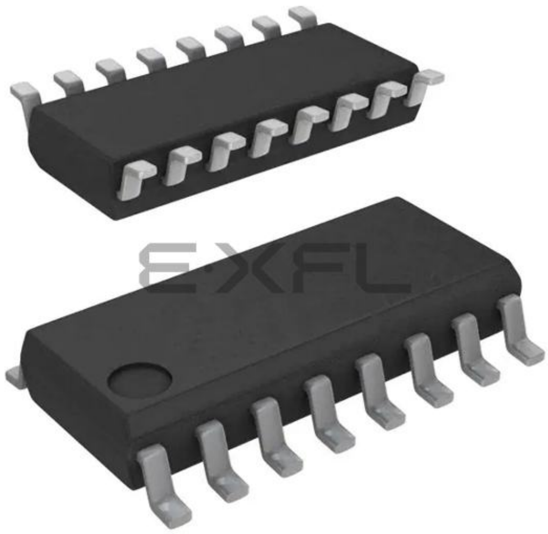


Welcome to E-XFL.COM

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	Cap Sense, POR, PWM, Temp Sensor, WDT
Number of I/O	13
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	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	16-SOIC (0.154", 3.90mm Width)
Supplier Device Package	16-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f815-gsr

List of Figures

1. System Overview

C8051F80x-83x devices are fully integrated, mixed-signal, system-on-a-chip capacitive sensing MCUs. Highlighted features are listed below. Refer to Table 2.1 for specific product feature selection and part ordering numbers.

- High-speed pipelined 8051-compatible microcontroller core (up to 25 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- Capacitive sense interface with 16 input channels
- 10-bit 500 ksps single-ended ADC with 16-channel analog multiplexer and integrated temperature sensor
- Precision calibrated 24.5 MHz internal oscillator
- 16 kb of on-chip Flash memory
- 512 bytes of on-chip RAM
- SMBus/I^{2}C, Enhanced UART, and Enhanced SPI serial interfaces implemented in hardware
- Three general-purpose 16-bit timers
- Programmable counter/timer array (PCA) with three capture/compare modules
- On-chip internal voltage reference
- On-chip Watchdog timer
- On-chip power-on reset and supply monitor
- On-chip voltage comparator
- 17 general purpose I/O

With on-chip power-on reset, V_{DD} monitor, watchdog timer, and clock oscillator, the C8051F80x-83x devices are truly stand-alone, system-on-a-chip solutions. The Flash memory can be reprogrammed even in-circuit, providing non-volatile data storage, and also allowing field upgrades of the 8051 firmware. User software has complete control of all peripherals, and may individually shut down any or all peripherals for power savings.

The C8051F80x-83x processors include Silicon Laboratories' 2-Wire C2 Debug and Programming interface, which allows non-intrusive (uses no on-chip resources), full speed, in-circuit debugging using the production MCU installed in the final application. This debug logic supports inspection of memory, viewing and modification of special function registers, setting breakpoints, single stepping, and run and halt commands. All analog and digital peripherals are fully functional while debugging using C2. The two C2 interface pins can be shared with user functions, allowing in-system debugging without occupying package pins.

Each device is specified for 1.8–3.6 V operation over the industrial temperature range (–45 to +85 °C). An internal LDO regulator is used to supply the processor core voltage at 1.8 V. The Port I/O and $\overline{\text{RST}}$ pins are tolerant of input signals up to 5 V. See Table 2.1 for ordering information. Block diagrams of the devices in the C8051F80x-83x family are shown in Figure 1.1 through Figure 1.9.

