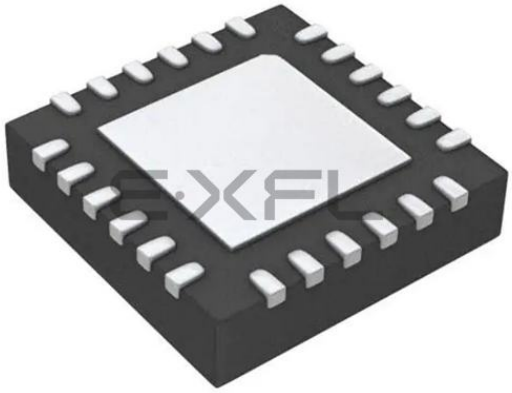

Welcome to E-XFL.COM

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	25MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	21
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	24-WFQFN Exposed Pad
Supplier Device Package	24-QFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f339-gm

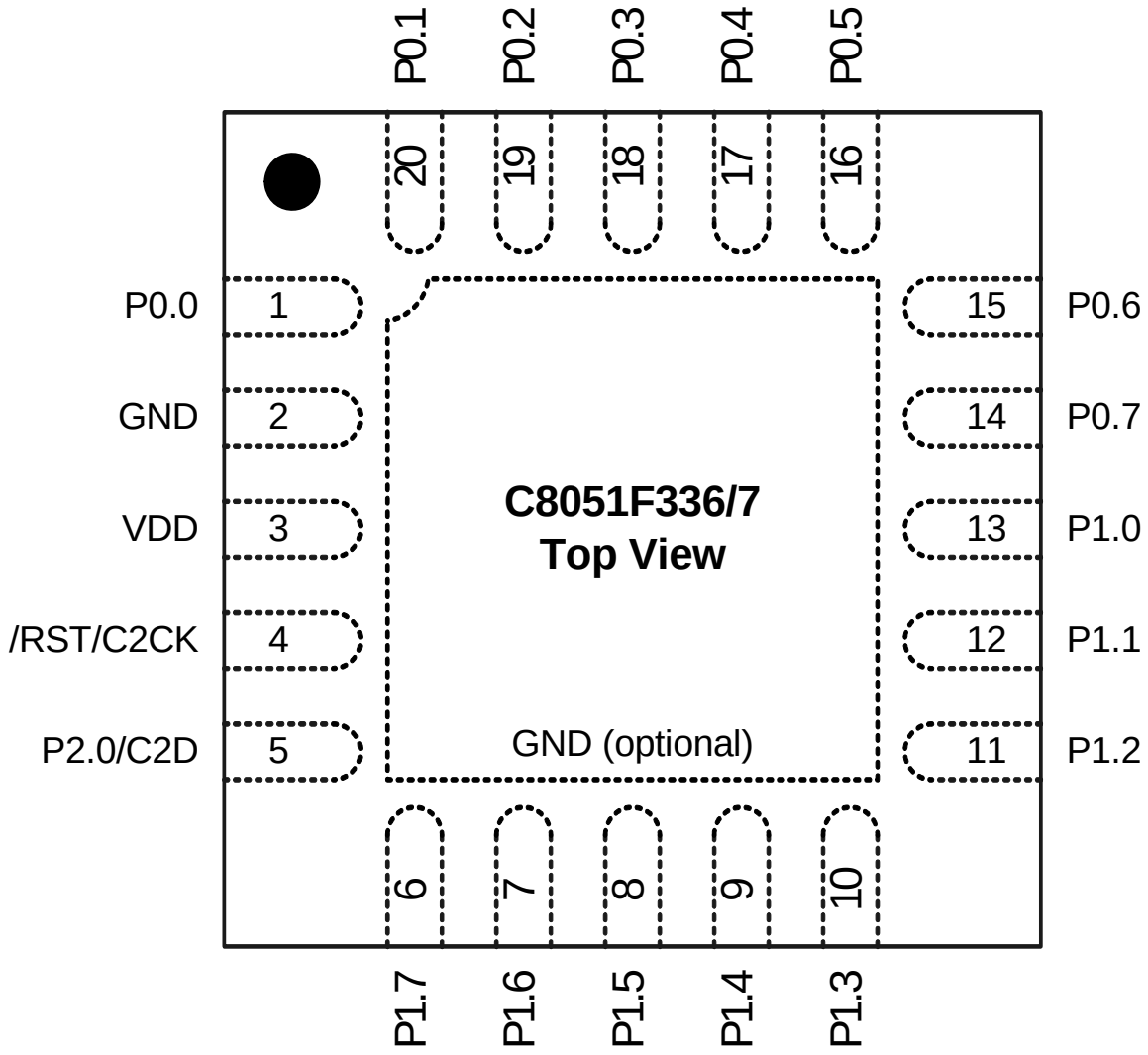

Figure 3.1. QFN-20 Pinout Diagram (Top View)

