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), UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	8
Program Memory Size	2KB (2K x 8)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	14-SOIC (0.154", 3.90mm Width)
Supplier Device Package	14-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051t605-gsr

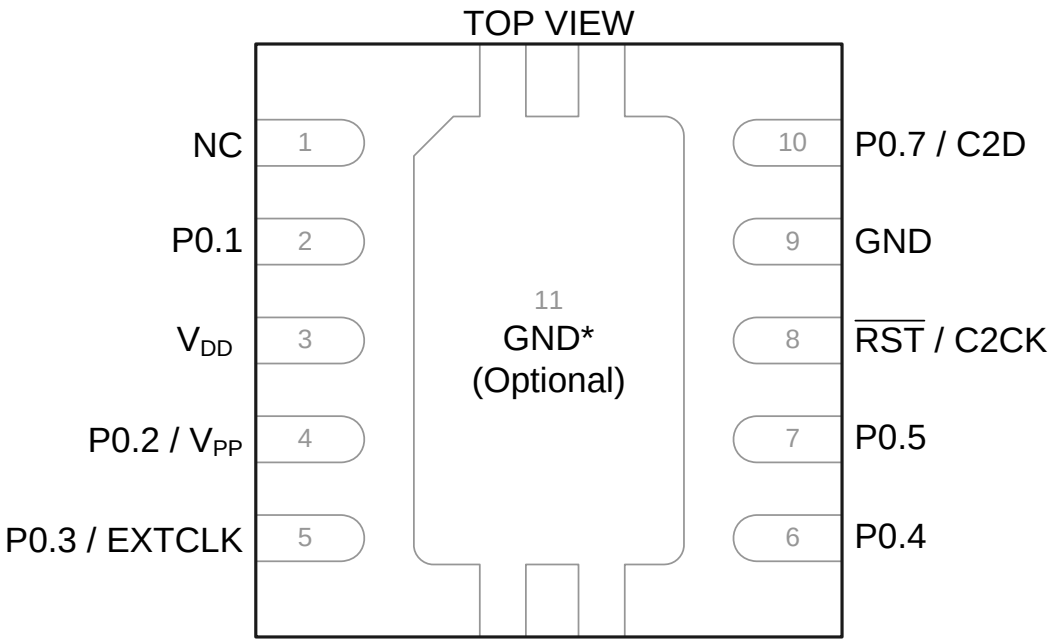

Figure 3.3. C8051T606-GM QFN11 Pinout Diagram (Top View)

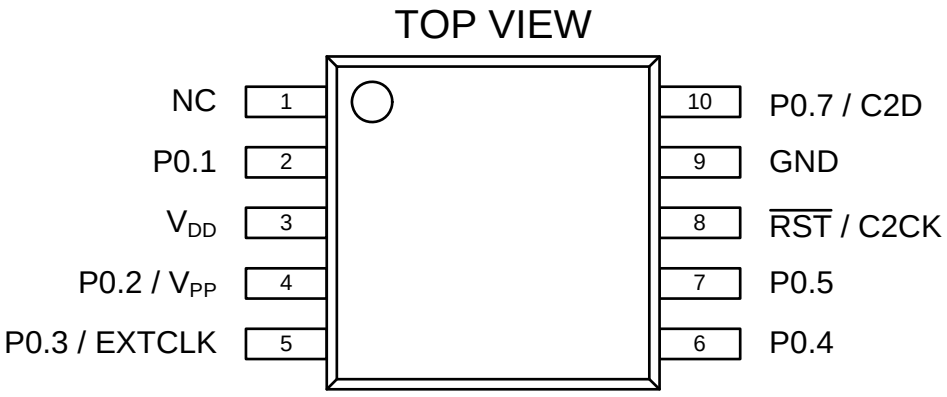

Figure 3.4. C8051T606-GT MSOP10 Pinout Diagram (Top View)

