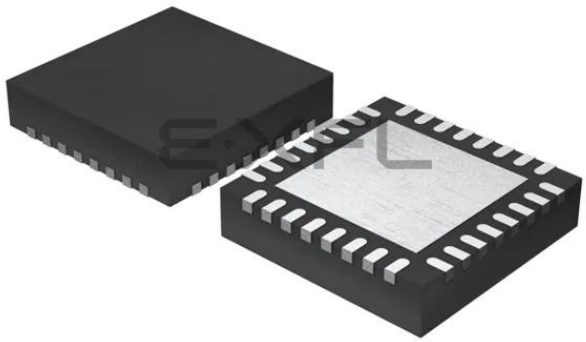

Welcome to E-XFL.COM

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	Active
Core Processor	8051
Core Size	8-Bit
Speed	48 MIPS
Connectivity	I²C, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 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/c8051f381-gmr

6.3.2. Tracking Modes

The AD0TM bit in register ADC0CN controls the ADC0 track-and-hold mode. In its default state, the ADC0 input is continuously tracked, except when a conversion is in progress. When the AD0TM bit is logic 1, ADC0 operates in low-power track-and-hold mode. In this mode, each conversion is preceded by a tracking period of 3 SAR clocks (after the start-of-conversion signal). When the CNVSTR signal is used to initiate conversions in low-power tracking mode, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR. See Figure 6.4 for track and convert timing details. Tracking can also be disabled (shutdown) when the device is in low power standby or sleep modes. Low-power track-and-hold mode is also useful when AMUX settings are frequently changed, due to the settling time requirements described in Section “6.3.3. Settling Time Requirements” on page 52.

