. Priority Crossbar Decoder”** on page 104 for details on configuring the Crossbar). The counter/timer is driven by a programmable timebase that can select between six sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0 overflow, 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-Bit PWM, or 16-Bit PWM (each mode is described in **Section “16.2. Capture/Compare Modules”** on page 157). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The basic PCA block diagram is shown in Figure 16.1.

**Important Note:** The PCA Module 2 may be used as a watchdog timer (WDT), and is enabled in this mode following a system reset. Access to certain PCA registers is restricted while WDT mode is enabled. See **Section 16.3** for details.



**Figure 16.1. PCA Block Diagram**



## SFR Definition 16.2. PCA0MD: PCA Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CIDL	WDTE	WDLCK	—	CPS2	CPS1	CPS0	ECF	01000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xD9

- Bit7: CIDL: PCA Counter/Timer Idle Control.  
Specifies PCA behavior when CPU is in Idle Mode.  
0: PCA continues to function normally while the system controller is in Idle Mode.  
1: PCA operation is suspended while the system controller is in Idle Mode.
- Bit6: WDTE: Watchdog Timer Enable  
If this bit is set, PCA Module 2 is used as the Watchdog Timer.  
0: Watchdog Timer disabled.  
1: PCA Module 2 enabled as Watchdog Timer.
- Bit5: WDLCK: Watchdog Timer Lock  
This bit locks/unlocks the Watchdog Timer Enable. When WDLCK is set, the Watchdog Timer may not be disabled until the next system reset.  
0: Watchdog Timer Enable unlocked.  
1: Watchdog Timer Enable locked.
- Bit4: UNUSED. Read = 0b, Write = don't care.
- Bits3—1: CPS2—CPS0: PCA Counter/Timer Pulse Select.  
These bits select the clock source for the PCA counter

CPS2	CPS1	CPS0	Timebase
0	0	0	System clock divided by 12
0	0	1	System clock divided by 4
0	1	0	Timer 0 overflow
0	1	1	High-to-low transitions on ECI (max rate = system clock divided by 4)
1	0	0	System clock
1	0	1	External clock divided by 8*
1	1	0	Reserved
1	1	1	Reserved

\*Note: External oscillator source divided by 8 is synchronized with the system clock.

- Bit0: ECF: PCA Counter/Timer Overflow Interrupt Enable.  
This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.  
0: Disable the CF interrupt.  
1: Enable a PCA Counter/Timer Overflow interrupt when CF (PCA0CN.7) is set.

**Note:** When the WDTE bit is set to '1', the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled.

SFR Definition 16.6. PCA0CPLn: PCA Capture Module Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xFB, 0xE9, 0xEB
PCA0CPLn Address:     PCA0CPL0 = 0xFB (n = 0) PCA0CPL1 = 0xE9 (n = 1) PCA0CPL2 = 0xEB (n = 2)								
Bits7–0: PCA0CPLn: PCA Capture Module Low Byte. The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture Module n.								

SFR Definition 16.7. PCA0CPHn: PCA Capture Module High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xFC, 0xEA, 0xEC
PCA0CPHn Address:     PCA0CPH0 = 0xFC (n = 0) PCA0CPH1 = 0xEA (n = 1) PCA0CPH2 = 0xEC(n = 2)								
Bits7–0: PCA0CPHn: PCA Capture Module High Byte. The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture Module n.								