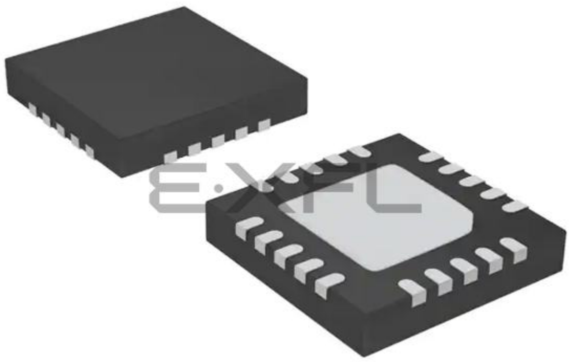

Welcome to E-XFL.COM

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	CIP-51™
Core Size	8-Bit
Speed	25MHz
Connectivity	I²C, SMBus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	15
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.2V ~ 3.6V
Data Converters	A/D 15x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-VFQFN Exposed Pad
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f850-b-gm

2.5. Communications and other Digital Peripherals

2.5.1. Universal Asynchronous Receiver/Transmitter (UART0)

The UART uses two signals (TX and RX) and a predetermined fixed baud rate to provide asynchronous communications with other devices.

The UART module provides the following features:

- Asynchronous transmissions and receptions.
- Baud rates up to SYSCLK / 2 (transmit) or SYSCLK / 8 (receive).
- 8- or 9-bit data.
- Automatic start and stop generation.

2.5.2. Serial Peripheral Interface (SPI0)

SPI is a 3- or 4-wire communication interface that includes a clock, input data, output data, and an optional select signal.

The SPI module includes the following features:

- Supports 3- or 4-wire master or slave modes.
- Supports external clock frequencies up to SYSCLK / 2 in master mode and SYSCLK / 10 in slave mode.
- Support for all clock phase and polarity modes.
- 8-bit programmable clock rate.
- Support for multiple masters on the same data lines.

2.5.3. System Management Bus / I2C (SMBus0)

The SMBus interface is a two-wire, bi-directional serial bus compatible with both I2C and SMBus protocols. The two clock and data signals operate in open-drain mode with external pull-ups to support automatic bus arbitration.

Reads and writes to the interface are byte-oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/8th of the system clock as a master or slave, which can be faster than allowed by the SMBus / I2C specification, depending on the clock source used. A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and start/stop control and generation.

The SMBus module includes the following features:

- Standard (up to 100 kbps) and Fast (400 kbps) transfer speeds.
- Support for master, slave, and multi-master modes.
- Hardware synchronization and arbitration for multi-master mode.
- Clock low extending (clock stretching) to interface with faster masters.
- Hardware support for 7-bit slave and general call address recognition.
- Firmware support for 10-bit slave address decoding.
- Ability to inhibit all slave states.
- Programmable data setup/hold times.

2.5.4. 16/32-bit CRC (CRC0)

The CRC module is designed to provide hardware calculations for flash memory verification and communications protocols. The CRC module supports the standard CCITT-16 16-bit polynomial (0x1021), and includes the following features:

- Support for four CCITT-16 polynomial.

