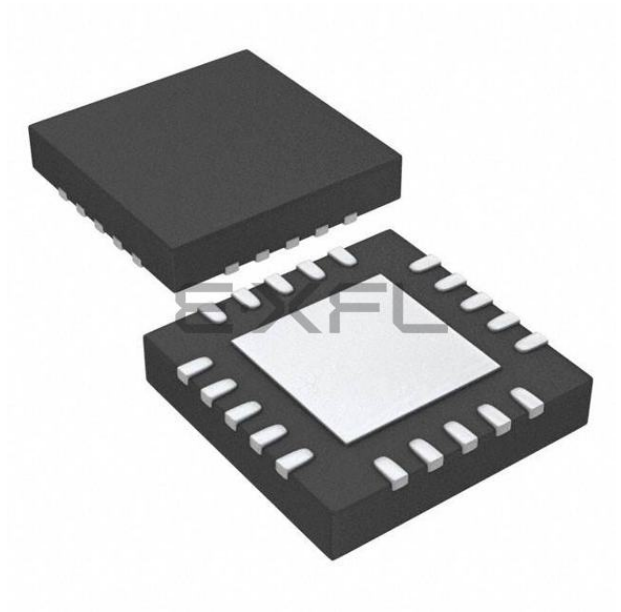

**Welcome to E-XFL.COM**

## 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 | 50MHz |
| Connectivity | SMBus (2-Wire/I²C), SPI, UART/USART |
| Peripherals | POR, PWM, Temp Sensor, WDT |
| Number of I/O | 17 |
| Program Memory Size | 8KB (8K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 1K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 3.6V |
| Data Converters | A/D 16x10b; D/A 2x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 105°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 20-VFQFN Exposed Pad |
| Supplier Device Package | 20-QFN (4x4) |
| Purchase URL | https://www.e-xfl.com/product-detail/silicon-labs/c8051f396-a-gmr |

## 5. QFN-20 Package Specifications

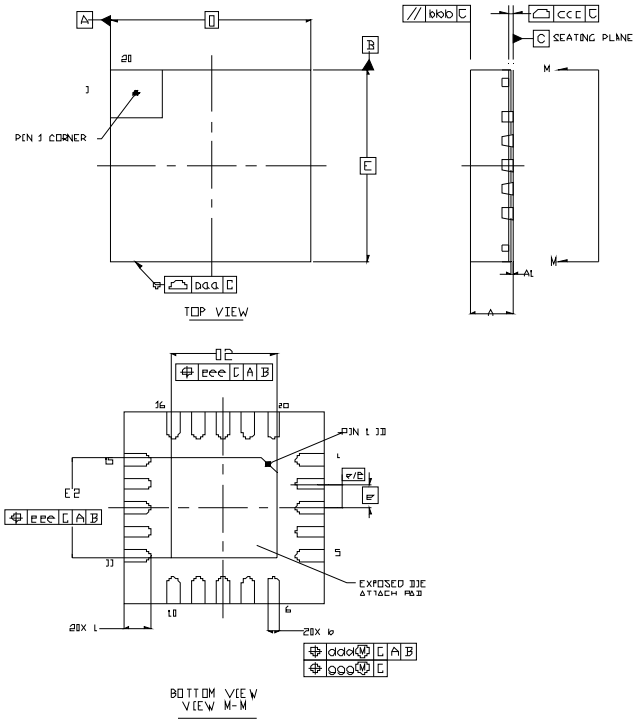


**Figure 5.1. QFN-20 Package Drawing**

**Table 5.1. QFN-20 Package Dimensions**

| Dimension | Min | Typ | Max | | Dimension | Min | Typ | Max |
|---|---|---|---|---|---|---|---|---|
| A | 0.80 | 0.85 | 0.90 | | L | 0.50 | 0.55 | 0.60 |
| A1 | 0.00 | 0.035 | 0.05 | | aaa | — | — | 0.10 |
| b | 0.20 | 0.25 | 0.30 | | bbb | — | — | 0.10 |
| D | 4.00 BSC. | | | | ccc | — | — | 0.08 |
| D2 | 2.00 | 2.10 | 2.20 | | ddd | — | — | 0.10 |
| e | 0.50 BSC. | | | | eee | — | — | 0.10 |
| E | 4.00 BSC. | | | | ggg | | | 0.05 |
| E2 | 2.00 | 2.10 | 2.20 | | | | | |

