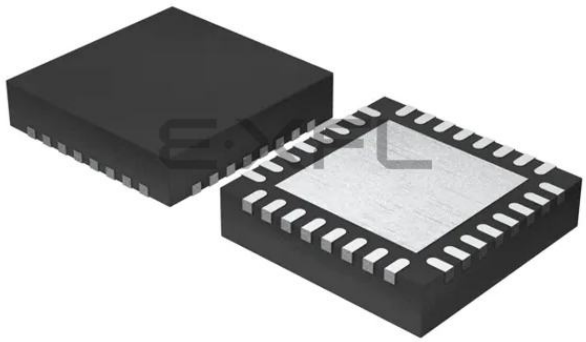

Welcome to E-XFL.COM

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
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2.25K 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/c8051f38b-b-gmr

Table 3.1. Pin Definitions for the C8051F388/9/A/B (Continued)

Name	Pin Numbers		Type	Description
	48-pin	32-pin		
P3.2	28	—	D I/O or A In	Port 3.2.
P3.3	27	—	D I/O or A In	Port 3.3.
P3.4	26	—	D I/O or A In	Port 3.4.
P3.5	25	—	D I/O or A In	Port 3.5.
P3.6	24	—	D I/O or A In	Port 3.6.
P3.7	23	—	D I/O or A In	Port 3.7.
P4.0	22	—	D I/O or A In	Port 4.0. See Section 20 for a complete description of Port 4.
P4.1	21	—	D I/O or A In	Port 4.1.
P4.2	20	—	D I/O or A In	Port 4.2.
P4.3	19	—	D I/O or A In	Port 4.3.
P4.4	18	—	D I/O or A In	Port 4.4.
P4.5	17	—	D I/O or A In	Port 4.5.
P4.6	16	—	D I/O or A In	Port 4.6.
P4.7	15	—	D I/O or A In	Port 4.7.

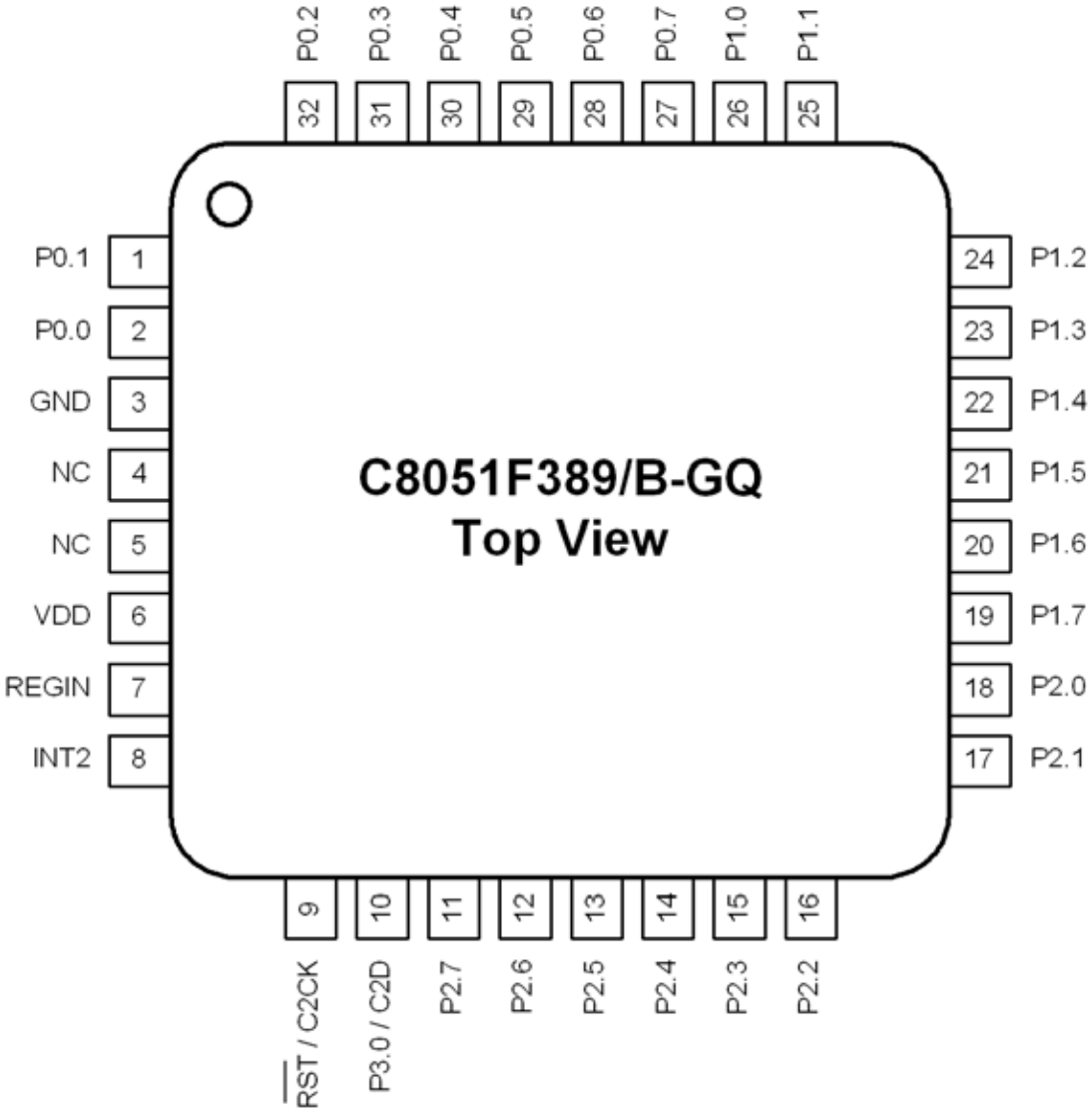

Figure 3.4. LQFP-32 (C8051F389/B) Pinout Diagram (Top View)