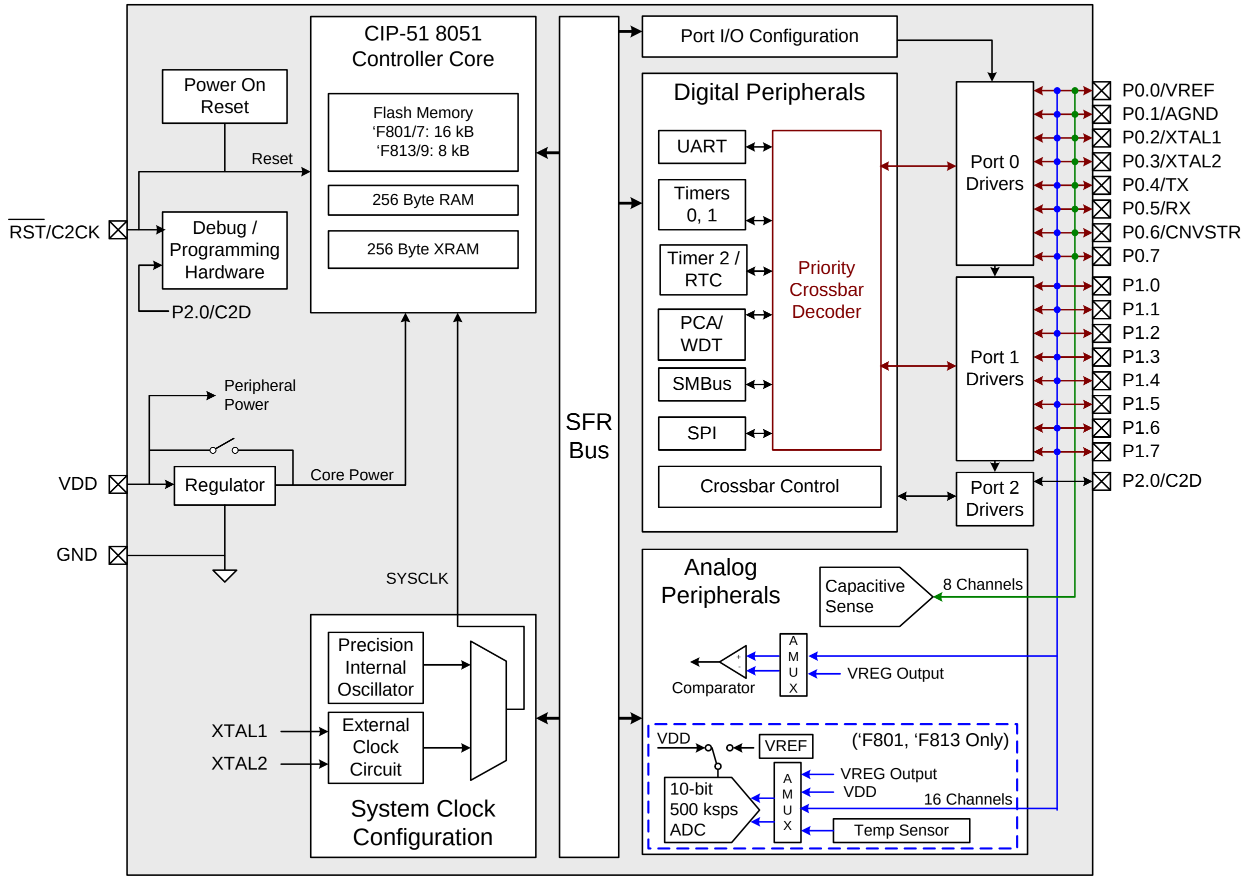


Figure 1.2. C8051F801, C8051F807, C8051F813, C8051F819 Block Diagram

8.5. ADC0 Analog Multiplexer

ADC0 on the C8051F800/1/2/3/4/5, C8051F812/3/4/5/6/7, C8051F824/5/6, and C8051F830/1/2 uses an analog input multiplexer to select the positive input to the ADC. Any of the following may be selected as the positive input: Port 0 or Port 1 I/O pins, the on-chip temperature sensor, or the positive power supply (V_{DD}). The ADC0 input channel is selected in the ADC0MX register described in SFR Definition 8.9.

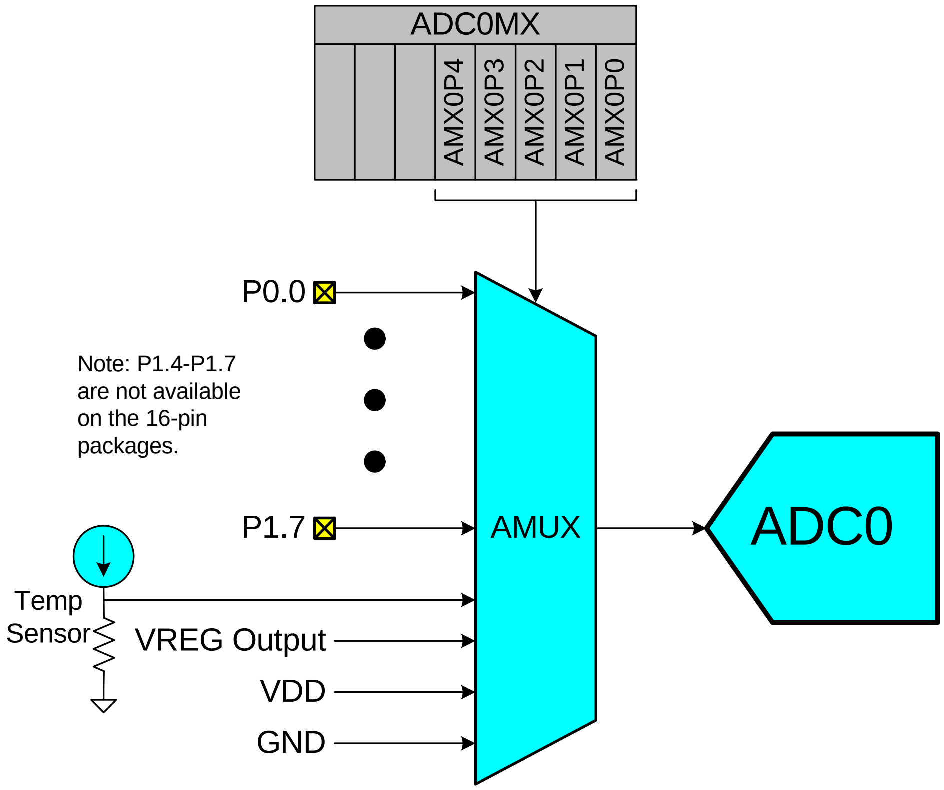


Figure 8.6. ADC0 Multiplexer Block Diagram

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 the corresponding bit in register PnMDIN to 0. To force the Crossbar to skip a Port pin, set the corresponding bit in register PnSKIP to 1. See Section “23. Port Input/Output” on page 138 for more Port I/O configuration details.