**Notes:**

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This drawing conforms to the JEDEC Solid State Outline MO-220, variation VGGD except for custom features D2, E2, Z, Y, and L which are toleranced per supplier designation.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

**Table 7.3. Port I/O DC Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

| Parameters | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| **Standard Port I/O** | | | | | |
| 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 |
| | $I_{OL}$ = 10 mA, 1.8 V ≤ $V_{DD}$ < 2.7 V | — | 0.8 | — | V |
| | $I_{OL}$ = 25 mA, 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | — | 1.0 | — | V |
| Input High Voltage | 1.8 V ≤ $V_{DD}$ < 2.7 V | $V_{DD}$ – 0.4 | — | — | V |
| | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | $V_{DD}$ – 0.5 | — | — | V |
| Input Low Voltage | 1.8 V ≤ $V_{DD}$ < 2.7 V | — | — | 0.5 | V |
| | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | — | — | 0.6 | V |
| Input Leakage Current | Weak Pullup Off | — | — | ±1 | µA |
| | Weak Pullup On, $V_{IN}$ = 0 V | — | 20 | 100 | µA |
| **EESDA and EESCL (C8051F37x Only)*** | | | | | |
| Output Low Voltage (EESDA) | $I_{OL}$ = 0.15 mA, $V_{DD}$ = 1.8 V | — | — | 0.2 | V |
| Output Low Voltage (EESDA) | $I_{OL}$ = 2.1 mA, $V_{DD}$ = 3 V | — | — | 0.4 | V |
| Output Leakage Current (EESDA) | EEPUE = 0, $V_{DD}$ = 3.6 V, 0 V ≤ $V_{OUT}$ ≤ $V_{DD}$ | — | — | 2 | µA |
| Input High Voltage | | $V_{DD}$ x 0.7 | — | — | V |
| Input Low Voltage | | — | — | $V_{DD}$ x 0.3 | V |
| Input Leakage Current | EEPUE = 0, Standby, $V_{DD}$ = 3.6 V, 0 V ≤ $V_{IN}$ ≤ $V_{DD}$ | — | — | ±3 | µA |
| **Note:** Applicable when interfacing to the C8051F37x EEPROM. Otherwise, standard port I/O characteristics apply. | | | | | |

**Table 7.4. Reset Electrical Characteristics**

–40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| $\overline{\text{RST}}$ Output Low Voltage | $I_{OL}$ = 4 mA,<br>$V_{DD}$ = 1.8 to 3.6 V | — | — | 0.6 | V |
| $\overline{\text{RST}}$ Input Low Voltage | | — | — | 0.6 | V |
| $\overline{\text{RST}}$ Input Pullup Current | $\overline{\text{RST}}$ = 0.0 V | — | 20 | 100 | µA |
| $V_{DD}$ POR Threshold ($V_{RST}$) | $V_{RST\_LOW}$ | 1.7 | 1.75 | 1.8 | V |
| | $V_{RST\_HIGH}$ | 2.4 | 2.55 | 2.7 | V |
| Missing Clock Detector Time-out | Time from last system clock rising edge to reset initiation | 80 | 580 | 800 | µs |
| Reset Time Delay | Delay between release of any reset source and code execution at location 0x0000 | — | — | 40 | µs |
| Minimum $\overline{\text{RST}}$ Low Time to Generate a System Reset | | 15 | — | — | µs |
| $V_{DD}$ Monitor Turn-on Time | | 100 | — | — | µs |
| $V_{DD}$ Monitor Supply Current | | — | 20 | 50 | µA |