7. SOIC-16 Package Specifications

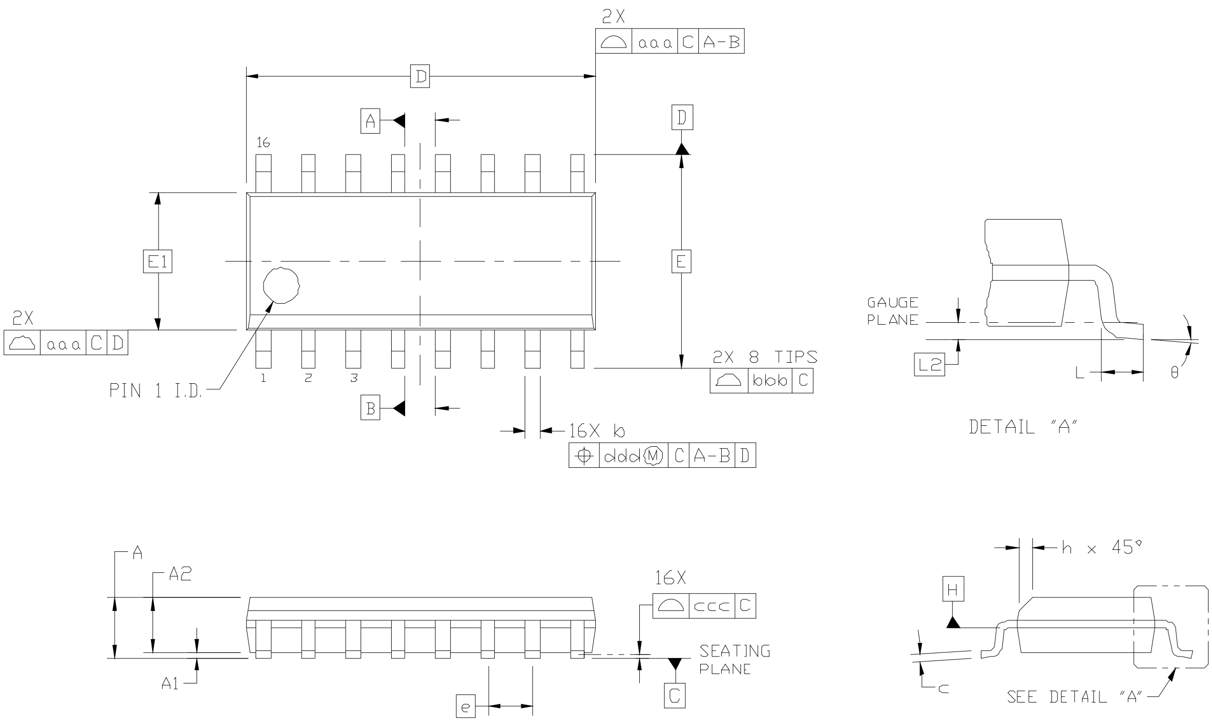

Figure 7.1. SOIC-16 Package Drawing

Table 7.1. SOIC-16 Package Dimensions

Dimension	Min	Nom	Max	Dimension	Min	Nom	Max
A	—		1.75	L	0.40		1.27
A1	0.10		0.25	L2	0.25 BSC		
A2	1.25		—	h	0.25		0.50
b	0.31		0.51	θ	0º		8º
c	0.17		0.25	aaa	0.10		
D	9.90 BSC			bbb	0.20		
E	6.00 BSC			ccc	0.10		
E1	3.90 BSC			ddd	0.25		
e	1.27 BSC						

Notes:
1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This drawing conforms to the JEDEC Solid State Outline MS-012, Variation AC.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

Register 12.3. EIE1: Extended Interrupt Enable 1

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ET3	ECP1	ECP0	EPCA0	EADC0	EWADC0	EMAT	ESMB0
Type	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xE6								

Table 12.4. EIE1 Register Bit Descriptions

Bit	Name	Function
7	ET3	**Enable Timer 3 Interrupt.** This bit sets the masking of the Timer 3 interrupt. 0: Disable Timer 3 interrupts. 1: Enable interrupt requests generated by the TF3L or TF3H flags.
6	ECP1	**Enable Comparator1 (CP1) Interrupt.** This bit sets the masking of the CP1 interrupt. 0: Disable CP1 interrupts. 1: Enable interrupt requests generated by the comparator 1 CPRIF or CPFIF flags.
5	ECP0	**Enable Comparator0 (CP0) Interrupt.** This bit sets the masking of the CP0 interrupt. 0: Disable CP0 interrupts. 1: Enable interrupt requests generated by the comparator 0 CPRIF or CPFIF flags.
4	EPCA0	**Enable Programmable Counter Array (PCA0) Interrupt.** This bit sets the masking of the PCA0 interrupts. 0: Disable all PCA0 interrupts. 1: Enable interrupt requests generated by PCA0.
3	EADC0	**Enable ADC0 Conversion Complete Interrupt.** This bit sets the masking of the ADC0 Conversion Complete interrupt. 0: Disable ADC0 Conversion Complete interrupt. 1: Enable interrupt requests generated by the ADINT flag.
2	EWADC0	**Enable Window Comparison ADC0 Interrupt.** This bit sets the masking of ADC0 Window Comparison interrupt. 0: Disable ADC0 Window Comparison interrupt. 1: Enable interrupt requests generated by ADC0 Window Compare flag (ADWINT).
1	EMAT	**Enable Port Match Interrupts.** This bit sets the masking of the Port Match Event interrupt. 0: Disable all Port Match interrupts. 1: Enable interrupt requests generated by a Port Match.

