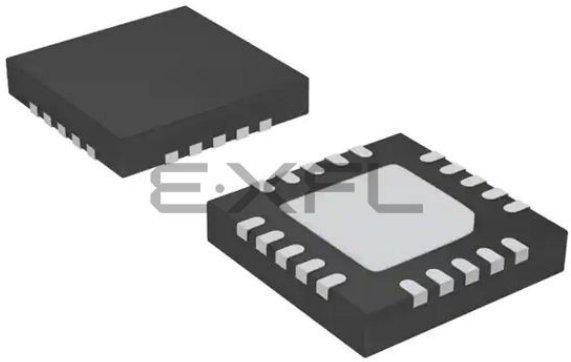

**Welcome to E-XFL.COM**

## 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 | Not For New Designs |
| 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 | 17 |
| Program Memory Size | 2KB (2K 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 | 20-VFQFN Exposed Pad |
| Supplier Device Package | 20-QFN (4x4) |
| Purchase URL | https://www.e-xfl.com/product-detail/silicon-labs/c8051f335-gm |

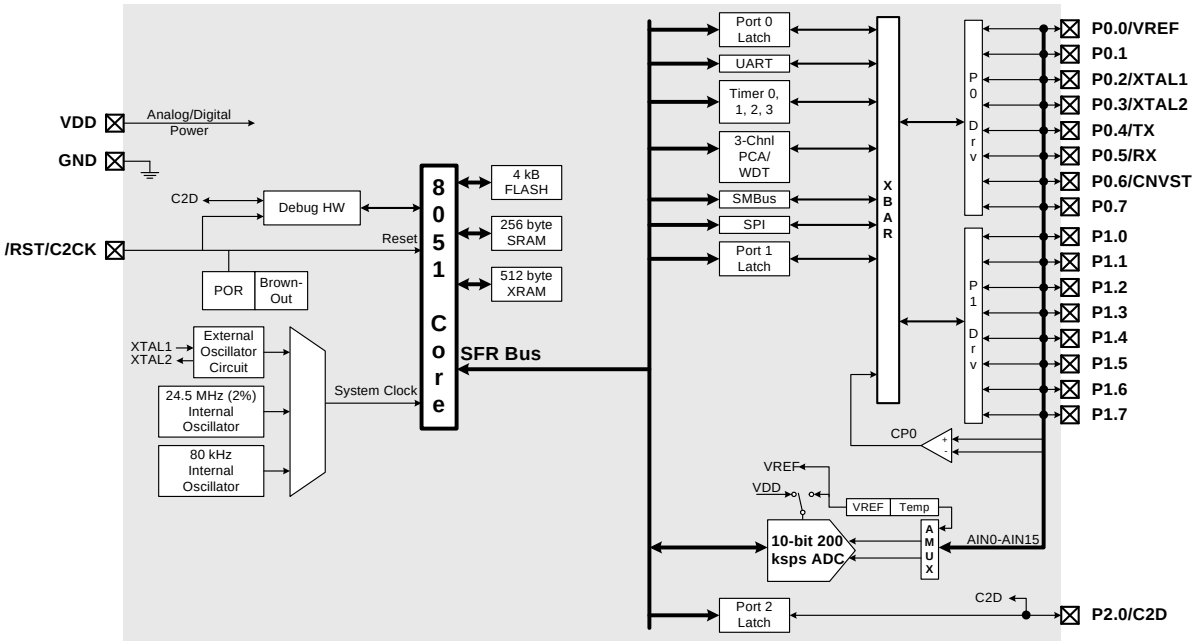


**Figure 1.3. C8051F332 Block Diagram**

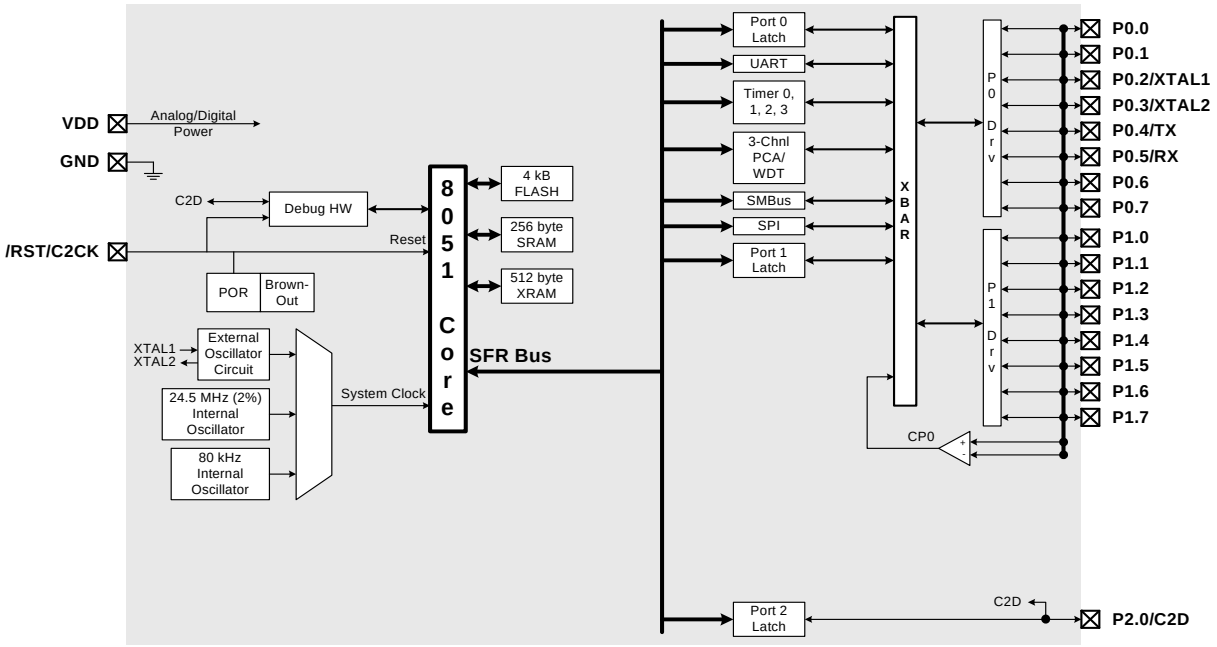


**Figure 1.4. C8051F333 Block Diagram**

measured from ‘0’ to VREF x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADC0H and ADC0L registers are set to ‘0’.