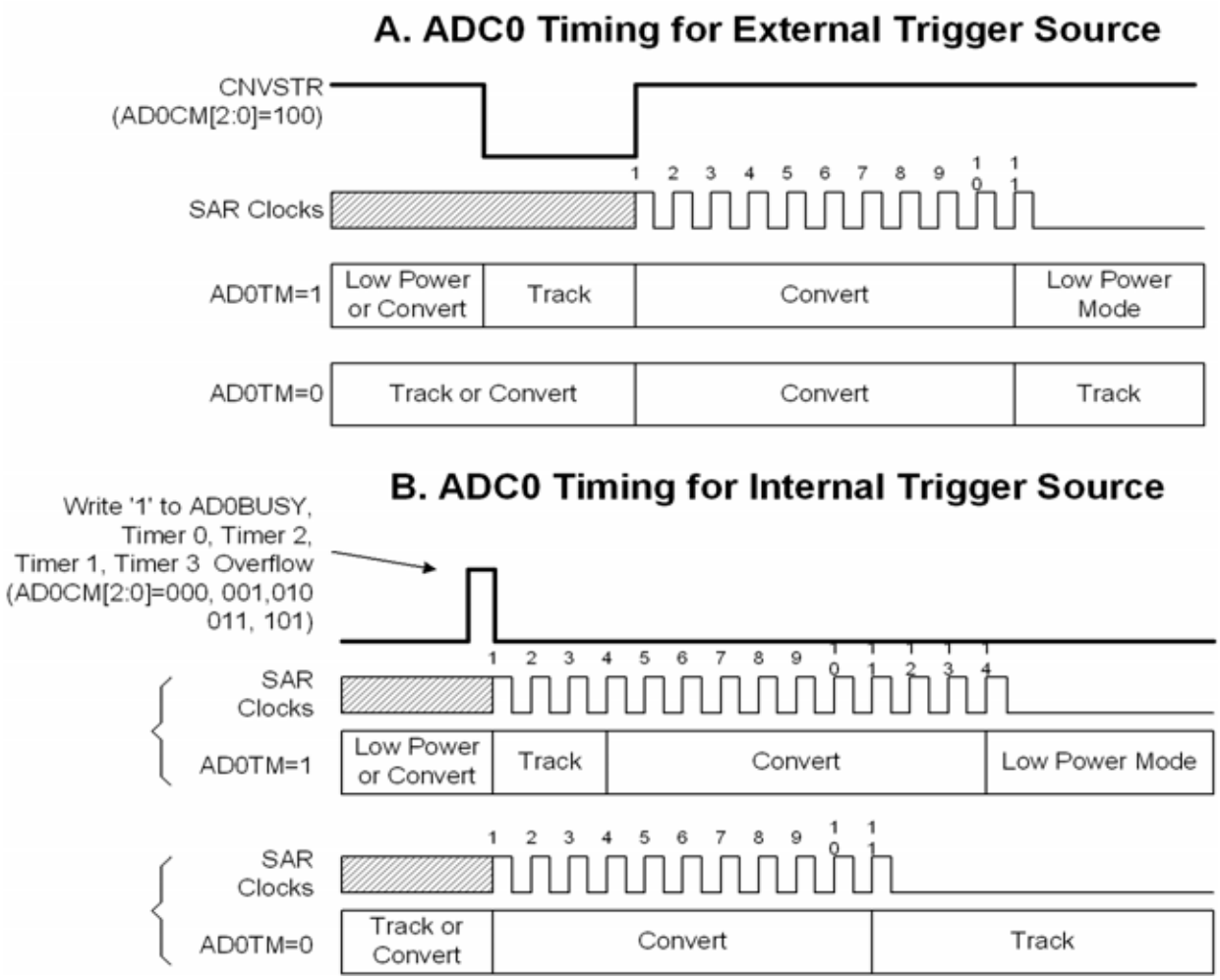

Figure 6.4. 10-Bit ADC Track and Conversion Example Timing

SFR Definition 6.7. ADC0LTH: ADC0 Less-Than Data High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC6; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0LTH[7:0]	**ADC0 Less-Than Data Word High-Order Bits.**

SFR Definition 6.8. ADC0LTL: ADC0 Less-Than Data Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC5; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0LTL[7:0]	**ADC0 Less-Than Data Word Low-Order Bits.**

SFR Definition 6.10. AMX0N: AMUX0 Negative Channel Select

Bit	7	6	5	4	3	2	1	0
Name			AMX0N[5:0]					
Type	R	R	R/W					
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBA; SFR Page = All Pages

Bit	Name	Function
7:6	Unused	Read = 00b; Write = don't care.
5:0	AMX0N[5:0]	**AMUX0 Negative Input Selection.**

AMX0N	32-pin Packages	48-pin Packages	AMX0N	32-pin Packages	48-pin Packages
000000:	P1.0	P2.0	010010:	P0.1	P0.4
000001:	P1.1	P2.1	010011:	P0.4	P1.1
000010:	P1.2	P2.2	010100:	P0.5	P1.2
000011:	P1.3	P2.3	010101:	Reserved	P1.0
000100:	P1.4	P2.5	010110:	Reserved	P1.3
000101:	P1.5	P2.6	010111:	Reserved	P1.6
000110:	P1.6	P3.0	011000:	Reserved	P1.7
000111:	P1.7	P3.1	011001:	Reserved	P2.4
001000:	P2.0	P3.4	011010:	Reserved	P2.7
001001:	P2.1	P3.5	011011:	Reserved	P3.2
001010:	P2.2	P3.7	011100:	Reserved	P3.3
001011:	P2.3	P4.0	011101:	Reserved	P3.6
001100:	P2.4	P4.3	011110:	VREF	VREF
001101:	P2.5	P4.4	011111:	GND (Single-Ended Measurement)	GND (Single-Ended Measurement)
001110:	P2.6	P4.5	100000:	Reserved	P4.1
001111:	P2.7	P4.6	100001:	Reserved	P4.2
010000:	P3.0	Reserved	100010:	Reserved	P4.7
010001:	P0.0	P0.3	100011 - 111111:	Reserved	Reserved

SFR Definition 14.1. EMI0CN: External Memory Interface Control

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

SFR Address = 0xAA; SFR Page = All Pages

Bit	Name	Function
7:0	PGSEL[7:0]	**XRAM Page Select Bits.** The XRAM Page Select Bits provide the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. 0x00: 0x0000 to 0x00FF 0x01: 0x0100 to 0x01FF ... 0xFE: 0xFE00 to 0xFEFF 0xFF: 0xFF00 to 0xFFFF