Register 14.2. ADC0CN1: ADC0 Control 1

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							ADCMBE
Type	R							RW
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xB2								

Table 14.5. ADC0CN1 Register Bit Descriptions

Bit	Name	Function
7:1	Reserved	Must write reset value.
0	ADCMBE	**Common Mode Buffer Enable.** 0: Disable the common mode buffer. This setting should be used only if the tracking time of the signal is greater than 1.5 us. 1: Enable the common mode buffer. This setting should be used in most cases, and will give the best dynamic ADC performance. The common mode buffer must be enabled if signal tracking time is less than or equal to 1.5 us.

Register 14.5. ADC0PWR: ADC0 Power Control

Bit	7	6	5	4	3	2	1	0
Name	ADBIAS		ADMXLP	ADLPM	ADPWR			
Type	RW		RW	RW	RW			
Reset	0	0	0	0	1	1	1	1
SFR Address: 0xDF								

Table 14.8. ADC0PWR Register Bit Descriptions

Bit	Name	Function
7:6	ADBIAS	**Bias Power Select.** This field can be used to adjust the ADC's power consumption based on the conversion speed. Higher bias currents allow for faster conversion times. 00: Select bias current mode 0. Recommended to use modes 1, 2, or 3. 01: Select bias current mode 1 (SARCLK <= 16 MHz). 10: Select bias current mode 2. 11: Select bias current mode 3 (SARCLK <= 4 MHz).
5	ADMXLP	**Mux and Reference Low Power Mode Enable.** Enables low power mode operation for the multiplexer and voltage reference buffers. 0: Low power mode disabled. 1: Low power mode enabled (SAR clock < 4 MHz).
4	ADLPM	**Low Power Mode Enable.** This bit can be used to reduce power to the ADC's internal common mode buffer. It can be set to 1 to reduce power when tracking times in the application are longer (slower sample rates). 0: Disable low power mode. 1: Enable low power mode (requires extended tracking time).
3:0	ADPWR	**Burst Mode Power Up Time.** This field sets the time delay allowed for the ADC to power up from a low power state. When ADTM is set, an additional 4 SARCLKs are added to this time. $T_{PWRTIME} = \frac{8 \times ADPWR}{F_{HFOSC}}$

Table 15.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
XRL A, Rn	Exclusive-OR Register to A	1	1
XRL A, direct	Exclusive-OR direct byte to A	2	2
XRL A, @Ri	Exclusive-OR indirect RAM to A	1	2
XRL A, #data	Exclusive-OR immediate to A	2	2
XRL direct, A	Exclusive-OR A to direct byte	2	2
XRL direct, #data	Exclusive-OR immediate to direct byte	3	3
CLR A	Clear A	1	1
CPL A	Complement A	1	1
RL A	Rotate A left	1	1
RLC A	Rotate A left through Carry	1	1
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 byte	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 DPTR 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

Register 15.3. SP: Stack Pointer

Bit	7	6	5	4	3	2	1	0
Name	SP							
Type	RW							
Reset	0	0	0	0	0	1	1	1
SFR Address: 0x81								

Table 15.4. SP Register Bit Descriptions

Bit	Name	Function
7:0	SP	**Stack Pointer.** The Stack Pointer holds the location of the top of the stack. The stack pointer is incremented before every PUSH operation. The SP register defaults to 0x07 after reset.

Register 17.4. CPT1CN: Comparator 1 Control

Bit	7	6	5	4	3	2	1	0
Name	CPEN	CPOUT	CPRIF	CPFIF	CPHYP		CPHYN	
Type	RW	R	RW	RW	RW		RW	
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xBF								

Table 17.8. CPT1CN Register Bit Descriptions

Bit	Name	Function
7	CPEN	**Comparator 1 Enable Bit.** 0: Comparator Disabled. 1: Comparator Enabled.
6	CPOUT	**Comparator 1 Output State Flag.** 0: Voltage on CP1P < CP1N. 1: Voltage on CP1P > CP1N.
5	CPRIF	**Comparator 1 Rising-Edge Flag. Must be cleared by software.** 0: No Comparator Rising Edge has occurred since this flag was last cleared. 1: Comparator Rising Edge has occurred.
4	CPFIF	**Comparator 1 Falling-Edge Flag. Must be cleared by software.** 0: No Comparator Falling Edge has occurred since this flag was last cleared. 1: Comparator Falling Edge has occurred.
3:2	CPHYP	**Comparator 1 Positive Hysteresis Control Bits.** 00: Positive Hysteresis Disabled. 01: Positive Hysteresis = 5 mV. 10: Positive Hysteresis = 10 mV. 11: Positive Hysteresis = 20 mV.
1:0	CPHYN	**Comparator 1 Negative Hysteresis Control Bits.** 00: Negative Hysteresis Disabled. 01: Negative Hysteresis = 5 mV. 10: Negative Hysteresis = 10 mV. 11: Negative Hysteresis = 20 mV.