9. Voltage Regulators (REG0 and REG1)

C8051F388/9/A/B devices include two internal voltage regulators: one regulates a voltage source on REGIN to 3.3 V (REG0), and the other regulates the internal core supply to 1.8 V from a V_{DD} supply of 1.8 to 3.6 V (REG1). When enabled, the REG0 output appears on the V_{DD} pin and can be used to power external devices. REG0 can be enabled/disabled by software using bit REG0DIS in register REG01CN (SFR Definition 9.1). REG1 has two power-saving modes built into the regulator to help reduce current consumption in low-power applications. These modes are accessed through the REG01CN register. Electrical characteristics for the on-chip regulators are specified in Table 5.5 on page 37.

9.1. Voltage Regulator (REG0)

See “4. Typical Connection Diagrams” for typical connection diagrams using the REG0 voltage regulator.

9.1.1. Regulator Mode Selection

REG0 offers a low power mode intended for use when the device is in suspend mode. In this low power mode, the REG0 output remains as specified; however the REG0 dynamic performance (response time) is degraded. See Table 5.5 for normal and low power mode supply current specifications. The REG0 mode selection is controlled via the REG0MD bit in register REG01CN.

9.1.2. External Interrupt 2 (INT2)

On C8051F388/9/A/B devices, the VBSTAT bit (register REG01CN) indicates the current logic level of the INT2 signal. If enabled, a INT2 interrupt will be generated when the INT2 signal has either a falling or rising edge. The INT2 interrupt is edge-sensitive, and has no associated interrupt pending flag. See Table 5.3 for INT2 input parameters.

9.2. Voltage Regulator (REG1)

Under default conditions, the internal REG1 regulator will remain on when the device enters STOP mode. This allows any enabled reset source to generate a reset for the device and bring the device out of STOP mode. For additional power savings, the STOPCF bit can be used to shut down the regulator and the internal power network of the device when the part enters STOP mode. When STOPCF is set to 1, the $\overline{\text{RST}}$ pin and a full power cycle of the device are the only methods of generating a reset.

REG1 offers an additional low power mode intended for use when the device is in suspend mode. This low power mode should not be used during normal operation or if the REG0 Voltage Regulator is disabled. See Table 5.5 for normal and low power mode supply current specifications. The REG1 mode selection is controlled via the REG1MD bit in register REG01CN.

Important Note: At least 12 clock instructions must occur after placing REG1 in low power mode before the Internal High Frequency Oscillator is Suspended (OSCICN.5 = 1b).

12. Prefetch Engine

The C8051F388/9/A/B family of devices incorporate a 2-byte prefetch engine. Because the access time of the Flash memory is 40 ns, and the minimum instruction time is roughly 20 ns, the prefetch engine is necessary for full-speed code execution. Instructions are read from Flash memory two bytes at a time by the prefetch engine and given to the CIP-51 processor core to execute. When running linear code (code without any jumps or branches), the prefetch engine allows instructions to be executed at full speed. When a code branch occurs, the processor may be stalled for up to two clock cycles while the next set of code bytes is retrieved from Flash memory. It is recommended that the prefetch be used for optimal code execution timing.

Note: The prefetch engine can be disabled when the device is in suspend mode to save power.

SFR Definition 12.1. PFE0CN: Prefetch Engine Control

Bit	7	6	5	4	3	2	1	0
Name			PFEN					FLBWE
Type	R	R	R/W	R	R	R	R	R/W
Reset	0	0	1	0	0	0	0	0

SFR Address = 0xAF; SFR Page = All Pages

Bit	Name	Function
7:6	Unused	Read = 00b, Write = don't care.
5	PFEN	**Prefetch Enable.** This bit enables the prefetch engine. 0: Prefetch engine is disabled. 1: Prefetch engine is enabled.
4:1	Unused	Read = 0000b. Write = don't care.
0	FLBWE	**Flash Block Write Enable.** This bit allows block writes to Flash memory from software. 0: Each byte of a software Flash write is written individually. 1: Flash bytes are written in groups of two.

13. Memory Organization

The memory organization of the CIP-51 System Controller is similar to that of a standard 8051. There are two separate memory spaces: program memory and data memory. Program and data memory share the same address space but are accessed via different instruction types. The CIP-51 memory organization is shown in Figure 13.1 and Figure 13.2.

