

Welcome to [E-XFL.COM](https://www.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	CIP-51 8051
Core Size	8-Bit
Speed	25MHz
Connectivity	SMBus (2-Wire/I <sup>2</sup> C), SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	17
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 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	<a href="https://www.e-xfl.com/product-detail/silicon-labs/c8051f987-c-gmr">https://www.e-xfl.com/product-detail/silicon-labs/c8051f987-c-gmr</a>

# C8051F99x-C8051F98x

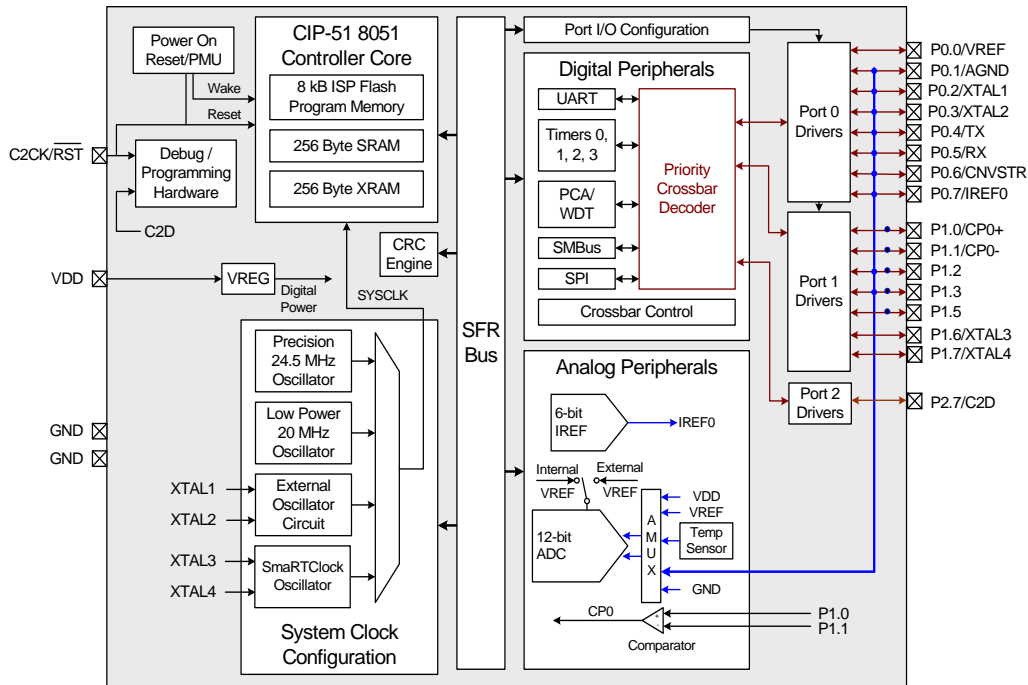


Figure 1.1. C8051F980 Block Diagram

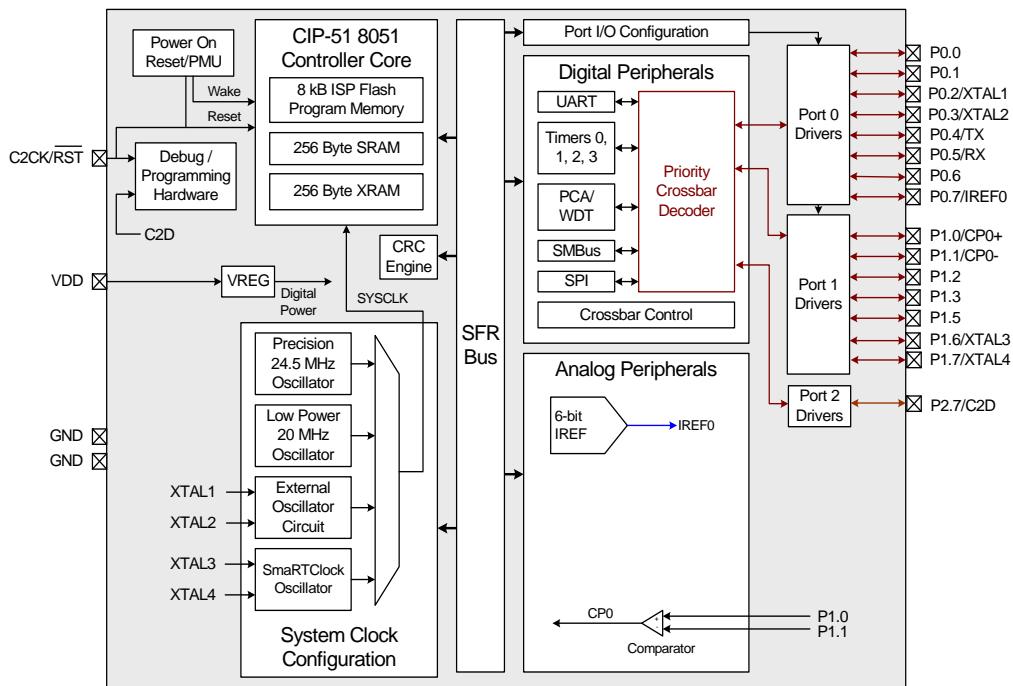


Figure 1.2. C8051F981 Block Diagram

# C8051F99x-C8051F98x

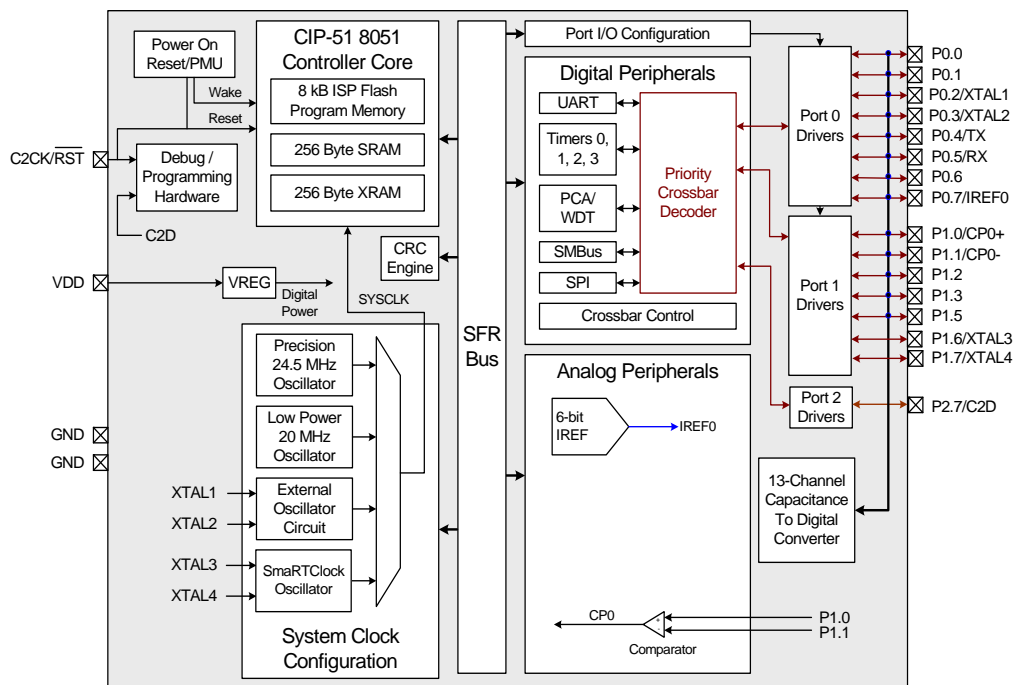


Figure 1.11. C8051F991 Block Diagram

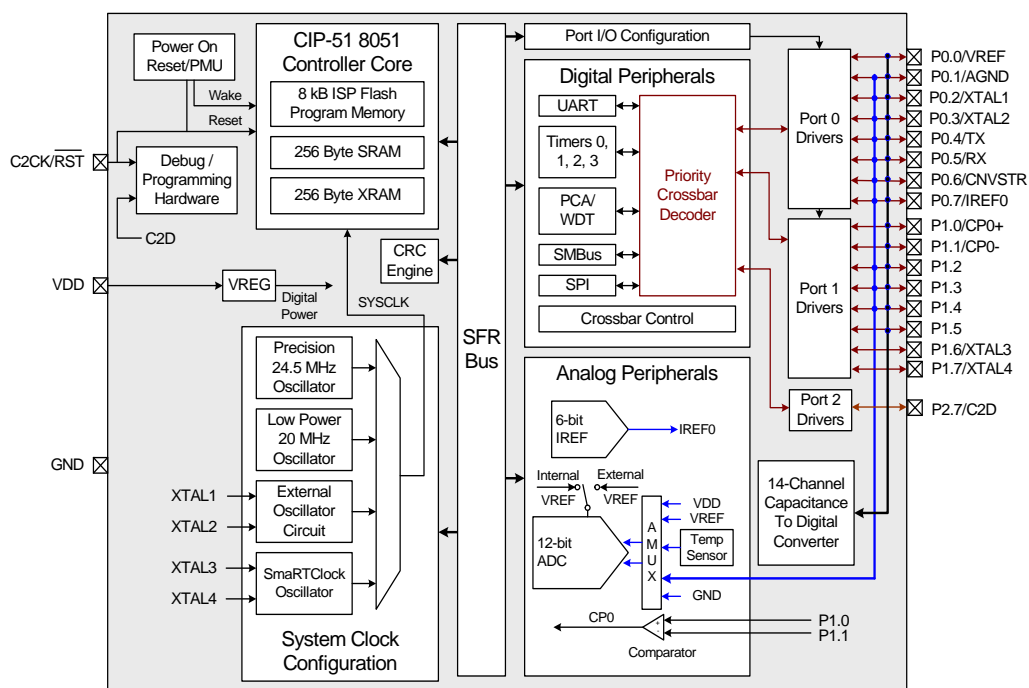


Figure 1.12. C8051F996 Block Diagram

## 5.2.4. Settling Time Requirements

A minimum amount of tracking time is required before each conversion can be performed, to allow the sampling capacitor voltage to settle. This tracking time is determined by the AMUX0 resistance, the ADC0 sampling capacitance, any