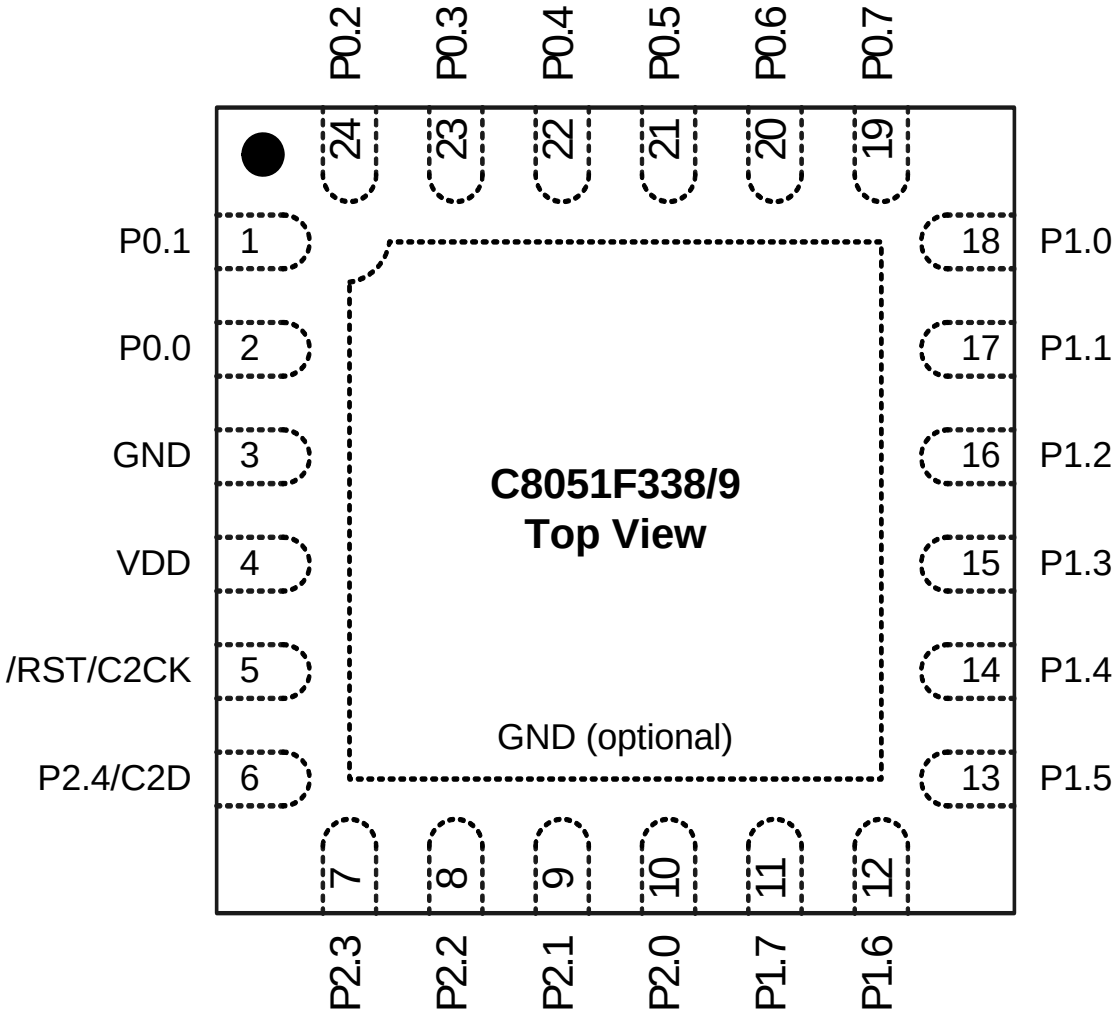

Figure 3.2. QFN-24 Pinout Diagram (Top View)

Table 6.3. Port I/O DC Electrical Characteristics

V_{DD} = 2.7 to 3.6 V, –40 to +85 °C unless otherwise specified.

Parameters	Conditions	Min	Typ	Max	Units
Output High Voltage	I_{OH} = –3 mA, Port I/O push-pull	V_{DD} – 0.7	—	—	V
	I_{OH} = –10 µA, Port I/O push-pull	V_{DD} – 0.1	—	—	V
	I_{OH} = –10 mA, Port I/O push-pull	—	V_{DD} – 0.8	—	V
Output Low Voltage	I_{OL} = 8.5 mA	—	—	0.6	V
	I_{OL} = 10 µA	—	—	0.1	V
	I_{OL} = 25 mA	—	1.0	—	V
Input High Voltage		2.0	—	—	V
Input Low Voltage		—	—	0.8	V
Input Leakage Current	Weak Pullup Off	—	—	±1	µA
	Weak Pullup On, V_{IN} = 0 V	—	50	100	µA

7.3.1. Window Detector In Single-Ended Mode

Figure 7.4 shows two example window comparisons for right-justified, single-ended data, with ADC0LTH:ADC0LTL = 0x0080 (128d) and ADC0GTH:ADC0GTL = 0x0040 (64d). In single-ended mode, the input voltage can range from '0' to VREF x (1023/1024) with respect to GND, and is represented by a 10-bit unsigned integer value. In the left example, an AD0WINT interrupt will be generated if the ADC0 conversion word (ADC0H:ADC0L) is within the range defined by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL (if 0x0040 < ADC0H:ADC0L < 0x0080). In the right example, and AD0WINT interrupt will be generated if the ADC0 conversion word is outside of the range defined by the ADC0GT and ADC0LT registers (if ADC0H:ADC0L < 0x0040 or ADC0H:ADC0L > 0x0080). Figure 7.5 shows an example using left-justified data with the same comparison values.