Register 20.6. PCA0L: PCA Counter/Timer Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0L							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xF9								

Table 20.8. PCA0L Register Bit Descriptions

Bit	Name	Function
7:0	PCA0L	**PCA Counter/Timer Low Byte.** The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.

21.3. Priority Crossbar Decoder

The priority crossbar decoder assigns a priority to each I/O function, starting at the top with UART0. When a digital resource is selected, the least-significant unassigned port pin is assigned to that resource (excluding UART0, which is always at pins P0.4 and P0.5). If a port pin is assigned, the crossbar skips that pin when assigning the next selected resource. Additionally, the crossbar will skip port pins whose associated bits in the PnSKIP registers are set. The PnSKIP registers allow software to skip port pins that are to be used for analog input, dedicated functions, or GPIO.

Important Note on Crossbar Configuration: If a port pin is claimed by a peripheral without use of the crossbar, its corresponding PnSKIP bit should be set. This applies to P0.0 if VREF is used, P0.1 if AGND is used, P0.3 if the EXTCLK input is enabled, P0.6 if the ADC is configured to use the external conversion start signal (CNVSTR), and any selected ADC or comparator inputs. The crossbar skips selected pins as if they were already assigned, and moves to the next unassigned pin.

Figure 21.2 shows all of the potential peripheral-to-pin assignments available to the crossbar. Note that this does not mean any peripheral can always be assigned to the highlighted pins. The actual pin assignments are determined by the priority of the enabled peripherals.

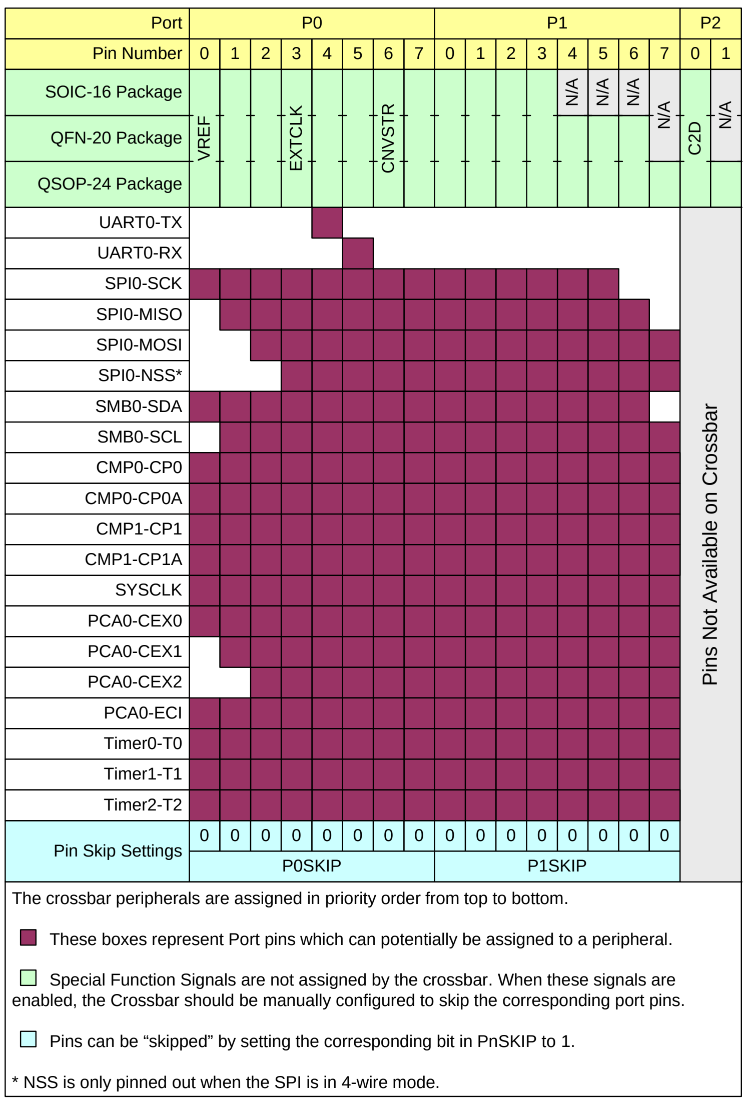

Figure 21.2. Crossbar Priority Decoder - Possible Pin Assignments

Register 21.3. XBR2: Port I/O Crossbar 2

Bit	7	6	5	4	3	2	1	0
Name	WEAKPUD	XBARE	Reserved					
Type	RW	RW	R					
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xE3								