external source resistance, and the accuracy required for the conversion. Note that in low-power tracking mode, three SAR clocks are used for tracking at the start of every conversion. For many applications, these three SAR clocks will meet the minimum tracking time requirements, and higher values for the external source impedance will increase the required tracking time.

Figure 5.4 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 5.1. When measuring the Temperature Sensor output or  $V_{DD}$  with respect to GND,  $R_{TOTAL}$  reduces to  $R_{MUX}$ . See Table 4.10 for ADC0 minimum settling time requirements as well as the mux impedance and sampling capacitor values.

$$t = \ln\left(\frac{2^n}{SA}\right) \times R_{TOTAL} C_{SAMPLE}$$

**Equation 5.1. ADC0 Settling Time Requirements**

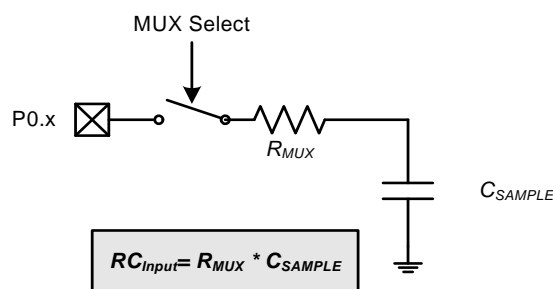
Where:

SA is the settling accuracy, given as a fraction of an LSB (for example, 0.25 to settle within 1/4 LSB)

$t$  is the required settling time in seconds

$R_{TOTAL}$  is the sum of the AMUX0 resistance and any external source resistance.

$n$  is the ADC resolution in bits (10).