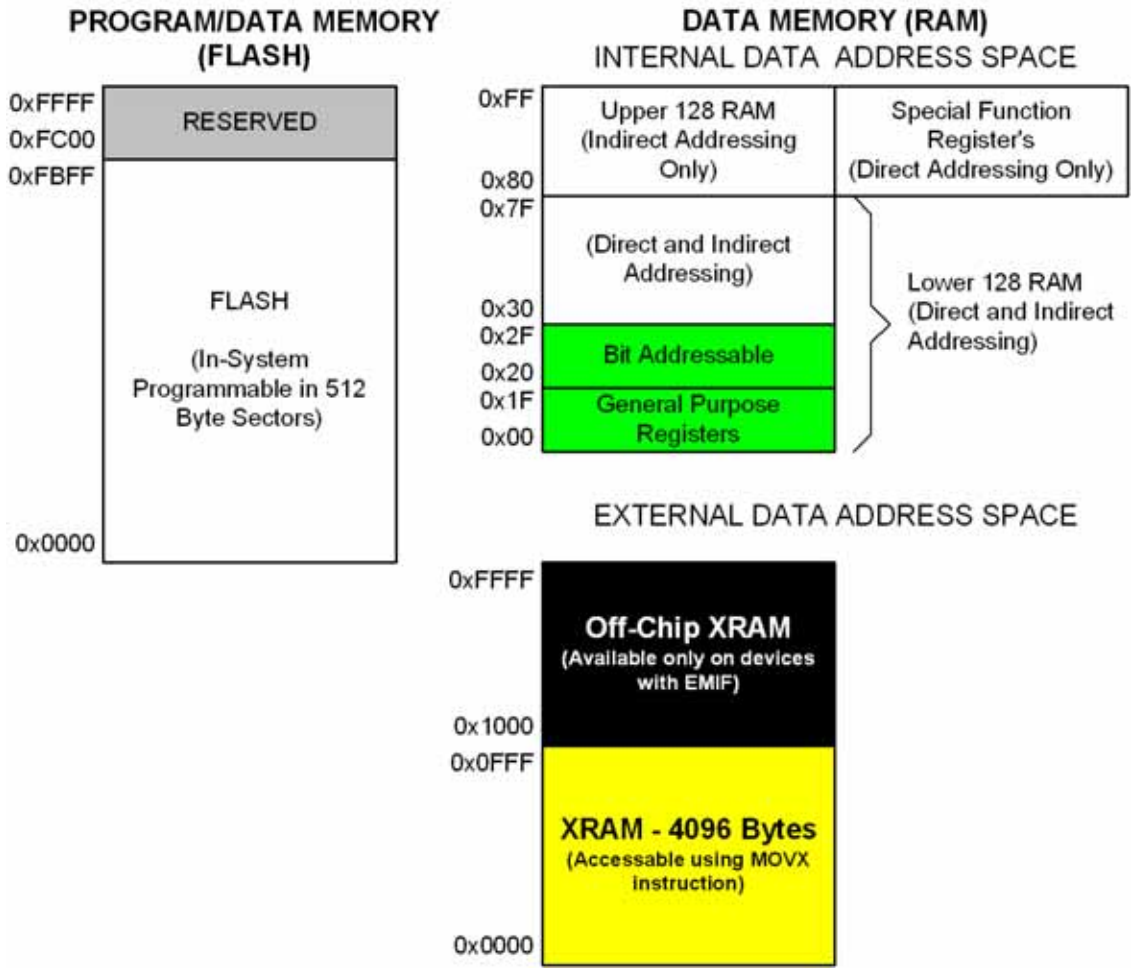

Figure 13.1. On-Chip Memory Map for 64 kB Devices (C8051F388/9)

14.5.3. Split Mode with Bank Select

When EMI0CF.[3:2] are set to 10, the XRAM memory map is split into two areas, on-chip space and off-chip space.

- Effective addresses below the internal XRAM size boundary will access on-chip XRAM space.
- Effective addresses above the internal XRAM size boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is on-chip or off-chip. The upper 8-bits of the Address Bus A[15:8] are determined by EMI0CN, and the lower 8-bits of the Address Bus A[7:0] are determined by R0 or R1. All 16-bits of the Address Bus A[15:0] are driven in “Bank Select” mode.
- 16-bit MOVX operations use the contents of DPTR to determine whether the memory access is on-chip or off-chip, and the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

14.5.4. External Only

When EMI0CF[3:2] are set to 11, all MOVX operations are directed to off-chip space. On-chip XRAM is not visible to the CPU. This mode is useful for accessing off-chip memory located between 0x0000 and the internal XRAM size boundary.

- 8-bit MOVX operations ignore the contents of EMI0CN. The upper Address bits A[15:8] are not driven (identical behavior to an off-chip access in “Split Mode without Bank Select” described above). This allows the user to manipulate the upper address bits at will by setting the Port state directly. The lower 8-bits of the effective address A[7:0] are determined by the contents of R0 or R1.
- 16-bit MOVX operations use the contents of DPTR to determine the effective address A[15:0]. The full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

15. Special Function Registers

The direct-access data memory locations from 0x80 to 0xFF constitute the special function registers (SFRs). The SFRs provide control and data exchange with the C8051F388/9/A/B's resources and peripherals. The CIP-51 controller core duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the C8051F388/9/A/B. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 15.1 lists the SFRs implemented in the C8051F388/9/A/B device family.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g. P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing these areas will have an indeterminate effect and should be avoided. Refer to the corresponding pages of the data sheet, as indicated in Table 15.2, for a detailed description of each register.

15.1. SFR Paging

The CIP-51 features SFR paging, allowing the device to map many SFRs into the 0x80 to 0xFF memory address space. The SFR memory space has 256 pages. In this way, each memory location from 0x80 to 0xFF can access up to 256 SFRs. The C8051F388/9/A/B devices utilize two SFR pages: 0x0, and 0xF. Most SFRs are available on both pages. SFR pages are selected using the Special Function Register Page Selection register, SFRPAGE. The procedure for reading and writing an SFR is as follows:

1. Select the appropriate SFR page number using the SFRPAGE register.
2. Use direct accessing mode to read or write the special function register (MOV instruction).

