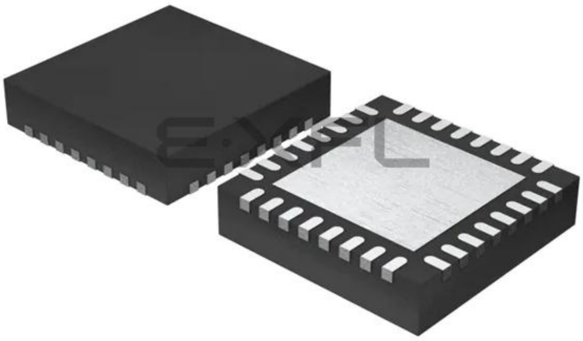
Silicon Labs - C8051T627-B-GM Datasheet



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	48 MIPS
Connectivity	I²C, SPI, UART/USART, USB
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	24
Program Memory Size	32KB (32K x 8)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	3.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.25V
Data Converters	A/D 21x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051t627-b-gm

Table 7.6. EPROM Electrical Characteristics

Parameters	Test Condition	Min	Typ	Max	Unit
EPROM Size (Note 1)	C8051T620/1 & C8051T320/1/2/3	16384	—	—	bytes
	C8051T626	65535	—	—	bytes
	C8051T627	32768	—	—	bytes
Write Cycle Time (per Byte) (Note 2)		105	155	205	µs
In-Application Programming Write Cycle Time (per Byte) (Note 3)	Capacitor on V_{PP} = 4.7 µF and fully discharged	—	37	—	ms
	Capacitor on V_{PP} = 4.7 µF and initially charged to 3.3 V	—	26	—	ms
Programming Voltage (V_{PP})		5.75	6.0	6.25	V
Capacitor on V_{PP} for In-application Programming		—	4.7	—	µF

Notes:

1. 512 bytes at location 0x3E00 to 0x3FFF are not available for program storage on the 16k devices, and 512 bytes at location 0xFE00 to 0xFFFF are not available for program storage on the C8051T626.
2. For devices with a Date Code prior to 1040, the programming time over the C2 interface is twice as long. See Section 18.1.1 for more information.
3. Duration of write time is largely dependent on VIO voltage, supply voltage, and residual charge on the VPP capacitor. The majority of the write time consists of charging the voltage on VPP to 6.0 V. These measurements include the VPP ramp time and VDD = VIO = 3.3 V

Table 7.7. Internal High-Frequency Oscillator Electrical Characteristics

V_{DD} = 2.7 to 3.6 V; T_A = –40 to +85 °C unless otherwise specified; Using factory-calibrated settings.

Parameters	Test Condition	Min	Typ	Max	Unit
Oscillator Frequency	IFCN = 11b	47.28	48	48.72	MHz
Oscillator Supply Current (from V_{DD})	25 °C, V_{DD} = 3.0 V, OSCICN.7 = 1, OSCICN.5 = 0				
	C8051T626/7/T320/1/2/3	—	900	1000	µA
	C8051T626/7	—	925	1100	µA
Power Supply Sensitivity	Constant Temperature	—	±0.02	—	%/V
Temperature Sensitivity	Constant Supply	—	±20	—	ppm/°C

Note: Represents mean ±1 standard deviation.

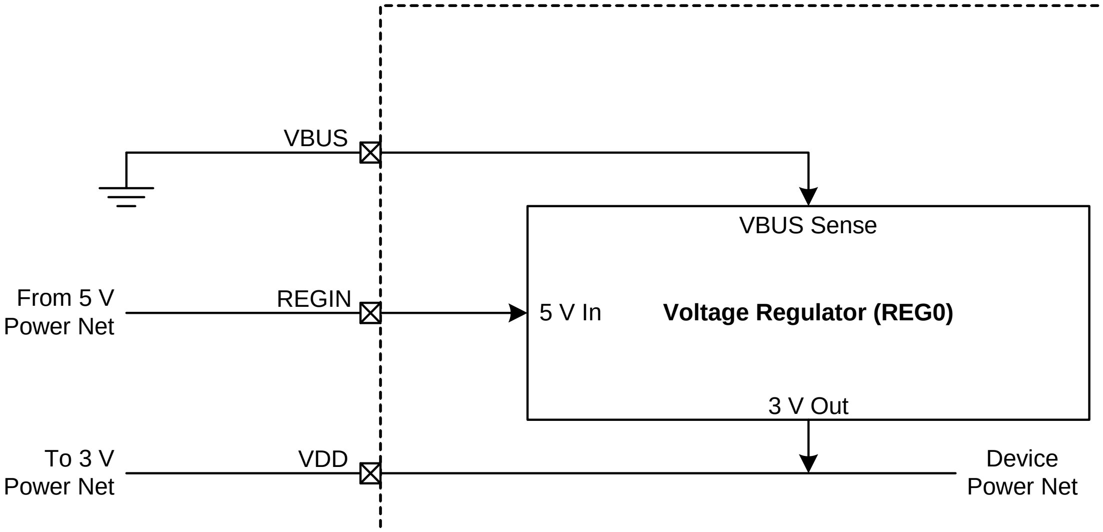

VBUS
VBUS Sense
From 5 V
Power Net
REGIN
5 V In
Voltage Regulator (REG0)
3 V Out
To 3 V
Power Net
VDD
Device
Power Net

Figure 11.4. REG0 Configuration: No USB Connection

12. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 30), and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control-system solution in a single integrated circuit.