**Note:** The value of CSAMPLE depends on the PGA Gain. See Table 4.10 for details.

**Figure 5.4. ADC0 Equivalent Input Circuits**

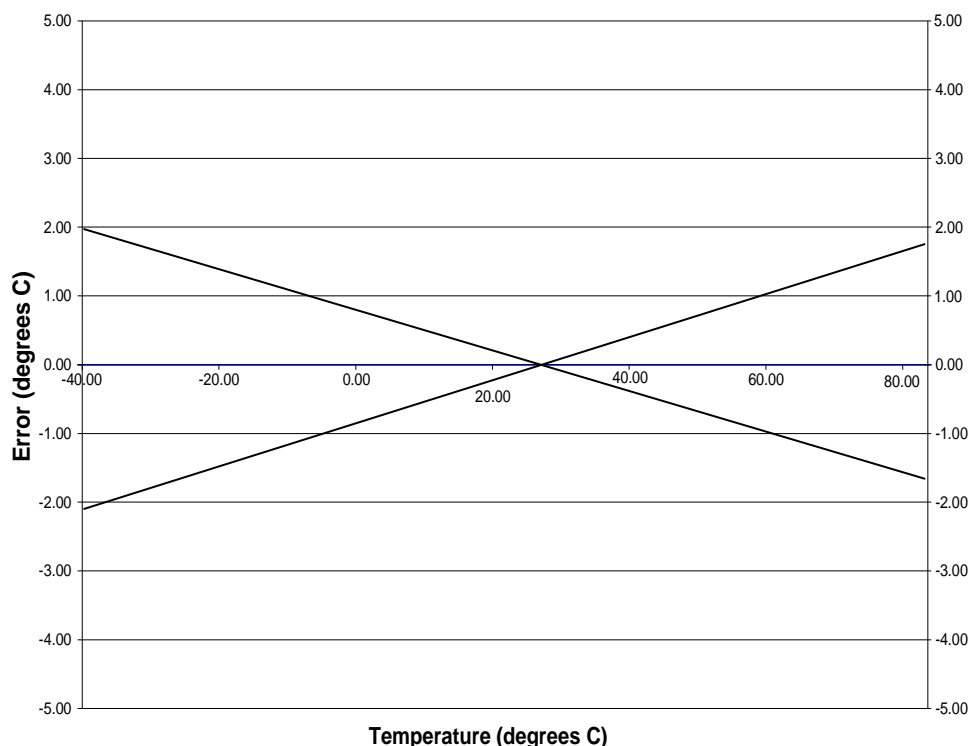
## 5.8.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements (see Table 4.11 for linearity specifications). For absolute temperature measurements, offset and/or gain calibration is recommended. Typically a 1-point (offset) calibration includes the following steps:

1. Control/measure the ambient temperature (this temperature must be known).
2. Power the device, and delay for a few seconds to allow for self-heating.
3. Perform an ADC conversion with the temperature sensor selected as the positive input and GND selected as the negative input.
4. Calculate the offset characteristics, and store this value in non-volatile memory for use with subsequent temperature sensor measurements.

Figure 5.9 shows the typical temperature sensor error assuming a 1-point calibration at 25 °C. **Parameters that affect ADC measurement, in particular the voltage reference value, will also affect temperature measurement.**

A single-point offset measurement of the temperature sensor is performed on each device during production test. The measurement is performed at 25 °C  $\pm$  5 °C, using the ADC with the internal high speed reference buffer selected as the Voltage Reference. The direct ADC result of the measurement is stored in the SFR registers TOFFH and TOFFL, shown in SFR Definition 5.13 and SFR Definition 5.14.



**Figure 5.9. Temperature Sensor Error with 1-Point Calibration ( $V_{REF} = 1.65$  V)**

## SFR Definition 6.2. IREF0CF: Current Reference Configuration

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

SFR Page = All; SFR Address = 0xB9

Bit	Name	Function
7	PWMEN	<b>PWM Enhanced Mode Enable.</b> Enables the PWM Enhanced Mode. 0: PWM Enhanced Mode disabled. 1: PWM Enhanced Mode enabled.
6:3	Unused	Read = 0000b, Write = don't care.
2:0	PWMSS[2:0]	<b>PWM Source Select.</b> Selects the PCA channel to use for the fine-tuning control signal. 000: CEX0 selected as fine-tuning control signal. 001: CEX1 selected as fine-tuning control signal. 010: CEX2 selected as fine-tuning control signal. All Other Values: Reserved.

## 6.2. IREF0 Specifications

See Table 4.13 on page 62 for a detailed listing of IREF0 specifications.

**SFR Definition 8.1. CS0CN: Capacitive Sense Control**

Bit	7	6	5	4	3	2	1	0
Name	CS0EN	CS0EOS	CS0INT	CS0BUSY	CS0CMPEN	Reserved	CS0PME	CS0CMPF
Type	R/W	R	R/W	R/W	R/W	R	R	R
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x0; SFR Address = 0xB0

Bit	Name	Description
7	CS0EN	<b>CS0 Enable.</b> 0: CS0 disabled and in low-power mode. 1: CS0 enabled and ready to convert.
6	CS0EOS	<b>CS0 End of Scan Interrupt Flag.</b> 0: CS0 has not completed a scan since the last time CS0EOS was cleared. 1: CS0 has completed a scan. This bit is not automatically cleared by hardware.
5	CS0INT	<b>CS0 Interrupt Flag.</b> 0: CS0 has not completed a data conversion since the last time CS0INT was cleared. 1: CS0 has completed a data conversion. This bit is not automatically cleared by hardware.
4	CS0BUSY	<b>CS0 Busy.</b> Read: 0: CS0 conversion is complete or a conversion is not currently in progress. 1: CS0 conversion is in progress. Write: 0: No effect. 1: Initiates CS0 conversion if CS0CM[2:0] = 000b, 110b, or 111b.
3	CS0CMPEN	<b>CS0 Digital Comparator Enable Bit.</b> Enables the digital comparator, which compares accumulated CS0 conversion output to the value stored in CS0THH:CS0THL. 0: CS0 digital comparator disabled. 1: CS0 digital comparator enabled.
2	Reserved	Read = Varies.
1	CS0PME	<b>CS0 Pin Monitor Event.</b> Set if any converter re-tries have occurred due to a pin monitor event. This bit remains set until cleared by firmware.
0	CS0CMPF	<b>CS0 Digital Comparator Interrupt Flag.</b> 0: CS0 result is smaller than the value set by CS0THH and CS0THL since the last time CS0CMPF was cleared. 1: CS0 result is greater than the value set by CS0THH and CS0THL since the last time CS0CMPF was cleared.

## SFR Definition 9.3. SP: Stack Pointer

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

SFR Page = All; SFR Address = 0x81

Bit	Name	Function
7:0	SP[7:0]	<b>Stack Pointer.</b> The Stack Pointer holds the location of the top of the stack. The stack pointer is incremented before every PUSH operation. The SP register defaults to 0x07 after reset.

## SFR Definition 9.4. ACC: Accumulator

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

SFR Page = All; SFR Address = 0xE0; Bit-Addressable

Bit	Name	Function
7:0	ACC[7:0]	<b>Accumulator.</b> This register is the accumulator for arithmetic operations.

## SFR Definition 9.5. B: B Register

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

SFR Page = All; SFR Address = 0xF0; Bit-Addressable

Bit	Name	Function
7:0	B[7:0]	<b>B Register.</b> This register serves as a second accumulator for certain arithmetic operations.