SFR Definition 12.1. CPT0CN: Comparator0 Control

Bit	7	6	5	4	3	2	1	0
Name	CP0EN	CP0OUT	CP0RIF	CP0FIF	CP0HYP[1:0]		CP0HYN[1:0]	
Type	R/W	R	R/W	R/W	R/W		R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9B

Bit	Name	Function
7	CP0EN	**Comparator0 Enable Bit.** 0: Comparator0 Disabled. 1: Comparator0 Enabled.
6	CP0OUT	**Comparator0 Output State Flag.** 0: Voltage on CP0+ < CP0–. 1: Voltage on CP0+ > CP0–.
5	CP0RIF	**Comparator0 Rising-Edge Flag. Must be cleared by software.** 0: No Comparator0 Rising Edge has occurred since this flag was last cleared. 1: Comparator0 Rising Edge has occurred.
4	CP0FIF	**Comparator0 Falling-Edge Flag. Must be cleared by software.** 0: No Comparator0 Falling-Edge has occurred since this flag was last cleared. 1: Comparator0 Falling-Edge has occurred.
3:2	CP0HYP[1:0]	**Comparator0 Positive Hysteresis Control Bits.** 00: Positive Hysteresis Disabled. 01: Positive Hysteresis = 5 mV. 10: Positive Hysteresis = 10 mV. 11: Positive Hysteresis = 20 mV.
1:0	CP0HYN[1:0]	**Comparator0 Negative Hysteresis Control Bits.** 00: Negative Hysteresis Disabled. 01: Negative Hysteresis = 5 mV. 10: Negative Hysteresis = 10 mV. 11: Negative Hysteresis = 20 mV.

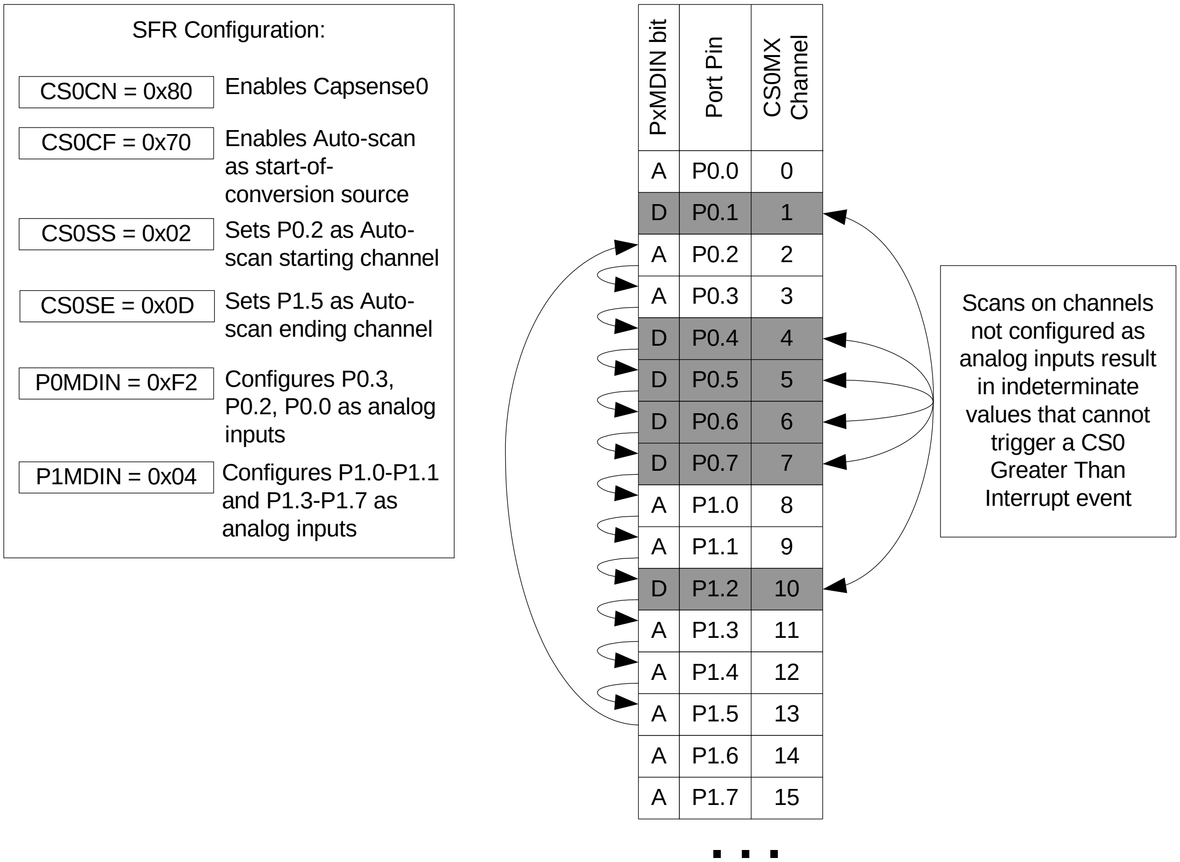


Figure 13.2. Auto-Scan Example

13.4. CS0 Comparator

The CS0 comparator compares the latest capacitive sense conversion result with the value stored in CS0THH:CS0THL. If the result is less than or equal to the stored value, the CS0CMPF bit(CS0CN:0) is set to 0. If the result is greater than the stored value, CS0CMPF is set to 1.

If the CS0 conversion accumulator is configured to accumulate multiple conversions, a comparison will not be made until the last conversion has been accumulated.

An interrupt will be generated if CS0 greater-than comparator interrupts are enabled by setting the ECS-GRT bit (EIE2.1) when the comparator sets CS0CMPF to 1.

If auto-scan is running when the comparator sets the CS0CMPF bit, no further auto-scan initiated conversions will start until firmware sets CS0BUSY to 1.