| Input Voltage | Right-Justified ADC0H:ADC0L (AD0LJST = 0) | Left-Justified ADC0H:ADC0L (AD0LJST = 1) |
|---|---|---|
| VREF x 1023/1024 | 0x03FF | 0xFFC0 |
| VREF x 512/1024 | 0x0200 | 0x8000 |
| VREF x 256/1024 | 0x0100 | 0x4000 |
| 0 | 0x0000 | 0x0000 |

When in Differential Mode, conversion codes are represented as 10-bit signed 2’s complement numbers. Inputs are measured from –VREF to VREF x 511/512. Example codes are shown below for both right-justified and left-justified data. For right-justified data, the unused MSBs of ADC0H are a sign-extension of the data word. For left-justified data, the unused LSBs in the ADC0L register are set to ‘0’.

| Input Voltage | Right-Justified ADC0H:ADC0L (AD0LJST = 0) | Left-Justified ADC0H:ADC0L (AD0LJST = 1) |
|---|---|---|
| VREF x 511/512 | 0x01FF | 0x7FC0 |
| VREF x 256/512 | 0x0100 | 0x4000 |
| 0 | 0x0000 | 0x0000 |
| –VREF x 256/512 | 0xFF00 | 0xC000 |
| –VREF | 0xFE00 | 0x8000 |

**Important Note About ADC0 Input Configuration:** Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to ‘0’ the corresponding bit in register PnMDIN (for n = 0,1). To force the Crossbar to skip a Port pin, set to ‘1’ the corresponding bit in register PnSKIP (for n = 0,1). See **Section “14. Port Input/Output” on page 123** for more Port I/O configuration details.

## 5.2. Temperature Sensor

The typical temperature sensor transfer function is shown in Figure 5.2. The output voltage ($V_{TEMP}$) is the positive ADC input when the temperature sensor is selected by bits AMX0P4–0 in register AMX0P.

### 5.3.2. Tracking Modes

Each ADC0 conversion must be preceded by a minimum tracking time in order for the converted result to be accurate. The minimum tracking time is given in Table 5.1. The AD0TM bit in register ADC0CN controls the ADC0 track-and-hold mode. In its default state, the ADC0 input is continuously tracked, except when a conversion is in progress. When the AD0TM bit is logic 1, ADC0 operates in low-power track-and-hold mode. In this mode, each conversion is preceded by a tracking period of 3 SAR clocks (after the start-of-conversion signal). When the CNVSTR signal is used to initiate conversions in low-power tracking mode, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR (see Figure 5.3). Tracking can also be disabled (shutdown) when the device is in low power standby or sleep modes. Low-power track-and-hold mode is also useful when AMUX settings are frequently changed, due to the settling time requirements described in **Section "5.3.3. Settling Time Requirements" on page 46**.