## 10.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051F99x-C8051F98x devices implement 8 kB (C8051F980/1/6/7, C8051F990/1/6/7), 4 kB (C8051F982/3/8/9), or 2 kB (C8051F985) of this program memory space as in-system, re-programmable Flash memory, organized in a contiguous block from addresses 0x0000 to 0x1FFF (8 kB devices), 0x0FFF (4 kB devices), or 0x07FF (2 kB devices). The last byte of this contiguous block of addresses serves as the security lock byte for the device. Any addresses above the lock byte are reserved.

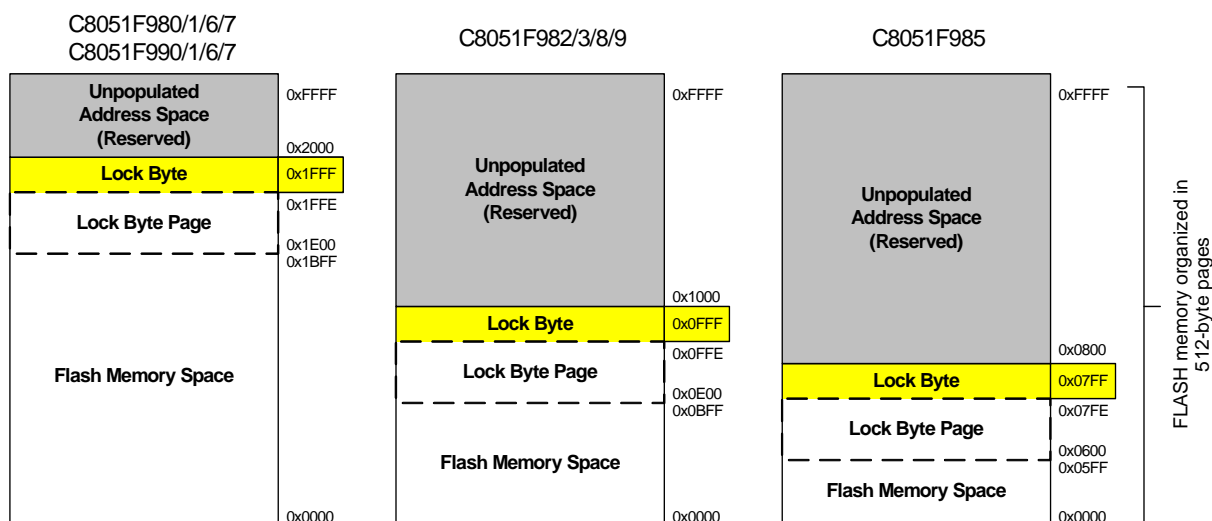


Figure 10.2. Flash Program Memory Map

### 10.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F99x-C8051F98x devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip Flash memory space. MOVC instructions are always used to read Flash memory, while MOVX write instructions are used to erase and write Flash. This Flash access feature provides a mechanism for the C8051F99x-C8051F98x to update program code and use the program memory space for non-volatile data storage. Refer to Section “14. Flash Memory” on page 150 for further details.

## 10.2. Data Memory

The C8051F99x-C8051F98x device family include 512 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. The remainder of this memory is on-chip “external” memory. The data memory map is shown in Figure 10.1 for reference.

### 10.2.1. Internal RAM

There are 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight byte-wide registers. The next 16 bytes, locations 0x20 through 0x2F, may either be addressed as bytes or as 128 bit locations accessible with the direct addressing mode.

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR

## 11. On-Chip XRAM

The C8051F99x-C8051F98x MCUs include on-chip RAM mapped into the external data memory space (XRAM). The external memory space may be accessed using the external move instruction (MOVX) with the target address specified in either the data pointer (DPTR), or with the target address low byte in R0 or R1. On C8051F99x-C8051F98x devices, the target address high byte is a don't care.

When using the MOVX instruction to access on-chip RAM, no additional initialization is required and the MOVX instruction execution time is as specified in the CIP-51 chapter.

**Important Note:** MOVX write operations can be configured to target Flash memory, instead of XRAM. See Section “14. Flash Memory” on page 150 for more details. The MOVX instruction accesses XRAM by default.

**Important Note:** On device reset or upon waking up from Sleep mode, address 0x0000 of external memory may be overwritten by an indeterminate value. The indeterminate value is 0x00 in most situations. A dummy variable should be placed at address 0x0000 in external memory to ensure that the application firmware does not store any data that needs to be retained during sleep or reset at this memory location.

### 11.1. Accessing XRAM

The XRAM memory space is accessed using the MOVX instruction. The MOVX instruction has two forms, both of which use an indirect addressing method. The first method uses the Data Pointer, DPTR, a 16-bit register which contains the effective address of the XRAM location to be read from or written to. The second method uses R0 or R1 to generate the effective XRAM address. Examples of both of these methods are given below.

#### 11.1.1. 16-Bit MOVX Example

The 16-bit form of the MOVX instruction accesses the memory location pointed to by the contents of the DPTR register. The following series of instructions reads the value of the byte at address 0x1234 into the accumulator A:

```
MOV    DPTR, #0034h      ; load DPTR with 16-bit address to read (0x0034)
MOVX   A, @DPTR          ; load contents of 0x0034 into accumulator A
```

The above example uses the 16-bit immediate MOV instruction to set the contents of DPTR. Alternately, the DPTR can be accessed through the SFR registers DPH, which contains the upper 8-bits of DPTR, and DPL, which contains the lower 8-bits of DPTR.

#### 11.1.2. 8-Bit MOVX Example

The 8-bit form of the MOVX instruction uses the contents of R0 or R1 to determine the 8-bits of the effective address to be accessed. The following series of instructions read the contents of the byte at address 0x0034 into the accumulator A.

```
MOV    R0, #34h          ; load low byte of address into R0 (or R1)
MOVX   a, @R0            ; load contents of 0x0034 into accumulator A
```

# C8051F99x-C8051F98x

## SFR Definition 16.5. CRC0CNT: CRC0 Automatic Flash Sector Count

Bit	7	6	5	4	3	2	1	0
Name				CRC0CNT[4:0]				
Type	R	R	R	R/W				
Reset	0	0	0	0	0	0	0	0

SFR Page = All; SFR Address = 0x9A

Bit	Name	Function
7:5	Unused	Read = 000b; Write = Don't Care.
4:0	CRC0CNT[4:0]	<b>Automatic CRC Calculation Block Count.</b> These bits specify the number of Flash blocks to include in an automatic CRC calculation. The last address of the last Flash block included in the automatic CRC calculation is $(CRC0ST + CRC0CNT) \times \text{Block Size} - 1$ .  <b>Notes:</b> <ol style="list-style-type: none"><li>1. The block size is 256 bytes.</li><li>2. The maximum number of blocks that may be computed in a single operation is 31. To compute a CRC on all 32 blocks, perform one operation on 31 blocks, then perform a second operation on 1 block without clearing the CRC result.</li></ol>

---

## 20.1.5. RTC0ADR Autoincrement Feature

For ease of reading and writing the 32-bit CAPTURE and ALARM values, RTC0ADR automatically increments after each read or write to a CAPTUREn or ALARMn register. This speeds up the process of setting an alarm or reading the current SmarTClock timer value. Autoincrement is always enabled.