Important Note: When reading or writing SFRs that are not available on all pages within an ISR, it is recommended to save the state of the SFRPAGE register on ISR entry, and restore state on exit.

SFR Definition 15.1. SFRPAGE: SFR Page

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

SFR Address = 0xBF; SFR Page = All Pages

Bit	Name	Function
7:0	SFRPAGE[7:0]	**SFR Page Bits.** Represents the SFR Page the C8051 core uses when reading or modifying SFRs. Write: Sets the SFR Page. Read: Byte is the SFR page the C8051 core is using.

Table 15.2. Special Function Registers (Continued)

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

Register	Address	Page	Description	Page
IE	0xA8	All Pages	Interrupt Enable	115
IP	0xB8	All Pages	Interrupt Priority	116
IT01CF	0xE4	0	INT0/INT1 Configuration	122
OSCICL	0xB3	All Pages	Internal Oscillator Calibration	139
OSCICN	0xB2	All Pages	Internal Oscillator Control	140
OSCLCN	0x86	All Pages	Internal Low-Frequency Oscillator Control	142
OSCXCN	0xB1	All Pages	External Oscillator Control	146
P0	0x80	All Pages	Port 0 Latch	156
P0MDIN	0xF1	All Pages	Port 0 Input Mode Configuration	156
P0MDOUT	0xA4	All Pages	Port 0 Output Mode Configuration	157
P0SKIP	0xD4	All Pages	Port 0 Skip	157
P1	0x90	All Pages	Port 1 Latch	158
P1MDIN	0xF2	All Pages	Port 1 Input Mode Configuration	158
P1MDOUT	0xA5	All Pages	Port 1 Output Mode Configuration	159
P1SKIP	0xD5	All Pages	Port 1 Skip	159
P2	0xA0	All Pages	Port 2 Latch	160
P2MDIN	0xF3	All Pages	Port 2 Input Mode Configuration	160
P2MDOUT	0xA6	All Pages	Port 2 Output Mode Configuration	161
P2SKIP	0xD6	All Pages	Port 2 Skip	161
P3	0xB0	All Pages	Port 3 Latch	162
P3MDIN	0xF4	All Pages	Port 3 Input Mode Configuration	162
P3MDOUT	0xA7	All Pages	Port 3 Output Mode Configuration	163
P3SKIP	0xDF	All Pages	Port 3Skip	163
P4	0xC7	All Pages	Port 4 Latch	164
P4MDIN	0xF5	All Pages	Port 4 Input Mode Configuration	164
P4MDOUT	0xAE	All Pages	Port 4 Output Mode Configuration	165
PCA0CN	0xD8	All Pages	PCA Control	272
PCA0CPH0	0xFC	All Pages	PCA Capture 0 High	276
PCA0CPH1	0xEA	All Pages	PCA Capture 1 High	276
PCA0CPH2	0xEC	All Pages	PCA Capture 2 High	276
PCA0CPH3	0xEE	All Pages	PCA Capture 3High	276
PCA0CPH4	0xFE	All Pages	PCA Capture 4 High	276
PCA0CPL0	0xFB	All Pages	PCA Capture 0 Low	276
PCA0CPL1	0xE9	All Pages	PCA Capture 1 Low	276

19.5.2. External RC Example

If an RC network is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 19.1, “RC Mode”. The capacitor should be no greater than 100 pF; however, for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, first select the RC network value to produce the desired frequency of oscillation, according to Equation 19.1, where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and R = the pull-up resistor value in kΩ.

$$f = 1.23 \times 10^3 \S (R \times C)$$

Equation 19.1. RC Mode Oscillator Frequency

For example: If the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

f = 1.23(10^3) / RC = 1.23 (10^3) / [246 x 50] = 0.1 MHz = 100 kHz

Referring to the table in SFR Definition 19.6, the required XFCN setting is 010b.

19.5.3. External Capacitor Example

If a capacitor is used as an external oscillator for the MCU, the circuit should be configured as shown in Figure 19.1, “C Mode”. The capacitor should be no greater than 100 pF; however, for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, select the capacitor to be used and find the frequency of oscillation according to Equation 19.2, where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and V_{DD} = the MCU power supply in Volts.

$$f = (KF)/(C \times V_{DD})$$

Equation 19.2. C Mode Oscillator Frequency

For example: Assume V_{DD} = 3.0 V and f = 150 kHz:

f = KF / (C x VDD)
0.150 MHz = KF / (C x 3.0)

Since the frequency of roughly 150 kHz is desired, select the K Factor from the table in SFR Definition 19.6 (OSCXCN) as KF = 22:

0.150 MHz = 22 / (C x 3.0)
C x 3.0 = 22 / 0.150 MHz
C = 146.6 / 3.0 pF = 48.8 pF

Therefore, the XFCN value to use in this example is 011b and C = 50 pF.

21. SMBus0 and SMBus1 (I^2C Compatible)

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I^2C serial bus. The C8051F388/9/A/B devices contain two SMBus interfaces, SMBus0 and SMBus1.

Reads and writes to the SMBus by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/20th of the system clock as a master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus 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 peripherals can be fully driven by software (i.e., software accepts/rejects slave addresses, and generates ACKs), or hardware slave address recognition and automatic ACK generation can be enabled to minimize software overhead. A block diagram of the SMBus0 peripheral and the associated SFRs is shown in Figure 21.1. SMBus1 is identical, with the exception of the available timer options for the clock source, and the timer used to implement the SCL low timeout feature. Refer to the specific SFR definitions for more details.