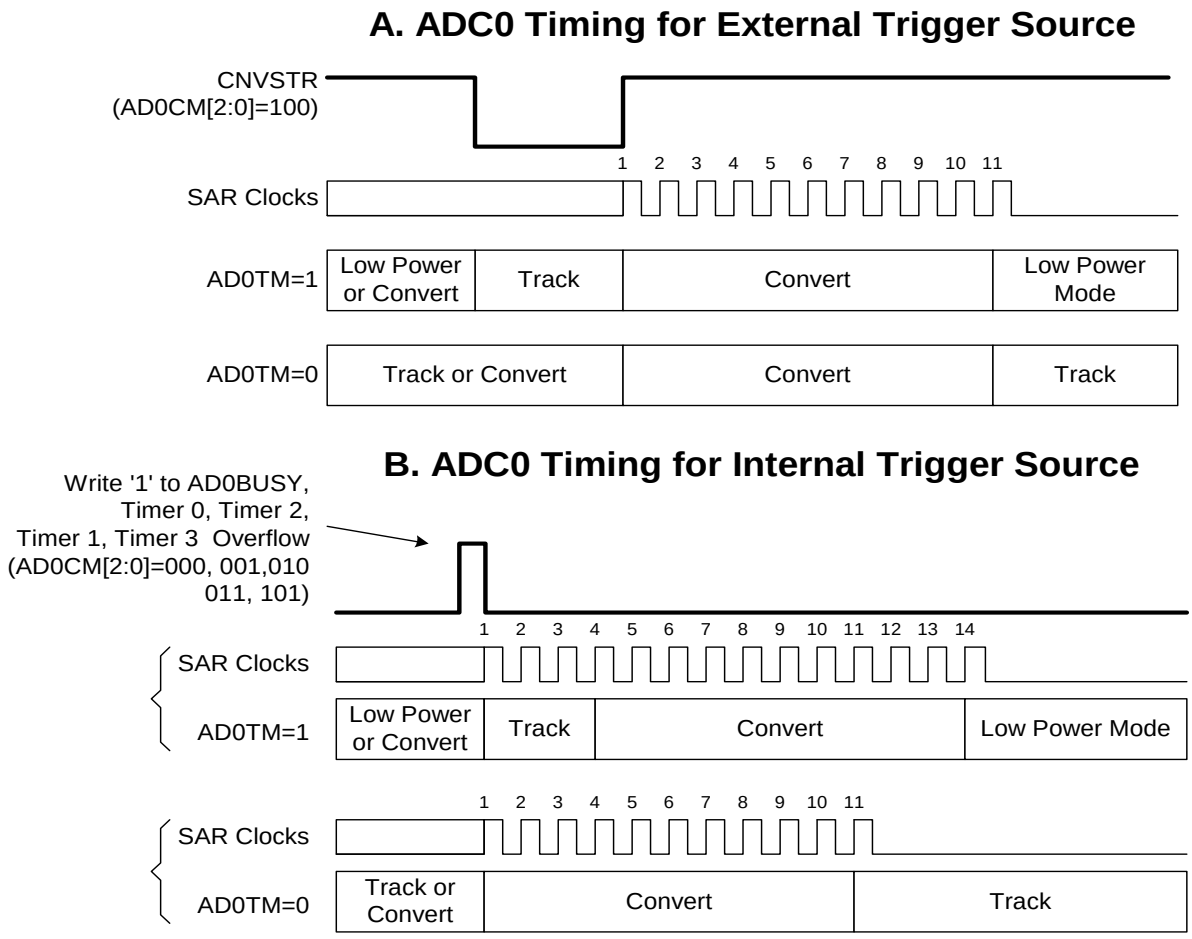


**Figure 5.3. 10-Bit ADC Track and Conversion Example Timing**

# 6. 10-Bit Current Mode DAC (IDA0, C8051F330 only)

The C8051F330 device includes a 10-bit current-mode Digital-to-Analog Converter (IDAC). The maximum current output of the IDAC can be adjusted for three different current settings; 0.5 mA, 1 mA, and 2 mA. The IDAC is enabled or disabled with the IDA0EN bit in the IDA0 Control Register (see SFR Definition 6.1). When IDA0EN is set to ‘0’, the IDAC port pin (P0.1) behaves as a normal GPIO pin. When IDA0EN is set to ‘1’, the digital output drivers and weak pullup for the IDAC pin are automatically disabled, and the pin is connected to the IDAC output. An internal bandgap bias generator is used to generate a reference current for the IDAC whenever it is enabled. When using the IDAC, bit 1 in the P0SKIP register should be set to ‘1’, to force the Crossbar to skip the IDAC pin.

## 6.1. IDA0 Output Scheduling

IDA0 features a flexible output update mechanism which allows for seamless full-scale changes and supports jitter-free updates for waveform generation. Three update modes are provided, allowing IDAC output updates on a write to IDA0H, on a Timer overflow, or on an external pin edge.

### 6.1.1. Update Output On-Demand

In its default mode (IDA0CN.[6:4] = ‘111’) the IDA0 output is updated “on-demand” on a write to the high-byte of the IDA0 data register (IDA0H). It is important to note that writes to IDA0L are held in this mode, and have no effect on the IDA0 output until a write to IDA0H takes place. If writing a full 10-bit word to the IDAC data registers, the 10-bit data word is written to the low byte (IDA0L) and high byte (IDA0H) data registers. Data is latched into IDA0 after a write to the IDA0H register, **so the write sequence should be IDA0L followed by IDA0H** if the full 10-bit resolution is required. The IDAC can be used in 8-bit mode by initializing IDA0L to the desired value (typically 0x00), and writing data to only IDA0H (see **Section 6.2** for information on the format of the 10-bit IDAC data word within the 16-bit SFR space).