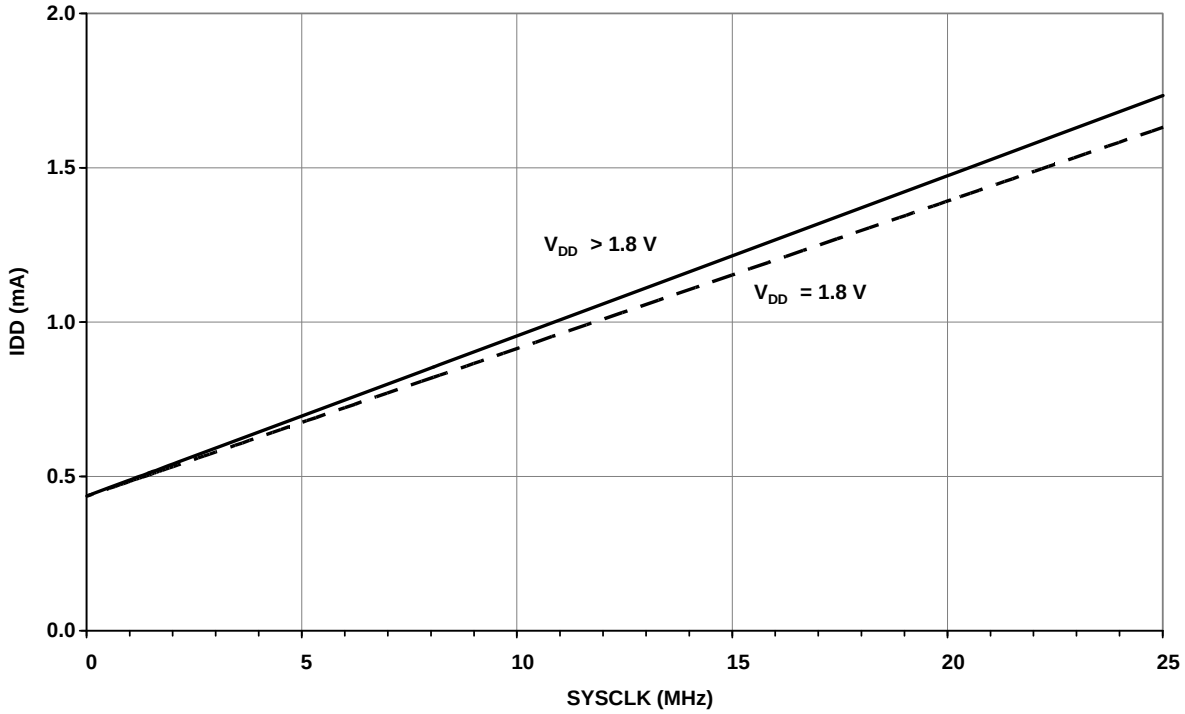

Figure 8.3. C8051T600/1/2/3/4/5 Idle Mode Supply Current vs. Frequency (MPCE = 1)

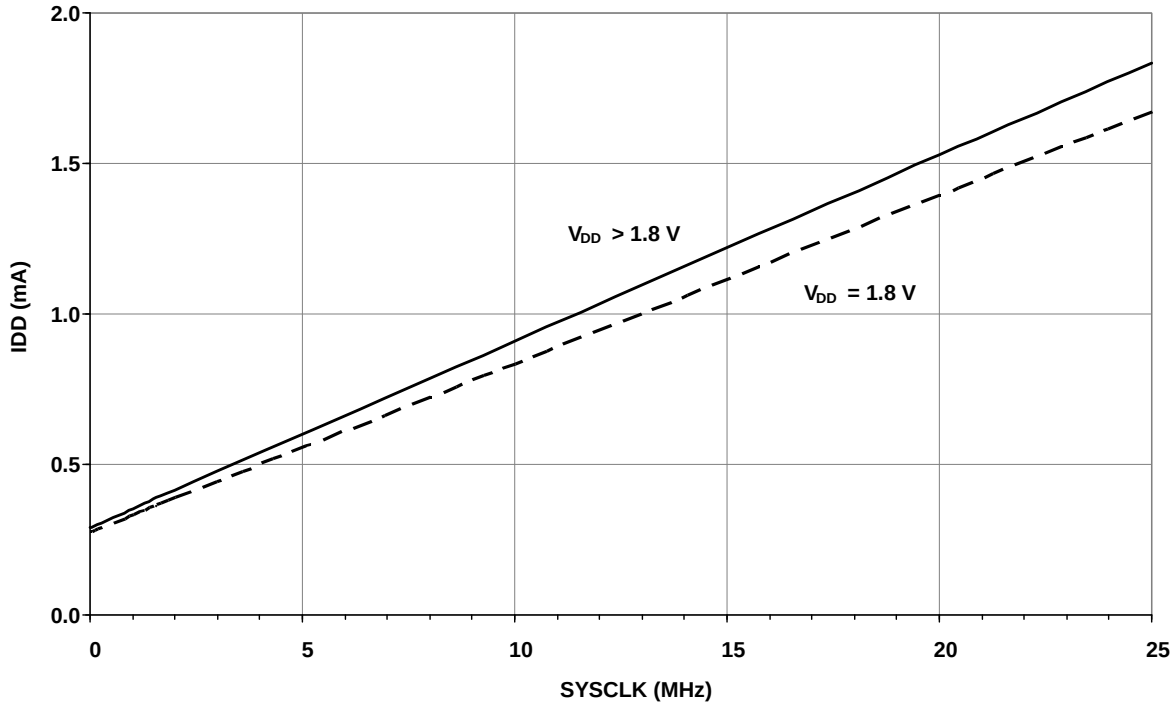

Figure 8.4. C8051T606 Idle Mode Digital Current vs. Frequency (MPCE = 1)

Table 14.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
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 is 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
Program Branching			
ACALL addr11	Absolute subroutine call	2	3
LCALL addr16	Long subroutine call	3	4
RET	Return from subroutine	1	5
RETI	Return from interrupt	1	5
AJMP addr11	Absolute jump	2	3
LJMP addr16	Long jump	3	4
SJMP rel	Short jump (relative address)	2	3
JMP @A+DPTR	Jump indirect relative to DPTR	1	3
JZ rel	Jump if A equals zero	2	2/3
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

SFR Definition 14.6. PSW: Program Status Word

Bit	7	6	5	4	3	2	1	0
Name	CY	AC	F0	RS[1:0]		OV	F1	PARITY
Type	R/W	R/W	R/W	R/W		R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD0; Bit-Addressable