### Table 7.7. Internal High-Frequency Oscillator Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), using factory-calibrated settings, unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Oscillator Frequency | C8051F390/1/2/3, C8051F370/1 | 48 | 49 | 50 | MHz |
| Oscillator Supply Current (from $V_{DD}$) | C8051F394/5/6/7, C8051F374/5 | — | 840 | 880 | µA |
| Power Supply Sensitivity | C8051F398/9 | — | 0.12 | — | %/V |
| Temperature Sensitivity | | — | 90 | — | ppm/°C |

### Table 7.8. Internal Low-Frequency Oscillator Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), using factory-calibrated settings, unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Oscillator Frequency | C8051F390/1/2/3, C8051F370/1 | 75 | 80 | 85 | kHz |
| Oscillator Supply Current (from $V_{DD}$) | C8051F394/5/6/7, C8051F374/5 | — | 5.5 | 12 | µA |
| Power Supply Sensitivity | C8051F398/9 | — | 0.05 | — | %/V |
| Temperature Sensitivity | | — | 160 | — | ppm/°C |

### Table 7.9. Internal Low-Power Oscillator Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), using factory-calibrated settings, unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Oscillator Frequency | C8051F390/1/2/3, C8051F370/1 | 18.5 | 20 | 21.5 | MHz |
| Power Supply Sensitivity | C8051F394/5/6/7, C8051F374/5 | — | 0.1 | — | %/V |
| Temperature Sensitivity | C8051F398/9 | — | 60 | — | ppm/°C |

## SFR Definition 9.9. AMX0P: AMUX0 Positive Channel Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | AMX0P[4:0] | | | | |
| Type | R | R | R | R/W | | | | |
| Reset | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

SFR Address = 0xBB; SFR Page = All Pages

| Bit | Name | Function |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't Care. |
| 4:0 | AMX0P[4:0] | **AMUX0 Positive Input Selection.**<br>00000: P0.0<br>00001: P0.1<br>00010: P0.2<br>00011: P0.3<br>00100: P0.4<br>00101: P0.5<br>00110: P0.6<br>00111: P0.7<br>01000: P1.0<br>01001: P1.1<br>01010: P1.2<br>01011: P1.3<br>01100: P1.4<br>01101: P1.5<br>01110: P1.6<br>01111: P1.7<br>10000: Temp Sensor<br>10001: $V_{DD}$<br>10010: P2.0 (C8051F390/1/4/5 and C8051F37x Only)<br>10011: P2.1 (C8051F390/1/4/5 and C8051F37x Only)<br>10100: P2.2 (C8051F390/1/4/5 and C8051F37x Only)<br>10101: P2.3 (C8051F390/1/4/5 and C8051F37x Only)<br>10110 – 11111: no input selected |

While in the SPI0 ISR, a PCA interrupt occurs. Recall the PCA interrupt is configured as a high priority interrupt, while the SPI0 interrupt is configured as a low priority interrupt. Thus, the CIP-51 will now vector to the high priority PCA ISR. Upon doing so, the value that was in the SFRPGIDX = 001b location before the PCA interrupt (in this case SFR page 0x0F for TS0CN) is pushed down to the SFRPGIDX = 010b location. Likewise, the value that was in the SFRPAGE register before the PCA interrupt (SFR page 0x00 for SPI0) is pushed down the stack into the SFRPGIDX = 001b location. Lastly, the CIP-51 will automatically places the SFR page needed to access the PCA0's special function registers into the SFRPAGE register, SFR page 0x00. See Figure 19.4.