Table 21.6. XBR2 Register Bit Descriptions

Bit	Name	Function
7	WEAKPUD	**Port I/O Weak Pullup Disable.** 0: Weak Pullups enabled (except for Ports whose I/O are configured for analog mode). 1: Weak Pullups disabled.
6	XBARE	**Crossbar Enable.** 0: Crossbar disabled. 1: Crossbar enabled.
5:0	Reserved	Must write reset value.

Register 21.18. P2MDOUT: Port 2 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	Reserved						P2MDOUT	
Type	R						RW	
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xA6								

Table 21.21. P2MDOUT Register Bit Descriptions

Bit	Name	Function
7:2	Reserved	Must write reset value.
1:0	P2MDOUT	**Port 2 Output Mode.** 0: Corresponding P2.n Output is open-drain. 1: Corresponding P2.n Output is push-pull.
Note: Port 2 consists of 2 bits (P2.0-P2.1) on QSOP24 devices and 1 bit (P2.0) on QFN20 and SOIC16 packages.		

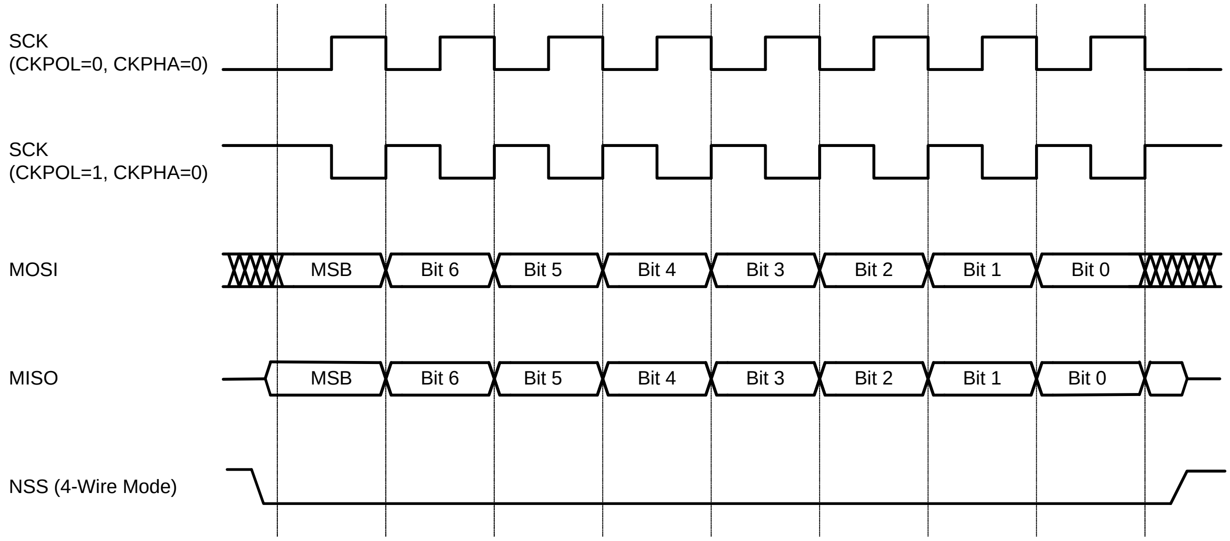

Figure 23.6. Slave Mode Data/Clock Timing (CKPHA = 0)