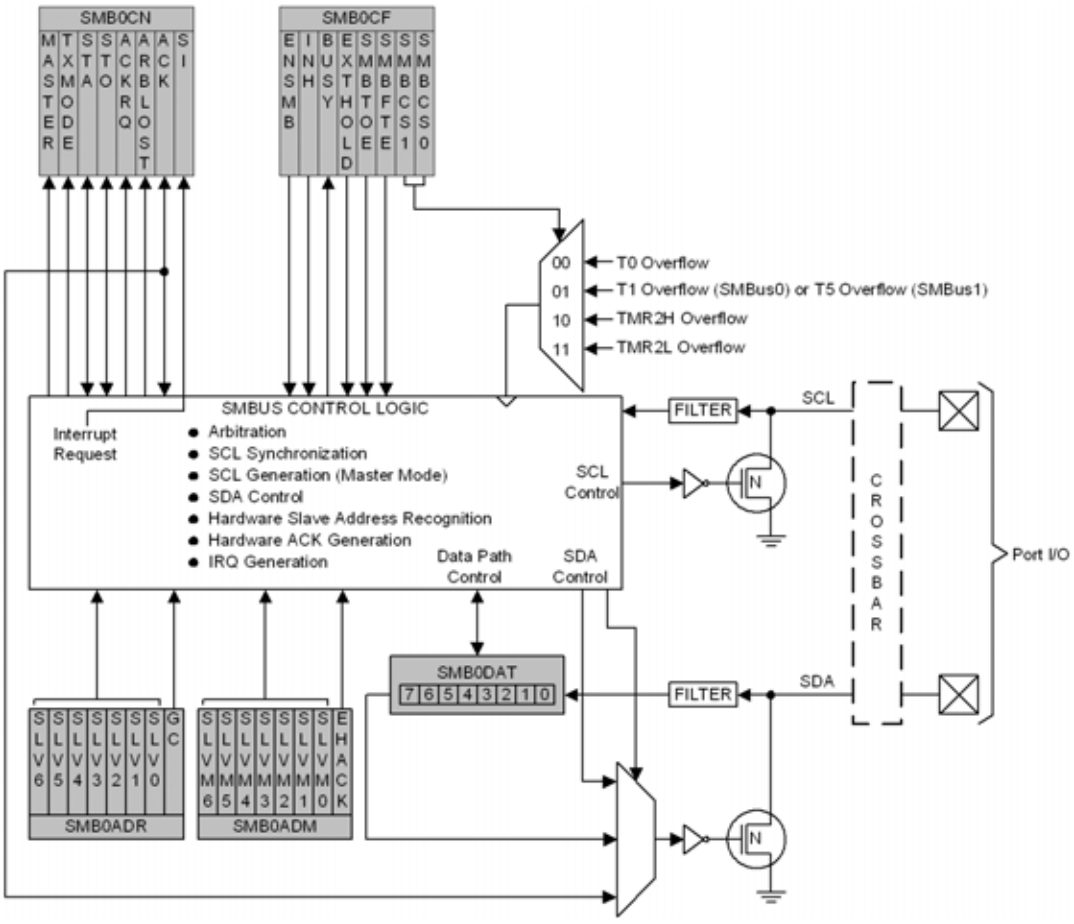

Figure 21.1. SMBus Block Diagram

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 21.3 illustrates a typical SMBus transaction.

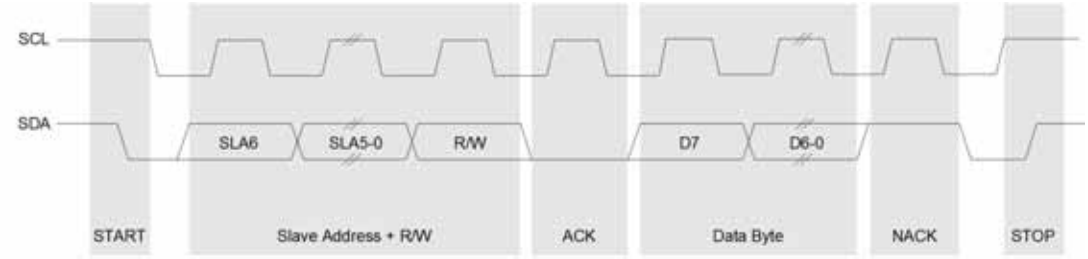

Figure 21.3. SMBus Transaction

21.3.1. Transmitter Vs. Receiver

On the SMBus communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

21.3.2. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see Section “21.3.5. SCL High (SMBus Free) Timeout” on page 169). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

21.3.3. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I2C, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

21.3.4. SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

For the SMBus0 interface, Timer 3 is used to implement SCL low timeouts. Timer 4 is used on the SMBus1 interface for SCL low timeouts. The SCL low timeout feature is enabled by setting the SMBnTOE bit in SMBnCF. The associated timer is forced to reload when SCL is high, and allowed to count when SCL is

SFR Definition 21.5. SMB1CN: SMBus Control

Bit	7	6	5	4	3	2	1	0
Name	MASTER1	TXMODE1	STA1	STO1	ACKRQ1	ARBLOST1	ACK1	SI1
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; SFR Page = F; Bit-Addressable