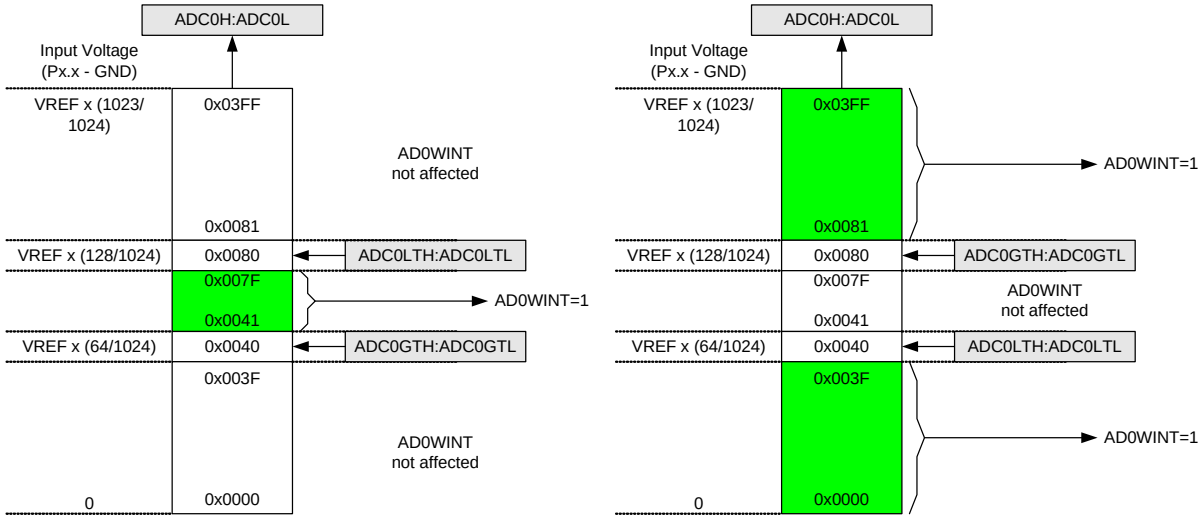

Figure 7.4. ADC Window Compare Example: Right-Justified Single-Ended Data

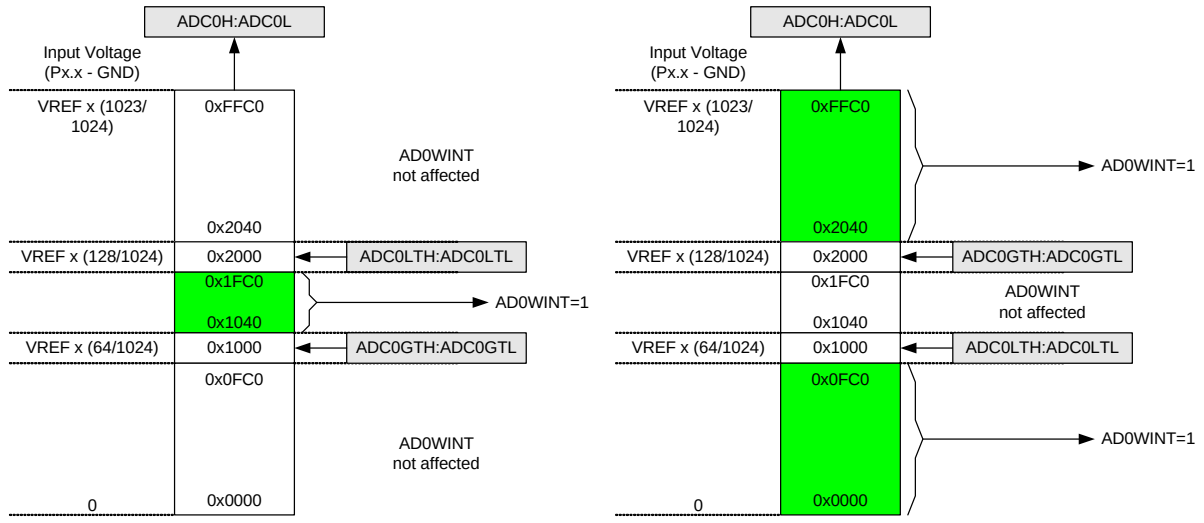

Figure 7.5. ADC Window Compare Example: Left-Justified Single-Ended Data

SFR Definition 11.2. CPT0MD: Comparator0 Mode Selection

Bit	7	6	5	4	3	2	1	0
Name			CP0RIE	CP0FIE			CP0MD[1:0]	
Type	R	R	R/W	R/W	R	R	R/W	
Reset	0	0	0	0	0	0	1	0