Recommended Instruction Timing for a multi-byte register read with short strobe and auto read enabled:

```
mov RTC0ADR, #0d0h
nop
nop
nop
nop
mov A, RTC0DAT
nop
nop
mov A, RTC0DAT
nop
nop
mov A, RTC0DAT
nop
nop
mov A, RTC0DAT
```

Recommended Instruction Timing for a multi-byte register write with short strobe enabled:

```
mov RTC0ADR, #010h
mov RTC0DAT, #05h
nop
mov RTC0DAT, #06h
nop
mov RTC0DAT, #07h
nop
mov RTC0DAT, #08h
nop
```

## 21. Port Input/Output

Digital and analog resources are available through 16 or 17 I/O pins. Port pins are organized as three byte-wide ports. Port pins P0.0–P1.7 can be defined as digital or analog I/O. Digital I/O pins can be assigned to one of the internal digital resources or used as general purpose I/O (GPIO). Analog I/O pins are used by the internal analog resources. P2.7 can be used as GPIO and is shared with the C2 Interface Data signal (C2D). See Section “27. C2 Interface” on page 319 for more details.

The designer has complete control over which digital and analog functions are assigned to individual Port pins, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a Priority Crossbar Decoder. See Section 21.3 for more information on the Crossbar.

All Port I/Os can tolerate voltages up to the supply rail when used as digital inputs or open-drain outputs. For Port I/Os configured as push-pull outputs, current is sourced from the  $V_{DD}$  supply. Port I/Os used for analog functions can operate up to the  $V_{DD}$  supply voltage. See Section 21.1 for more information on Port I/O operating modes and the electrical specifications chapter for detailed electrical specifications.

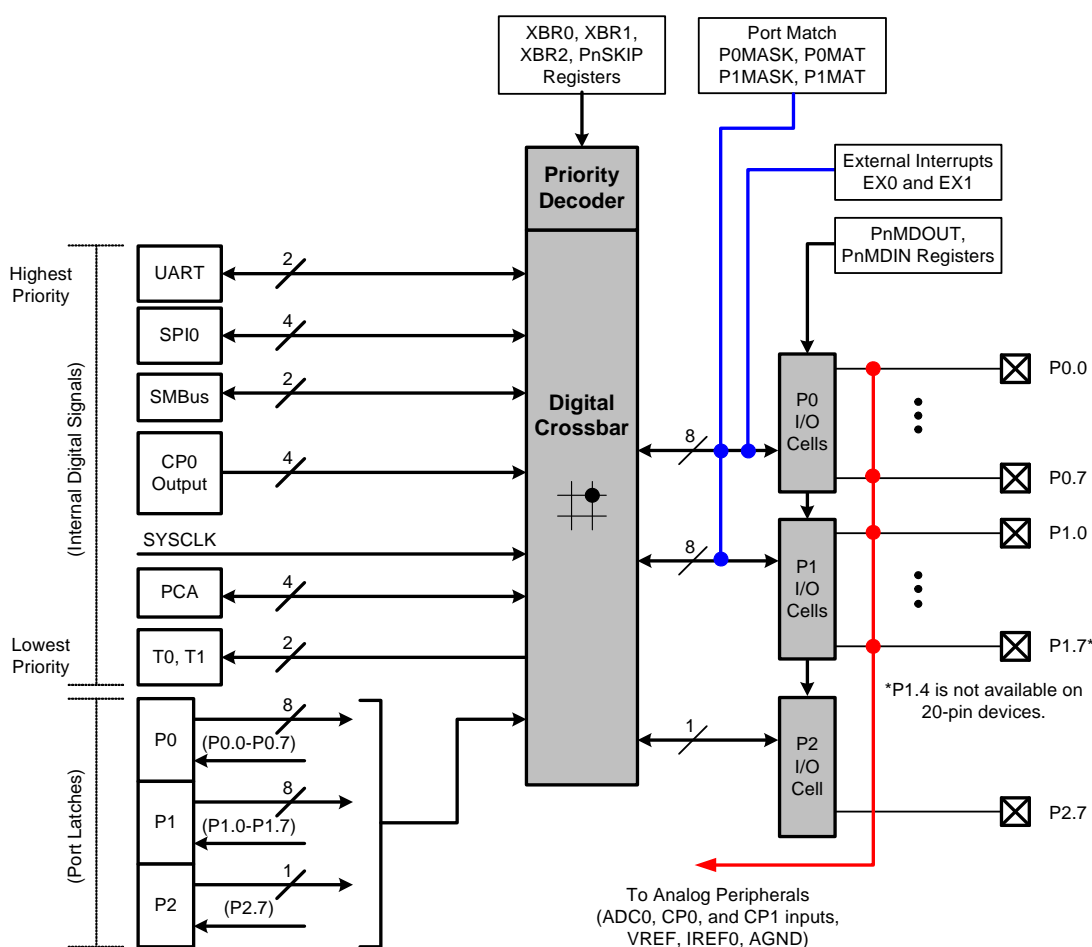


Figure 21.1. Port I/O Functional Block Diagram

## 21.1. Port I/O Modes of Operation

Port pins P0.0–P1.7 use the Port I/O cell shown in Figure 21.2. Each Port I/O cell can be configured by software for analog I/O or digital I/O using the PnMDIN registers. On reset, all Port I/O cells default to a digital high impedance state with weak pull-ups enabled.

### 21.1.1. Port Pins Configured for Analog I/O

Any pins to be used as Comparator or ADC input, oscillator input/output, or AGND, VREF, or Current Reference output should be configured for analog I/O (PnMDIN.n = 0). When a pin is configured for analog I/O, its weak pullup and digital receiver are disabled. In most cases, software should also disable the digital output drivers. Port pins configured for analog I/O will always read back a value of 0 regardless of the actual voltage on the pin.