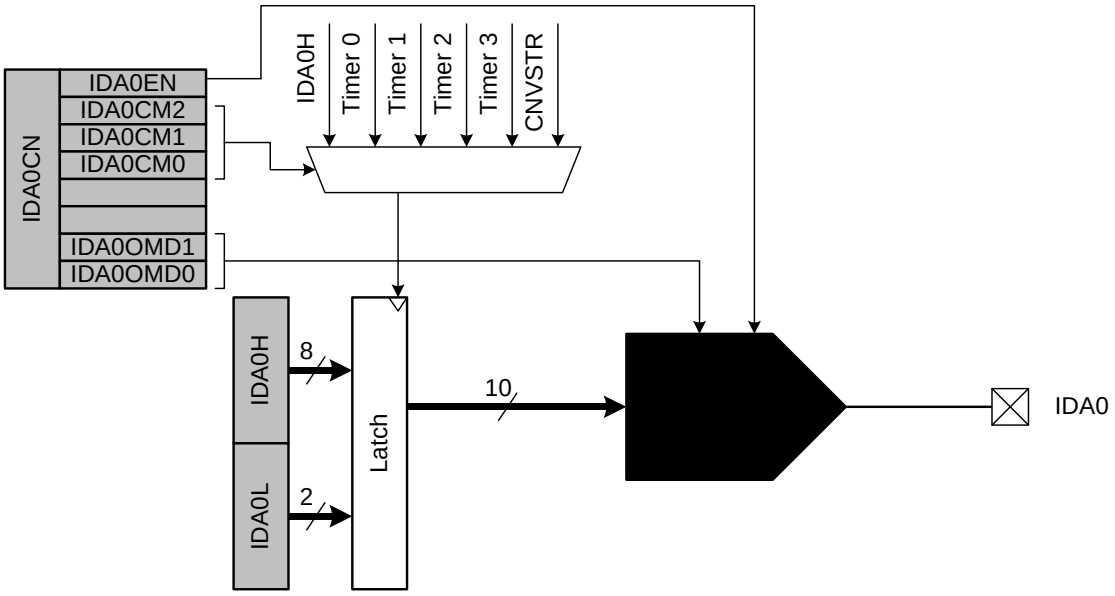


**Figure 6.1. IDA0 Functional Block Diagram**

| Notes on Registers, Operands and Addressing Modes: |
|---|
| **Rn** - Register R0–R7 of the currently selected register bank.<br><br>**@Ri** - Data RAM location addressed indirectly through R0 or R1.<br><br>**rel** - 8-bit, signed (two's complement) offset relative to the first byte of the following instruction. Used by SJMP and all conditional jumps.<br><br>**direct** - 8-bit internal data location's address. This could be a direct-access Data RAM location (0x00–0x7F) or an SFR (0x80–0xFF).<br><br>**#data** - 8-bit constant<br><br>**#data16** - 16-bit constant<br><br>**bit** - Direct-accessed bit in Data RAM or SFR<br><br>**addr11** - 11-bit destination address used by ACALL and AJMP. The destination must be within the same 2 kB page of program memory as the first byte of the following instruction.<br><br>**addr16** - 16-bit destination address used by LCALL and LJMP. The destination may be anywhere within the 8 kB program memory space.<br><br>There is one unused opcode (0xA5) that performs the same function as NOP.<br>All mnemonics copyrighted © Intel Corporation 1980. |

## 9.2. 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 9.2

```
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.
```

If an interrupt is posted during the execution phase of a "CLR EA" opcode (or any instruction which clears the EA bit), and the instruction is followed by a single-cycle instruction, the interrupt may be taken. However, a read of the EA bit will return a '0' inside the interrupt service routine. When the "CLR EA" 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.

### 9.3.1. MCU Interrupt Sources and Vectors

The MCUs support 13 interrupt sources. Software can simulate an interrupt by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag. MCU interrupt sources, associated vector addresses, priority order and control bits are summarized in Table 9.4 on page 88. Refer to the datasheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

**Table 9.4. Interrupt Summary**

| Interrupt Source | Interrupt Vector | Priority Order | Pending Flag | Bit addressable? | Cleared by HW? | Enable Flag | Priority Control |
|---|---|---|---|---|---|---|---|
| Reset | 0x0000 | Top | None | N/A | N/A | Always Enabled | Always Highest |
| External Interrupt 0 (/INT0) | 0x0003 | 0 | IE0 (TCON.1) | Y | Y | EX0 (IE.0) | PX0 (IP.0) |
| Timer 0 Overflow | 0x000B | 1 | TF0 (TCON.5) | Y | Y | ET0 (IE.1) | PT0 (IP.1) |
| External Interrupt 1 (/INT1) | 0x0013 | 2 | IE1 (TCON.3) | Y | Y | EX1 (IE.2) | PX1 (IP.2) |
| Timer 1 Overflow | 0x001B | 3 | TF1 (TCON.7) | Y | Y | ET1 (IE.3) | PT1 (IP.3) |
| UART0 | 0x0023 | 4 | RI0 (SCON0.0)<br>TI0 (SCON0.1) | Y | N | ES0 (IE.4) | PS0 (IP.4) |
| Timer 2 Overflow | 0x002B | 5 | TF2H (TMR2CN.7)<br>TF2L (TMR2CN.6) | Y | N | ET2 (IE.5) | PT2 (IP.5) |
| SPI0 | 0x0033 | 6 | SPIF (SPI0CN.7)<br>WCOL (SPI0CN.6)<br>MODF (SPI0CN.5)<br>RXOVRN (SPI0CN.4) | Y | N | ESPI0 (IE.6) | PSPI0 (IP.6) |
| SMB0 | 0x003B | 7 | SI (SMB0CN.0) | Y | N | ESMB0 (EIE1.0) | PSMB0 (EIP1.0) |
| RESERVED | 0x0043 | 8 | N/A | N/A | N/A | N/A | N/A |
| ADC0 Window Compare | 0x004B | 9 | AD0WINT (ADC0CN.3) | Y | N | EWADC0 (EIE1.2) | PWADC0 (EIP1.2) |
| ADC0 Conversion Complete | 0x0053 | 10 | AD0INT (ADC0CN.5) | Y | N | EADC0 (EIE1.3) | PADC0 (EIP1.3) |
| Programmable Counter Array | 0x005B | 11 | CF (PCA0CN.7)<br>CCFn (PCA0CN.n) | Y | N | EPCA0 (EIE1.4) | PPCA0 (EIP1.4) |
| Comparator0 | 0x0063 | 12 | CP0FIF (CPT0CN.4)<br>CP0RIF (CPT0CN.5) | N | N | ECP0 (EIE1.5) | PCP0 (EIP1.5) |
| RESERVED | 0x006B | 13 | N/A | N/A | N/A | N/A | N/A |
| Timer 3 Overflow | 0x0073 | 14 | TF3H (TMR3CN.7)<br>TF3L (TMR3CN.6) | N | N | ET3 (EIE1.7) | PT3 (EIP1.7) |

## 10.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin is driven low until $V_{DD}$ settles above $V_{RST}$. A delay occurs before the device is released from reset; the delay decreases as the $V_{DD}$ ramp time increases ($V_{DD}$ ramp time is defined as how fast $V_{DD}$ ramps from 0 V to $V_{RST}$). Figure 10.2. plots the power-on and $V_{DD}$ monitor reset timing. The maximum $V_{DD}$ ramp time is 1 ms; slower ramp times may cause the device to be released from reset before $V_{DD}$ reaches the $V_{RST}$ level. For ramp times less than 1 ms, the power-on reset delay ($T_{PORDelay}$) is typically less than 0.3 ms.