A CS0 greater-than comparator event can wake a device from suspend mode. This feature is useful in systems configured to continuously sample one or more capacitive sense channels. The device will remain in the low-power suspend state until the captured value of one of the scanned channels causes a CS0 greater-than comparator event to occur. It is not necessary to have CS0 comparator interrupts enabled in order to wake a device from suspend with a greater-than event.

Note: On waking from suspend mode due to a CS0 greater-than comparator event, the CS0CN register should be accessed only after at least two system clock cycles have elapsed.

For a summary of behavior with different CS0 comparator, auto-scan, and auto accumulator settings, please see Table 13.1.

14. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 30), and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control-system solution in a single integrated circuit.

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability (see Figure 14.1 for a block diagram). The CIP-51 includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 25 MIPS Peak Throughput with 25 MHz Clock
- 0 to 25 MHz Clock Frequency
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

Performance

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. In a standard 8051, all instructions except for MUL and DIV take 12 or 24 system clock cycles to execute, and usually have a maximum system clock of 12 MHz. By contrast, the CIP-51 core executes 70% of its instructions in one or two system clock cycles, with no instructions taking more than eight system clock cycles.

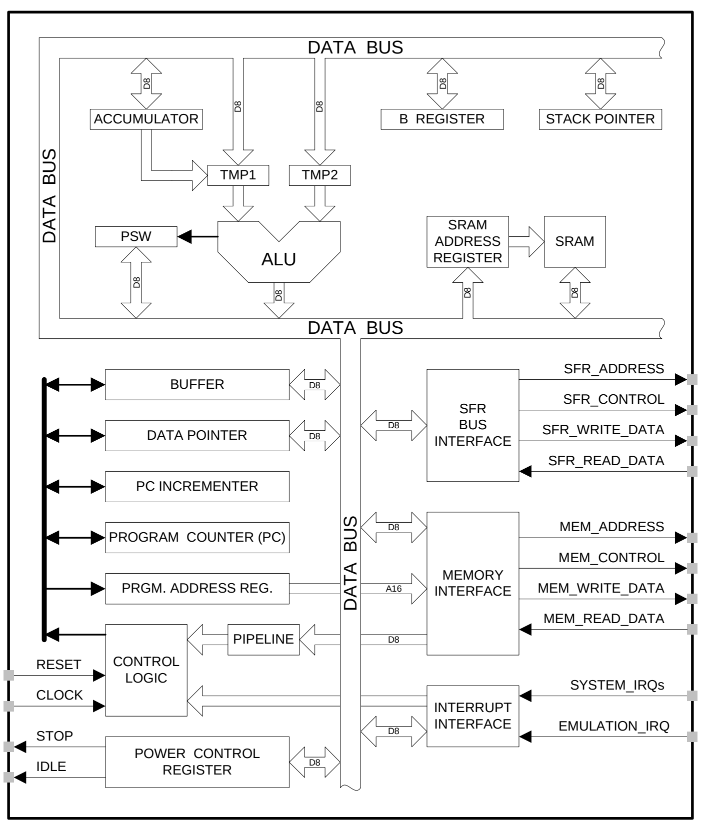


Figure 14.1. CIP-51 Block Diagram

Table 17.2. Special Function Registers (Continued)

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

Register	Address	Description	Page
CS0CF	0x9E	CS0 Configuration	76
CS0MX	0x9C	CS0 Mux	81
CS0SE	0xBA	Auto Scan End Channel	78
CS0SS	0xB9	Auto Scan Start Channel	78
DERIVID	0xAD	Derivative Identification	96
DPH	0x83	Data Pointer High	88
DPL	0x82	Data Pointer Low	88
EIE1	0xE6	Extended Interrupt Enable 1	107
EIE2	0xE7	Extended Interrupt Enable 2	108
EIP1	0xF3	Extended Interrupt Priority 1	109
EIP2	0xF4	Extended Interrupt Priority 2	110
FLKEY	0xB7	Flash Lock And Key	119
HWID	0xB5	Hardware Identification	95
IE	0xA8	Interrupt Enable	105
IP	0xB8	Interrupt Priority	106
IT01CF	0xE4	INT0/INT1 Configuration	112
OSCICL	0xB3	Internal Oscillator Calibration	131
OSCICN	0xB2	Internal Oscillator Control	132
OSCXCN	0xB1	External Oscillator Control	134
P0	0x80	Port 0 Latch	153
P0MASK	0xFE	Port 0 Mask	151
P0MAT	0xFD	Port 0 Match	151
P0MDIN	0xF1	Port 0 Input Mode Configuration	154
P0MDOUT	0xA4	Port 0 Output Mode Configuration	154
P0SKIP	0xD4	Port 0 Skip	155
P1	0x90	Port 1 Latch	155
P1MASK	0xEE	P0 Mask	152

SFR Definition 18.1. IE: Interrupt Enable

Bit	7	6	5	4	3	2	1	0
Name	EA	ESPI0	ET2	ES0	ET1	EX1	ET0	EX0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