Configuring pins as analog I/O saves power and isolates the Port pin from digital interference. Port pins configured as digital inputs may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

### 21.1.2. Port Pins Configured For Digital I/O

Any pins to be used by digital peripherals (UART, SPI, SMBus, etc.), external digital event capture functions, or as GPIO should be configured as digital I/O (PnMDIN.n = 1). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PnMDOUT registers.

Push-pull outputs (PnMDOUT.n = 1) drive the Port pad to the VDD or GND supply rails based on the output logic value of the Port pin. Open-drain outputs have the high side driver disabled; therefore, they only drive the Port pad to GND when the output logic value is 0 and become high impedance inputs (both high and low drivers turned off) when the output logic value is 1.

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the Port pad to the VDD supply voltage to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven to GND to minimize power consumption and may be globally disabled by setting WEAKPUD to 1. The user must ensure that digital I/O are always internally or externally pulled or driven to a valid logic state. Port pins configured for digital I/O always read back the logic state of the Port pad, regardless of the output logic value of the Port pin.

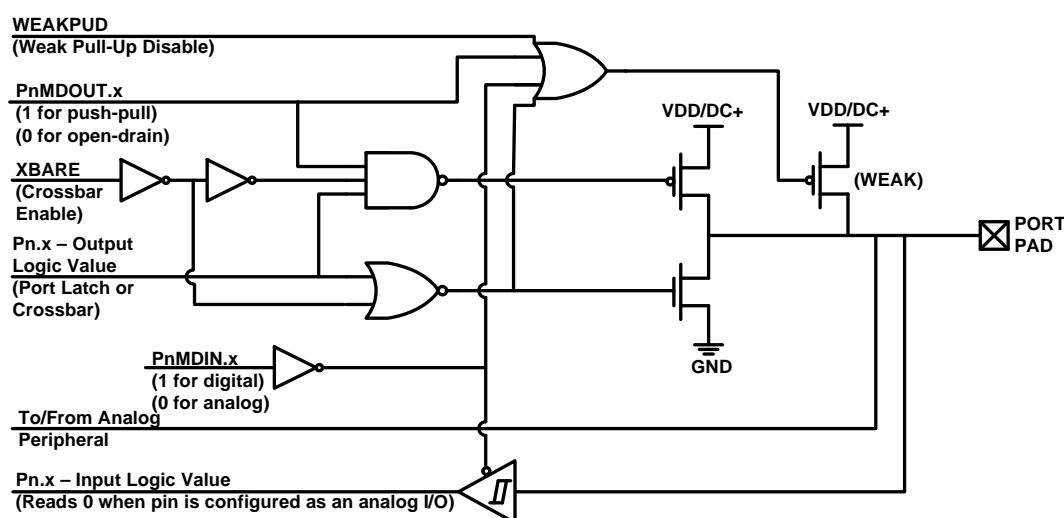


Figure 21.2. Port I/O Cell Block Diagram

## SFR Definition 21.6. P1MASK: Port1 Mask Register

Bit	7	6	5	4	3	2	1	0
Name	P1MASK[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Page= 0x0; SFR Address = 0xBF

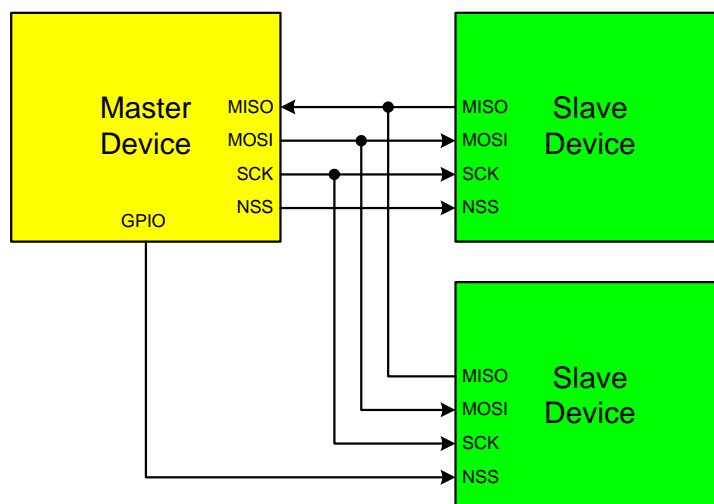
Bit	Name	Function
7:0	P1MASK[7:0]	<b>Port 1 Mask Value.</b> Selects P1 pins to be compared to the corresponding bits in P1MAT. 0: P1.n pin logic value is ignored and cannot cause a Port Mismatch event. 1: P1.n pin logic value is compared to P1MAT.n.

## SFR Definition 21.7. P1MAT: Port1 Match Register

Bit	7	6	5	4	3	2	1	0
Name	P1MAT[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Page = 0x0; SFR Address = 0xCF

Bit	Name	Function
7:0	P1MAT[7:0]	<b>Port 1 Match Value.</b> Match comparison value used on Port 1 for bits in P1MASK which are set to 1. 0: P1.n pin logic value is compared with logic LOW. 1: P1.n pin logic value is compared with logic HIGH.