Bit	Name	Function
7	CY	**Carry Flag.** This bit is set when the last arithmetic operation resulted in a carry (addition) or a borrow (subtraction). It is cleared to logic 0 by all other arithmetic operations.
6	AC	**Auxiliary Carry Flag.** This bit is set when the last arithmetic operation resulted in a carry into (addition) or a borrow from (subtraction) the high order nibble. It is cleared to logic 0 by all other arithmetic operations.
5	F0	**User Flag 0.** This is a bit-addressable, general purpose flag for use under software control.
4:3	RS[1:0]	**Register Bank Select.** These bits select which register bank is used during register accesses. 00: Bank 0, Addresses 0x00-0x07 01: Bank 1, Addresses 0x08-0x0F 10: Bank 2, Addresses 0x10-0x17 11: Bank 3, Addresses 0x18-0x1F
2	OV	**Overflow Flag.** This bit is set to 1 under the following circumstances: ■ An ADD, ADDC, or SUBB instruction causes a sign-change overflow. ■ A MUL instruction results in an overflow (result is greater than 255). ■ A DIV instruction causes a divide-by-zero condition. The OV bit is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases.
1	F1	**User Flag 1.** This is a bit-addressable, general purpose flag for use under software control.
0	PARITY	**Parity Flag.** This bit is set to logic 1 if the sum of the eight bits in the accumulator is odd and cleared if the sum is even.

15.2. Data Memory

The C8051T600/1/2/3/4/5 devices include 256 bytes of RAM, and the C8051T606 devices include 128 bytes of RAM. This memory is mapped into the internal data memory space of the 8051 controller core. The RAM memory organization of the C8051T600/1/2/3/4/5/6 device family is shown in Figure 15.2

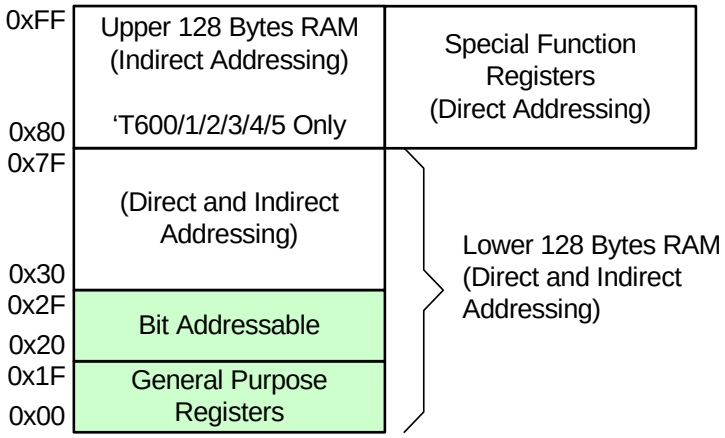

Figure 15.2. RAM Memory Map

15.2.1. Internal RAM

The 256 bytes of internal RAM on the C8051T600/1/2/3/4/5 are mapped into the data memory space from 0x00 through 0xFF. The 128 bytes of internal RAM on the C8051T606 are mapped into the data memory space from 0x00 through 0x7F. The 128 bytes of data memory from 0x00 to 0x7F on all devices are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access these 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 available on the C8051T600/1/2/3/4/5 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 15.2 illustrates the data memory organization of the C8051T600/1/2/3/4/5/6.

17. Interrupts

The C8051T600/1/2/3/4/5/6 includes an extended interrupt system supporting a total of 12 interrupt sources with two priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

If interrupts are enabled for the source, an interrupt request is generated when the interrupt-pending flag is set. As soon as execution of the current instruction is complete, the CPU generates an LCALL to a predetermined address to begin execution of an interrupt service routine (ISR). Each ISR must end with an RETI instruction, which returns program execution to the next instruction that would have been executed if the interrupt request had not occurred. If interrupts are not enabled, the interrupt-pending flag is ignored by the hardware and program execution continues as normal. (The interrupt-pending flag is set to logic 1 regardless of the interrupt's enable/disable state.)

Each interrupt source can be individually enabled or disabled through the use of an associated interrupt enable bit in an SFR (IE–EIE1). However, interrupts must first be globally enabled by setting the EA bit (IE.7) to logic 1 before the individual interrupt enables are recognized. Setting the EA bit to logic 0 disables all interrupt sources regardless of the individual interrupt-enable settings.

Note: Any instruction that clears a bit to disable an interrupt should be immediately followed by an instruction that has two or more opcode bytes. Using EA (global interrupt enable) as an example:

```
// in 'C':
EA = 0; // clear EA bit.
EA = 0; // this is a dummy instruction with two-byte opcode.

; in assembly:
CLR EA ; clear EA bit.
CLR EA ; this is a dummy instruction with two-byte opcode.
```

For example, if an interrupt is posted during the execution phase of a "CLR EA" opcode (or any instruction which clears a bit to disable an interrupt source), and the instruction is followed by a single-cycle instruction, the interrupt may be taken. However, a read of the enable bit will return a '0' inside the interrupt service routine. When the bit-clearing opcode is followed by a multi-cycle instruction, the interrupt will not be taken.