Bit	Name	Description	Read	Write
7	MASTER1	**SMBus1 Master/Slave Indicator.** This read-only bit indicates when the SMBus1 is operating as a master.	0: SMBus1 operating in slave mode. 1: SMBus1 operating in master mode.	N/A
6	TXMODE1	**SMBus1 Transmit Mode Indicator.** This read-only bit indicates when the SMBus1 is operating as a transmitter.	0: SMBus1 in Receiver Mode. 1: SMBus1 in Transmitter Mode.	N/A
5	STA1	**SMBus1 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	STO1	**SMBus1 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	ACKRQ1	**SMBus1 Acknowledge Request.**	0: No ACK requested 1: ACK requested	N/A
2	ARBLOST1	**SMBus1 Arbitration Lost Indicator.**	0: No arbitration error. 1: Arbitration Lost	N/A
1	ACK1	**SMBus1 Acknowledge.**	0: NACK received. 1: ACK received.	0: Send NACK 1: Send ACK
0	SI1	**SMBus1 Interrupt Flag.** This bit is set by hardware under the conditions listed in Table 15.3. SI1 must be cleared by software. While SI1 is set, SCL1 is held low and the SMBus1 is stalled.	0: No interrupt pending 1: Interrupt Pending	0: Clear interrupt, and initiate next state machine event. 1: Force interrupt.

SFR Definition 21.9. SMB1ADM: SMBus1 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM1[6:0]							EHACK1
Type	R/W							R/W
Reset	1	1	1	1	1	1	1	0

SFR Address = 0xCE; SFR Page = F

Bit	Name	Function
7:1	SLVM1[6:0]	**SMBus1 Slave Address Mask.** Defines which bits of register SMB1ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM1[6:0] enables comparisons with the corresponding bit in SLV1[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
0	EHACK1	**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.