On exit from a power-on reset, the PORSF flag (RSTSRC.1) is set by hardware to logic 1. When PORSF is set, all of the other reset flags in the RSTSRC Register are indeterminate (PORSF is cleared by all other resets). Since all resets cause program execution to begin at the same location (0x0000) software can read the PORSF flag to determine if a power-up was the cause of reset. The content of internal data memory should be assumed to be undefined after a power-on reset. The $V_{DD}$ monitor is disabled following a power-on reset.

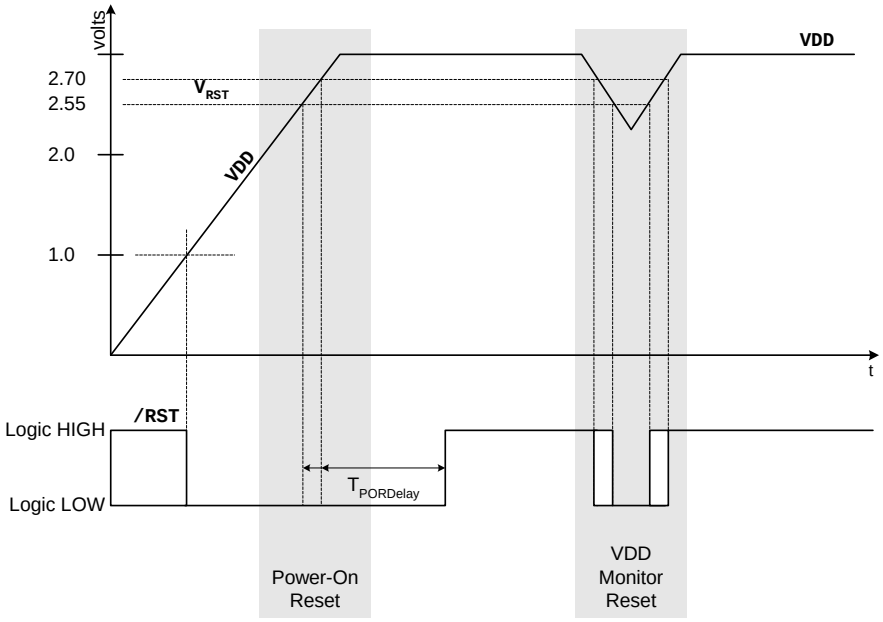


**Figure 10.2. Power-On and $V_{DD}$ Monitor Reset Timing**

## 10.2. 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 10.2). 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 disabled after power-on resets; however its defined state (enabled/disabled) is not altered by any other reset source. For example, if the $V_{DD}$ monitor is enabled and a software reset is performed, the $V_{DD}$ monitor will still be enabled after the reset.

## 16.1. Enhanced Baud Rate Generation

The UART0 baud rate is generated by Timer 1 in 8-bit auto-reload mode. The TX clock is generated by TL1; the RX clock is generated by a copy of TL1 (shown as RX Timer in Figure 16.2), which is not user-accessible. Both TX and RX Timer overflows are divided by two to generate the TX and RX baud rates. The RX Timer runs when Timer 1 is enabled, and uses the same reload value (TH1). However, an RX Timer reload is forced when a START condition is detected on the RX pin. This allows a receive to begin any time a START is detected, independent of the TX Timer state.

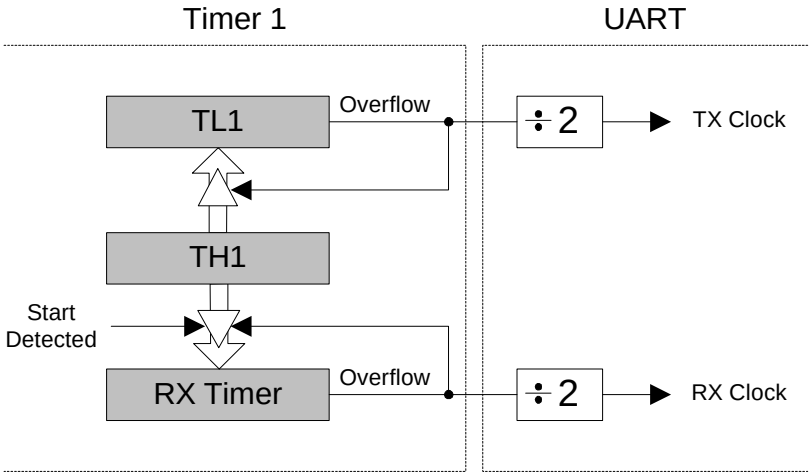


**Figure 16.2. UART0 Baud Rate Logic**

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see **Section "18.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload" on page 179**). The Timer 1 reload value should be set so that overflows will occur at two times the desired UART baud rate frequency. Note that Timer 1 may be clocked by one of six sources: SYSCLK, SYSCLK / 4, SYSCLK / 12, SYSCLK / 48, the external oscillator clock / 8, or an external input T1. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 16.1-A and Equation 16.1-B.

A) $$\text{UartBaudRate} = \frac{1}{2} \times \text{T1\_Overflow\_Rate}$$

B) $$\text{T1\_Overflow\_Rate} = \frac{\text{T1}_{\text{CLK}}}{256 - \text{TH1}}$$

**Equation 16.1. UART0 Baud Rate**

Where $T1_{CLK}$ is the frequency of the clock supplied to Timer 1, and *T1H* is the high byte of Timer 1 (reload value). Timer 1 clock frequency is selected as described in **Section "18. Timers" on page 177**. A quick reference for typical baud rates and system clock frequencies is given in Table 16.1 through Table 16.6. Note that the internal oscillator may still generate the system clock when the external oscillator is driving Timer 1.