14.7.2. Multiplexed Mode

14.7.2.1. 16-bit MOVX: EMI0CF[4:2] = 001, 010, or 011

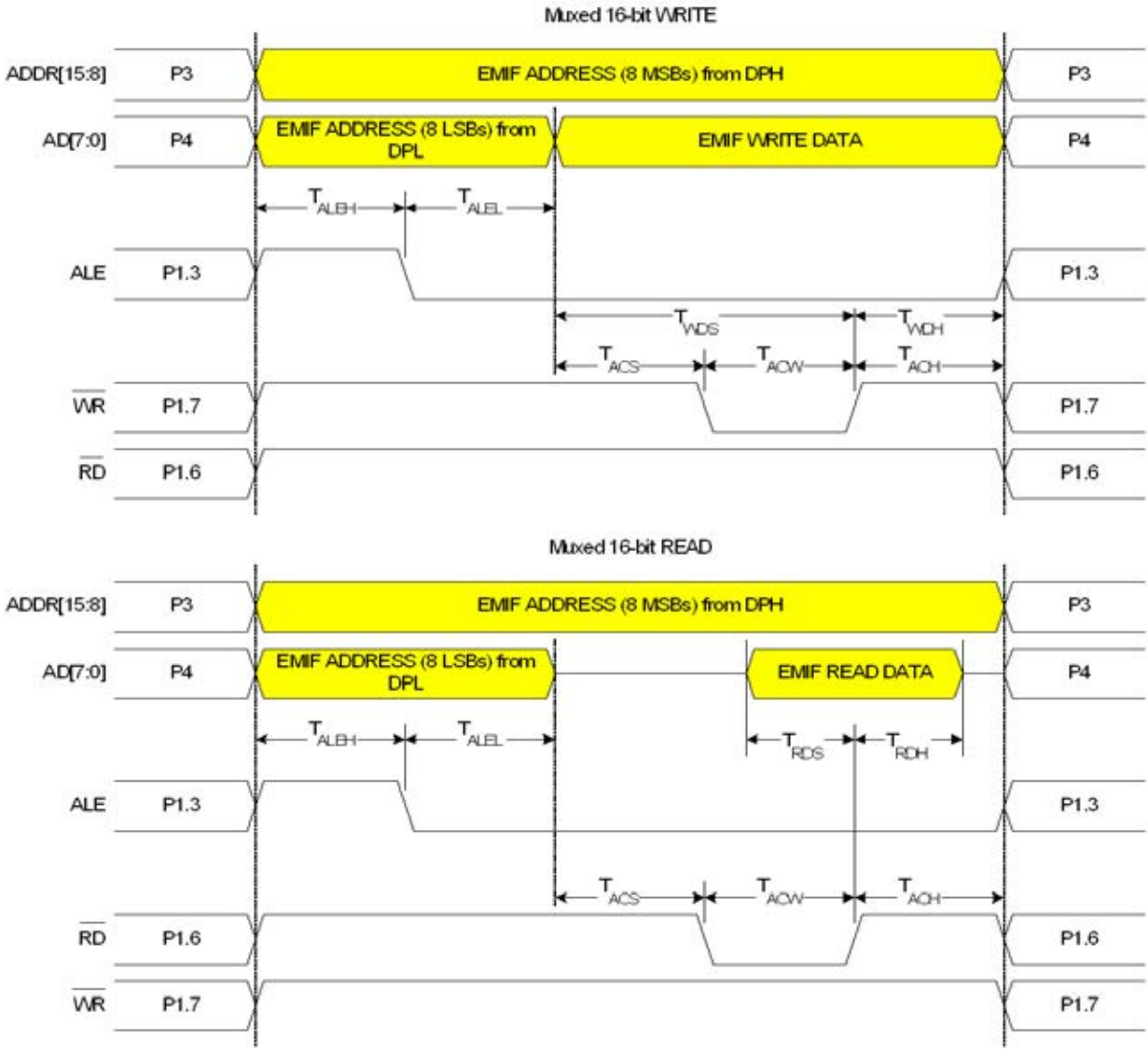

Figure 14.8. Multiplexed 16-bit MOVX Timing

Table 15.2. Special Function Registers

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

Register	Address	Page	Description	Page
ACC	0xE0	All Pages	Accumulator	86
ADC0CF	0xBC	All Pages	ADC0 Configuration	53
ADC0CN	0xE8	All Pages	ADC0 Control	55
ADC0GTH	0xC4	All Pages	ADC0 Greater-Than Compare High	56
ADC0GTL	0xC3	All Pages	ADC0 Greater-Than Compare Low	56
ADC0H	0xBE	All Pages	ADC0 High	54
ADC0L	0xBD	All Pages	ADC0 Low	54
ADC0LTH	0xC6	All Pages	ADC0 Less-Than Compare Word High	57
ADC0LTL	0xC5	All Pages	ADC0 Less-Than Compare Word Low	57
AMX0N	0xBA	All Pages	AMUX0 Negative Channel Select	61
AMX0P	0xBB	All Pages	AMUX0 Positive Channel Select	60
B	0xF0	All Pages	B Register	86
CKCON	0x8E	All Pages	Clock Control	264
CKCON1	0xE4	F	Clock Control 1	265
CLKMUL	0xB9	0	Clock Multiplier	147
CLKSEL	0xA9	All Pages	Clock Select	144
CPT0CN	0x9B	All Pages	Comparator0 Control	67
CPT0MD	0x9D	All Pages	Comparator0 Mode Selection	68
CPT0MX	0x9F	All Pages	Comparator0 MUX Selection	72
CPT1CN	0x9A	All Pages	Comparator1 Control	69
CPT1MD	0x9C	All Pages	Comparator1 Mode Selection	70
CPT1MX	0x9E	All Pages	Comparator1 MUX Selection	73
DPH	0x83	All Pages	Data Pointer High	85
DPL	0x82	All Pages	Data Pointer Low	85
EIE1	0xE6	All Pages	Extended Interrupt Enable 1	123
EIE2	0xE7	All Pages	Extended Interrupt Enable 2	125
EIP1	0xF6	All Pages	Extended Interrupt Priority 1	124
EIP2	0xF7	All Pages	Extended Interrupt Priority 2	126
EMI0CF	0x85	All Pages	External Memory Interface Configuration	97
EMI0CN	0xAA	All Pages	External Memory Interface Control	96
EMI0TC	0x84	All Pages	External Memory Interface Timing	103
FLKEY	0xB7	All Pages	Flash Lock and Key	140
FLSCL	0xB6	All Pages	Flash Scale	141

SFR Definition 16.1. IE: Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	EA	ESPI0	ET2	ES0	ET1	EX1	ET0	EX0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	EA	**Enable All Interrupts.** Globally enables/disables all interrupts. It overrides individual interrupt mask settings. 0: Disable all interrupt sources. 1: Enable each interrupt according to its individual mask setting.