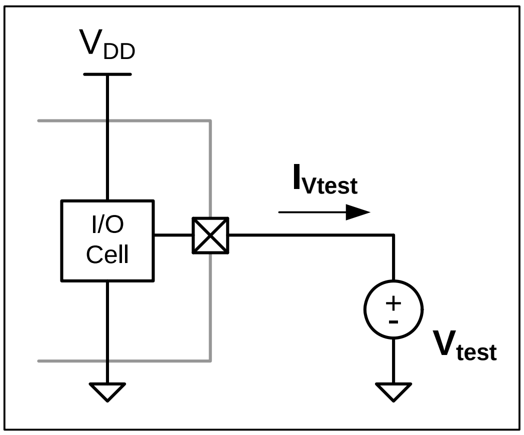
SFR Address = 0xA8; Bit-Addressable

Bit	Name	Function
7	EA	**Enable All Interrupts.** Globally enables/disables all interrupts. It overrides individual interrupt mask settings. 0: Disable all interrupt sources. 1: Enable each interrupt according to its individual mask setting.
6	ESPI0	**Enable Serial Peripheral Interface (SPI0) Interrupt.** This bit sets the masking of the SPI0 interrupts. 0: Disable all SPI0 interrupts. 1: Enable interrupt requests generated by SPI0.
5	ET2	**Enable Timer 2 Interrupt.** This bit sets the masking of the Timer 2 interrupt. 0: Disable Timer 2 interrupt. 1: Enable interrupt requests generated by the TF2L or TF2H flags.
4	ES0	**Enable UART0 Interrupt.** This bit sets the masking of the UART0 interrupt. 0: Disable UART0 interrupt. 1: Enable UART0 interrupt.
3	ET1	**Enable Timer 1 Interrupt.** This bit sets the masking of the Timer 1 interrupt. 0: Disable all Timer 1 interrupt. 1: Enable interrupt requests generated by the TF1 flag.
2	EX1	**Enable External Interrupt 1.** This bit sets the masking of External Interrupt 1. 0: Disable external interrupt 1. 1: Enable interrupt requests generated by the $\overline{INT1}$ input.
1	ET0	**Enable Timer 0 Interrupt.** This bit sets the masking of the Timer 0 interrupt. 0: Disable all Timer 0 interrupt. 1: Enable interrupt requests generated by the TF0 flag.
0	EX0	**Enable External Interrupt 0.** This bit sets the masking of External Interrupt 0. 0: Disable external interrupt 0. 1: Enable interrupt requests generated by the $\overline{INT0}$ input.

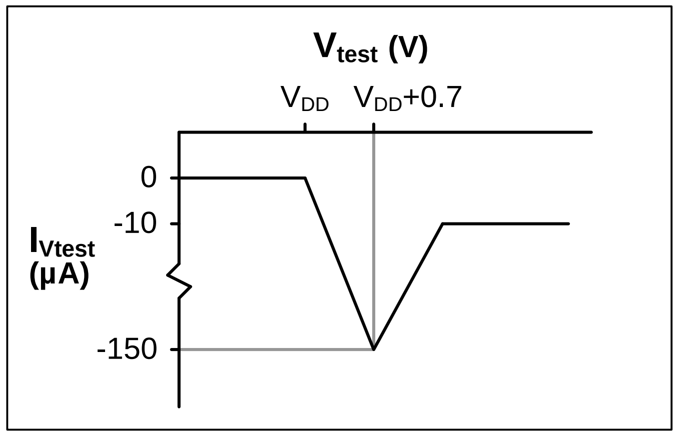
23.1.3. Interfacing Port I/O to 5 V Logic

All Port I/O configured for digital, open-drain operation are capable of interfacing to digital logic operating at a supply voltage up to 2 V higher than VDD and less than 5.25 V. An external pull-up resistor to the higher supply voltage is typically required for most systems.

Important Note: In a multi-voltage interface, the external pull-up resistor should be sized to allow a current of at least 150 µA to flow into the Port pin when the supply voltage is between (VDD + 0. 6V) and (VDD + 1.0V). Once the Port pin voltage increases beyond this range, the current flowing into the Port pin is minimal. Figure 23.3 shows the input current characteristics of port pins driven above VDD. The port pin requires 150 µA peak overdrive current when its voltage reaches approximately (VDD + 0.7 V).



Port I/O Overdrive Test Circuit



Port I/O Overdrive Current vs. Voltage

Figure 23.3. Port I/O Overdrive Current

23.2. Assigning Port I/O Pins to Analog and Digital Functions

Port I/O pins P0.0–P1.7 can be assigned to various analog, digital, and external interrupt functions. The Port pins assigned to analog functions should be configured for analog I/O, and Port pins assigned to digital or external interrupt functions should be configured for digital I/O.

23.2.1. Assigning Port I/O Pins to Analog Functions

Table 23.1 shows all available analog functions that require Port I/O assignments. **Port pins selected for these analog functions should have their corresponding bit in PnSKIP set to 1.** This reserves the pin for use by the analog function and does not allow it to be claimed by the Crossbar. **Any selected pins should also have their corresponding bit in the Port Latch set to 1 (Pn.n = 1).** This prevents the low port I/O drive circuit from pulling the pin low. Table 23.1 shows the potential mapping of Port I/O to each analog function.

SFR Definition 23.3. P0MASK: Port 0 Mask Register

Bit	7	6	5	4	3	2	1	0
Name	P0MASK[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFE

Bit	Name	Function
7:0	P0MASK[7:0]	**Port 0 Mask Value.** Selects P0 pins to be compared to the corresponding bits in P0MAT. 0: P0.n pin logic value is ignored and cannot cause a Port Mismatch event. 1: P0.n pin logic value is compared to P0MAT.n.

SFR Definition 23.4. P0MAT: Port 0 Match Register

Bit	7	6	5	4	3	2	1	0
Name	P0MAT[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xFD

Bit	Name	Function
7:0	P0MAT[7:0]	**Port 0 Match Value.** Match comparison value used on Port 0 for bits in P0MASK which are set to 1. 0: P0.n pin logic value is compared with logic LOW. 1: P0.n pin logic value is compared with logic HIGH.

24.2. 32-bit CRC Algorithm

The C8051F80x-83x CRC unit calculates the 32-bit CRC using a poly of 0x04C11DB7. The CRC-32 algorithm is "reflected", meaning that all of the input bytes and the final 32-bit output are bit-reversed in the processing engine. The following is a description of a simplified CRC algorithm that produces results identical to the hardware:

1. XOR the least-significant byte of the current CRC result with the input byte. If this is the first iteration of the CRC unit, the current CRC result will be the set initial value (0x00000000 or 0xFFFFFFFF).
2. Right-shift the CRC result.
3. If the LSB of the CRC result is set, XOR the CRC result with the reflected polynomial (0xEDB88320).
4. Repeat at Step 2 for the number of input bits (8).

For example, the 32-bit C8051F80x-83x CRC algorithm can be described by the following code:

```
unsigned long UpdateCRC (unsigned long CRC_acc, unsigned char CRC_input){
    unsigned char i; // loop counter
    #define POLY 0xEDB88320 // bit-reversed version of the poly 0x04C11DB7
    // Create the CRC "dividend" for polynomial arithmetic (binary arithmetic
    // with no carries)
    CRC_acc = CRC_acc ^ CRC_input;
    // "Divide" the poly into the dividend using CRC XOR subtraction
    // CRC_acc holds the "remainder" of each divide
    // Only complete this division for 8 bits since input is 1 byte
    for (i = 0; i < 8; i++)
    {
        // Check if the MSB is set (if MSB is 1, then the POLY can "divide"
        // into the "dividend")
        if ((CRC_acc & 0x00000001) == 0x00000001)
        {
            // if so, shift the CRC value, and XOR "subtract" the poly
            CRC_acc = CRC_acc >> 1;
            CRC_acc ^= POLY;
        }
        else
        {
            // if not, just shift the CRC value
            CRC_acc = CRC_acc >> 1;
        }
    }
    return CRC_acc; // Return the final remainder (CRC value)
}
```

Table 24.2 lists example input values and the associated outputs using the 32-bit C8051F80x-83x CRC algorithm (an initial value of 0xFFFFFFFF is used):

Table 24.2. Example 32-bit CRC Outputs

Input	Output
0x63	0xF9462090
0xAA, 0xBB, 0xCC	0x41B207B3
0x00, 0x00, 0xAA, 0xBB, 0xCC	0x78D129BC

24.3. Preparing for a CRC Calculation

To prepare CRC0 for a CRC calculation, software should select the desired polynomial and set the initial value of the result. Two polynomials are available: 0x1021 (16-bit) and 0x04C11DB7 (32-bit). The CRC0 result may be initialized to one of two values: 0x00000000 or 0xFFFFFFFF. The following steps can be used to initialize CRC0.

1. Select a polynomial (Set CRC0SEL to 0 for 32-bit or 1 for 16-bit).
2. Select the initial result value (Set CRC0VAL to 0 for 0x00000000 or 1 for 0xFFFFFFFF).
3. Set the result to its initial value (Write 1 to CRC0INIT).

24.4. Performing a CRC Calculation

Once CRC0 is initialized, the input data stream is sequentially written to CRC0IN, one byte at a time. The CRC0 result is automatically updated after each byte is written. The CRC engine may also be configured to automatically perform a CRC on one or more Flash sectors. The following steps can be used to automatically perform a CRC on Flash memory.

1. Prepare CRC0 for a CRC calculation as shown above.
2. Write the index of the starting page to CRC0AUTO.
3. Set the AUTOEN bit in CRC0AUTO.
4. Write the number of Flash sectors to perform in the CRC calculation to CRC0CNT.

Note: Each Flash sector is 512 bytes.

5. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes.
6. Clear the AUTOEN bit in CRC0AUTO.
7. Read the CRC result using the procedure below.

24.5. Accessing the CRC0 Result

The internal CRC0 result is 32-bits (CRC0SEL = 0b) or 16-bits (CRC0SEL = 1b). The CRC0PNT bits select the byte that is targeted by read and write operations on CRC0DAT and increment after each read or write. The calculation result will remain in the internal CR0 result register until it is set, overwritten, or additional data is written to CRC0IN.