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability (see Figure 12.1 for a block diagram). The CIP-51 includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 48 MIPS Peak Throughput with 48 MHz Clock
- 0 to 48 MHz Clock Frequency
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

Performance

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. In a standard 8051, all instructions except for MUL and DIV take 12 or 24 system clock cycles to execute, and usually have a maximum system clock of 12 MHz. By contrast, the CIP-51 core executes 70% of its instructions in one or two system clock cycles, with no instructions taking more than eight system clock cycles.

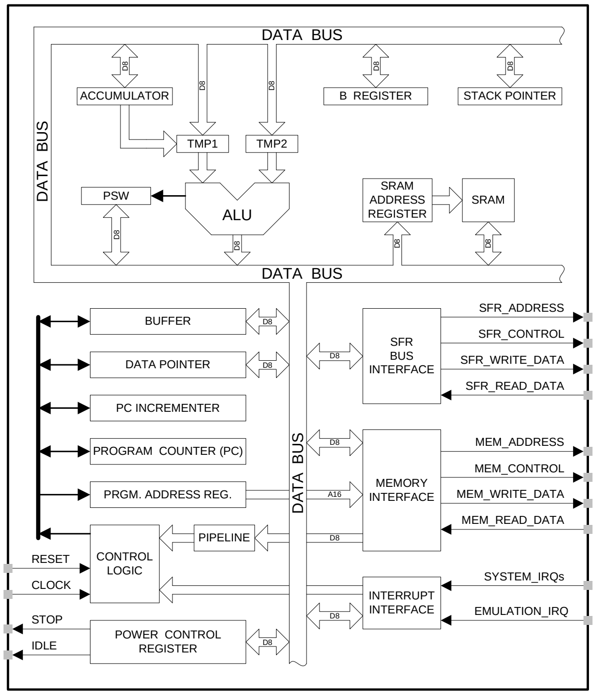

DATA BUS
D8
ACCUMULATOR
B REGISTER
STACK POINTER
TMP1
TMP2
PSW
ALU
SRAM ADDRESS REGISTER
SRAM
DATA BUS
BUFFER
DATA POINTER
PC INCREMENTER
PROGRAM COUNTER (PC)
PRGM. ADDRESS REG.
PIPELINE
CONTROL LOGIC
RESET
CLOCK
STOP
IDLE
POWER CONTROL REGISTER
A16
SFR BUS INTERFACE
SFR_ADDRESS
SFR_CONTROL
SFR_WRITE_DATA
SFR_READ_DATA
MEMORY INTERFACE
MEM_ADDRESS
MEM_CONTROL
MEM_WRITE_DATA
MEM_READ_DATA
INTERRUPT INTERFACE
SYSTEM_IRQs
EMULATION_IRQ

Figure 12.1. CIP-51 Block Diagram

Table 16.2. Special Function Registers (Continued)

SFRs are listed in alphabetical order. All undefined SFR locations are reserved

Register	Address	Description	Page
PCA0CPH0	0xFC	PCA Capture 0 High	287
PCA0CPH1	0xEA	PCA Capture 1 High	287
PCA0CPH2	0xEC	PCA Capture 2 High	287
PCA0CPH3	0xEE	PCA Capture 3 High	287
PCA0CPH4	0xFE	PCA Capture 4 High	287
PCA0CPL0	0xFB	PCA Capture 0 Low	287
PCA0CPL1	0xE9	PCA Capture 1 Low	287
PCA0CPL2	0xEB	PCA Capture 2 Low	287
PCA0CPL3	0xED	PCA Capture 3 Low	287
PCA0CPL4	0xFD	PCA Capture 4 Low	287
PCA0CPM0	0xDA	PCA Module 0 Mode Register	285
PCA0CPM1	0xDB	PCA Module 1 Mode Register	285
PCA0CPM2	0xDC	PCA Module 2 Mode Register	285
PCA0CPM3	0xDD	PCA Module 3 Mode Register	285
PCA0CPM4	0xDE	PCA Module 4 Mode Register	285
PCA0H	0xFA	PCA Counter High	286
PCA0L	0xF9	PCA Counter Low	286
PCA0MD	0xD9	PCA Mode	283
PCA0PWM	0xF4	PCA PWM Configuration	284
PCON	0x87	Power Control	120
PFE0CN	0xAF	Prefetch Engine Control	76
PSCTL	0x8F	Program Store R/W Control	116
PSW	0xD0	Program Status Word	75
REF0CN	0xD1	Voltage Reference Control	60
REG01CN	0xC9	Voltage Regulator Control	65
RSTSRC	0xEF	Reset Source Configuration/Status	126
SBCON1	0xAC	UART1 Baud Rate Generator Control	231
SBRLH1	0xB5	UART1 Baud Rate Generator High Byte	231

17.1. MCU Interrupt Sources and Vectors

The C8051T620/1/6/7 & C8051T320/1/2/3 MCUs support 18 interrupt sources. Software can simulate an interrupt by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag. MCU interrupt sources, associated vector addresses, priority order and control bits are summarized in Table 17.1. Refer to the datasheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

17.1.1. Interrupt Priorities

Each interrupt source can be individually programmed to one of two priority levels: low or high. A low priority interrupt service routine can be preempted by a high priority interrupt. A high priority interrupt cannot be preempted. Each interrupt has an associated interrupt priority bit in an SFR (IP, EIP1, or EIP2) used to configure its priority level. Low priority is the default. If two interrupts are recognized simultaneously, the interrupt with the higher priority is serviced first. If both interrupts have the same priority level, a fixed priority order is used to arbitrate, given in Table 17.1.