6	ESPI0	**Enable Serial Peripheral Interface (SPI0) Interrupt.** This bit sets the masking of the SPI0 interrupts. 0: Disable all SPI0 interrupts. 1: Enable interrupt requests generated by SPI0.
5	ET2	**Enable Timer 2 Interrupt.** This bit sets the masking of the Timer 2 interrupt. 0: Disable Timer 2 interrupt. 1: Enable interrupt requests generated by the TF2L or TF2H flags.
4	ES0	**Enable UART0 Interrupt.** This bit sets the masking of the UART0 interrupt. 0: Disable UART0 interrupt. 1: Enable UART0 interrupt.
3	ET1	**Enable Timer 1 Interrupt.** This bit sets the masking of the Timer 1 interrupt. 0: Disable all Timer 1 interrupt. 1: Enable interrupt requests generated by the TF1 flag.
2	EX1	**Enable External Interrupt 1.** This bit sets the masking of External Interrupt 1. 0: Disable external interrupt 1. 1: Enable interrupt requests generated by the $\overline{\text{INT1}}$ input.
1	ET0	**Enable Timer 0 Interrupt.** This bit sets the masking of the Timer 0 interrupt. 0: Disable all Timer 0 interrupt. 1: Enable interrupt requests generated by the TF0 flag.
0	EX0	**Enable External Interrupt 0.** This bit sets the masking of External Interrupt 0. 0: Disable external interrupt 0. 1: Enable interrupt requests generated by the $\overline{\text{INT0}}$ input.

19. Oscillators and Clock Selection

C8051F380/1/2/3/4/5/6/7/C devices include a programmable internal high-frequency oscillator, a programmable internal low-frequency oscillator, and an external oscillator drive circuit. The internal high-frequency oscillator can be enabled/disabled and calibrated using the OSCICN and OSCICL registers, as shown in Figure 19.1. The internal low-frequency oscillator can be enabled/disabled and calibrated using the OSCLCN register. The system clock can be sourced by the external oscillator circuit or either internal oscillator. Both internal oscillators offer a selectable post-scaling feature. The USB clock (USBCLK) can be derived from the internal oscillators or external oscillator.