**Figure 24.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram**

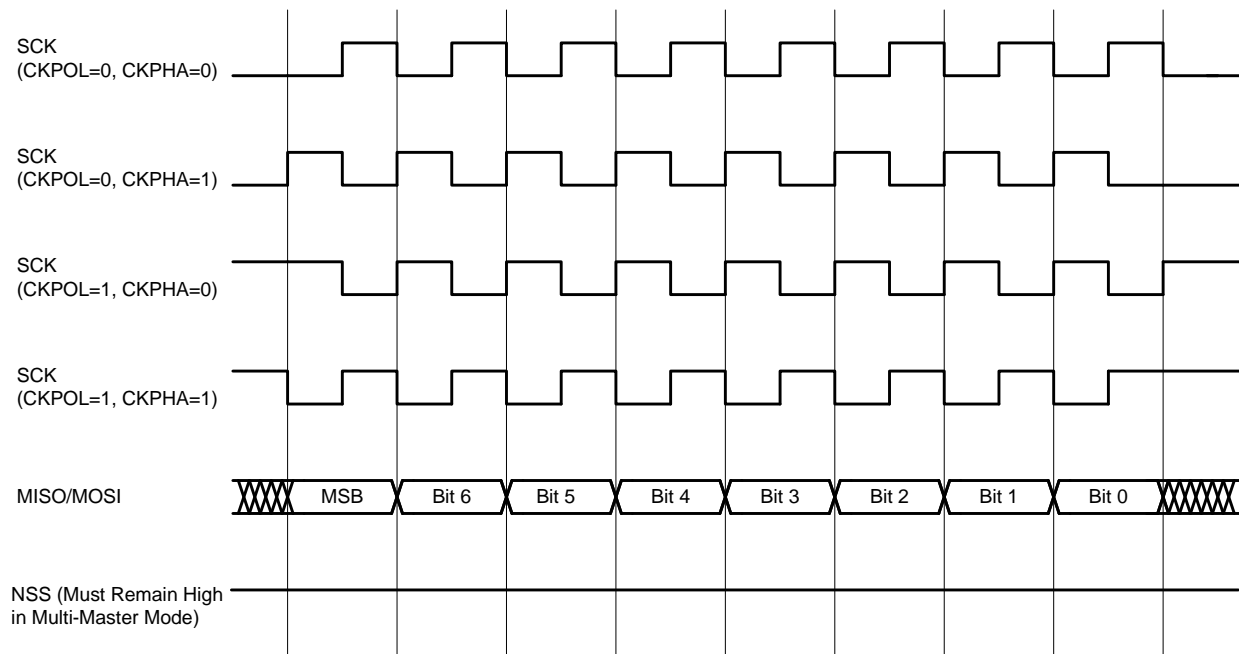
### 24.3. SPI0 Slave Mode Operation

When SPI0 is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted through the shift register, the SPIF flag is set to logic 1, and the byte is copied into the receive buffer. Data is read from the receive buffer by reading SPI0DAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the shift register by writing to SPI0DAT. Writes to SPI0DAT are double-buffered, and are placed in the transmit buffer first. If the shift register is empty, the contents of the transmit buffer will immediately be transferred into the shift register. When the shift register already contains data, the SPI will load the shift register with the transmit buffer's contents after the last SCK edge of the next (or current) SPI transfer.

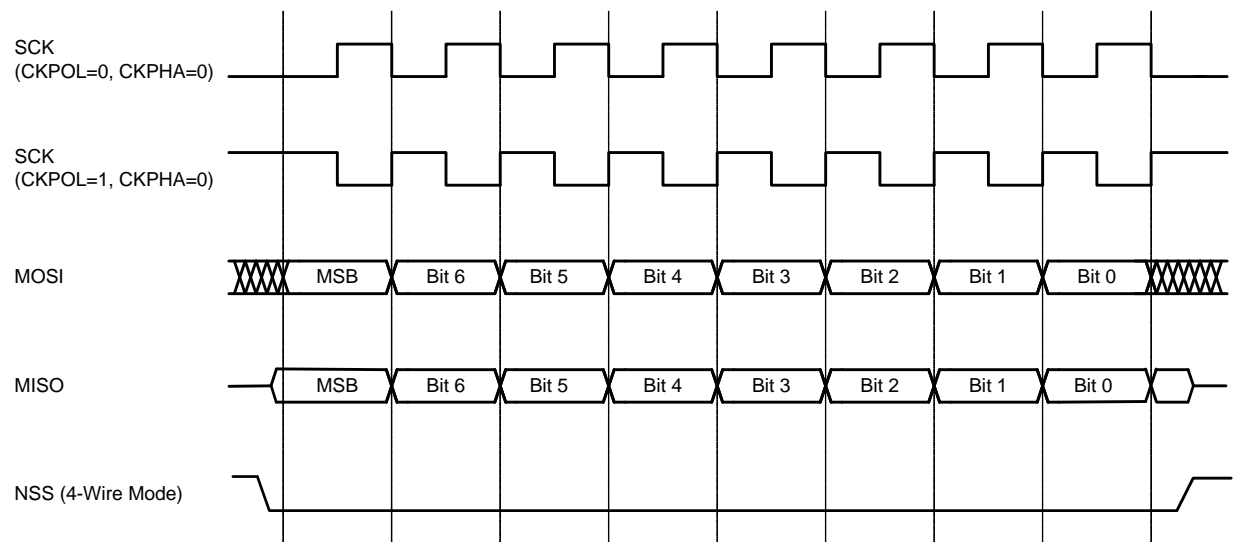
When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. The default, 4-wire slave mode, is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In 4-wire mode, the NSS signal is routed to a port pin and configured as a digital input. SPI0 is enabled when NSS is logic 0, and disabled when NSS is logic 1. The bit counter is reset on a falling edge of NSS. Note that the NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 24.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and is not mapped to an external port pin through the crossbar. Since there is no way of uniquely addressing the device in 3-wire slave mode, SPI0 must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling SPI0 with the SPIEN bit. Figure 24.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.





**Figure 24.5. Master Mode Data/Clock Timing**



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

## SFR Definition 24.3. SPI0CKR: SPI0 Clock Rate

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

SFR Page = 0x0; SFR Address = 0xA2

Bit	Name	Function
7:0	SCR[7:0]	<p><b>SPI0 Clock Rate.</b></p> <p>These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where <i>SYSCCLK</i> is the system clock frequency and <i>SPI0CKR</i> is the 8-bit value held in the SPI0CKR register.</p> $f_{SCK} = \frac{SYSCCLK}{2 \times (SPI0CKR[7:0] + 1)}$ <p>for <math>0 \leq SPI0CKR \leq 255</math></p> <p>Example: If <i>SYSCCLK</i> = 2 MHz and <i>SPI0CKR</i> = 0x04,</p> $f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$ $f_{SCK} = 200kHz$

## SFR Definition 24.4. SPI0DAT: SPI0 Data

Bit	7	6	5	4	3	2	1	0
Name	SPI0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x0; SFR Address = 0xA3

Bit	Name	Function
7:0	SPI0DAT[7:0]	<p><b>SPI0 Transmit and Receive Data.</b></p> <p>The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0DAT places the data into the transmit buffer and initiates a transfer when in Master Mode. A read of SPI0DAT returns the contents of the receive buffer.</p>

## SFR Definition 25.6. TH0: Timer 0 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x0; SFR Address = 0x8C

Bit	Name	Function
7:0	TH0[7:0]	<b>Timer 0 High Byte.</b> The TH0 register is the high byte of the 16-bit Timer 0.