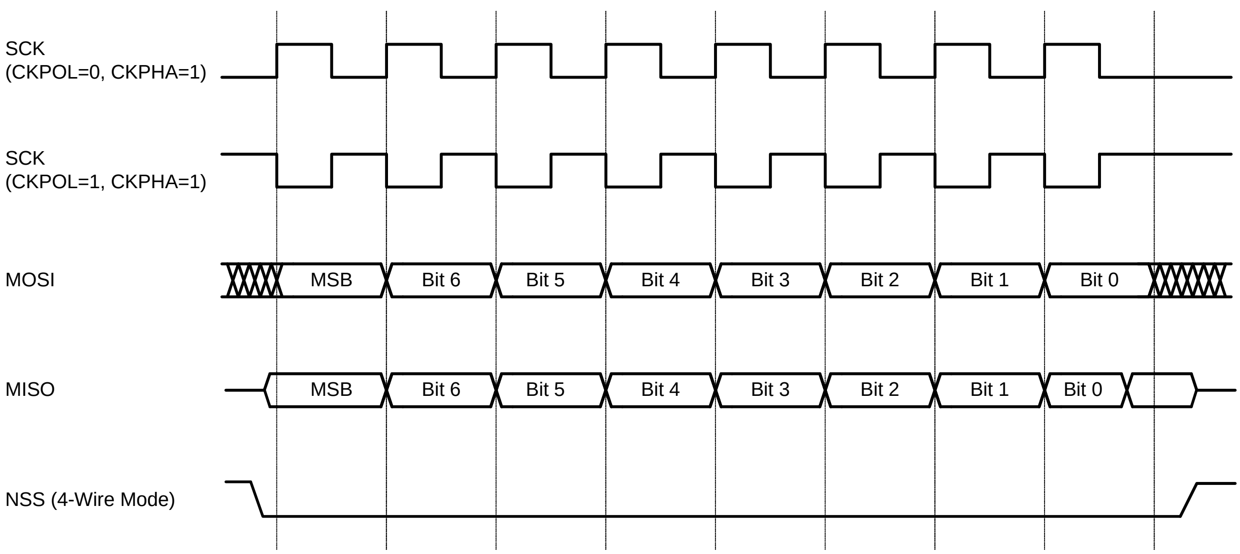

Figure 23.7. Slave Mode Data/Clock Timing (CKPHA = 1)

23.6. SPI Special Function Registers

SPI0 is accessed and controlled through four special function registers in the system controller: SPI0CN Control Register, SPI0DAT Data Register, SPI0CFG Configuration Register, and SPI0CKR Clock Rate Register. The four special function registers related to the operation of the SPI0 Bus are described in the following figures.

For the SMBus0 interface, Timer 3 is used to implement SCL low timeouts. The SCL low timeout feature is enabled by setting the SMB0TOE bit in SMB0CF. The associated timer is forced to reload when SCL is high, and allowed to count when SCL is low. With the associated timer enabled and configured to overflow after 25 ms (and SMB0TOE set), the timer interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

24.3.5. SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMB0FTE bit in SMB0CF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods (as defined by the timer configured for the SMBus clock source). If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. A clock source is required for free timeout detection, even in a slave-only implementation.

24.4. Using the SMBus

The SMBus can operate in both Master and Slave modes. The interface provides timing and shifting control for serial transfers; higher level protocol is determined by user software. The SMBus interface provides the following application-independent features:

- Byte-wise serial data transfers
- Clock signal generation on SCL (Master Mode only) and SDA data synchronization
- Timeout/bus error recognition, as defined by the SMB0CF configuration register
- START/STOP timing, detection, and generation
- Bus arbitration
- Interrupt generation
- Status information
- Optional hardware recognition of slave address and automatic acknowledgement of address/data

SMBus interrupts are generated for each data byte or slave address that is transferred. When hardware acknowledgement is disabled, the point at which the interrupt is generated depends on whether the hardware is acting as a data transmitter or receiver. When a transmitter (i.e., sending address/data, receiving an ACK), this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data (i.e., receiving address/data, sending an ACK), this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. If hardware acknowledgement is enabled, these interrupts are always generated after the ACK cycle. See Section 24.5 for more details on transmission sequences.

Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN (SMBus Control register) to find the cause of the SMBus interrupt. Table 24.5 provides a quick SMB0CN decoding reference.

24.4.1. SMBus Configuration Register

The SMBus Configuration register (SMB0CF) is used to enable the SMBus Master and/or Slave modes, select the SMBus clock source, and select the SMBus timing and timeout options. When the ENSMB bit is set, the SMBus is enabled for all master and slave events. Slave events may be disabled by setting the INH bit. With slave events inhibited, the SMBus interface will still monitor the SCL and SDA pins; however, the interface will NACK all received addresses and will not generate any slave interrupts. When the INH bit is set, all slave events will be inhibited following the next START (interrupts will continue for the duration of the current transfer).

25.1. Timer 0 and Timer 1

Timer 0 and Timer 1 are each implemented as a16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register. Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register. Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently for the operating modes described below.