Some interrupt-pending flags are automatically cleared by the hardware when the CPU vectors to the ISR. However, most are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the return-from-interrupt (RETI) instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

SFR Definition 17.4. EIP1: Extended Interrupt Priority 1

Bit	7	6	5	4	3	2	1	0
Name			PCP0R	PCP0F	PPCA0	PADC0	PWADC0	PSMB0
Type	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	0	0	0	0	0	0

SFR Address = 0xF6

Bit	Name	Function
7:6	Unused	Unused. Read = 11b; Write = Don't Care.
5	PCP0R	**Comparator0 (CP0) Rising Edge Interrupt Priority Control.** This bit sets the priority of the CP0 rising edge interrupt. 0: CP0 rising edge interrupt set to low priority level. 1: CP0 rising edge interrupt set to high priority level.
4	PCP0F	**Comparator0 (CP0) Falling Edge Interrupt Priority Control.** This bit sets the priority of the CP0 falling edge interrupt. 0: CP0 falling edge interrupt set to low priority level. 1: CP0 falling edge interrupt set to high priority level.
3	PPCA0	**Programmable Counter Array (PCA0) Interrupt Priority Control.** This bit sets the priority of the PCA0 interrupt. 0: PCA0 interrupt set to low priority level. 1: PCA0 interrupt set to high priority level.
2	PADC0	**ADC0 Conversion Complete Interrupt Priority Control.** This bit sets the priority of the ADC0 Conversion Complete interrupt. 0: ADC0 Conversion Complete interrupt set to low priority level. 1: ADC0 Conversion Complete interrupt set to high priority level.
1	PWADC0	**ADC0 Window Comparator Interrupt Priority Control.** This bit sets the priority of the ADC0 Window interrupt. 0: ADC0 Window interrupt set to low priority level. 1: ADC0 Window interrupt set to high priority level.
0	PSMB0	**SMBus (SMB0) Interrupt Priority Control.** This bit sets the priority of the SMB0 interrupt. 0: SMB0 interrupt set to low priority level. 1: SMB0 interrupt set to high priority level.

19.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin is driven low until V_{DD} settles above V_{RST}. A delay occurs before the device is released from reset; the delay decreases as the V_{DD} ramp time increases (V_{DD} ramp time is defined as how fast V_{DD} ramps from 0 V to V_{RST}). Figure 19.2. plots the power-on and V_{DD} monitor event timing. The maximum V_{DD} ramp time is 1 ms; slower ramp times may cause the device to be released from reset before V_{DD} reaches the V_{RST} level. For ramp times less than 1 ms, the power-on reset delay ($T_{PORDelay}$) is typically less than 0.3 ms.

On exit from a power-on or V_{DD} monitor reset, the PORSF flag (RSTSRC.1) is set by hardware to logic 1. When PORSF is set, all of the other reset flags in the RSTSRC Register are indeterminate (PORSF is cleared by all other resets). Since all resets cause program execution to begin at the same location (0x0000) software can read the PORSF flag to determine if a power-up was the cause of reset. The content of internal data memory should be assumed to be undefined after a power-on reset. The V_{DD} monitor is disabled following a power-on reset.

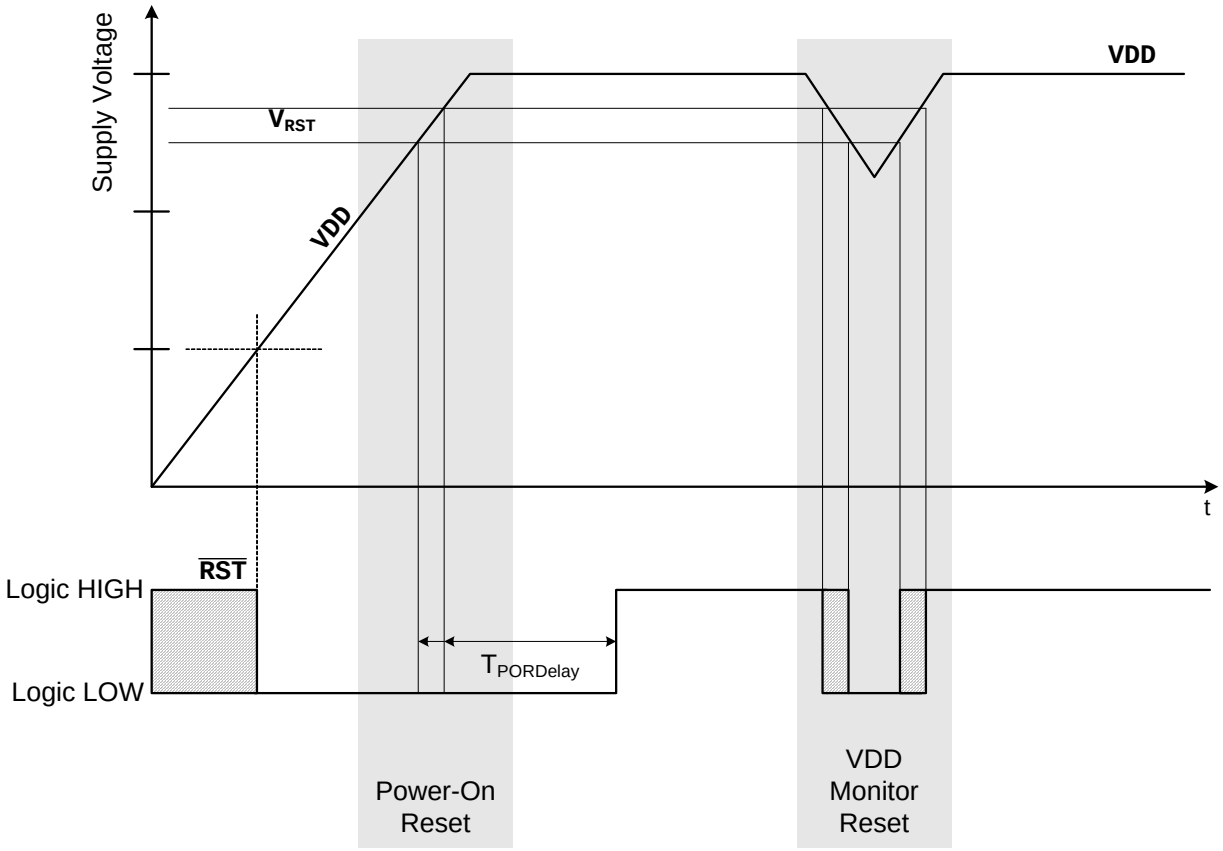

Figure 19.2. Power-On and V_{DD} Monitor Reset Timing

19.2. Power-Fail Reset/V_{DD} Monitor

When a power-down transition or power irregularity causes V_{DD} to drop below V_{RST}, the power supply monitor will drive the $\overline{RST}$ pin low and hold the CIP-51 in a reset state (see Figure 19.2). When V_{DD} returns to a level above V_{RST}, the CIP-51 will be released from the reset state. Note that even though internal data memory contents are not altered by the power-fail reset, it is impossible to determine if V_{DD} dropped below the level required for data retention. If the PORSF flag reads 1, the data may no longer be valid. The V_{DD} monitor is disabled after power-on resets. Its defined state (enabled/disabled) is not altered by any other reset source. For example, if the V_{DD} monitor is enabled by code and a software reset is performed, the V_{DD} monitor will still be enabled after the reset.

Important Note: If the V_{DD} monitor is being turned on from a disabled state, it has the potential to generate a system reset. The V_{DD} monitor is enabled and selected as a reset source by writing the PORSF flag in RSTSRC to 1.

See Figure 19.2 for V_{DD} monitor timing; note that the power-on-reset delay is not incurred after a V_{DD} monitor reset. See Table 8.4 for complete electrical characteristics of the V_{DD} monitor.

19.3. External Reset

The external $\overline{RST}$ pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the $\overline{RST}$ pin generates a reset; an external pullup and/or decoupling of the $\overline{RST}$ pin may be necessary to avoid erroneous noise-induced resets. See Table 8.4 for complete $\overline{RST}$ pin specifications. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

19.4. Missing Clock Detector Reset

The Missing Clock Detector (MCD) is a one-shot circuit that is triggered by the system clock. If the system clock remains high or low for more than the time specified in Section “8. Electrical Characteristics” on page 30, the one-shot will time out and generate a reset. After a MCD reset, the MCDRSF flag (RSTSRC.2) will read 1, signifying the MCD as the reset source; otherwise, this bit reads 0. Writing a 1 to the MCDRSF bit enables the Missing Clock Detector; writing a 0 disables it. The state of the $\overline{RST}$ pin is unaffected by this reset.

19.5. Comparator0 Reset

Comparator0 can be configured as a reset source by writing a 1 to the C0RSEF flag (RSTSRC.5). Comparator0 should be enabled and allowed to settle prior to writing to C0RSEF to prevent any turn-on chatter on the output from generating an unwanted reset. The Comparator0 reset is active-low: if the non-inverting input voltage (on CP0+) is less than the inverting input voltage (on CP0-), the device is put into the reset state. After a Comparator0 reset, the C0RSEF flag (RSTSRC.5) will read 1 signifying Comparator0 as the reset source; otherwise, this bit reads 0. The state of the $\overline{RST}$ pin is unaffected by this reset.

19.6. PCA Watchdog Timer Reset

The watchdog timer (WDT) function of the programmable counter array (PCA) can be used to prevent software from running out of control during a system malfunction. The PCA WDT function can be enabled or disabled by software as described in Section “26.4. Watchdog Timer Mode” on page 170; the WDT is enabled and clocked by SYSCLK/12 following any reset. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit (RSTSRC.5) is set to 1. The state of the $\overline{RST}$ pin is unaffected by this reset.

19.7. EPROM Error Reset

If an EPROM read or write targets an illegal address, a system reset is generated. This may occur due to any of the following:

- Programming hardware attempts to write or read an EPROM location which is above the user code space address limit.
- An EPROM read from firmware is attempted above user code space. This occurs when a MOVC operation is attempted above the user code space address limit.
- A Program read is attempted above user code space. This occurs when user code attempts to branch to an address above the user code space address limit.

The MEMERR bit (RSTSRC.6) is set following an EPROM error reset. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

19.8. Software Reset

Software may force a reset by writing a 1 to the SWRSF bit (RSTSRC.4). The SWRSF bit will read 1 following a software forced reset. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

21.3. External Oscillator Drive Circuit

The external oscillator circuit may drive an external capacitor or RC network. A CMOS clock may also provide a clock input. In RC, capacitor, or CMOS clock configuration, the clock source should be wired to the EXTCLK pin as shown in Figure 21.1. The type of external oscillator must be selected in the OSCXCN register, and the frequency control bits (XFCN) must be selected appropriately (see SFR Definition 21.3).