17.1.2. Interrupt Latency

Interrupt response time depends on the state of the CPU when the interrupt occurs. Pending interrupts are sampled and priority decoded each system clock cycle. Therefore, the fastest possible response time is 6 system clock cycles: 1 clock cycle to detect the interrupt and 5 clock cycles to complete the LCALL to the ISR. If an interrupt is pending when a RETI is executed, a single instruction is executed before an LCALL is made to service the pending interrupt. Therefore, the maximum response time for an interrupt (when no other interrupt is currently being serviced or the new interrupt is of greater priority) occurs when the CPU is performing an RETI instruction followed by a DIV as the next instruction. In this case, the response time is 20 system clock cycles: 1 clock cycle to detect the interrupt, 6 clock cycles to execute the RETI, 8 clock cycles to complete the DIV instruction and 5 clock cycles to execute the LCALL to the ISR. If the CPU is executing an ISR for an interrupt with equal or higher priority, the new interrupt will not be serviced until the current ISR completes, including the RETI and following instruction.

Note that the CPU is stalled during EPROM write operations and USB FIFO MOVX accesses (see Section 15.2.3. Accessing USB FIFO Space on page 91). Interrupt service latency will be increased for interrupts occurring while the CPU is stalled. The latency for these situations will be determined by the standard interrupt service procedure (as described above) and the amount of time the CPU is stalled.

17.2. Interrupt Register Descriptions

The SFRs used to enable the interrupt sources and set their priority level are described in this section. Refer to the data sheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

Table 17.1. Interrupt Summary

Interrupt Source	Interrupt Vector	Priority Order	Pending Flag	Bit addressable?	Cleared by HW?	Enable Flag	Priority Control
Reset	0x0000	Top	None	N/A	N/A	Always Enabled	Always Highest
External Interrupt 0 ($\overline{\text{INT0}}$)	0x0003	0	IE0 (TCON.1)	Y	Y	EX0 (IE.0)	PX0 (IP.0)
Timer 0 Overflow	0x000B	1	TF0 (TCON.5)	Y	Y	ET0 (IE.1)	PT0 (IP.1)
External Interrupt 1 ($\overline{\text{INT1}}$)	0x0013	2	IE1 (TCON.3)	Y	Y	EX1 (IE.2)	PX1 (IP.2)
Timer 1 Overflow	0x001B	3	TF1 (TCON.7)	Y	Y	ET1 (IE.3)	PT1 (IP.3)
UART0	0x0023	4	RI0 (SCON0.0) TI0 (SCON0.1)	Y	N	ES0 (IE.4)	PS0 (IP.4)
Timer 2 Overflow	0x002B	5	TF2H (TMR2CN.7) TF2L (TMR2CN.6)	Y	N	ET2 (IE.5)	PT2 (IP.5)
SPI0	0x0033	6	SPIF (SPI0CN.7) WCOL (SPI0CN.6) MODF (SPI0CN.5) RXOVRN (SPI0CN.4)	Y	N	ESPI0 (IE.6)	PSPI0 (IP.6)
SMB0	0x003B	7	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)	PSMB0 (EIP1.0)
USB0	0x0043	8	Special	N	N	EUSB0 (EIE1.0)	PUSB0 (EIP1.1)
ADC0 Window Compare	0x004B	9	AD0WINT (ADC0CN.3)	Y	N	EWADC0 (EIE1.2)	PWADC0 (EIP1.2)
ADC0 Conversion Complete	0x0053	10	AD0INT (ADC0CN.5)	Y	N	EADC0 (EIE1.3)	PADC0 (EIP1.3)
Programmable Counter Array	0x005B	11	CF (PCA0CN.7) CCFn (PCA0CN.n) COVF (PCA0PWM.6)	Y	N	EPCA0 (EIE1.4)	PPCA0 (EIP1.4)
Comparator0	0x0063	12	CP0FIF (CPT0CN.4) CP0RIF (CPT0CN.5)	N	N	ECP0 (EIE1.5)	PCP0 (EIP1.5)
Comparator1	0x006B	13	CP1FIF (CPT1CN.4) CP1RIF (CPT1CN.5)	N	N	ECP1 (EIE1.6)	PCP1 (EIP1.6)
Timer 3 Overflow	0x0073	14	TF3H (TMR3CN.7) TF3L (TMR3CN.6)	N	N	ET3 (EIE1.7)	PT3 (EIP1.7)
VBUS Level	0x007B	15	N/A	N/A	N/A	EVBUS (EIE2.0)	PVBUS (EIP2.0)
UART1	0x0083	16	RI1 (SCON1.0) TI1 (SCON1.1)	N	N	ES1 (EIE2.1)	PS1 (EIP2.1)
Reserved	0x008B	17	N/A	N/A	N/A	N/A	N/A
Port Match	0x0093	18	None	N/A	N/A	EMAT (EIE2.3)	PMAT (EIP2.3)
Reserved		19	N/A	N/A	N/A	N/A	N/A

22.4. Port I/O Initialization

Port I/O initialization consists of the following steps:

1. Select the input mode (analog or digital) for all Port pins, using the Port Input Mode register (PnMDIN).
2. Select the output mode (open-drain or push-pull) for all Port pins, using the Port Output Mode register (PnMDOUT).
3. Select any pins to be skipped by the I/O Crossbar using the Port Skip registers (PnSKIP).
4. Assign Port pins to desired peripherals (XBR0, XBR1, XBR2).
5. Enable the Crossbar (XBARE = 1).

All Port pins must be configured as either analog or digital inputs. Any pins to be used as Comparator or ADC inputs should be configured as an analog inputs. When a pin is configured as an analog input, its weak pullup, digital driver, and digital receiver are disabled. This process saves power and reduces noise on the analog input. Pins configured as digital inputs may still be used by analog peripherals; however this practice is not recommended.