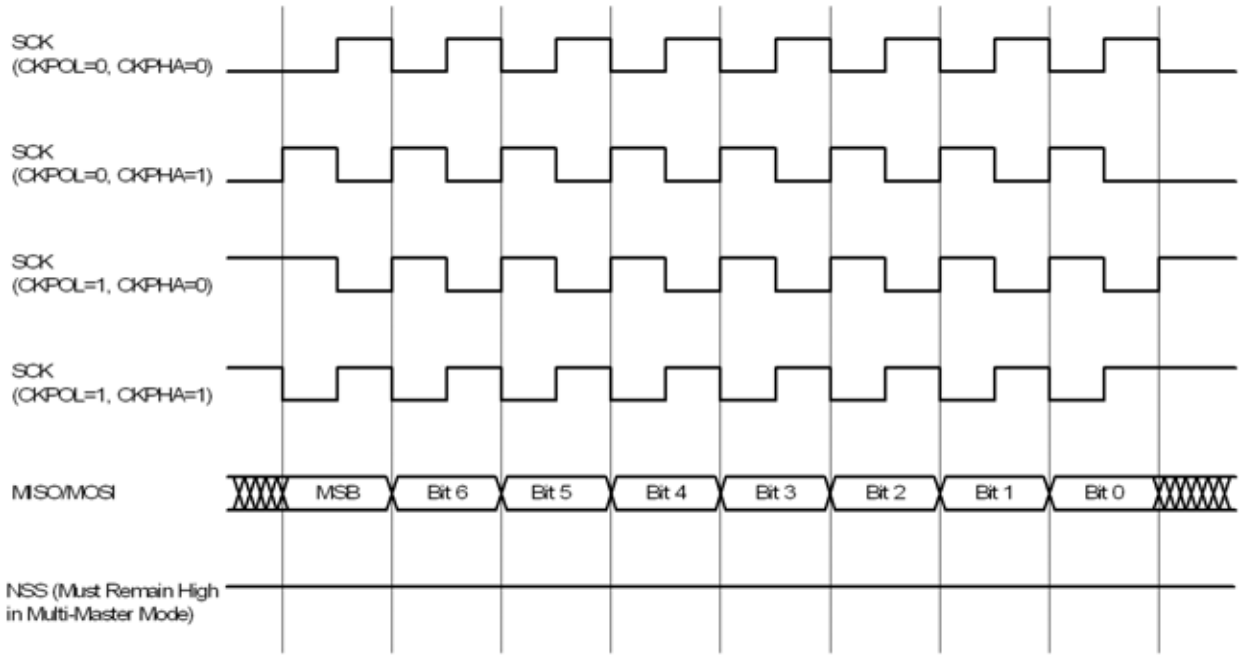

Figure 24.5. Master Mode Data/Clock Timing

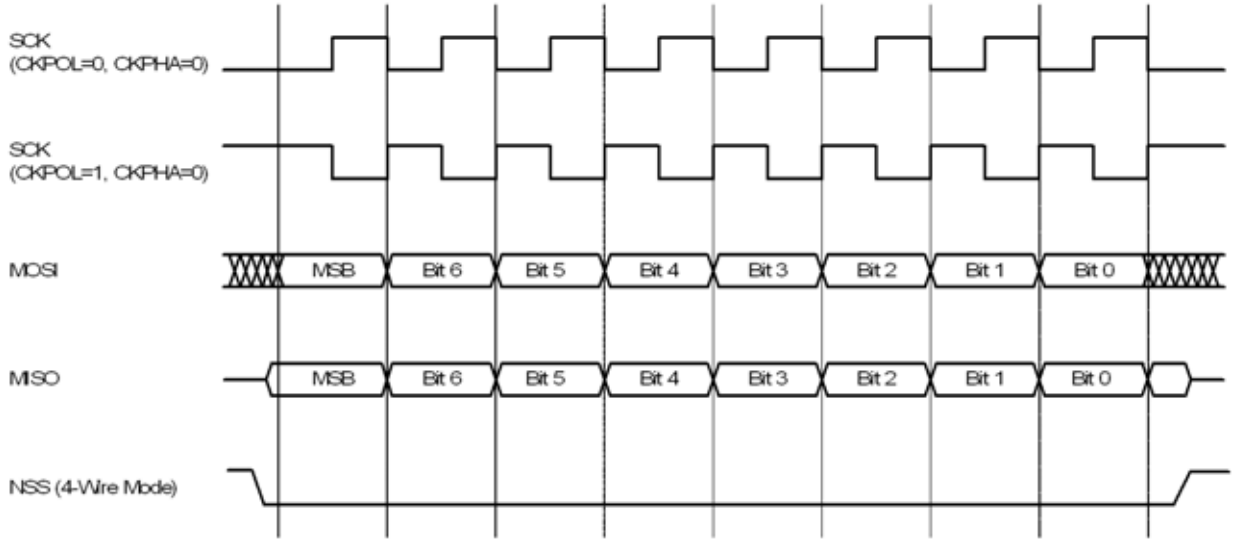

Figure 24.6. Slave Mode Data/Clock Timing (CKPHA = 0)

SFR Definition 25.2. CKCON1: Clock Control 1

Bit	7	6	5	4	3	2	1	0
Name					T5MH	T5ML	T4MH	T4ML
Type	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE4; SFR Page = F

Bit	Name	Function
7:4	Unused	Read = 0000b; Write = don't care
3	T5MH	**Timer 5 High Byte Clock Select.** Selects the clock supplied to the Timer 5 high byte (split 8-bit timer mode only). 0: Timer 5 high byte uses the clock defined by the T5XCLK bit in TMR5CN. 1: Timer 5 high byte uses the system clock.
2	T5ML	**Timer 5 Low Byte Clock Select.** Selects the clock supplied to Timer 5. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode. 0: Timer 5 low byte uses the clock defined by the T5XCLK bit in TMR5CN. 1: Timer 5 low byte uses the system clock.
1	T4MH	**Timer 4 High Byte Clock Select.** Selects the clock supplied to the Timer 4 high byte (split 8-bit timer mode only). 0: Timer 4 high byte uses the clock defined by the T4XCLK bit in TMR4CN. 1: Timer 4 high byte uses the system clock.
0	T4ML	**Timer 4 Low Byte Clock Select.** Selects the clock supplied to Timer 4. If Timer 4 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer. 0: Timer 4 low byte uses the clock defined by the T4XCLK bit in TMR4CN. 1: Timer 4 low byte uses the system clock.

SFR Definition 25.9. TMR2CN: Timer 2 Control

Bit	7	6	5	4	3	2	1	0
Name	TF2H	TF2L	TF2LEN	TF2CEN	T2SPLIT	TR2	T2CSS	T2XCLK
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 = 0xC8; SFR Page = 0; 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	TF2CEN	**Timer 2 Low-Frequency Oscillator Capture Enable.** When set to 1, this bit enables Timer 2 Low-Frequency Oscillator Capture Mode. If TF2CEN is set and Timer 2 interrupts are enabled, an interrupt will be generated on a falling edge of the low-frequency oscillator output, and the current 16-bit timer value in TMR2H:TMR2L will be copied to TMR2RLH:TMR2RLL.
3	T2SPLIT	**Timer 2 Split Mode Enable.** When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload.
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	T2CSS	**Timer 2 Capture Source Select.** This bit selects the source of a capture event when bit T2CE is set to 1. 0: Reserved. 1: Capture source is falling edge of Low-Frequency Oscillator.
0	T2XCLK	**Timer 2 External Clock Select.** This bit selects the external clock source for Timer 2. 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).