Important Note on External Oscillator Usage: Port pins must be configured when using the external oscillator circuit. When the external oscillator drive circuit is enabled in capacitor, RC, or CMOS clock mode, Port pin P0.3 is used as EXTCLK. The Port I/O Crossbar should be configured to skip the Port pin used by the oscillator circuit; see Section “22.3. Priority Crossbar Decoder” on page 111 for Crossbar configuration. Additionally, when using the external oscillator circuit in capacitor or RC mode, the associated Port pin should be configured as an **analog input**. In CMOS clock mode, the associated pin should be configured as a **digital input**. See Section “22.4. Port I/O Initialization” on page 114 for details on Port input mode selection.

SFR Definition 23.1. SMB0CF: SMBus Clock/Configuration

Bit	7	6	5	4	3	2	1	0
Name	ENSMB	INH	BUSY	EXTHOLD	SMBTOE	SMBFTE	SMBCS[1:0]	
Type	R/W	R/W	R	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC1

Bit	Name	Function
7	ENSMB	**SMBus Enable.** This bit enables the SMBus interface when set to 1. When enabled, the interface constantly monitors the SDA and SCL pins.
6	INH	**SMBus Slave Inhibit.** When this bit is set to logic 1, the SMBus does not generate an interrupt when slave events occur. This effectively removes the SMBus slave from the bus. Master Mode interrupts are not affected.
5	BUSY	**SMBus Busy Indicator.** This bit is set to logic 1 by hardware when a transfer is in progress. It is cleared to logic 0 when a STOP or free-timeout is sensed.
4	EXTHOLD	**SMBus Setup and Hold Time Extension Enable.** This bit controls the SDA setup and hold times according to Table 23.2. 0: SDA Extended Setup and Hold Times disabled. 1: SDA Extended Setup and Hold Times enabled.
3	SMBTOE	**SMBus SCL Timeout Detection Enable.** This bit enables SCL low timeout detection. If set to logic 1, the SMBus forces Timer 3 to reload while SCL is high and allows Timer 3 to count when SCL goes low. If Timer 3 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL is high. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus communication.
2	SMBFTE	**SMBus Free Timeout Detection Enable.** When this bit is set to logic 1, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods.
1:0	SMBCS[1:0]	**SMBus Clock Source Selection.** These two bits select the SMBus clock source, which is used to generate the SMBus bit rate. The selected device should be configured according to Equation 23.1. 00: Timer 0 Overflow 01: Timer 1 Overflow 10: Timer 2 High Byte Overflow 11: Timer 2 Low Byte Overflow

SFR Definition 23.2. SMB0CN: SMBus Control

Bit	7	6	5	4	3	2	1	0
Name	MASTER	TXMODE	STA	STO	ACKRQ	ARBLOST	ACK	SI
Type	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC0; Bit-Addressable

Bit	Name	Description	Read	Write
7	MASTER	**SMBus Master/Slave Indicator.** This read-only bit indicates when the SMBus is operating as a master.	0: SMBus operating in slave mode. 1: SMBus operating in master mode.	N/A
6	TXMODE	**SMBus Transmit Mode Indicator.** This read-only bit indicates when the SMBus is operating as a transmitter.	0: SMBus in Receiver Mode. 1: SMBus in Transmitter Mode.	N/A
5	STA	**SMBus Start Flag.**	0: No Start or repeated Start detected. 1: Start or repeated Start detected.	0: No Start generated. 1: When Configured as a Master, initiates a START or repeated START.
4	STO	**SMBus Stop Flag.**	0: No Stop condition detected. 1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).	0: No STOP condition is transmitted. 1: When configured as a Master, causes a STOP condition to be transmitted after the next ACK cycle. Cleared by Hardware.
3	ACKRQ	**SMBus Acknowledge Request.**	0: No Ack requested 1: ACK requested	N/A
2	ARBLOST	**SMBus Arbitration Lost Indicator.**	0: No arbitration error. 1: Arbitration Lost	N/A
1	ACK	**SMBus Acknowledge.**	0: NACK received. 1: ACK received.	0: Send NACK 1: Send ACK
0	SI	**SMBus Interrupt Flag.** This bit is set by hardware under the conditions listed in Table 15.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled.	0: No interrupt pending 1: Interrupt Pending	0: Clear interrupt, and initiate next state machine event. 1: Force interrupt.

23.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. Upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK or ignore the received slave address with a NACK.

If the received slave address is ignored by software (by NACKing the address), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters slave transmitter mode and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit. If the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (an error condition may be generated if SMB0DAT is written following a received NACK while in slave transmitter mode). The interface exits slave transmitter mode after receiving a STOP. Note that the interface will switch to slave receiver mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 23.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode.