24.6. CRC0 Bit Reverse Feature

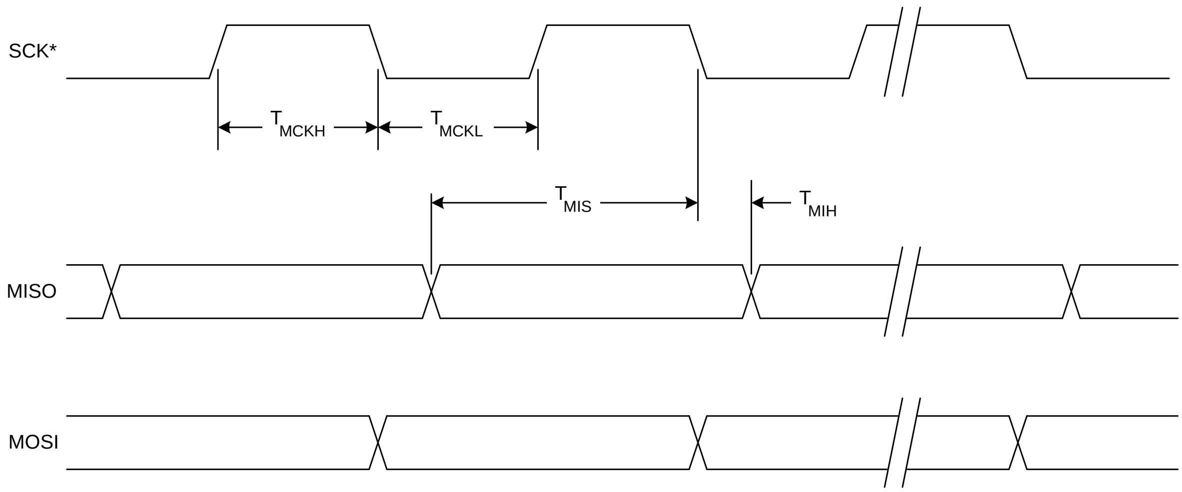
CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 24.1. Each byte of data written to CRC0FLIP is read back bit reversed. For example, if 0xC0 is written to CRC0FLIP, the data read back is 0x03. Bit reversal is a useful mathematical function used in algorithms such as the FFT.

SFR Definition 24.6. CRC0FLIP: CRC Bit Flip

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CRC0FLIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

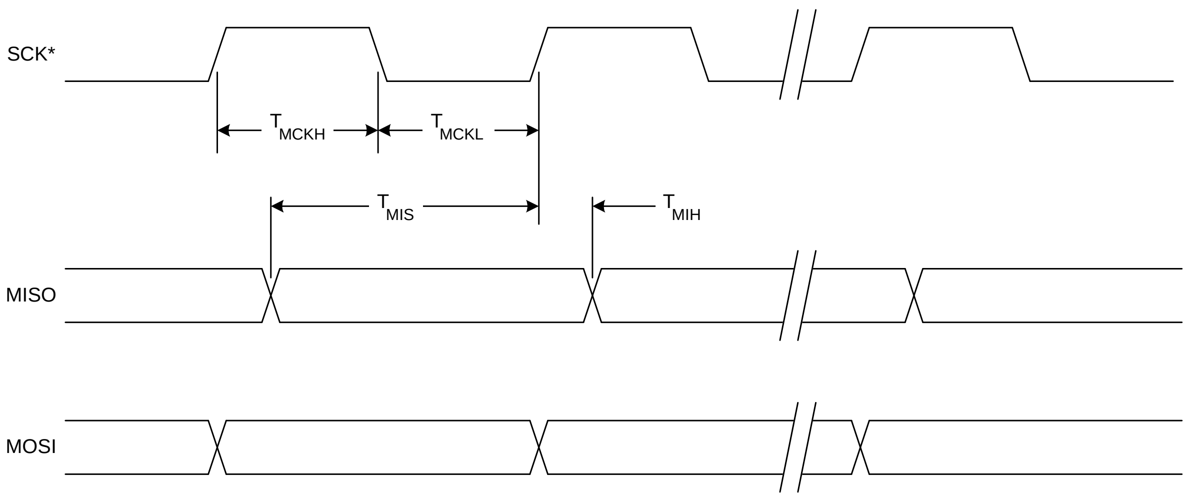
SFR Address = 0xCF

Bit	Name	Function
7:0	CRC0FLIP[7:0]	**CRC0 Bit Flip.** Any byte written to CRC0FLIP is read back in a bit-reversed order, i.e. the written LSB becomes the MSB. For example: If 0xC0 is written to CRC0FLIP, the data read back will be 0x03. If 0x05 is written to CRC0FLIP, the data read back will be 0xA0.



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 25.8. SPI Master Timing (CKPHA = 0)



* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 25.9. SPI Master Timing (CKPHA = 1)

29.3.3. High-Speed Output Mode

In high-speed output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and 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 TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the high-speed output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

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.