SFR Address = 0x9D

Bit	Name	Function
7:6	Unused	Unused. Read = 00b, Write = Don't Care.
5	CP0RIE	**Comparator0 Rising-Edge Interrupt Enable.** 0: Comparator0 Rising-edge interrupt disabled. 1: Comparator0 Rising-edge interrupt enabled.
4	CP0FIE	**Comparator0 Falling-Edge Interrupt Enable.** 0: Comparator0 Falling-edge interrupt disabled. 1: Comparator0 Falling-edge interrupt enabled.
3:2	Unused	Unused. Read = 00b, Write = don't care.
1:0	CP0MD[1:0]	**Comparator0 Mode Select.** These bits affect the response time and power consumption for Comparator0. 00: Mode 0 (Fastest Response Time, Highest Power Consumption) 01: Mode 1 10: Mode 2 11: Mode 3 (Slowest Response Time, Lowest Power Consumption)

12.2. CIP-51 Register Descriptions

Following are descriptions of SFRs related to the operation of the CIP-51 System Controller. Reserved bits should always be written to the value indicated in the SFR description. Future product versions may use these bits to implement new features in which case the reset value of the bit will be the indicated value, selecting the feature's default state. Detailed descriptions of the remaining SFRs are included in the sections of the data sheet associated with their corresponding system function.

SFR Definition 12.1. DPL: Data Pointer Low Byte

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

SFR Address = 0x82

Bit	Name	Function
7:0	DPL[7:0]	**Data Pointer Low.** The DPL register is the low byte of the 16-bit DPTR.

SFR Definition 12.2. DPH: Data Pointer High Byte

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

SFR Address = 0x83

Bit	Name	Function
7:0	DPH[7:0]	**Data Pointer High.** The DPH register is the high byte of the 16-bit DPTR.

space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFRs. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the upper 128 bytes of data memory. Figure 13.1 illustrates the data memory organization of the C8051F336/7/8/9.

13.2.1.1. General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in SFR Definition 12.6). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

13.2.1.2. Bit Addressable Locations

In addition to direct access to data memory organized as bytes, the sixteen data memory locations at 0x20 through 0x2F are also accessible as 128 individually addressable bits. Each bit has a bit address from 0x00 to 0x7F. Bit 0 of the byte at 0x20 has bit address 0x00 while bit7 of the byte at 0x20 has bit address 0x07. Bit 7 of the byte at 0x2F has bit address 0x7F. A bit access is distinguished from a full byte access by the type of instruction used (bit source or destination operands as opposed to a byte source or destination).

The MCS-51™ assembly language allows an alternate notation for bit addressing of the form XX.B where XX is the byte address and B is the bit position within the byte. For example, the instruction:

```
MOV    C, 22.3h
```

moves the Boolean value at 0x13 (bit 3 of the byte at location 0x22) into the Carry flag.

13.2.1.3. Stack

A programmer's stack can be located anywhere in the 256-byte data memory. The stack area is designated using the Stack Pointer (SP) SFR. The SP will point to the last location used. The next value pushed on the stack is placed at SP+1 and then SP is incremented. A reset initializes the stack pointer to location 0x07. Therefore, the first value pushed on the stack is placed at location 0x08, which is also the first register (R0) of register bank 1. Thus, if more than one register bank is to be used, the SP should be initialized to a location in the data memory not being used for data storage. The stack depth can extend up to 256 bytes.

13.2.2. External RAM

There are 512 bytes of on-chip RAM mapped into the external data memory space. All of these address locations may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using MOVX indirect addressing mode. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN as shown in SFR Definition 13.1). Note: the MOVX instruction is also used for writes to the Flash memory. See **Section “16. Flash Memory” on page 91** for details. The MOVX instruction accesses XRAM by default.

For a 16-bit MOVX operation (@DPTR), the upper 7 bits of the 16-bit external data memory address word are "don't cares". As a result, the 512-byte RAM is mapped modulo style over the entire 64 k external data memory address range. For example, the XRAM byte at address 0x0000 is shadowed at addresses 0x0200, 0x0400, 0x0600, 0x0800, etc. This is a useful feature when performing a linear memory fill, as the address pointer doesn't have to be reset when reaching the RAM block boundary.

Table 14.2. Special Function Registers (Continued)

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

Register	Address	Description	Page
REF0CN	0xD1	Voltage Reference Control	57
RSTSRC	0xEF	Reset Source Configuration/Status	105
SBUF0	0x99	UART0 Data Buffer	165
SCON0	0x98	UART0 Control	164
SMB0ADM	0xE7	SMBus Slave Address Mask	149
SMB0ADR	0xD7	SMBus Slave Address	148
SMB0CF	0xC1	SMBus Configuration	144
SMB0CN	0xC0	SMBus Control	146
SMB0DAT	0xC2	SMBus Data	150
SP	0x81	Stack Pointer	72
SPI0CFG	0xA1	SPI Configuration	174
SPI0CKR	0xA2	SPI Clock Rate Control	176
SPI0CN	0xF8	SPI Control	175
SPI0DAT	0xA3	SPI Data	176
TCON	0x88	Timer/Counter Control	186
TH0	0x8C	Timer/Counter 0 High	189
TH1	0x8D	Timer/Counter 1 High	189
TL0	0x8A	Timer/Counter 0 Low	188
TL1	0x8B	Timer/Counter 1 Low	188
TMOD	0x89	Timer/Counter Mode	187
TMR2CN	0xC8	Timer/Counter 2 Control	193
TMR2H	0xCD	Timer/Counter 2 High	195
TMR2L	0xCC	Timer/Counter 2 Low	194
TMR2RLH	0xCB	Timer/Counter 2 Reload High	194
TMR2RLL	0xCA	Timer/Counter 2 Reload Low	194
TMR3CN	0x91	Timer/Counter 3Control	199
TMR3H	0x95	Timer/Counter 3 High	201
TMR3L	0x94	Timer/Counter 3Low	200
TMR3RLH	0x93	Timer/Counter 3 Reload High	200
TMR3RLL	0x92	Timer/Counter 3 Reload Low	200
VDM0CN	0xFF	V_{DD} Monitor Control	103
XBR0	0xE1	Port I/O Crossbar Control 0	127
XBR1	0xE2	Port I/O Crossbar Control 1	128