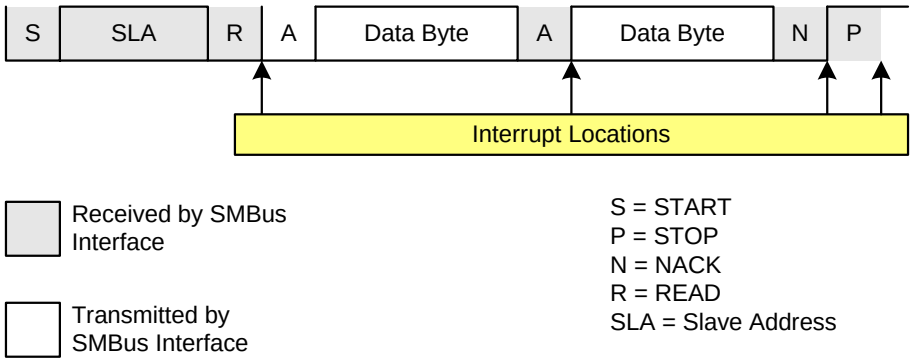

Figure 23.8. Typical Slave Read Sequence

23.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. Table 23.4 describes the typical actions taken by firmware on each condition. In the table, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the SMBus specification. Highlighted responses are allowed by hardware but do not conform to the SMBus specification.

24.2. 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 24.3.

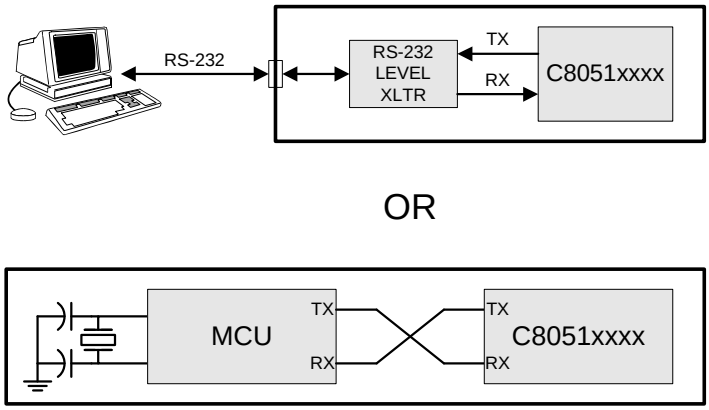

Figure 24.3. UART Interconnect Diagram

24.2.1. 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.

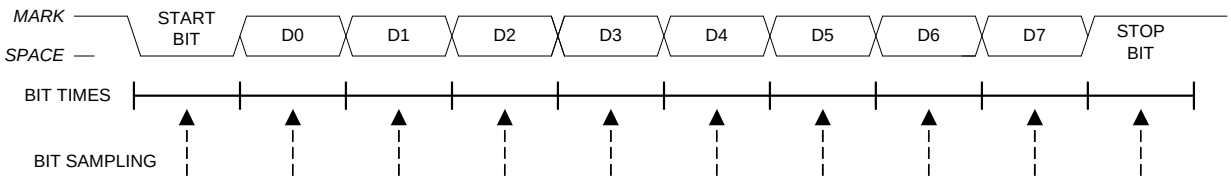

Figure 24.4. 8-Bit UART Timing Diagram

SFR Definition 25.8. TMR2CN: Timer 2 Control

Bit	7	6	5	4	3	2	1	0
Name	TF2H	TF2L	TF2LEN		T2SPLIT	TR2		T2XCLK
Type	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC8; Bit-Addressable

Bit	Name	Function
7	TF2H	**Timer 2 High Byte Overflow Flag.** Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF2L	**Timer 2 Low Byte Overflow Flag.** Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. TF2L will be set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware.
5	TF2LEN	**Timer 2 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 2 low byte interrupts. If Timer 2 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 2 overflows.
4	Unused	Unused. Read = 0b; Write = Don't Care
3	T2SPLIT	**Timer 2 Split Mode Enable.** When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload. 0: Timer 2 operates in 16-bit auto-reload mode. 1: Timer 2 operates as two 8-bit auto-reload timers.
2	TR2	**Timer 2 Run Control.** Timer 2 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in split mode.
1	Unused	Unused. Read = 0b; Write = Don't Care
0	T2XCLK	**Timer 2 External Clock Select.** This bit selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 2 Clock Select bits (T2MH and T2ML in register CKCON) may still be used to select between the external clock and the system clock for either timer. 0: Timer 2 clock is the system clock divided by 12. 1: Timer 2 clock is the external clock divided by 8 (synchronized with SYSCLK).

26.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte (MSB) of the 16-bit counter/timer and PCA0L is the low byte (LSB). Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register. **Reading the PCA0L register first guarantees an accurate reading of the entire 16-bit PCA0 counter.** Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer as shown in Table 26.1.

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle Mode.

Table 26.1. PCA Timebase Input Options

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 oscillator source divided by 8*
1	1	x	Reserved
Note: External oscillator source divided by 8 is synchronized with the system clock.			

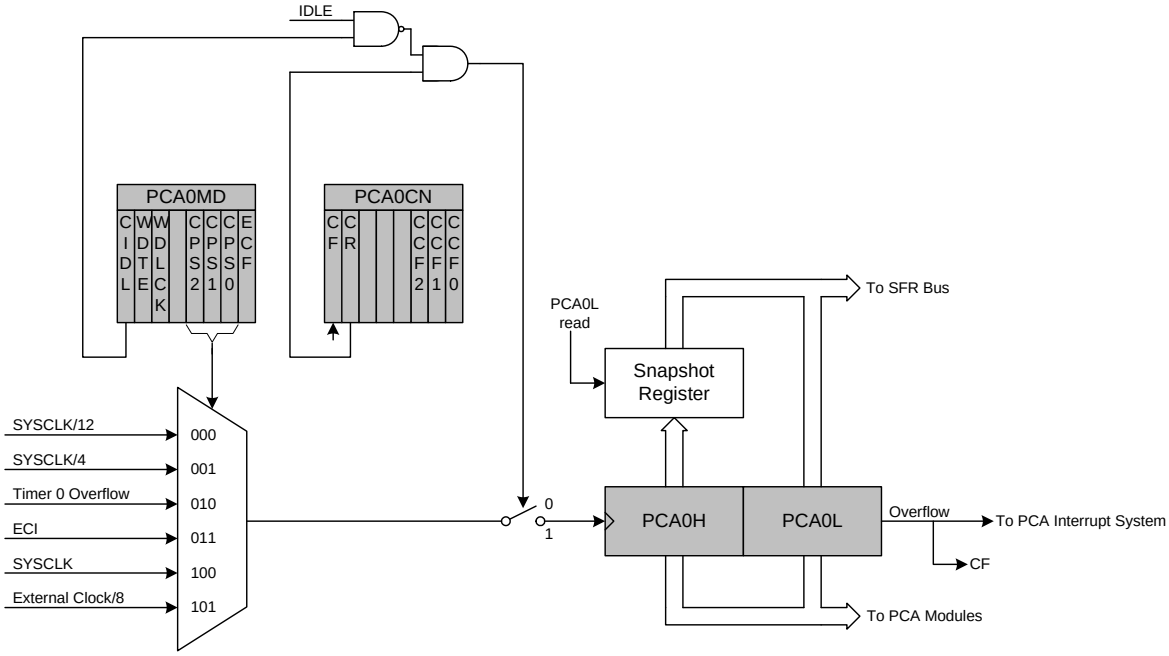

Figure 26.2. PCA Counter/Timer Block Diagram

The 8-bit offset held in PCA0CPH2 is compared to the upper byte of the 16-bit PCA counter. This offset value is the number of PCA0L overflows before a reset. Up to 256 PCA clocks may pass before the first PCA0L overflow occurs, depending on the value of the PCA0L when the update is performed. The total offset is then given (in PCA clocks) by Equation 26.4, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL2) + (256 - PCA0L)$$

Equation 26.4. Watchdog Timer Offset in PCA Clocks

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH2 and PCA0H. Software may force a WDT reset by writing a 1 to the CCF2 flag (PCA0CN.2) while the WDT is enabled.

26.4.2. Watchdog Timer Usage

To configure the WDT, perform the following tasks:

1. Disable the WDT by writing a 0 to the WDTE bit.
2. Select the desired PCA clock source (with the CPS2–CPS0 bits).
3. Load PCA0CPL2 with the desired WDT update offset value.
4. Configure the PCA Idle Mode (set CIDL if the WDT should be suspended while the CPU is in Idle Mode).
5. Enable the WDT by setting the WDTE bit to 1.
6. Reset the WDT timer by writing to PCA0CPH2.

The PCA clock source and Idle Mode select cannot be changed while the WDT is enabled. The Watchdog Timer is enabled by setting the WDTE or WDLCK bits in the PCA0MD register. When WDLCK is set, the WDT cannot be disabled until the next system reset. If WDLCK is not set, the WDT is disabled by clearing the WDTE bit.

The WDT is enabled following any reset. The PCA0 counter clock defaults to the system clock divided by 12, PCA0L defaults to 0x00, and PCA0CPL2 defaults to 0x00. Using Equation 26.4, this results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. Table 26.3 lists some example timeout intervals for typical system clocks.

27. C2 Interface

C8051T600/1/2/3/4/5/6 devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow EPROM programming and in-system debugging with the production part installed in the end application. The C2 interface operates using only two pins: a bi-directional data signal (C2D), and a clock input (C2CK). See the C2 Interface Specification for details on the C2 protocol.

27.1. C2 Interface Registers

The following describes the C2 registers necessary to perform EPROM programming functions through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

C2 Register Definition 27.1. C2ADD: C2 Address

Bit	7	6	5	4	3	2	1	0
Name	C2ADD[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
7:0	C2ADD[7:0]	**Write: C2 Address.** Selects the target Data register for C2 Data Read and Data Write commands according to the following list.

Address	Name	Description
0x00	DEVICEID	Selects the Device ID Register (read only)
0x01	REVID	Selects the Revision ID Register (read only)
0x02	DEVCTL	Selects the C2 Device Control Register
0xDF	EPCTL	Selects the C2 EPROM Programming Control Register
0xBF	EPDAT	Selects the C2 EPROM Data Register
0xB7	EPSTAT	Selects the C2 EPROM Status Register
0xAF	EPADDRH	Selects the C2 EPROM Address High Byte Register
0xAE	EPADDRL	Selects the C2 EPROM Address Low Byte Register
0xA9	CRC0	Selects the CRC0 Register
0xAA	CRC1	Selects the CRC1 Register
0xAB	CRC2	Selects the CRC2 Register
0xAC	CRC3	Selects the CRC3 Register

Read: C2 Status

Returns status information on the current programming operation.

When the MSB (bit 7) is set to ‘1’, a read or write operation is in progress. All other bits can be ignored by the programming tools.