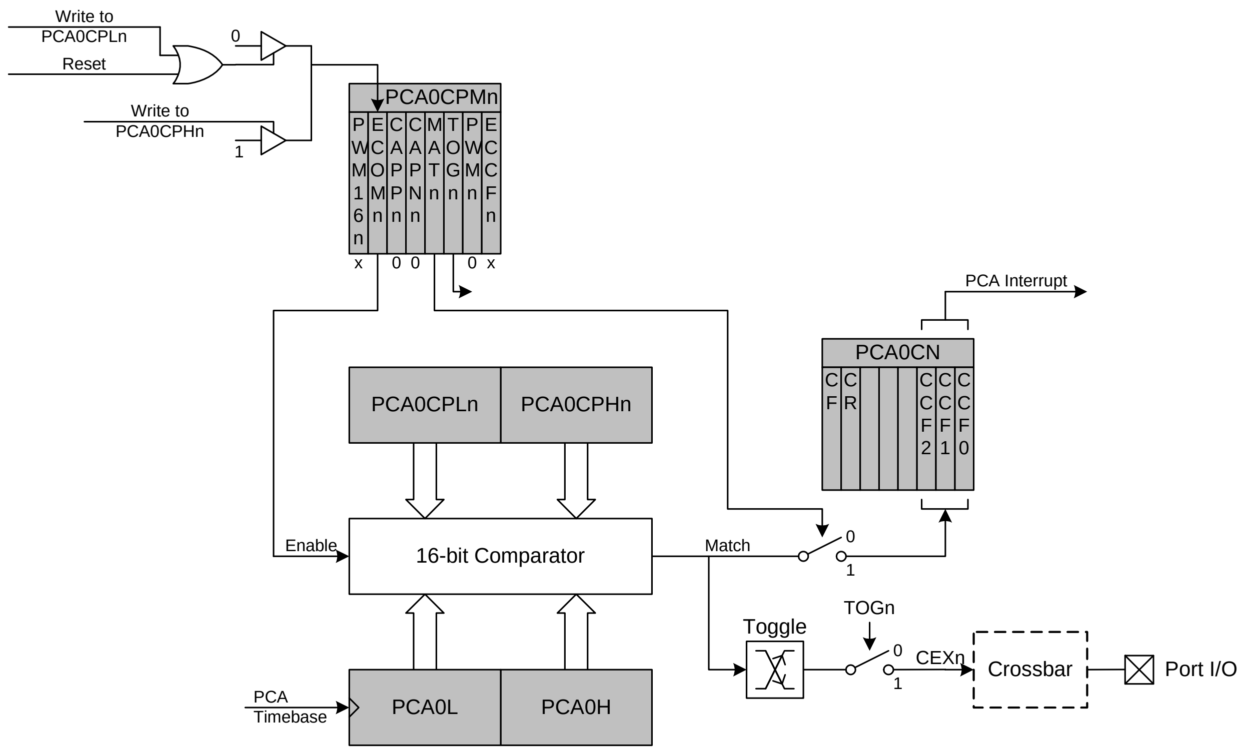


Figure 29.6. PCA High-Speed Output Mode Diagram

29.3.6. 16-Bit Pulse Width Modulator Mode

A PCA module may be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8-bit through 15-bit) PWM modes. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is asserted high; when the 16-bit counter overflows, CEXn is asserted low. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register.

The duty cycle of the PWM output signal can be varied by writing to an “Auto-Reload” Register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0. This synchronous update feature allows software to asynchronously write a new PWM high time, which will then take effect on the following PWM period.

For backwards-compatibility with the 16-bit PWM mode available on other devices, the PWM duty cycle can also be changed without using the “Auto-Reload” register. To output a varying duty cycle without using the “Auto-Reload” register, new value writes should be synchronized with PCA CCFn match interrupts. Match interrupts should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a 16-bit comparator match (rising edge) occurs. The CF flag in PCA0CN can be used to detect the overflow (falling edge). The duty cycle for 16-Bit PWM Mode is given by Equation 29.4.

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.

$$\text{Duty Cycle} = \frac{(65536 - PCA0CPn)}{65536}$$

Equation 29.4. 16-Bit PWM Duty Cycle

Using Equation 29.4, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

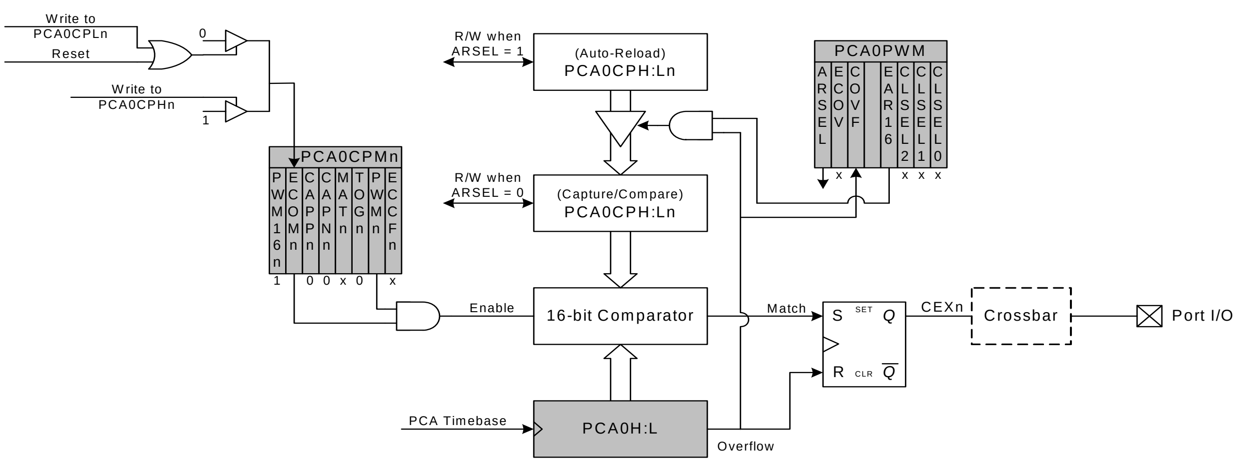


Figure 29.10. PCA 16-Bit PWM Mode

SFR Definition 29.5. PCA0L: PCA0 Counter/Timer Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF9

Bit	Name	Function
7:0	PCA0[7:0]	**PCA Counter/Timer Low Byte.** The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.

Note: When the WDTE bit is set to 1, the PCA0L register cannot be modified by software. To change the contents of the PCA0L register, the Watchdog Timer must first be disabled.

SFR Definition 29.6. PCA0H: PCA0 Counter/Timer High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0[15:8]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFA

Bit	Name	Function
7:0	PCA0[15:8]	**PCA Counter/Timer High Byte.** The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a “snapshot” register, whose contents are updated only when the contents of PCA0L are read (see Section 29.1).

Note: When the WDTE bit is set to 1, the PCA0H register cannot be modified by software. To change the contents of the PCA0H register, the Watchdog Timer must first be disabled.