25.1.1. Mode 0: 13-bit Counter/Timer

Timer 0 and Timer 1 operate as 13-bit counter/timers in Mode 0. The following describes the configuration and operation of Timer 0. However, both timers operate identically, and Timer 1 is configured in the same manner as described for Timer 0.

The TH0 register holds the eight MSBs of the 13-bit counter/timer. TL0 holds the five LSBs in bit positions TL0.4–TL0.0. The three upper bits of TL0 (TL0.7–TL0.5) are indeterminate and should be masked out or ignored when reading. As the 13-bit timer register increments and overflows from 0x1FFF (all ones) to 0x0000, the timer overflow flag TF0 in TCON is set and an interrupt will occur if Timer 0 interrupts are enabled.

The CT0 bit in the TMOD register selects the counter/timer's clock source. When CT0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register. Clearing CT selects the clock defined by the T0M bit in register CKCON. When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON.

Setting the TR0 bit enables the timer when either GATE0 in the TMOD register is logic 0 or the input signal INT0 is active as defined by bit IN0PL in register IT01CF. Setting GATE0 to 1 allows the timer to be controlled by the external input signal INT0, facilitating pulse width measurements.

TR0	GATE0	$\overline{\text{INT0}}$	Counter/Timer
0	X	X	Disabled
1	0	X	Enabled
1	1	0	Disabled
1	1	1	Enabled
Note: X = Don't Care			

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal INT1 is used with Timer 1; the /INT1 polarity is defined by bit IN1PL in register IT01CF.

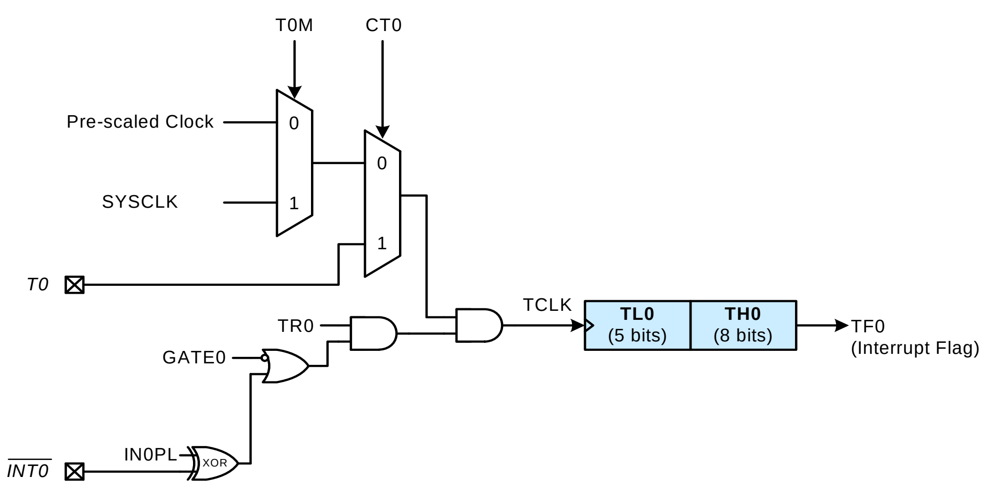

Figure 25.1. T0 Mode 0 Block Diagram

25.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload

Mode 2 configures Timer 0 and Timer 1 to operate as 8-bit counter/timers with automatic reload of the start value. TL0 holds the count and TH0 holds the reload value. When the counter in TL0 overflows from all ones to 0x00, the timer overflow flag TF0 in the TCON register is set and the counter in TL0 is reloaded from TH0. If Timer 0 interrupts are enabled, an interrupt will occur when the TF0 flag is set. The reload value in TH0 is not changed. TL0 must be initialized to the desired value before enabling the timer for the first count to be correct. When in Mode 2, Timer 1 operates identically to Timer 0.

Both counter/timers are enabled and configured in Mode 2 in the same manner as Mode 0. Setting the TR0 bit enables the timer when either GATE0 in the TMOD register is logic 0 or when the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF.