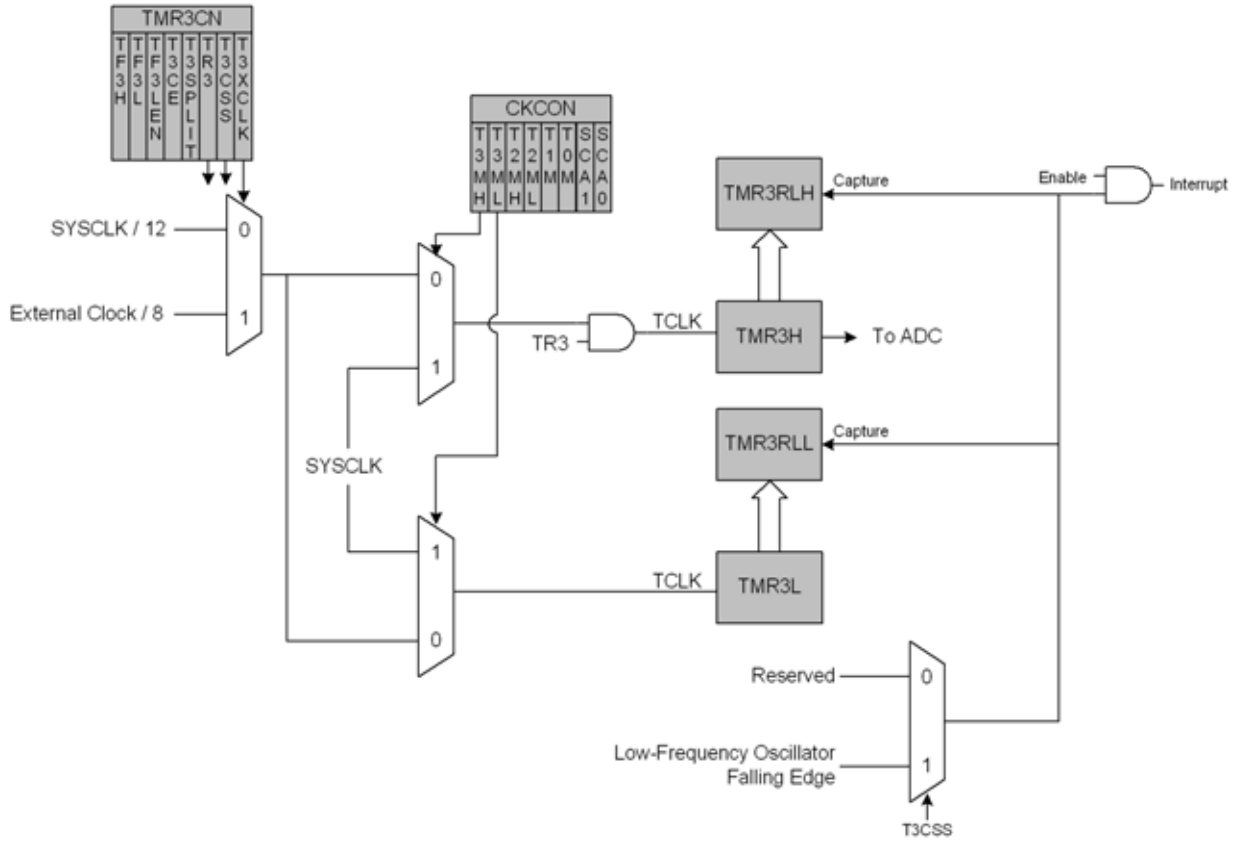

Figure 25.11. Timer 3 Capture Mode (T3SPLIT = 0)

SFR Definition 25.20. TMR4RLL: Timer 4 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x92; SFR Page = F

Bit	Name	Function
7:0	TMR4RLL[7:0]	**Timer 4 Reload Register Low Byte.** TMR4RLL holds the low byte of the reload value for Timer 4.

SFR Definition 25.21. TMR4RLH: Timer 4 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x93; SFR Page = F

Bit	Name	Function
7:0	TMR4RLH[7:0]	**Timer 4 Reload Register High Byte.** TMR4RLH holds the high byte of the reload value for Timer 4.

SFR Definition 25.22. TMR4L: Timer 4 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x94; SFR Page = F

Bit	Name	Function
7:0	TMR4L[7:0]	**Timer 4 Low Byte.** In 16-bit mode, the TMR4L register contains the low byte of the 16-bit Timer 4. In 8-bit mode, TMR4L contains the 8-bit low byte timer value.

26.3.4. Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module's associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined by Equation 26.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

Note: A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

Equation 26.1. Square Wave Frequency Output

Where F_{PCA} is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register, PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register. Note that the MATn bit should normally be set to 0 in this mode. If the MATn bit is set to 1, the CCFn flag for the channel will be set when the 16-bit PCA0 counter and the 16-bit capture/compare register for the channel are equal.

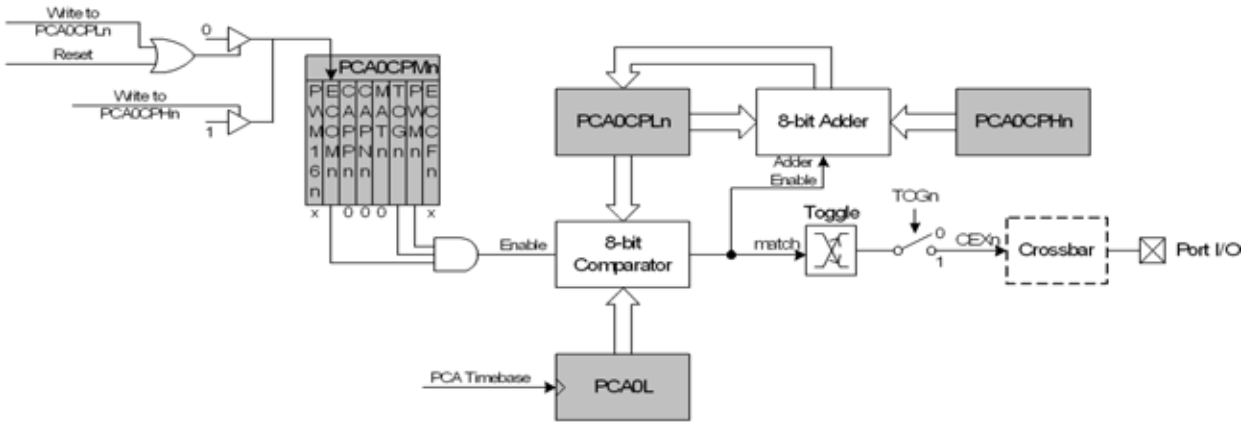

Figure 26.7. PCA Frequency Output Mode

27. C2 Interface

C8051F388/9/A/B devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow Flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. 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 Flash programming 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]	**C2 Address.** The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands.	
		Address	**Description**
		0x00	Selects the Device ID register for Data Read instructions
		0x01	Selects the Revision ID register for Data Read instructions
		0x02	Selects the C2 Flash Programming Control register for Data Read/Write instructions
		0xAD	Selects the C2 Flash Programming Data register for Data Read/Write instructions