16.1.3. Flash Write Procedure

Flash bytes are programmed by software with the following sequence:

1. Disable interrupts (recommended).
2. Erase the 512-byte Flash page containing the target location, as described in **Section 16.1.2**.
3. Set the PSWE bit (register PSCTL).
4. Clear the PSEE bit (register PSCTL).
5. Write the first key code to FLKEY: 0xA5.
6. Write the second key code to FLKEY: 0xF1.
7. Using the MOVX instruction, write a single data byte to the desired location within the 512-byte sector.
8. Clear the PSWE bit.

Steps 5–7 must be repeated for each byte to be written. After Flash writes are complete, PSWE should be cleared so that MOVX instructions do not target program memory.

16.2. Non-volatile Data Storage

The Flash memory can be used for non-volatile data storage as well as program code. This allows data such as calibration coefficients to be calculated and stored at run time. Data is written using the MOVX write instruction and read using the MOVC instruction. Note: MOVX read instructions always target XRAM.

16.4. Flash Write and Erase Guidelines

Any system which contains routines which write or erase Flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of V_{DD}, system clock frequency, or temperature. This accidental execution of Flash modifying code can result in alteration of Flash memory contents causing a system failure that is only recoverable by re-Flashing the code in the device.

The following guidelines are recommended for any system which contains routines which write or erase Flash from code.

16.4.1. V_{DD} Maintenance and the V_{DD} monitor

1. If the system power supply is subject to voltage or current "spikes," add sufficient transient protection devices to the power supply to ensure that the supply voltages listed in the Absolute Maximum Ratings table are not exceeded.
2. Make certain that the minimum V_{DD} rise time specification of 1 ms is met. If the system cannot meet this rise time specification, then add an external V_{DD} brownout circuit to the $\overline{RST}$ pin of the device that holds the device in reset until V_{DD} reaches 2.7 V and re-asserts $\overline{RST}$ if V_{DD} drops below 2.7 V.
3. Enable the on-chip V_{DD} monitor and enable the V_{DD} monitor as a reset source as early in code as possible. This should be the first set of instructions executed after the Reset Vector. For 'C'-based systems, this will involve modifying the startup code added by the 'C' compiler. See your compiler documentation for more details. Make certain that there are no delays in software between enabling the V_{DD} monitor and enabling the V_{DD} monitor as a reset source. Code examples showing this can be found in “AN201: Writing to Flash from Firmware", available from the Silicon Laboratories web site.
4. As an added precaution, explicitly enable the V_{DD} monitor and enable the V_{DD} monitor as a reset source inside the functions that write and erase Flash memory. The V_{DD} monitor enable instructions should be placed just after the instruction to set PSWE to a '1', but before the Flash write or erase operation instruction.
5. Make certain that all writes to the RSTSRC (Reset Sources) register use direct assignment operators and explicitly DO NOT use the bit-wise operators (such as AND or OR). For example, "RSTSRC = 0x02" is correct. "RSTSRC |= 0x02" is incorrect.
6. Make certain that all writes to the RSTSRC register explicitly set the PORSF bit to a '1'. Areas to check are initialization code which enables other reset sources, such as the Missing Clock Detector or Comparator, for example, and instructions which force a Software Reset. A global search on "RSTSRC" can quickly verify this.

16.4.2. PSWE Maintenance

7. Reduce the number of places in code where the PSWE bit (b0 in PSCTL) is set to a '1'. There should be exactly one routine in code that sets PSWE to a '1' to write Flash bytes and one routine in code that sets PSWE and PSEE both to a '1' to erase Flash pages.
8. Minimize the number of variable accesses while PSWE is set to a '1'. Handle pointer address updates and loop variable maintenance outside the "PSWE = 1;... PSWE = 0;" area. Code examples showing this can be found in *AN201, "Writing to Flash from Firmware"*, available from the Silicon Laboratories web site.
9. Disable interrupts prior to setting PSWE to a '1' and leave them disabled until after PSWE has been reset to '0'. Any interrupts posted during the Flash write or erase operation will be serviced in priority order after the Flash operation has been completed and interrupts have been re-enabled by software.
10. Make certain that the Flash write and erase pointer variables are not located in XRAM. See your compiler documentation for instructions regarding how to explicitly locate variables in different memory

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

When a power-down transition or power irregularity causes V_{DD} to drop below V_{RST}, the power supply monitor will drive the $\overline{RST}$ pin low and hold the CIP-51 in a reset state (see Figure 17.2). When V_{DD} returns to a level above V_{RST}, the CIP-51 will be released from the reset state. Note that even though internal data memory contents are not altered by the power-fail reset, it is impossible to determine if V_{DD} dropped below the level required for data retention. If the PORSF flag reads '1', the data may no longer be valid. The V_{DD} monitor is enabled after power-on resets. Its defined state (enabled/disabled) is not altered by any other reset source. For example, if the V_{DD} monitor is disabled by code and a software reset is performed, the V_{DD} monitor will still be disabled after the reset.

Important Note: If the V_{DD} monitor is being turned on from a disabled state, it should be enabled before it is selected as a reset source. Selecting the V_{DD} monitor as a reset source before it is enabled and stabilized may cause a system reset. In some applications, this reset may be undesirable. If this is not desirable in the application, a delay should be introduced between enabling the monitor and selecting it as a reset source. The procedure for enabling the V_{DD} monitor and configuring it as a reset source from a disabled state is shown below:

1. Enable the V_{DD} monitor (VDMEN bit in VDM0CN = '1').
2. If necessary, wait for the V_{DD} monitor to stabilize.
3. Select the V_{DD} monitor as a reset source (PORSF bit in RSTSRC = '1').

See Figure 17.2 for V_{DD} monitor timing; note that the power-on-reset delay is not incurred after a V_{DD} monitor reset. See **Section "6. Electrical Characteristics" on page 27** for complete electrical characteristics of the V_{DD} monitor.

20.2.3. Assigning Port I/O Pins to External Event Trigger Functions

External event trigger functions can be used to trigger an interrupt or wake the device from a low power mode when a transition occurs on a digital I/O pin. The event trigger functions do not require dedicated pins and will function on both GPIO pins (PnSKIP = ’1’) and pins in use by the Crossbar (PnSKIP = ‘0’). External event trigger functions cannot be used on pins configured for analog I/O. Table 20.3 shows all available external event trigger functions.

Table 20.3. Port I/O Assignment for External Event Trigger Functions

Event Trigger Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
External Interrupt 0	P0.0 - P0.7	IT01CF
External Interrupt 1	P0.0 - P0.7	IT01CF
Port Match	P0.0 - P1.7	P0MASK, P0MAT P1MASK, P1MAT

23.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

23.1.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to slave devices. It is used to serially transfer data from the master to the slave. This signal is an output when SPI0 is operating as a master and an input when SPI0 is operating as a slave. Data is transferred most-significant bit first. When configured as a master, MOSI is driven by the MSB of the shift register in both 3- and 4-wire mode.

23.1.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to the master device. It is used to serially transfer data from the slave to the master. This signal is an input when SPI0 is operating as a master and an output when SPI0 is operating as a slave. Data is transferred most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled and when the SPI operates in 4-wire mode as a slave that is not selected. When acting as a slave in 3-wire mode, MISO is always driven by the MSB of the shift register.

23.1.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. SPI0 generates this signal when operating as a master. The SCK signal is ignored by a SPI slave when the slave is not selected (NSS = 1) in 4-wire slave mode.

23.1.4. Slave Select (NSS)

The function of the slave-select (NSS) signal is dependent on the setting of the NSSMD1 and NSSMD0 bits in the SPI0CN register. There are three possible modes that can be selected with these bits:

1. NSSMD[1:0] = 00: 3-Wire Master or 3-Wire Slave Mode: SPI0 operates in 3-wire mode, and NSS is disabled. When operating as a slave device, SPI0 is always selected in 3-wire mode. Since no select signal is present, SPI0 must be the only slave on the bus in 3-wire mode. This is intended for point-to-point communication between a master and one slave.
2. NSSMD[1:0] = 01: 4-Wire Slave or Multi-Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an input. When operating as a slave, NSS selects the SPI0 device. When operating as a master, a 1-to-0 transition of the NSS signal disables the master function of SPI0 so that multiple master devices can be used on the same SPI bus.
3. NSSMD[1:0] = 1x: 4-Wire Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an output. The setting of NSSMD0 determines what logic level the NSS pin will output. This configuration should only be used when operating SPI0 as a master device.

See Figure 23.2, Figure 23.3, and Figure 23.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device. See Section “20. Port Input/Output” on page 119 for general purpose port I/O and crossbar information.

SFR Definition 24.4. TL0: Timer 0 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8A

Bit	Name	Function
7:0	TL0[7:0]	**Timer 0 Low Byte.** The TL0 register is the low byte of the 16-bit Timer 0.

SFR Definition 24.5. TL1: Timer 1 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8B

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

24.2.2. 8-bit Timers with Auto-Reload

When T2SPLIT is set, Timer 2 operates as two 8-bit timers (TMR2H and TMR2L). Both 8-bit timers operate in auto-reload mode as shown in Figure 24.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L 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 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bit (T2XCLK in TMR2CN), as follows:

T2MH	T2XCLK	TMR2H Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

T2ML	T2XCLK	TMR2L Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LEN (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LEN is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

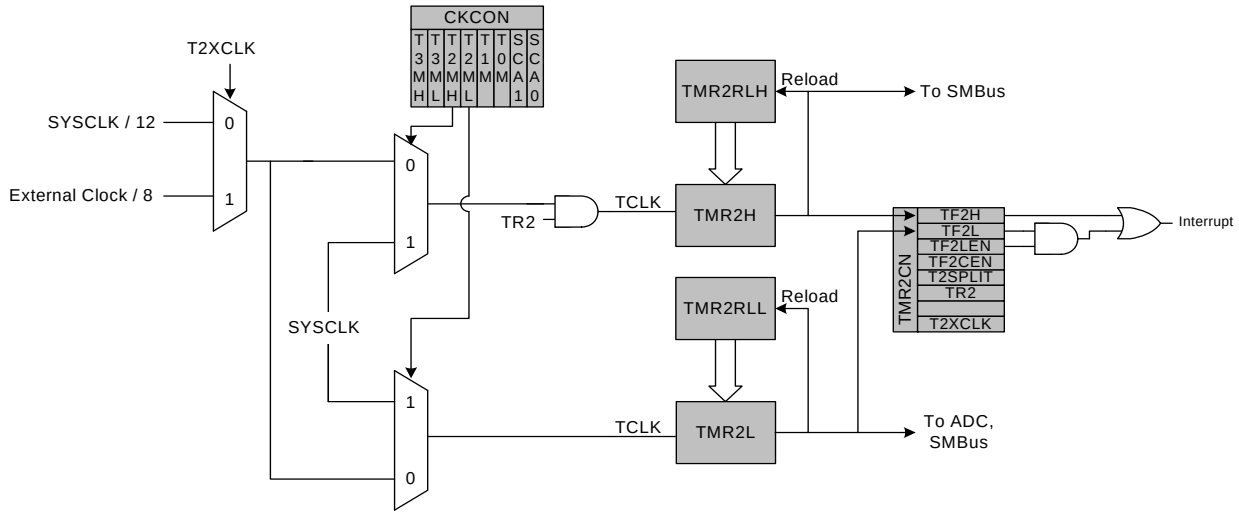

Figure 24.5. Timer 2 8-Bit Mode Block Diagram

SFR Definition 24.8. TMR2CN: Timer 2 Control

Bit	7	6	5	4	3	2	1	0
Name	TF2H	TF2L	TF2LEN	TF2CEN	T2SPLIT	TR2		T2XCLK
Type	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC8; Bit-Addressable

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

24.3.2. 8-bit Timers with Auto-Reload

When T3SPLIT is set, Timer 3 operates as two 8-bit timers (TMR3H and TMR3L). Both 8-bit timers operate in auto-reload mode as shown in Figure 24.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, the external oscillator clock source divided by 8, or the internal Low-frequency Oscillator. The Timer 3 Clock Select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK[1:0] in TMR3CN), as follows:

T3MH	T3XCLK[1:0]	TMR3H Clock Source
0	00	SYSCLK / 12
0	01	External Clock / 8
0	10	Reserved
0	11	Internal LFO
1	X	SYSCLK

T3ML	T3XCLK[1:0]	TMR3L Clock Source
0	00	SYSCLK / 12
0	01	External Clock / 8
0	10	Reserved
0	11	Internal LFO
1	X	SYSCLK

The TF3H bit is set when TMR3H overflows from 0xFF to 0x00; the TF3L bit is set when TMR3L overflows from 0xFF to 0x00. When Timer 3 interrupts are enabled, an interrupt is generated each time TMR3H overflows. If Timer 3 interrupts are enabled and TF3LEN (TMR3CN.5) is set, an interrupt is generated each time either TMR3L or TMR3H overflows. When TF3LEN is enabled, software must check the TF3H and TF3L flags to determine the source of the Timer 3 interrupt. The TF3H and TF3L interrupt flags are not cleared by hardware and must be manually cleared by software.