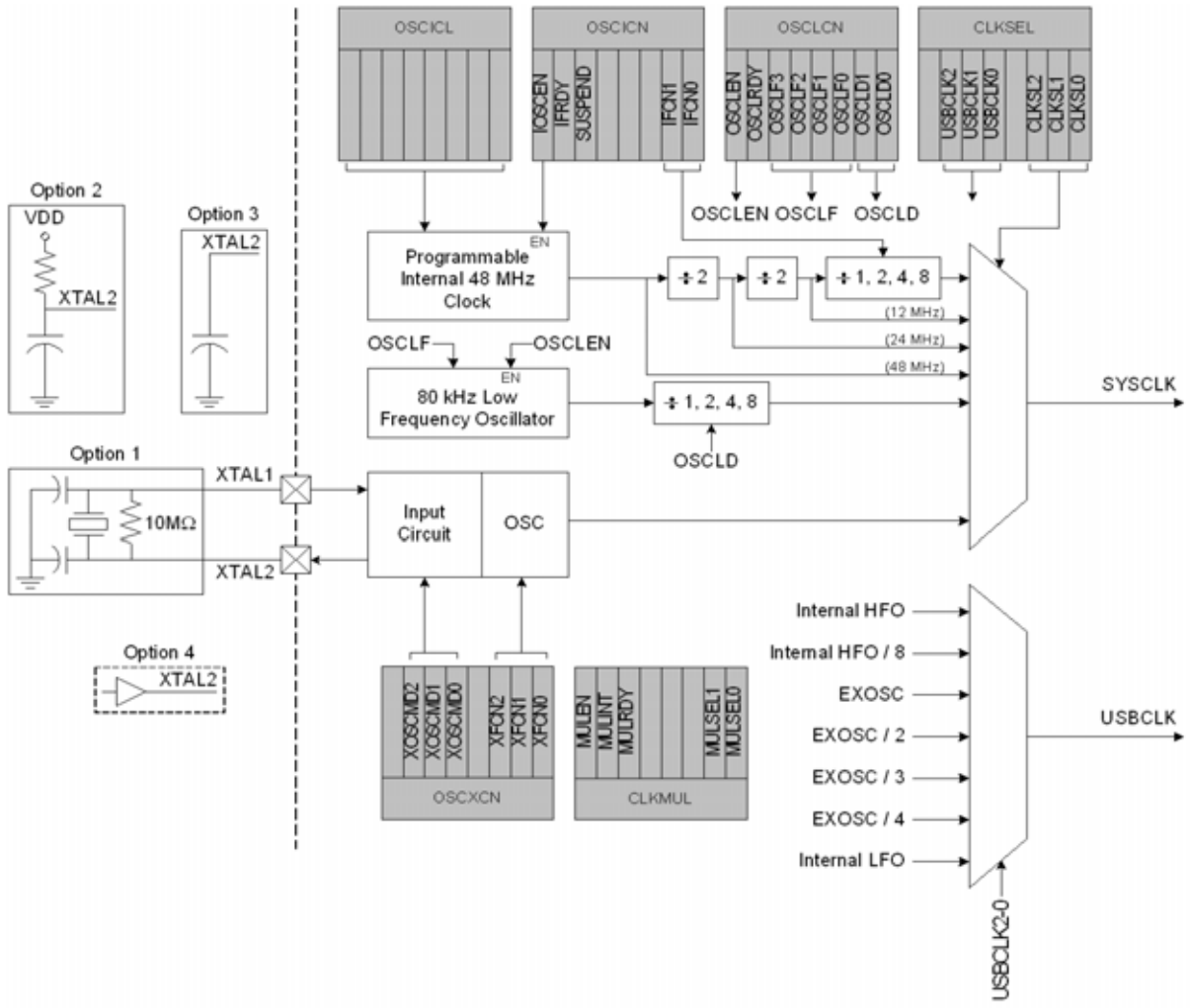

Figure 19.1. Oscillator Options

SFR Definition 21.2. USB0ADR: USB0 Indirect Address

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

SFR Address = 0x96; SFR Page = All Pages

Bit	Name	Description	Write	Read
7	BUSY	**USB0 Register Read Busy Flag.** This bit is used during indirect USB0 register accesses.	0: No effect. 1: A USB0 indirect register read is initiated at the address specified by the USBADDR bits.	0: USB0DAT register data is valid. 1: USB0 is busy accessing an indirect register; USB0DAT register data is invalid.
6	AUTORD	**USB0 Register Auto-read Flag.** This bit is used for block FIFO reads. 0: BUSY must be written manually for each USB0 indirect register read. 1: The next indirect register read will automatically be initiated when software reads USB0DAT (USBADDR bits will not be changed).		
5:0	USBADDR[5:0]	**USB0 Indirect Register Address Bits.** These bits hold a 6-bit address used to indirectly access the USB0 core registers. Table 21.2 lists the USB0 core registers and their indirect addresses. Reads and writes to USB0DAT will target the register indicated by the USBADDR bits.		

imum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

Table 22.2. Minimum SDA Setup and Hold Times

EXTHOLD	Minimum SDA Setup Time	Minimum SDA Hold Time
0	T_{low} – 4 system clocks or 1 system clock + s/w delay*	3 system clocks
1	11 system clocks	12 system clocks
Note: Setup Time for ACK bit transmissions and the MSB of all data transfers. When using software acknowledgement, the s/w delay occurs between the time SMB0DAT or ACK is written and when SI is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero.		