Additionally, all analog input pins should be configured to be skipped by the Crossbar (accomplished by setting the associated bits in PnSKIP). Port input mode is set in the PnMDIN register, where a 1 indicates a digital input, and a 0 indicates an analog input. All pins default to digital inputs on reset. See SFR Definition 22.9, SFR Definition 22.13, and SFR Definition 22.17 for the PnMDIN register details.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). Each Port Output driver can be configured as either open drain or push-pull. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is the SMBus (SDA, SCL) pins, which are configured as open-drain regardless of the PnMDOUT settings. When the WEAKPUD bit in XBR1 is 0, a weak pullup is enabled for all Port I/O configured as open-drain. WEAKPUD does not affect the push-pull Port I/O. Furthermore, the weak pullup is turned off on an output that is driving a 0 to avoid unnecessary power dissipation.

Registers XBR0, XBR1, and XBR2 must be loaded with the appropriate values to select the digital I/O functions required by the design. Setting the XBARE bit in XBR1 to 1 enables the Crossbar. Until the Crossbar is enabled, the external pins remain as standard Port I/O (in input mode), regardless of the XBRn Register settings. For given XBRn Register settings, one can determine the I/O pin-out using the Priority Decode Table; as an alternative, the Configuration Wizard utility of the Silicon Labs IDE software will determine the Port I/O pin-assignments based on the XBRn Register settings.

The Crossbar must be enabled to use Port pins as standard Port I/O in output mode. Port output drivers are disabled while the Crossbar is disabled.

SFR Definition 23.1. USB0XCN: USB0 Transceiver Control

Bit	7	6	5	4	3	2	1	0
Name	PREN	PHYEN	SPEED	PHYTST[1:0]		DFREC	Dp	Dn
Type	R/W	R/W	R/W	R/W		R	R	R
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD7

Bit	Name	Function
7	PREN	Internal Pull-up Resistor Enable. The location of the pull-up resistor (D+ or D-) is determined by the SPEED bit. 0: Internal pull-up resistor disabled (device effectively detached from USB network). 1: Internal pull-up resistor enabled when VBUS is present (device attached to the USB network).
6	PHYEN	Physical Layer Enable. 0: USB0 physical layer Transceiver disabled (suspend). 1: USB0 physical layer Transceiver enabled (normal).
5	SPEED	USB0 Speed Select. This bit selects the USB0 speed. 0: USB0 operates as a Low Speed device. If enabled, the internal pull-up resistor appears on the D– line. 1: USB0 operates as a Full Speed device. If enabled, the internal pull-up resistor appears on the D+ line.
4:3	PHYTST[1:0]	Physical Layer Test Bits. 00: Mode 0: Normal (non-test mode) (D+ = X, D- = X) 01: Mode 1: Differential 1 Forced (D+ = 1, D- = 0) 10: Mode 2: Differential 0 Forced (D+ = 0, D- = 1) 11: Mode 3: Single-Ended 0 Forced (D+ = 0, D– = 0)
2	DFREC	Differential Receiver Bit The state of this bit indicates the current differential value present on the D+ and D- lines when PHYEN = 1. 0: Differential 0 signalling on the bus. 1: Differential 1 signalling on the bus.
1	Dp	D+ Signal Status. This bit indicates the current logic level of the D+ pin. 0: D+ signal currently at logic 0. 1: D+ signal currently at logic 1.
0	Dn	D– Signal Status. This bit indicates the current logic level of the D- pin. 0: D– signal currently at logic 0. 1: D– signal currently at logic 1.

24.5.3. Write Sequence (Slave)

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. 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. If hardware ACK generation is disabled, 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 hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), 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.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.

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. Figure 24.7 shows a typical slave write sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the data byte transferred interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs before the ACK with hardware ACK generation disabled, and after the ACK when hardware ACK generation is enabled.

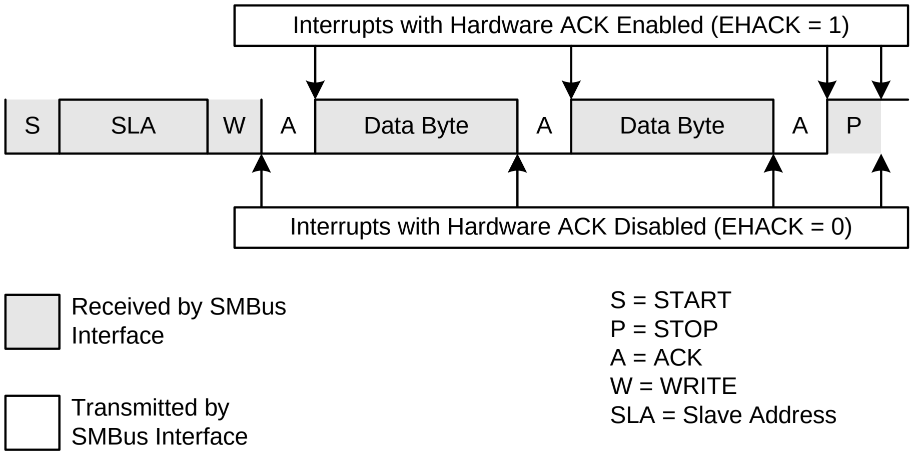

Interrupts with Hardware ACK Enabled (EHACK = 1)
S
SLA
W
A
Data Byte
A
Data Byte
A
P
Interrupts with Hardware ACK Disabled (EHACK = 0)
Received by SMBus Interface
Transmitted by SMBus Interface
S = START
P = STOP
A = ACK
W = WRITE
SLA = Slave Address

Figure 24.7. Typical Slave Write Sequence

25.2. 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 25.3.