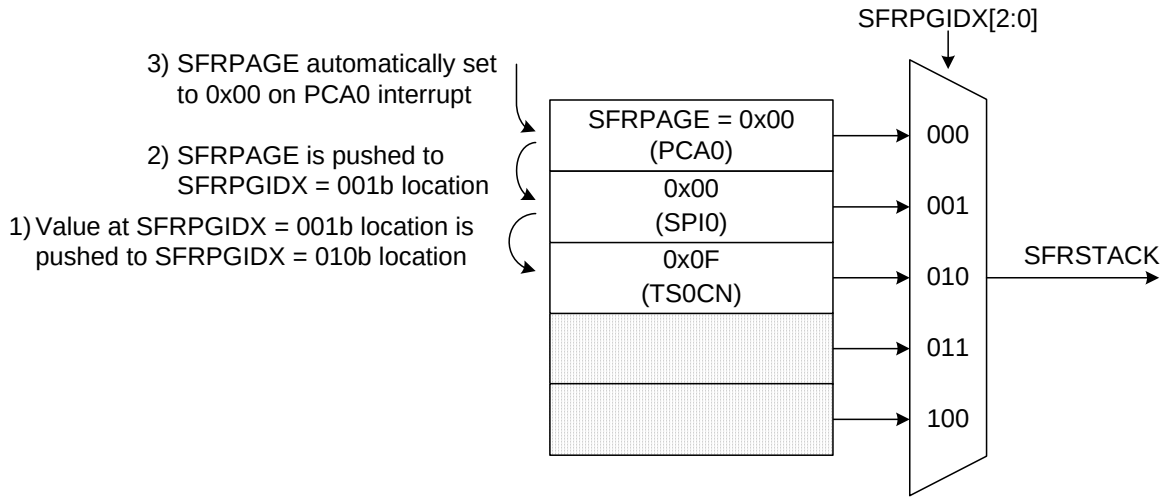


**Figure 19.4. SFR Page Stack Upon PCA Interrupt Occurring During a SPI0 ISR**

# 20. Interrupts

The C8051F39x/37x includes an extended interrupt system supporting multiple interrupt sources with four priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

If interrupts are enabled for the source, an interrupt request is generated when the interrupt-pending flag is set. As soon as execution of the current instruction is complete, the CPU generates an LCALL to a predetermined address to begin execution of an interrupt service routine (ISR). Each ISR must end with an RETI instruction, which returns program execution to the next instruction that would have been executed if the interrupt request had not occurred. If interrupts are not enabled, the interrupt-pending flag is ignored by the hardware and program execution continues as normal. (The interrupt-pending flag is set to logic 1 regardless of the interrupt's enable/disable state.)

Each interrupt source can be individually enabled or disabled through the use of an associated interrupt enable bit in an SFR (IE, EIE1, and EIE2). However, interrupts must first be globally enabled by setting the EA bit (IE.7) to logic 1 before the individual interrupt enables are recognized. Setting the EA bit to logic 0 disables all interrupt sources regardless of the individual interrupt-enable settings.

**Note:** Any instruction that clears a bit to disable an interrupt should be immediately followed by an instruction that has two or more opcode bytes. Using EA (global interrupt enable) as an example:

```
// in 'C':
EA = 0; // clear EA bit.
EA = 0; // this is a dummy instruction with two-byte opcode.

; in assembly:
CLR EA ; clear EA bit.
CLR EA ; this is a dummy instruction with two-byte opcode.
```

For example, if an interrupt is posted during the execution phase of a "CLR EA" opcode (or any instruction which clears a bit to disable an interrupt source), and the instruction is followed by a single-cycle instruction, the interrupt may be taken. However, a read of the enable bit will return a '0' inside the interrupt service routine. When the bit-clearing opcode is followed by a multi-cycle instruction, the interrupt will not be taken.

Some interrupt-pending flags are automatically cleared by the hardware when the CPU vectors to the ISR. However, most are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the return-from-interrupt (RETI) instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

## 21.4. 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.

### 21.4.1. $V_{DD}$ 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.

### 21.4.2. 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 areas.
11. Add address bounds checking to the routines that write or erase Flash memory to ensure that a routine called with an illegal address does not result in modification of the Flash.