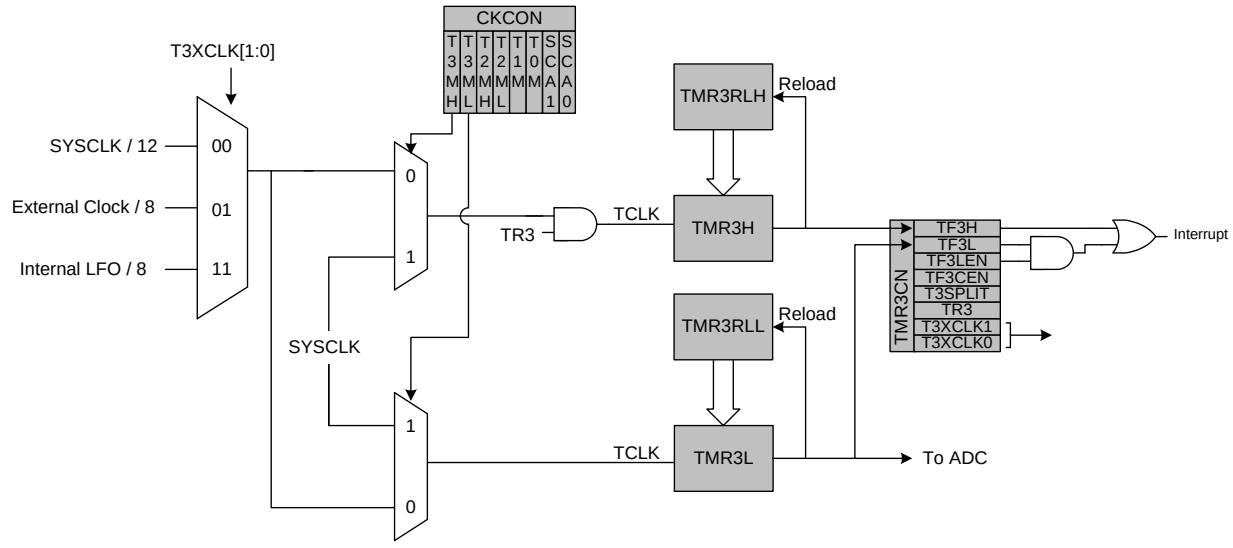

Figure 24.8. Timer 3 8-Bit Mode Block Diagram

SFR Definition 24.13. TMR3CN: Timer 3 Control

Bit	7	6	5	4	3	2	1	0
Name	TF3H	TF3L	TF3LEN	TF3CEN	T3SPLIT	TR3	T3XCLK[1:0]	
Type	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 = 0x91

Bit	Name	Function
7	TF3H	**Timer 3 High Byte Overflow Flag.** Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF3L	**Timer 3 Low Byte Overflow Flag.** Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. TF3L will be set when the low byte overflows regardless of the Timer 3 mode. This bit is not automatically cleared by hardware.
5	TF3LEN	**Timer 3 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows.
4	TF3CEN	**Timer 3 Low-Frequency Oscillator Capture Enable.** When set to 1, this bit enables Timer 3 Low-Frequency Oscillator Capture Mode. If TF3CEN is set and Timer 3 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 TMR3H:TMR3L will be copied to TMR3RLH:TMR3RLL.
3	T3SPLIT	**Timer 3 Split Mode Enable.** When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload. 0: Timer 3 operates in 16-bit auto-reload mode. 1: Timer 3 operates as two 8-bit auto-reload timers.
2	TR3	**Timer 3 Run Control.** Timer 3 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in split mode.
1:0	T3XCLK[1:0]	**Timer 3 External Clock Select.** This bit selects the “external” clock source for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the external clock and the system clock for either timer. 00: System clock divided by 12. 01: External clock divided by 8 (synchronized with SYSCLK when not in suspend). 10: Reserved. 11: Internal LFO/8 (synchronized with SYSCLK when not in suspend).

25. Programmable Counter Array

The Programmable Counter Array (PCA0) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. The PCA consists of a dedicated 16-bit counter/timer and three 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between six sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section "25.3. Capture/Compare Modules" on page 205). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 25.1

Important Note: The PCA Module 2 may be used as a watchdog timer (WDT), and is enabled in this mode following a system reset. **Access to certain PCA registers is restricted while WDT mode is enabled**. See Section 25.4 for details.

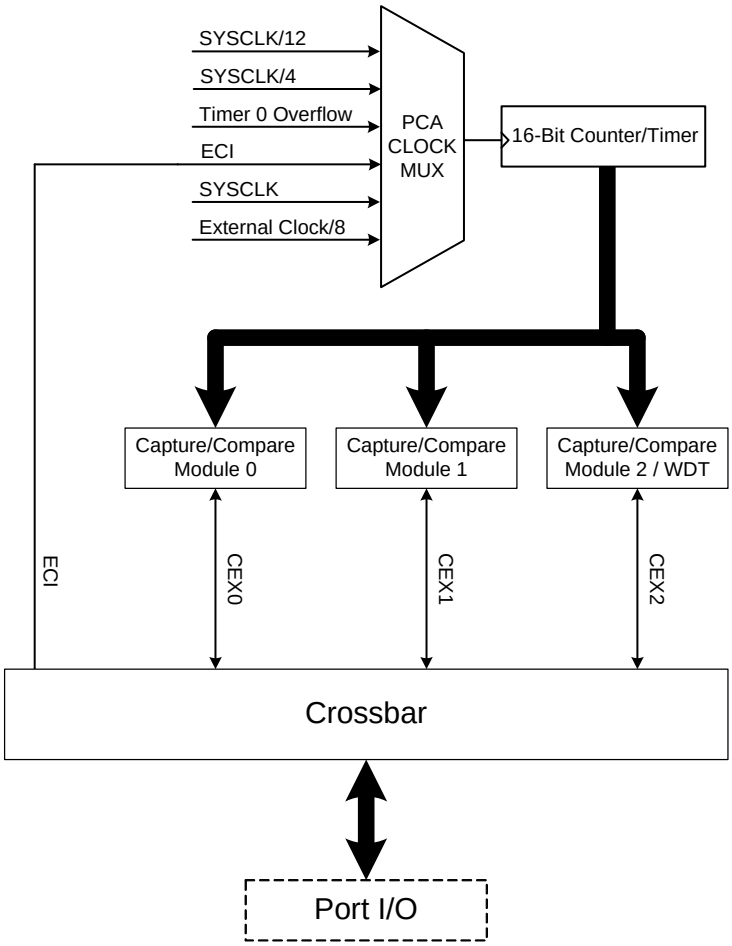

Figure 25.1. PCA Block Diagram