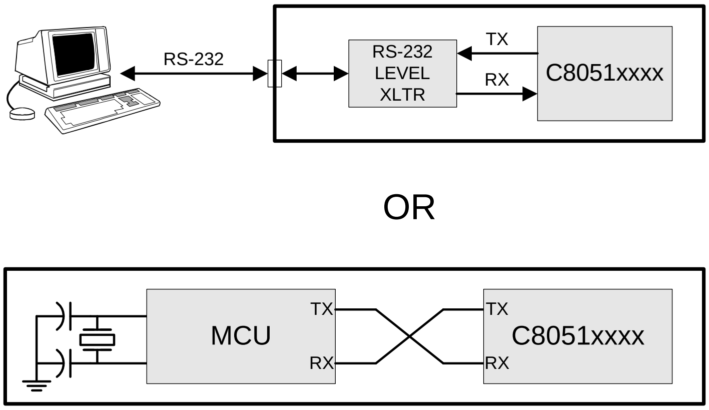


Figure 25.3. UART Interconnect Diagram

25.2.1. 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.

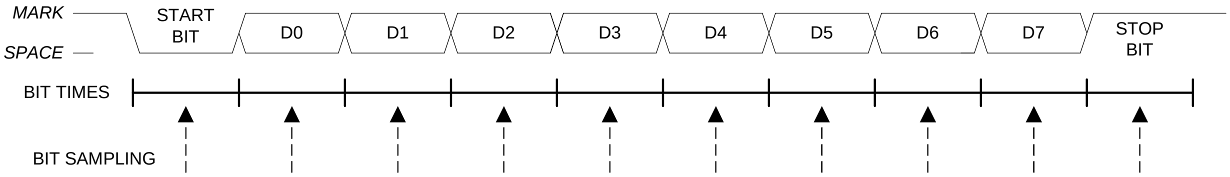


Figure 25.4. 8-Bit UART Timing Diagram

27. Enhanced Serial Peripheral Interface (SPI0)

The Enhanced Serial Peripheral Interface (SPI0) provides access to a flexible, full-duplex synchronous serial bus. SPI0 can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select SPI0 in slave mode, or to disable Master Mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a chip-select output in master mode, or disabled for 3-wire operation. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

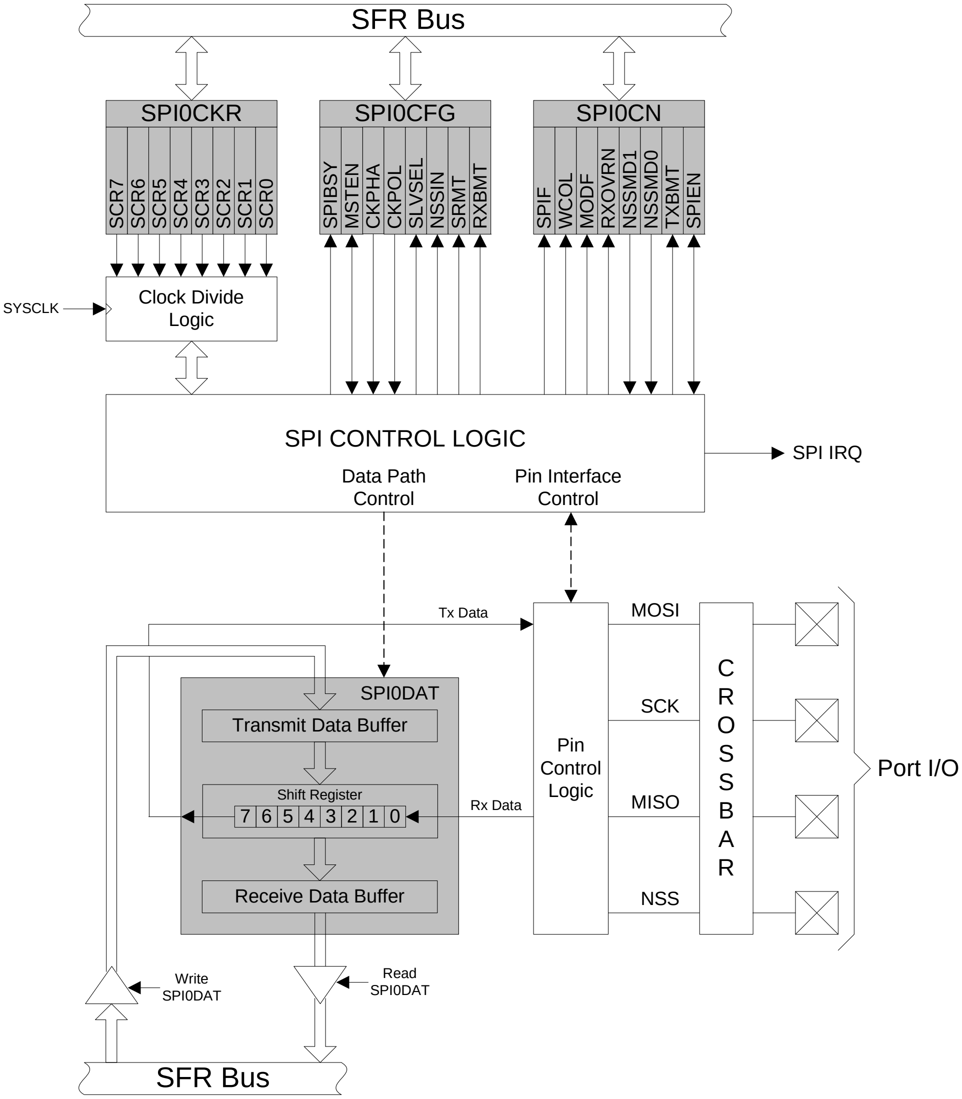

SFR Bus
SPI0CKR
SCR7
SCR6
SCR5
SCR4
SCR3
SCR2
SCR1
SCR0
SPI0CFG
SPIBSY
MSTEN
CKPHA
CKPOL
SLVSEL
NSSIN
SRMT
RXBMT
SPI0CN
SPIF
WCOL
MODF
RXOVRN
NSSMD1
NSSMD0
TXBMT
SPIEN
SYSCLK
Clock Divide Logic
SPI CONTROL LOGIC
Data Path Control
Pin Interface Control
SPI IRQ
Tx Data
SPI0DAT
Transmit Data Buffer
Shift Register
7 6 5 4 3 2 1 0
Rx Data
Receive Data Buffer
Pin Control Logic
MOSI
SCK
MISO
NSS
CROSSBAR
Port I/O
Write SPI0DAT
Read SPI0DAT
SFR Bus