## SFR Definition 21.2. FLKEY: Flash Lock and Key

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

SFR Address = 0xB7; SFR Page = All Pages

| Bit | Name | Function |
|---|---|---|
| 7:0 | FLKEY[7:0] | **Flash Lock and Key Register.**<br>Write:<br>This register provides a lock and key function for Flash erasures and writes. Flash writes and erases are enabled by writing 0xA5 followed by 0xF1 to the FLKEY register. Flash writes and erases are automatically disabled after the next write or erase is complete. If any writes to FLKEY are performed incorrectly, or if a Flash write or erase operation is attempted while these operations are disabled, the Flash will be permanently locked from writes or erasures until the next device reset. If an application never writes to Flash, it can intentionally lock the Flash by writing a non-0xA5 value to FLKEY from software.<br>Read:<br>When read, bits 1–0 indicate the current Flash lock state.<br>00: Flash is write/erase locked.<br>01: The first key code has been written (0xA5).<br>10: Flash is unlocked (writes/erases allowed).<br>11: Flash writes/erases disabled until the next reset. |

## SFR Definition 24.1. VDM0CN: $V_{DD}$ Monitor Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | VDMEN | VDDSTAT | VDMLVL | | | | | |
| Type | R/W | R | R/W | R | R | R | R | R |
| Reset | Varies | Varies | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address = 0xFF; SFR Page = All Pages

| Bit | Name | Function |
|---|---|---|
| 7 | VDMEN | **$V_{DD}$ Monitor Enable.**<br>This bit turns the $V_{DD}$ monitor circuit on/off. The $V_{DD}$ Monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC (SFR Definition 24.2). Selecting the $V_{DD}$ monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the $V_{DD}$ Monitor and selecting it as a reset source.<br>0: $V_{DD}$ Monitor Disabled.<br>1: $V_{DD}$ Monitor Enabled. |
| 6 | VDDSTAT | **$V_{DD}$ Status.**<br>This bit indicates the current power supply status ($V_{DD}$ Monitor output).<br>0: $V_{DD}$ is at or below the $V_{DD}$ monitor threshold.<br>1: $V_{DD}$ is above the $V_{DD}$ monitor threshold. |
| 5 | VDMLVL | **VDD Monitor Level Select.**<br>0: VDD Monitor Threshold is set to VRST-LOW.<br>1: VDD Monitor Threshold is set to VRST-HIGH. This setting is required for any system with firmware that writes and/or erases Flash. |
| 4:0 | Unused | Read = 000000b; Write = Don't care. |

## 25.2. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the controller core to enter Stop mode as soon as the instruction that sets the bit completes execution. Before entering stop mode, the system clock must be sourced by the internal high-frequency oscillator. In stop mode the internal oscillator, CPU, and all digital peripherals are stopped; the state of the external oscillator circuit is not affected. Each analog peripheral (including the external oscillator circuit) may be shut down individually prior to entering stop mode. Stop mode can only be terminated by an internal or external reset. On reset, the device performs the normal reset sequence and begins program execution at address 0x0000.

If enabled, the Missing Clock Detector will cause an internal reset and thereby terminate the stop mode. The Missing Clock Detector should be disabled if the CPU is to be put to in STOP mode for longer than the MCD timeout.

By default, when in stop mode the internal regulator is still active. However, the regulator can be configured to shut down while in stop mode to save power. To shut down the regulator in stop mode, the STOPCF bit in register REG01CN should be set to 1 prior to setting the STOP bit (see SFR Definition 25.1). If the regulator is shut down using the STOPCF bit, only the $\overline{\text{RST}}$ pin or a full power cycle are capable of resetting the device.

## 25.3. Suspend Mode

Setting the SUSPEND bit (OSCICN.5) causes the hardware to halt the CPU and the high-frequency internal oscillator, and go into suspend mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. Most digital peripherals are not active in suspend mode. The exception to this is the Port Match feature and Timer 3, when it is run from an external oscillator source or the internal low-frequency oscillator.

Suspend mode can be terminated by four types of events, a port match (described in Section “27.5. Port Match” on page 183), a Timer 3 overflow (described in Section “31.3. Timer 3” on page 259), a Comparator low output (if enabled), or a device reset event. Note that in order to run Timer 3 in suspend mode, the timer must be configured to clock from either the external clock source or the internal low-frequency oscillator source. When suspend mode is terminated, the device will continue execution on the instruction following the one that set the SUSPEND bit. If the wake event (port match or Timer 3 overflow) was configured to generate an interrupt, the interrupt will be serviced upon waking the device. If suspend mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

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 28.2. Minimum SDA Setup and Hold Times**

| EXTHOLD | Minimum SDA Setup Time | Minimum SDA Hold Time |
|---|---|---|
| 0 | $T_{low}$ – 4 system clocks<br>or<br>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 “28.3.4. SCL Low Timeout” on page 194). 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 28.4).

### 28.4.2. SMBus Pin Swap

The SMBus peripherals are assigned to pins using the priority crossbar decoder. By default, the SMBus signals are assigned to port pins starting with SDA on the lower-numbered pin, and SCL on the next available pin. The SMBnSWAP bits in the SMBTC register can be set to 1 to reverse the order in which the SMBus signals are assigned.

### 28.4.3. SMBus Timing Control

The SMBnSDD field in the SMBTC register are 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 SMBnSDD field 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.

### 28.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ 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 transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters slave transmitter mode, and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (an error condition may be generated if SMB0DAT is written following a received NACK while in slave transmitter mode). The interface exits slave transmitter mode after receiving a STOP. The interface will switch to slave receiver mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 28.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

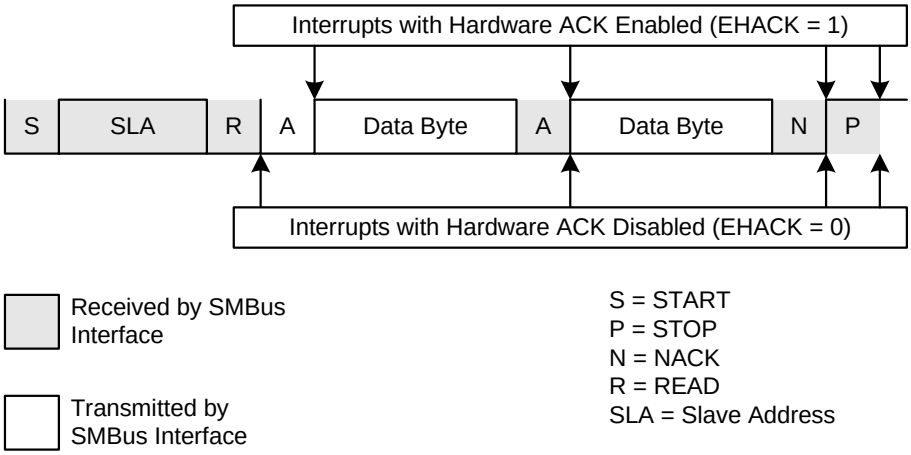


**Figure 28.8. Typical Slave Read Sequence**

## 28.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. The appropriate actions to take in response to an SMBus event depend on whether hardware slave address recognition and ACK generation is enabled or disabled. Table 28.5 describes the typical actions when hardware slave address recognition and ACK generation is disabled. Table 28.6 describes the typical actions when hardware slave address recognition and ACK generation is enabled. In the tables, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the SMBus specification. Highlighted responses are allowed by hardware but do not conform to the SMBus specification.