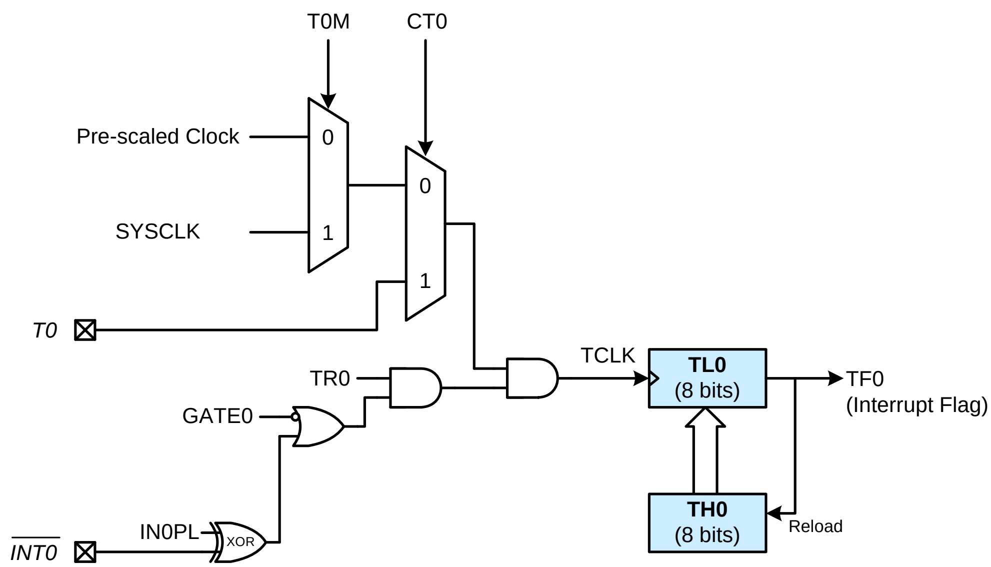

Figure 25.2. T0 Mode 2 Block Diagram

Register 25.5. TL1: Timer 1 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL1							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Address: 0x8B								

Table 25.7. TL1 Register Bit Descriptions

Bit	Name	Function
7:0	TL1	**Timer 1 Low Byte.** The TL1 register is the low byte of the 16-bit Timer 1.

Register 25.12. TMR2H: Timer 2 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2H							
Type	RW							
Reset	0	0	0	0	0	0	0	0
SFR Address: 0xCD								

Table 25.14. TMR2H Register Bit Descriptions

Bit	Name	Function
7:0	TMR2H	**Timer 2 High Byte.** In 16-bit mode, the TMR2H register contains the high byte of the 16-bit Timer 2. In 8-bit mode, TMR2H contains the 8-bit high byte timer value.

27. Watchdog Timer (WDT0)

The C8051F85x/86x family includes a programmable Watchdog Timer (WDT) running off the low-frequency oscillator. A WDT overflow will force the MCU into the reset state. To prevent the reset, the WDT must be restarted by application software before overflow. If the system experiences a software or hardware malfunction preventing the software from restarting the WDT, the WDT will overflow and cause a reset.

Following a reset the WDT is automatically enabled and running with the default maximum time interval. 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 $\overline{RST}$ pin is unaffected by this reset.

The WDT consists of an internal timer running from the low-frequency oscillator. The timer measures the period between specific writes to its control register. If this period exceeds the programmed limit, a WDT reset is generated. The WDT can be enabled and disabled as needed in software, or can be permanently enabled if desired. When the WDT is active, the low-frequency oscillator is forced on. All watchdog features are controlled via the Watchdog Timer Control Register (WDTCN).

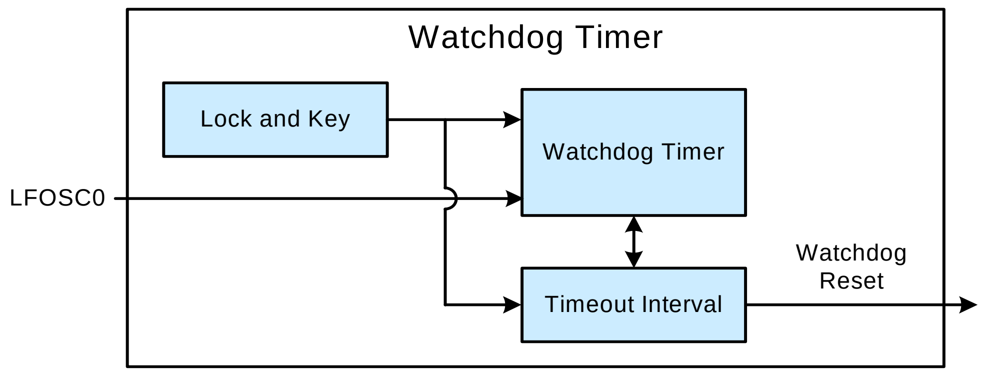

Figure 27.1. Watchdog Timer Block Diagram