## SFR Definition 25.7. TH1: Timer 1 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Page = 0x0; SFR Address = 0x8D

Bit	Name	Function
7:0	TH1[7:0]	<b>Timer 1 High Byte.</b> The TH1 register is the high byte of the 16-bit Timer 1.

## 25.3.3. SmarTClock/External Oscillator Capture Mode

The Capture Mode in Timer 3 allows either SmarTClock or the external oscillator period to be measured against the system clock or the system clock divided by 12. SmarTClock and the external oscillator period can also be compared against each other.

Setting TF3CEN to 1 enables the SmarTClock/External Oscillator Capture Mode for Timer 3. In this mode, T3SPLIT should be set to 0, as the full 16-bit timer is used.

When Capture Mode is enabled, a capture event will be generated either every SmarTClock rising edge or every 8 external clock cycles, depending on the T3XCLK1 setting. When the capture event occurs, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set (triggering an interrupt if Timer 3 interrupts are enabled). By recording the difference between two successive timer capture values, the SmarTClock or external clock period can be determined with respect to the Timer 3 clock. The Timer 3 clock should be much faster than the capture clock to achieve an accurate reading.

For example, if T3ML = 1b, T3XCLK1 = 0b, and TF3CEN = 1b, Timer 3 will clock every SYSCLK and capture every SmarTClock rising edge. If SYSCLK is 24.5 MHz and the difference between two successive captures is 350 counts, then the SmarTClock period is as follows:

$$350 \times (1 / 24.5 \text{ MHz}) = 14.2 \mu\text{s}.$$

This mode allows software to determine the exact frequency of the external oscillator in C and RC mode or the time between consecutive SmarTClock rising edges, which is useful for determining the SmarTClock frequency.

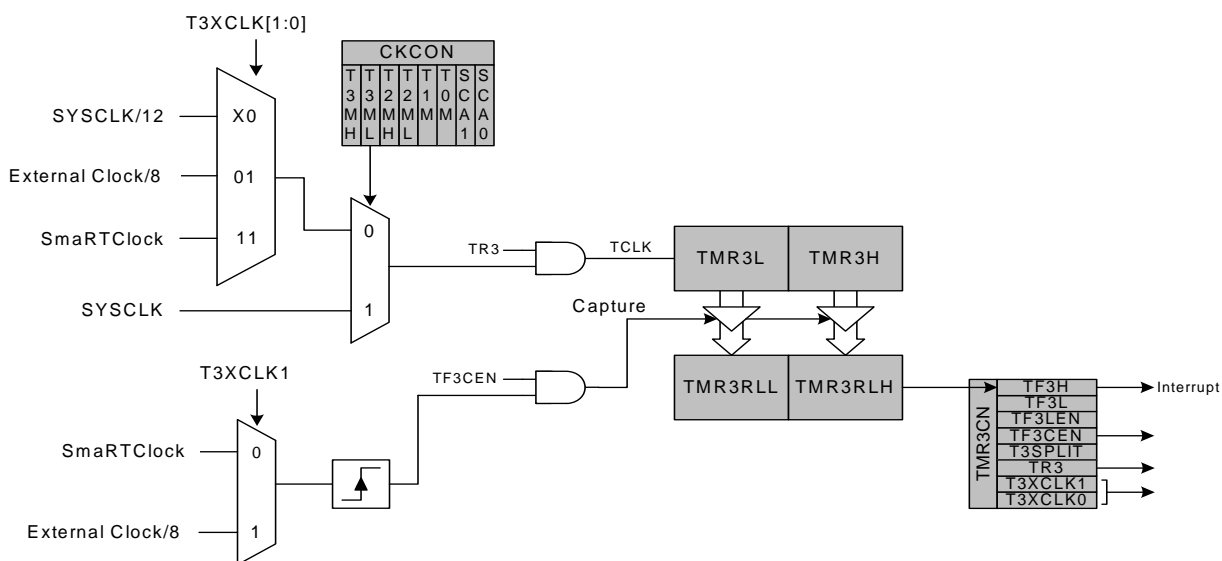
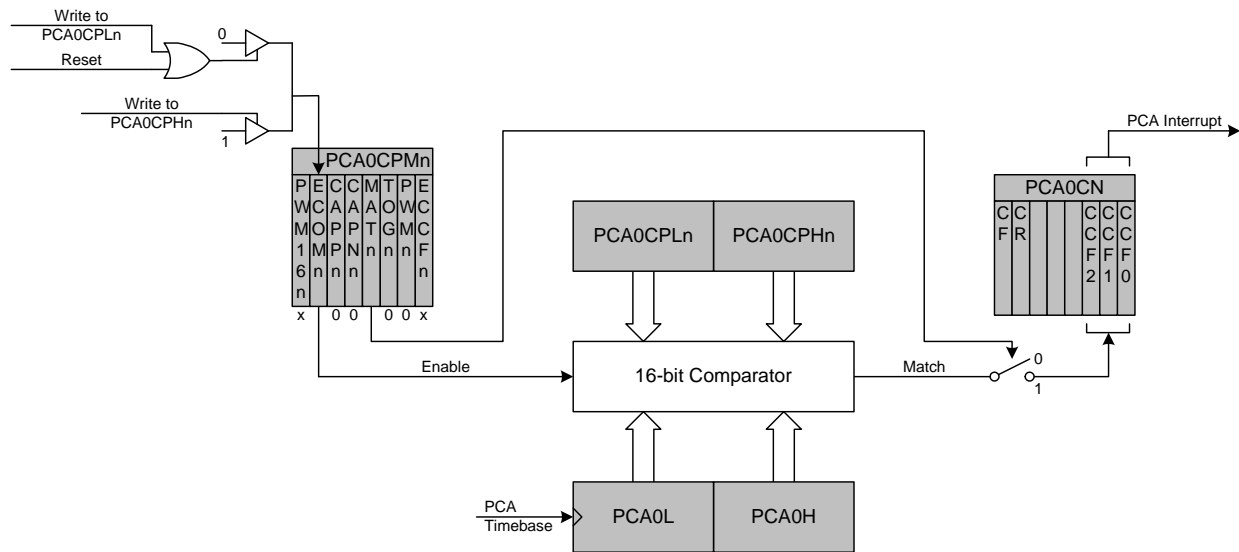


Figure 25.9. Timer 3 Capture Mode Block Diagram

## 26.3.2. 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. 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 26.5. PCA Software Timer Mode Diagram**