**Table 30.1. SPI Slave Timing Parameters**

| Parameter | Description | Min | Max | Units |
|---|---|---|---|---|
| **Master Mode Timing** (See Figure 30.8 and Figure 30.9) | | | | |
| $T_{MCKH}$ | 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 30.10 and Figure 30.11) | | | | |
| $T_{SE}$ | 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). | | | | |

## SFR Definition 31.15. TMR3RLL: Timer 3 Reload Register Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address = 0x92; SFR Page = 0

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLL[7:0] | **Timer 3 Reload Register Low Byte.**<br>TMR3RLL holds the low byte of the reload value for Timer 3. |

## SFR Definition 31.16. TMR3RLH: Timer 3 Reload Register High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address = 0x93; SFR Page = 0

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLH[7:0] | **Timer 3 Reload Register High Byte.**<br>TMR3RLH holds the high byte of the reload value for Timer 3. |

## SFR Definition 31.17. TMR3L: Timer 3 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3L[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address = 0x94; SFR Page = 0

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3L[7:0] | **Timer 3 Low Byte.**<br>In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value. |

## 32.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte (MSB) of the 16-bit counter/timer and PCA0L is the low byte (LSB). Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register. **Reading the PCA0L Register first guarantees an accurate reading of the entire 16-bit PCA0 counter.** Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer as shown in Table 32.1.

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

**Table 32.1. PCA Timebase Input Options**

| CPS2 | CPS1 | CPS0 | Timebase |
|---|---|---|---|
| 0 | 0 | 0 | System clock divided by 12 |
| 0 | 0 | 1 | System clock divided by 4 |
| 0 | 1 | 0 | Timer 0 overflow |
| 0 | 1 | 1 | High-to-low transitions on ECI (max rate = system clock divided by 4) |
| 1 | 0 | 0 | System clock |
| 1 | 0 | 1 | External oscillator source divided by 8* |
| 1 | 1 | 0 | Low frequency oscillator divided by 8* |
| 1 | 1 | 1 | Reserved |
| **Note:** Synchronized with the system clock. | | | |