**Table 16.3. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator**

| | Frequency: 22.1184 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1–SCA0 (pre-scale select)[1] | T1M[1] | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 96 | SYSCLK | XX[2] | 1 | 0xD0 |
| | 115200 | 0.00% | 192 | SYSCLK | XX | 1 | 0xA0 |
| | 57600 | 0.00% | 384 | SYSCLK | XX | 1 | 0x40 |
| | 28800 | 0.00% | 768 | SYSCLK / 12 | 00 | 0 | 0xE0 |
| | 14400 | 0.00% | 1536 | SYSCLK / 12 | 00 | 0 | 0xC0 |
| | 9600 | 0.00% | 2304 | SYSCLK / 12 | 00 | 0 | 0xA0 |
| | 2400 | 0.00% | 9216 | SYSCLK / 48 | 10 | 0 | 0xA0 |
| | 1200 | 0.00% | 18432 | SYSCLK / 48 | 10 | 0 | 0x40 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 96 | EXTCLK / 8 | 11 | 0 | 0xFA |
| | 115200 | 0.00% | 192 | EXTCLK / 8 | 11 | 0 | 0xF4 |
| | 57600 | 0.00% | 384 | EXTCLK / 8 | 11 | 0 | 0xE8 |
| | 28800 | 0.00% | 768 | EXTCLK / 8 | 11 | 0 | 0xD0 |
| | 14400 | 0.00% | 1536 | EXTCLK / 8 | 11 | 0 | 0xA0 |
| | 9600 | 0.00% | 2304 | EXTCLK / 8 | 11 | 0 | 0x70 |

**Notes:**
1. SCA1–SCA0 and T1M bit definitions can be found in **Section 18.1**.
2. X = Don't care.

**Table 16.4. Timer Settings for Standard Baud Rates Using an External 18.432 MHz Oscillator**

| | Frequency: 18.432 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1–SCA0 (pre-scale select)[1] | T1M[1] | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 80 | SYSCLK | XX[2] | 1 | 0xD8 |
| | 115200 | 0.00% | 160 | SYSCLK | XX | 1 | 0xB0 |
| | 57600 | 0.00% | 320 | SYSCLK | XX | 1 | 0x60 |
| | 28800 | 0.00% | 640 | SYSCLK / 4 | 01 | 0 | 0xB0 |
| | 14400 | 0.00% | 1280 | SYSCLK / 4 | 01 | 0 | 0x60 |
| | 9600 | 0.00% | 1920 | SYSCLK / 12 | 00 | 0 | 0xB0 |
| | 2400 | 0.00% | 7680 | SYSCLK / 48 | 10 | 0 | 0xB0 |
| | 1200 | 0.00% | 15360 | SYSCLK / 48 | 10 | 0 | 0x60 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 80 | EXTCLK / 8 | 11 | 0 | 0xFB |
| | 115200 | 0.00% | 160 | EXTCLK / 8 | 11 | 0 | 0xF6 |
| | 57600 | 0.00% | 320 | EXTCLK / 8 | 11 | 0 | 0xEC |
| | 28800 | 0.00% | 640 | EXTCLK / 8 | 11 | 0 | 0xD8 |
| | 14400 | 0.00% | 1280 | EXTCLK / 8 | 11 | 0 | 0xB0 |
| | 9600 | 0.00% | 1920 | EXTCLK / 8 | 11 | 0 | 0x88 |

**Notes:**
1. SCA1–SCA0 and T1M bit definitions can be found in **Section 18.1**.
2. X = Don't care.

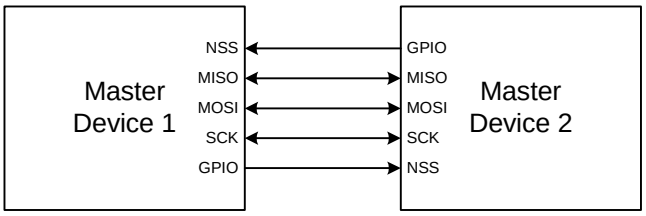


**Figure 17.2. Multiple-Master Mode Connection Diagram**

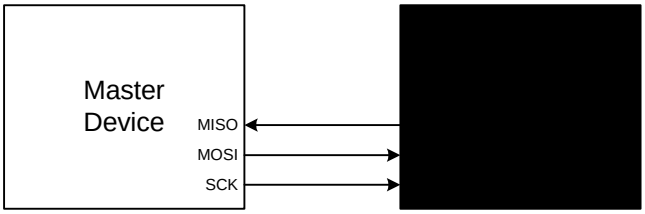


**Figure 17.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram**

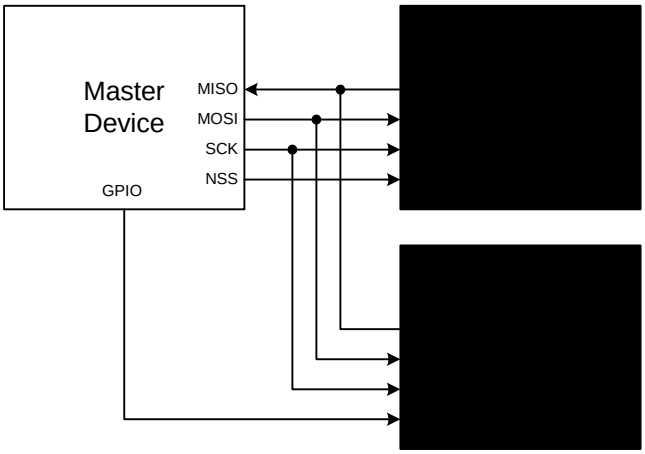


**Figure 17.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram**

## 17.5. Serial Clock Timing

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPI0 Configuration Register (SPI0CFG). The CKPHA bit (SPI0CFG.5) selects one of two clock phases (edge used to latch the data). The CKPOL bit (SPI0CFG.4) selects between an active-high or active-low clock. Both master and slave devices must be configured to use the same clock phase and polarity. SPI0 should be disabled (by clearing the SPIEN bit, SPI0CN.0) when changing the clock phase or polarity. The clock and data line relationships for master mode are shown in Figure 17.5. For slave mode, the clock and data relationships are shown in Figure 17.6 and Figure 17.7. Note that CKPHA must be set to '0' on both the master and slave SPI when communicating between two of the following devices: C8051F04x, C8051F06x, C8051F12x, C8051F31x, C8051F32x, and C8051F33x