Figure 27.1. SPI Block Diagram

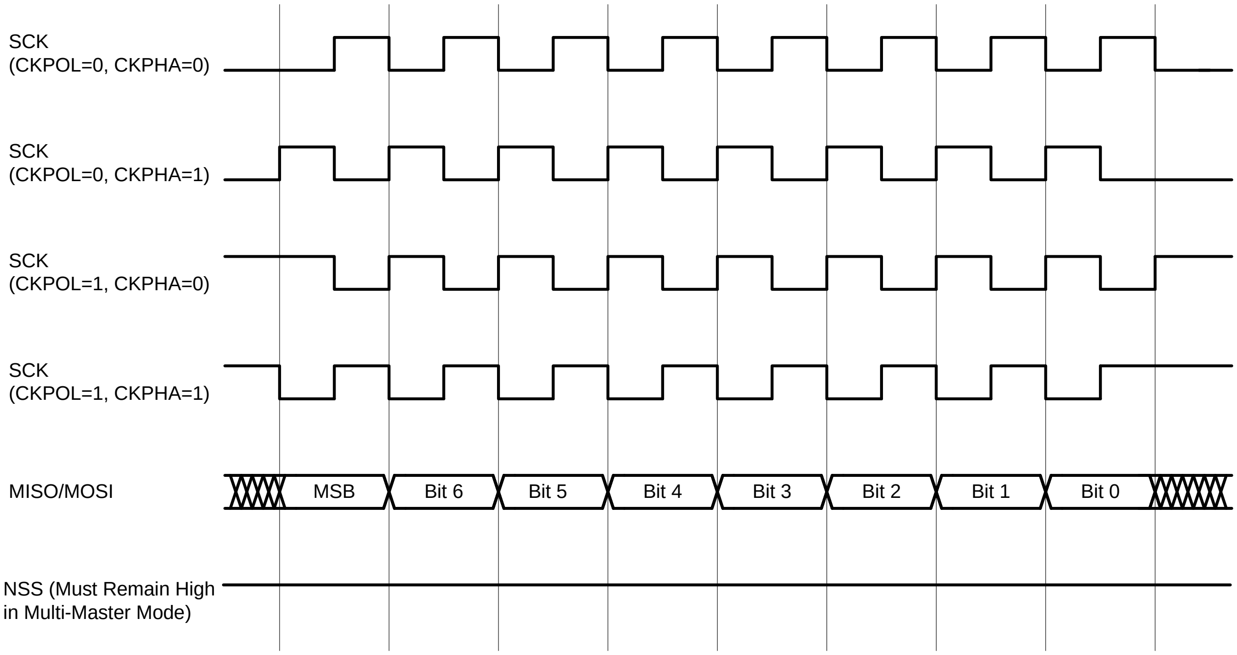

SCK
(CKPOL=0, CKPHA=0)
SCK
(CKPOL=0, CKPHA=1)
SCK
(CKPOL=1, CKPHA=0)
SCK
(CKPOL=1, CKPHA=1)
MISO/MOSI
MSB
Bit 6
Bit 5
Bit 4
Bit 3
Bit 2
Bit 1
Bit 0
NSS (Must Remain High
in Multi-Master Mode)

Figure 27.5. Master Mode Data/Clock Timing

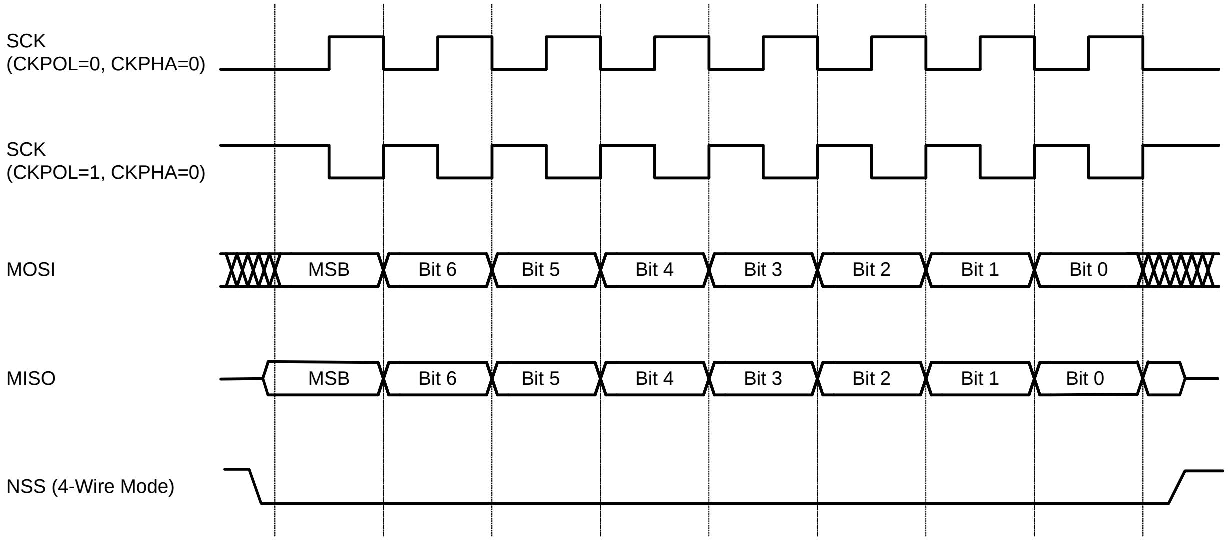

SCK
(CKPOL=0, CKPHA=0)
SCK
(CKPOL=1, CKPHA=0)
MOSI
MSB
Bit 6
Bit 5
Bit 4
Bit 3
Bit 2
Bit 1
Bit 0
MISO
MSB
Bit 6
Bit 5
Bit 4
Bit 3
Bit 2
Bit 1
Bit 0
NSS (4-Wire Mode)

Figure 27.6. Slave Mode Data/Clock Timing (CKPHA = 0)

Table 27.1. SPI Slave Timing Parameters

Parameter	Description	Min	Max	Units
Master Mode Timing (See Figure 27.8 and Figure 27.9)				
T_{MCKH}	SCK High Time	1 x T_{SYSCLK}		ns
T_{MCKL}	SCK Low Time	1 x T_{SYSCLK}		ns
T_{MIS}	MISO Valid to SCK Shift Edge	1 x T_{SYSCLK} + 20		ns
T_{MIH}	SCK Shift Edge to MISO Change	0		ns
Slave Mode Timing (See Figure 27.10 and Figure 27.11)				
T_{SE}	NSS Falling to First SCK Edge	2 x T_{SYSCLK}		ns
T_{SD}	Last SCK Edge to NSS Rising	2 x T_{SYSCLK}		ns
T_{SEZ}	NSS Falling to MISO Valid		4 x T_{SYSCLK}	ns
T_{SDZ}	NSS Rising to MISO High-Z		4 x T_{SYSCLK}	ns
T_{CKH}	SCK High Time	5 x T_{SYSCLK}		ns
T_{CKL}	SCK Low Time	5 x T_{SYSCLK}		ns
T_{SIS}	MOSI Valid to SCK Sample Edge	2 x T_{SYSCLK}		ns
T_{SIH}	SCK Sample Edge to MOSI Change	2 x T_{SYSCLK}		ns
T_{SOH}	SCK Shift Edge to MISO Change		4 x T_{SYSCLK}	ns
T_{SLH}	Last SCK Edge to MISO Change (CKPHA = 1 ONLY)	6 x T_{SYSCLK}	8 x T_{SYSCLK}	ns
Note: T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).				

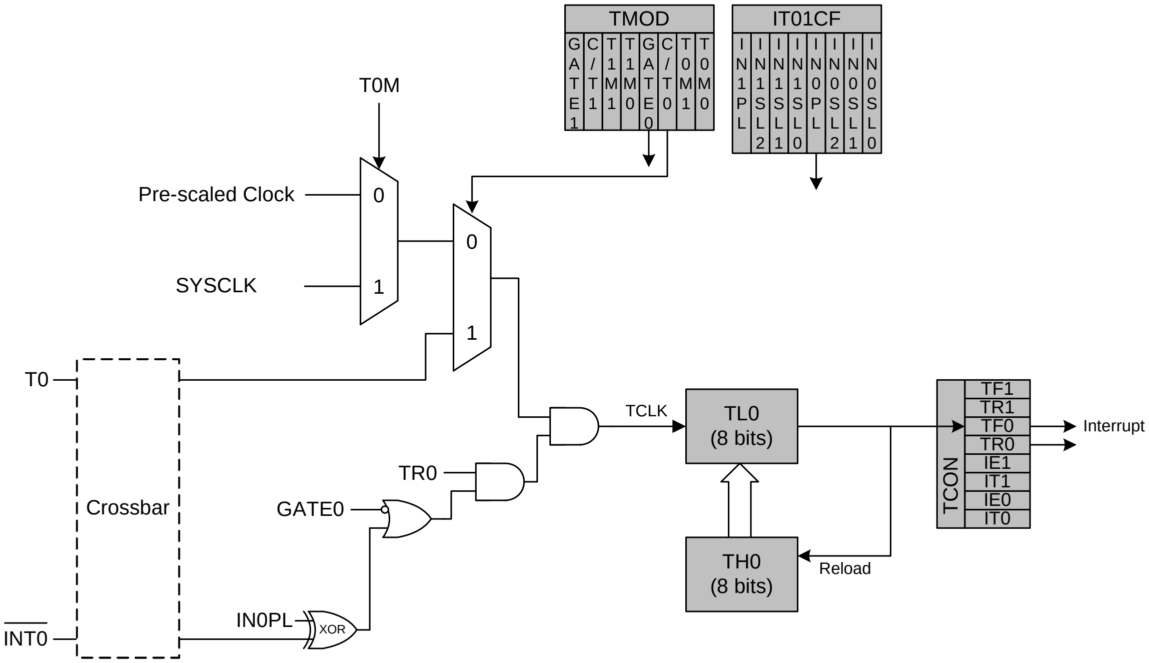

TMOD
GATE1
C/T1
T1M1
T1M0
GATE0
C/T0
T0M1
T0M0
IT01CF
IN1PL
IN1SL2
IN1SL1
IN1SL0
IN0PL
IN0SL2
IN0SL1
IN0SL0
T0M
Pre-scaled Clock
SYSCLK
0
1
0
1
T0
Crossbar
$\overline{\text{INT0}}$
TR0
GATE0
IN0PL
XOR
TCLK
TL0
(8 bits)
TH0
(8 bits)
Reload
TCON
TF1
TR1
TF0
TR0
IE1
IT1
IE0
IT0
Interrupt

Figure 28.2. T0 Mode 2 Block Diagram

28.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, C/T0, GATE0 and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates or overflow conditions for other peripherals. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

SFR Definition 28.9. TMR2RLL: Timer 2 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCA

Bit	Name	Function
7:0	TMR2RLL[7:0]	Timer 2 Reload Register Low Byte. TMR2RLL holds the low byte of the reload value for Timer 2.