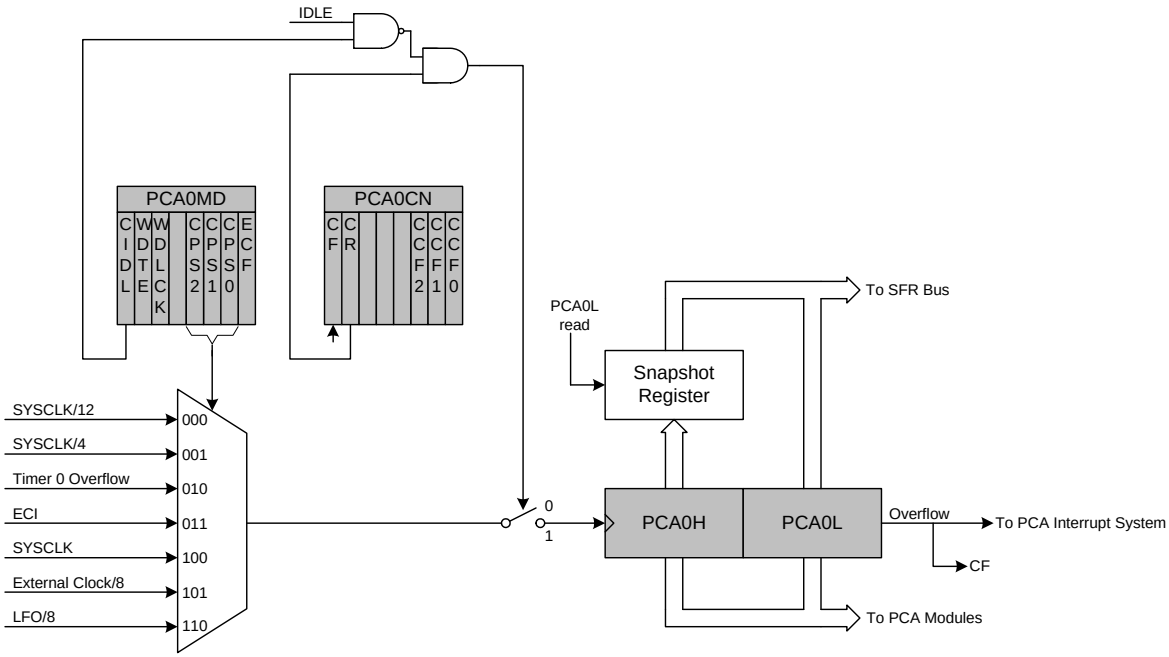


**Figure 32.2. PCA Counter/Timer Block Diagram**

### 32.3.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

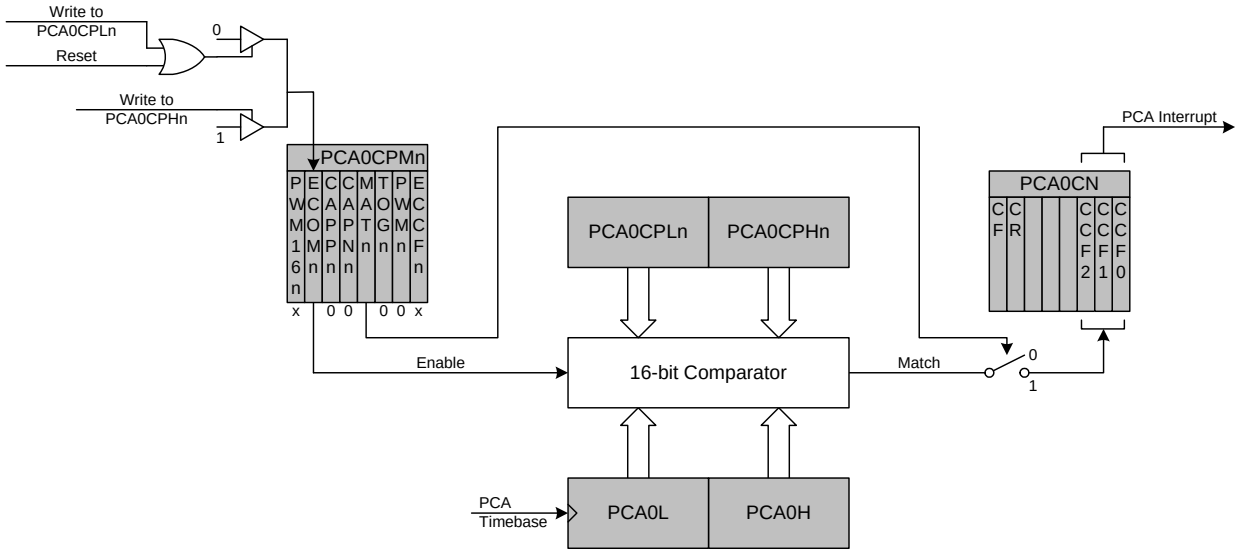


**Figure 32.5. PCA Software Timer Mode Diagram**

## SFR Definition 32.3. PCA0PWM: PCA PWM Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ARSEL | ECOV | COVF | | | | CLSEL[1:0] | |
| Type | R/W | R/W | R/W | R | R | R | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address = 0xF7; SFR Page = All Pages

| Bit | Name | Function |
|---|---|---|
| 7 | ARSEL | **Auto-Reload Register Select.**<br>This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function.<br>0: Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn.<br>1: Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn. |
| 6 | ECOV | **Cycle Overflow Interrupt Enable.**<br>This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt.<br>0: COVF will not generate PCA interrupts.<br>1: A PCA interrupt will be generated when COVF is set. |
| 5 | COVF | **Cycle Overflow Flag.**<br>This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software.<br>0: No overflow has occurred since the last time this bit was cleared.<br>1: An overflow has occurred since the last time this bit was cleared. |
| 4:2 | Unused | Unused. Read = 000b; Write = Don't care. |
| 1:0 | CLSEL[1:0] | **Cycle Length Select.**<br>When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode.<br>00: 8 bits.<br>01: 9 bits.<br>10: 10 bits.<br>11: 11 bits. |