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 17.3 controls the master mode serial clock frequency. This register is ignored when operating in slave mode. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower. When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS (in 4-wire slave mode), and the serial input data synchronously with the slave's system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less than 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the slave's system clock.

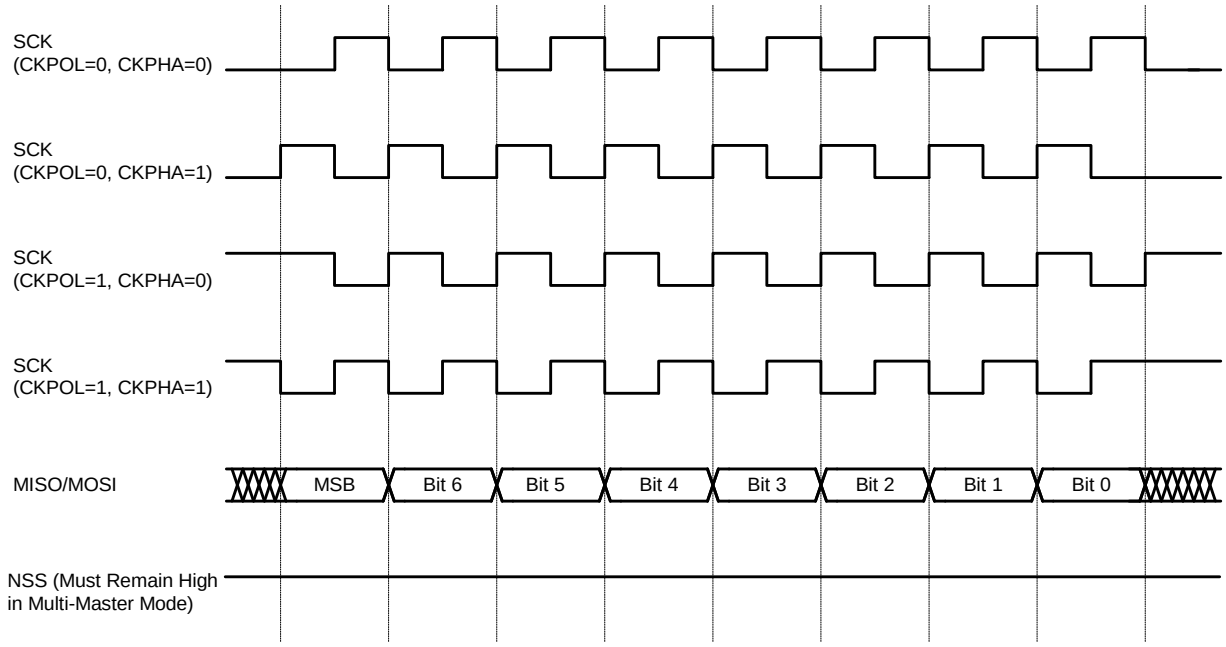


**Figure 17.5. Master Mode Data/Clock Timing**

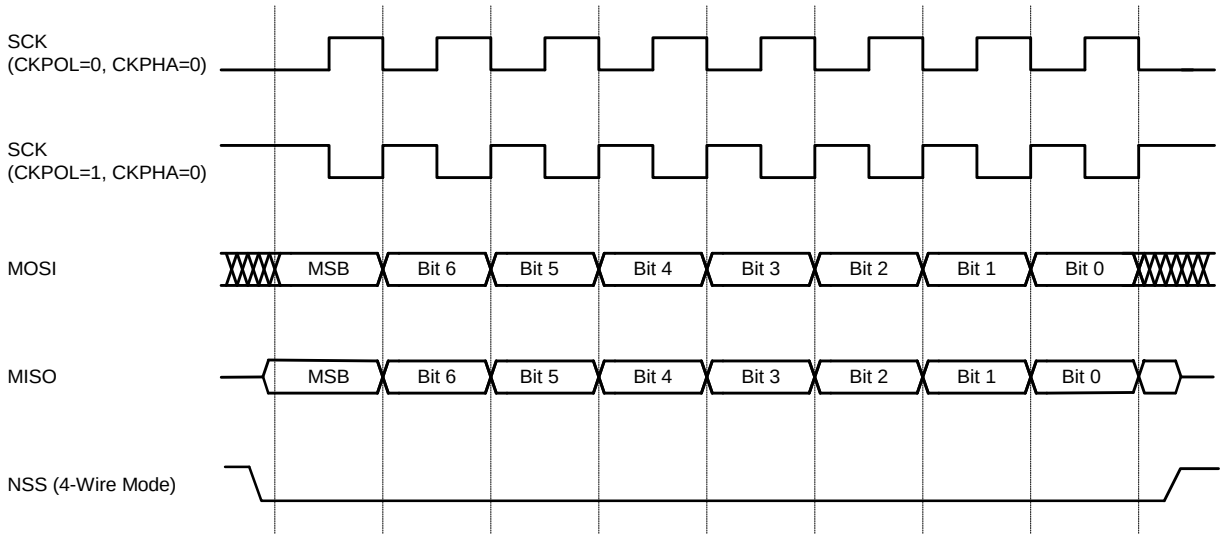


**Figure 17.6. Slave Mode Data/Clock Timing (CKPHA = 0)**

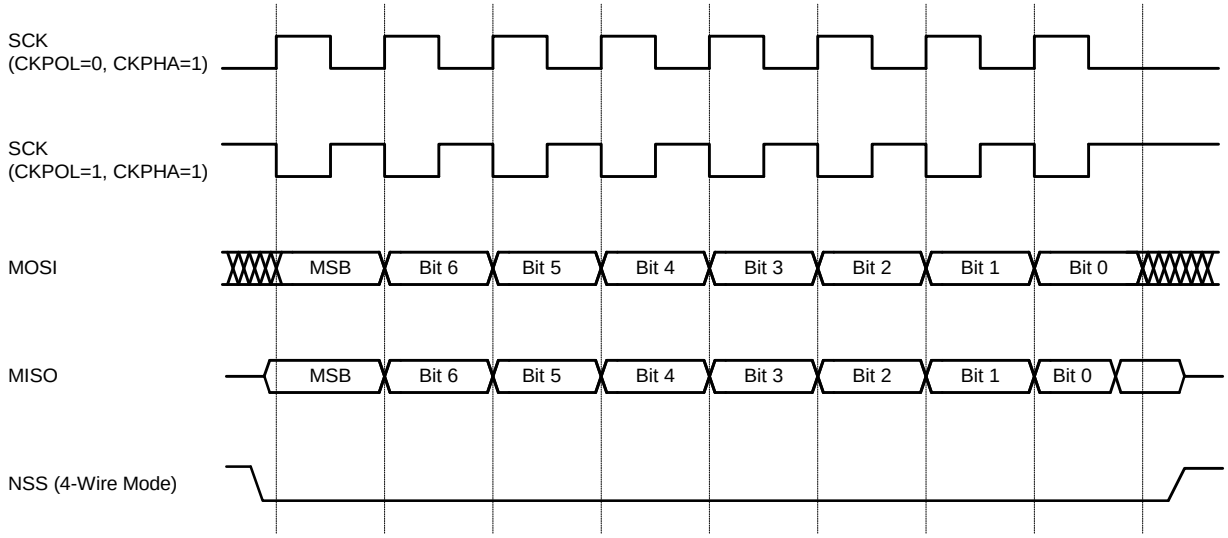


**Figure 17.7. Slave Mode Data/Clock Timing (CKPHA = 1)**

## 17.6. SPI Special Function Registers

SPI0 is accessed and controlled through four special function registers in the system controller: SPI0CN Control Register, SPI0DAT Data Register, SPI0CFG Configuration Register, and SPI0CKR Clock Rate Register. The four special function registers related to the operation of the SPI0 Bus are described in the following figures.

## SFR Definition 18.2. TMOD: Timer Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| GATE1 | C/T1 | T1M1 | T1M0 | GATE0 | C/T0 | T0M1 | T0M0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: |
| | | | | | | | | 0x89 |

Bit7: GATE1: Timer 1 Gate Control.
0: Timer 1 enabled when TR1 = 1 irrespective of /INT1 logic level.
1: Timer 1 enabled only when TR1 = 1 AND /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 9.11).

Bit6: C/T1: Counter/Timer 1 Select.
0: Timer Function: Timer 1 incremented by clock defined by T1M bit (CKCON.3).
1: Counter Function: Timer 1 incremented by high-to-low transitions on external input pin (T1).

Bits5–4: T1M1–T1M0: Timer 1 Mode Select.
These bits select the Timer 1 operation mode.