SFR Definition 28.10. TMR2RLH: Timer 2 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCB

Bit	Name	Function
7:0	TMR2RLH[7:0]	Timer 2 Reload Register High Byte. TMR2RLH holds the high byte of the reload value for Timer 2.

SFR Definition 28.11. TMR2L: Timer 2 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCC

Bit	Name	Function
7:0	TMR2L[7:0]	Timer 2 Low Byte. In 16-bit mode, the TMR2L register contains the low byte of the 16-bit Timer 2. In 8-bit mode, TMR2L contains the 8-bit low byte timer value.

28.3. Timer 3

Timer 3 is a 16-bit timer formed by two 8-bit SFRs: TMR3L (low byte) and TMR3H (high byte). Timer 3 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T3SPLIT bit (TMR3CN.3) defines the Timer 3 operation mode.

Timer 3 may be clocked by the system clock, the system clock divided by 12, the external oscillator source divided by 8, or the internal low-frequency oscillator divided by 8. The external clock mode is ideal for real-time clock (RTC) functionality, where the internal high-frequency oscillator drives the system clock while Timer 3 is clocked by an external oscillator source. Note that the external oscillator source divided by 8 and the LFO source divided by 8 are synchronized with the system clock when in all operating modes except suspend. When the internal oscillator is placed in suspend mode, The external clock/8 signal or the LFO/8 output can directly drive the timer. This allows the use of an external clock or the LFO to wake up the device from suspend mode. The timer will continue to run in suspend mode and count up. When the timer overflow occurs, the device will wake from suspend mode, and begin executing code again. The timer value may be set prior to entering suspend, to overflow in the desired amount of time (number of clocks) to wake the device. If a wake-up source other than the timer wakes the device from suspend mode, it may take up to three timer clocks before the timer registers can be read or written. During this time, the STSYNC bit in register OSCICN will be set to 1, to indicate that it is not safe to read or write the timer registers.

Important Note: In internal LFO/8 mode, the divider for the internal LFO must be set to 1 for proper functionality. The timer will not operate if the LFO divider is not set to 1.

28.3.1. 16-bit Time r with Auto-Reload

When T3SPLIT (TMR3CN.3) is zero, Timer 3 operates as a 16-bit timer with auto-reload. Timer 3 can be clocked by SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 3 reload registers (TMR3RLH and TMR3RLL) is loaded into the Timer 3 register as shown in Figure 28.7, and the Timer 3 High Byte Overflow Flag (TMR3CN.7) is set. If Timer 3 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 3 overflow. Additionally, if Timer 3 interrupts are enabled and the TF3LEN bit is set (TMR3CN.5), an interrupt will be generated each time the lower 8 bits (TMR3L) overflow from 0xFF to 0x00.

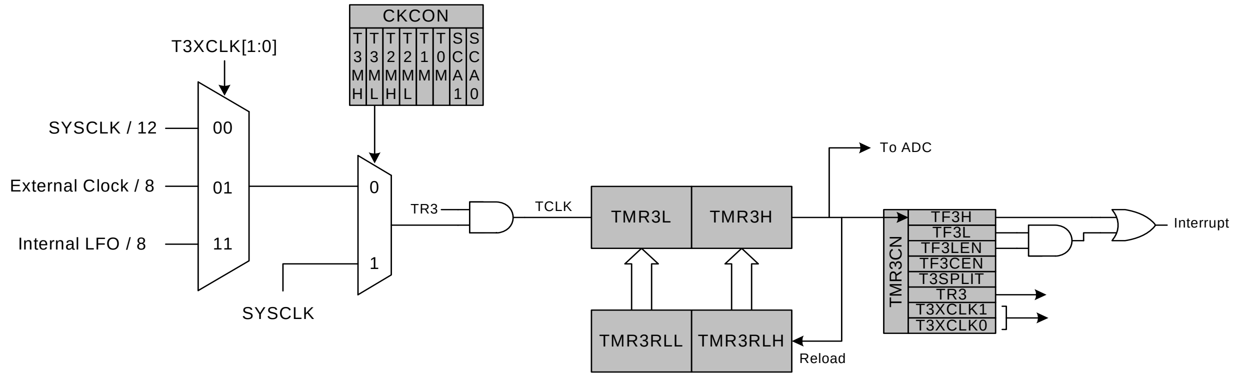


Figure 28.7. Timer 3 16-Bit Mode Block Diagram

C2 Register Definition 30.8. EP ADDRH: C2 EPROM Address High Byte

Bit	7	6	5	4	3	2	1	0
Name	EPADDR[15:8]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

C2 Address: 0xAF

Bit	Name	Function
7:0	EPADDR[15:8]	C2 EPROM Address High Byte. This register is used to set the EPROM address location during C2 EPROM operations.

C2 Register Definition 30.9. EP ADDRL: C2 EPROM Address Low Byte

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

C2 Address: 0xAE

Bit	Name	Function
7:0	EPADDR[15:8]	C2 EPROM Address Low Byte. This register is used to set the EPROM address location during C2 EPROM operations.