With the SMBnTOE bit set, Timer 3 (SMBus0) and Timer 5 (SMBus1) should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “22.3.4. SCL Low Timeout” on page 207). The SMBus interface will force the associated timer to reload while SCL is high, and allow the timer to count when SCL is low. The timer interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBnFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods (see Figure 22.4).

22.4.2. SMBus Timing Control Register

The SMBus Timing Control Register (SMBTC)is used to restrict the detection of a START condition under certain circumstances. In some systems where there is significant mis-match between the impedance or the capacitance on the SDA and SCL lines, it may be possible for SCL to fall after SDA during an address or data transfer. Such an event can cause a false START detection on the bus. These kind of events are not expected in a standard SMBus or I2C-compliant system. **In most systems this parameter should not be adjusted, and it is recommended that it be left at its default value.**

By default, if the SCL falling edge is detected after the falling edge of SDA (i.e. one SYSCLK cycle or more), the device will detect this as a START condition. The SMBTC register is used to increase the amount of hold time that is required between SDA and SCL falling before a START is recognized. An additional 2, 4, or 8 SYSCLKs can be added to prevent false START detection in systems where the bus conditions warrant this.

the incoming slave address. Additionally, if the GCn bit in register SMBnADR is set to 1, hardware will recognize the General Call Address (0x00). Table 22.4 shows some example parameter settings and the slave addresses that will be recognized by hardware under those conditions.

Table 22.4. Hardware Address Recognition Examples (EHACK = 1)

Hardware Slave Address SLVn[6:0]	Slave Address Mask SLVMn[6:0]	GCn bit	Slave Addresses Recognized by Hardware
0x34	0x7F	0	0x34
0x34	0x7F	1	0x34, 0x00 (General Call)
0x34	0x7E	0	0x34, 0x35
0x34	0x7E	1	0x34, 0x35, 0x00 (General Call)
0x70	0x73	0	0x70, 0x74, 0x78, 0x7C

SFR Definition 22.6. SMB0ADR: SMBus0 Slave Address

Bit	7	6	5	4	3	2	1	0
Name	SLV0[6:0]							GC0
Type	R/W							R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCF; SFR Page = 0

Bit	Name	Function
7:1	SLV0[6:0]	**SMBus Hardware Slave Address.** Defines the SMBus0 Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM0[6:0] are checked against the incoming address. This allows multiple addresses to be recognized.
0	GC0	**General Call Address Enable.** When hardware address recognition is enabled (EHACK0 = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware. 0: General Call Address is ignored. 1: General Call Address is recognized.

SFR Definition 22.7. SMB0ADM: SMBus0 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM0[6:0]							EHACK0
Type	R/W							R/W
Reset	1	1	1	1	1	1	1	0

SFR Address = 0xCE; SFR Page = 0

Bit	Name	Function
7:1	SLVM0[6:0]	**SMBus0 Slave Address Mask.** Defines which bits of register SMB0ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM0[6:0] enables comparisons with the corresponding bit in SLV0[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
0	EHACK0	**Hardware Acknowledge Enable.** Enables hardware acknowledgement of slave address and received data bytes. 0: Firmware must manually acknowledge all incoming address and data bytes. 1: Automatic Slave Address Recognition and Hardware Acknowledge is Enabled.

SFR Definition 22.8. SMB1ADR: SMBus1 Slave Address

Bit	7	6	5	4	3	2	1	0
Name	SLV1[6:0]							GC1
Type	R/W							R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCF; SFR Page = F

Bit	Name	Function
7:1	SLV1[6:0]	**SMBus1 Hardware Slave Address.** Defines the SMBus1 Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM1[6:0] are checked against the incoming address. This allows multiple addresses to be recognized.
0	GC1	**General Call Address Enable.** When hardware address recognition is enabled (EHACK1 = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware. 0: General Call Address is ignored. 1: General Call Address is recognized.

22.4.5. Data Register

The SMBus Data register SMBnDAT holds a byte of serial data to be transmitted or one that has just been received. Software may safely read or write to the data register when the SIn flag is set. Software should not attempt to access the SMBnDAT register when the SMBus is enabled and the SIn flag is cleared to logic 0, as the interface may be in the process of shifting a byte of data into or out of the register.

Data in SMBnDAT is always shifted out MSB first. After a byte has been received, the first bit of received data is located at the MSB of SMBnDAT. While data is being shifted out, data on the bus is simultaneously being shifted in. SMBnDAT always contains the last data byte present on the bus. In the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data or address in SMBnDAT.

SFR Definition 22.10. SMB0DAT: SMBus Data

Bit	7	6	5	4	3	2	1	0
Name	SMB0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC2; SFR Page = 0

Bit	Name	Function
7:0	SMB0DAT[7:0]	**SMBus0 Data.** The SMB0DAT register contains a byte of data to be transmitted on the SMBus0 serial interface or a byte that has just been received on the SMBus0 serial interface. The CPU can read from or write to this register whenever the SI0 serial interrupt flag (SMB0CN.0) is set to logic 1. The serial data in the register remains stable as long as the SI0 flag is set. When the SI0 flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register.

22.5.2. Read Sequence (Master)

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. 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.

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.**

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 for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 22.6 shows a typical master read 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.

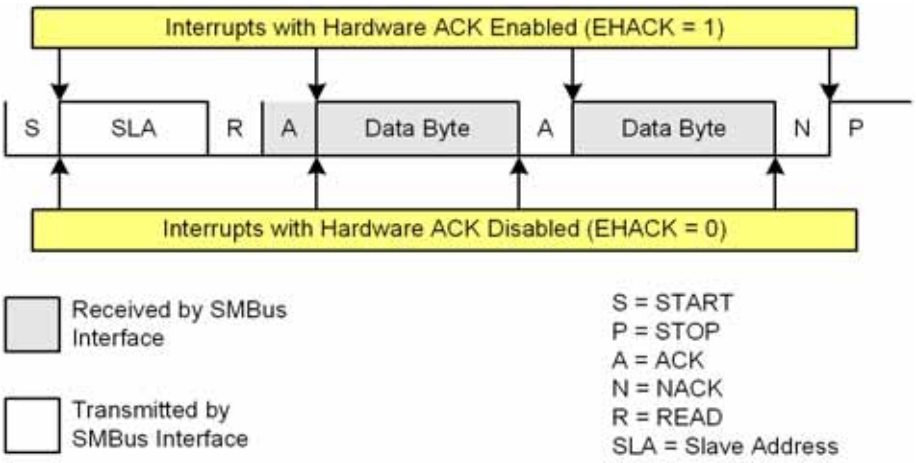

Figure 22.6. Typical Master Read Sequence

Table 22.6. SMBus Status Decoding: Hardware ACK Enabled (EHACK = 1) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Receiver	1000	0	0	1	A master data byte was received; ACK sent.	Set ACK for next data byte; Read SMB0DAT.	0	0	1	1000
						Set NACK to indicate next data byte as the last data byte; Read SMB0DAT.	0	0	0	1000
						Initiate repeated START.	1	0	0	1110
						Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	X	1100
		0	0	0	A master data byte was received; NACK sent (last byte).	Read SMB0DAT; send STOP.	0	1	0	—
						Read SMB0DAT; Send STOP followed by START.	1	1	0	1110
						Initiate repeated START.	1	0	0	1110
						Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	X	1100
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X	0001
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X	0100
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X	0001
	0101	0	X	X	An illegal STOP or bus error was detected while a Slave Transmission was in progress.	Clear STO.	0	0	X	—

26.4.2. 8-bit Timers with Auto-Reload

When T4SPLIT is 1 and T4CE = 0, Timer 4 operates as two 8-bit timers (TMR4H and TMR4L). Both 8-bit timers operate in auto-reload mode as shown in Figure 26.13. TMR4RLL holds the reload value for TMR4L; TMR4RLH holds the reload value for TMR4H. The TR4 bit in TMR4CN handles the run control for TMR4H. TMR4L 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 4 Clock Select bits (T4MH and T4ML in CKCON1) select either SYSCLK or the clock defined by the Timer 4 External Clock Select bit (T4XCLK in TMR4CN), as follows:

T4MH	T4XCLK	TMR4H Clock Source
0	0	SYSCLK/12
0	1	External Clock/8
1	X	SYSCLK

T4ML	T4XCLK	TMR4L Clock Source
0	0	SYSCLK/12
0	1	External Clock/8
1	X	SYSCLK

The TF4H bit is set when TMR4H overflows from 0xFF to 0x00; the TF4L bit is set when TMR4L overflows from 0xFF to 0x00. When Timer 4 interrupts are enabled, an interrupt is generated each time TMR4H overflows. If Timer 4 interrupts are enabled and TF4LEN (TMR4CN.5) is set, an interrupt is generated each time either TMR4L or TMR4H overflows. When TF4LEN is enabled, software must check the TF4H and TF4L flags to determine the source of the Timer 4 interrupt. The TF4H and TF4L interrupt flags are not cleared by hardware and must be manually cleared by software.

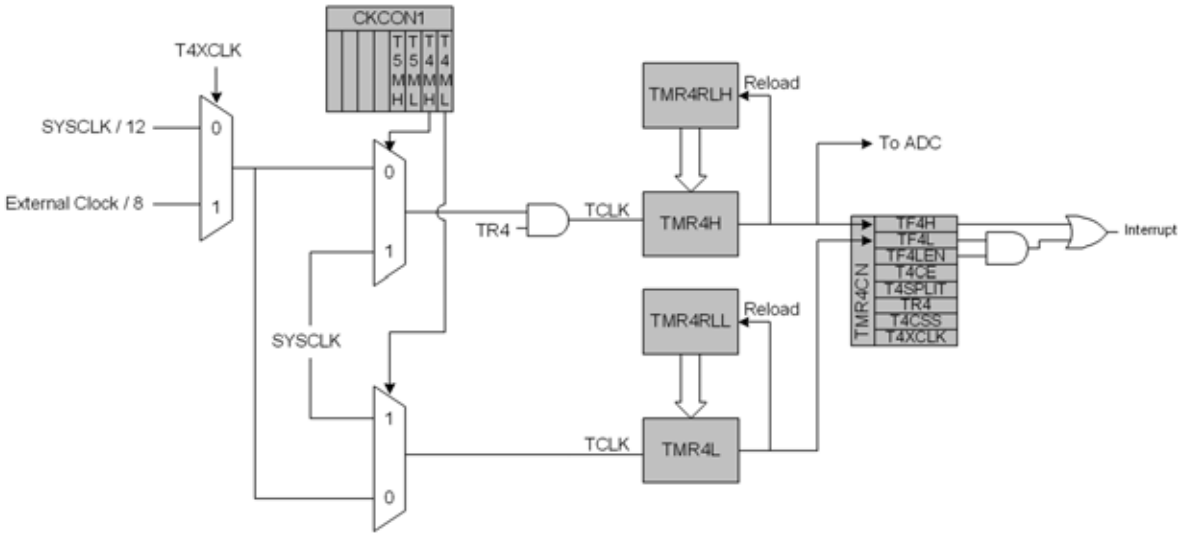

Figure 26.13. Timer 4 8-Bit Mode Block Diagram

SFR Definition 26.25. TMR5RLL: Timer 5 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCA; SFR Page = F

Bit	Name	Function
7:0	TMR5RLL[7:0]	**Timer 5 Reload Register Low Byte.** TMR5RLL holds the low byte of the reload value for Timer 5.

SFR Definition 26.26. TMR5RLH: Timer 5 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCB; SFR Page = F

Bit	Name	Function
7:0	TMR5RLH[7:0]	**Timer 5 Reload Register High Byte.** TMR5RLH holds the high byte of the reload value for Timer 5.

SFR Definition 26.27. TMR5L: Timer 5 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCC; SFR Page = F

Bit	Name	Function
7:0	TMR5L[7:0]	**Timer 5 Low Byte.** In 16-bit mode, the TMR5L register contains the low byte of the 16-bit Timer 5. In 8-bit mode, TMR5L contains the 8-bit low byte timer value.

The 8-bit offset held in PCA0CPH4 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 27.4, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL4) + (256 - PCA0L)$$

Equation 27.4. Watchdog Timer Offset in PCA Clocks

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH4 and PCA0H. Software may force a WDT reset by writing a 1 to the CCF4 flag (PCA0CN.4) while the WDT is enabled.

27.4.2. 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 PCA0CPL4 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 PCA0CPH4.

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 PCA0CPL4 defaults to 0x00. Using Equation 27.4, this results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. Table 27.3 lists some example timeout intervals for typical system clocks.