| T1M1 | T1M0 | Mode |
|---|---|---|
| 0 | 0 | Mode 0: 13-bit counter/timer |
| 0 | 1 | Mode 1: 16-bit counter/timer |
| 1 | 0 | Mode 2: 8-bit counter/timer with auto-reload |
| 1 | 1 | Mode 3: Timer 1 inactive |

Bit3: GATE0: Timer 0 Gate Control.
0: Timer 0 enabled when TR0 = 1 irrespective of /INT0 logic level.
1: Timer 0 enabled only when TR0 = 1 AND /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 9.11).

Bit2: C/T0: Counter/Timer Select.
0: Timer Function: Timer 0 incremented by clock defined by T0M bit (CKCON.2).
1: Counter Function: Timer 0 incremented by high-to-low transitions on external input pin (T0).

Bits1–0: T0M1–T0M0: Timer 0 Mode Select.
These bits select the Timer 0 operation mode.

| T0M1 | T0M0 | Mode |
|---|---|---|
| 0 | 0 | Mode 0: 13-bit counter/timer |
| 0 | 1 | Mode 1: 16-bit counter/timer |
| 1 | 0 | Mode 2: 8-bit counter/timer with auto-reload |
| 1 | 1 | Mode 3: Two 8-bit counter/timers |

## SFR Definition 19.1. PCA0CN: PCA Control

| R/W | R/W | R | R | R | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CF | CR | - | - | - | CCF2 | CCF1 | CCF0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xD8

Bit7: CF: PCA Counter/Timer Overflow Flag.
Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

Bit6: CR: PCA Counter/Timer Run Control.
This bit enables/disables the PCA Counter/Timer.
0: PCA Counter/Timer disabled.
1: PCA Counter/Timer enabled.

Bits5–3: UNUSED. Read = 000b, Write = don't care.

Bit2: CCF2: PCA Module 2 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF2 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

Bit1: CCF1: PCA Module 1 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF1 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

Bit0: CCF0: PCA Module 0 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF0 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

## NOTES: