Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	25MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	24
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4.25K x 8
Voltage - Supply (Vcc/Vdd)	0.9V ~ 3.6V
Data Converters	A/D 23x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f930-g-gm

21.5.Special Function Registers for Accessing and Configuring Port I/O 229
22. SMBus 238
22.1.Supporting Documents 239
22.2.SMBus Configuration 239
22.3.SMBus Operation 240
22.3.1.Transmitter Vs. Receiver 240
22.3.2.Arbitration 241
22.3.3.Clock Low Extension 241
22.3.4.SCL Low Timeout 241
22.3.5.SCL High (SMBus Free) Timeout 241
22.4.Using the SMBus 242
22.4.1.SMBus Configuration Register 243
22.4.2.SMB0CN Control Register 246
22.4.3.Hardware Slave Address Recognition 249
22.4.4.Data Register 251
22.5.SMBus Transfer Modes 252
22.5.1.Write Sequence (Master) 252
22.5.2.Read Sequence (Master) 253
22.5.3.Write Sequence (Slave) 254
22.5.4.Read Sequence (Slave) 255
22.6.SMBus Status Decoding 255
23. UART0 260
23.1.Enhanced Baud Rate Generation 261
23.2.Operational Modes 262
23.2.1.8-Bit UART 262
23.2.2.9-Bit UART 263
23.3.Multiprocessor Communications 263
24. Enhanced Serial Peripheral Interface (SPI0 and SPI1) 268
24.1.Signal Descriptions 269
24.1.1.Master Out, Slave In (MOSI) 269
24.1.2.Master In, Slave Out (MISO) 269
24.1.3.Serial Clock (SCK) 269
24.1.4.Slave Select (NSS) 269
24.2.SPI Master Mode Operation 270
24.3.SPI Slave Mode Operation 272
24.4.SPI Interrupt Sources 272
24.5.Serial Clock Phase and Polarity 273
24.6.SPI Special Function Registers 275
25. Timers 283
25.1.Timer 0 and Timer 1 285
25.1.1.Mode 0: 13-bit Counter/Timer 285
25.1.2.Mode 1: 16-bit Counter/Timer 286
25.1.3.Mode 2: 8-bit Counter/Timer with Auto-Reload 287
25.1.4.Mode 3: Two 8-bit Counter/Timers (Timer 0 Only) 288
25.2.Timer 2 293

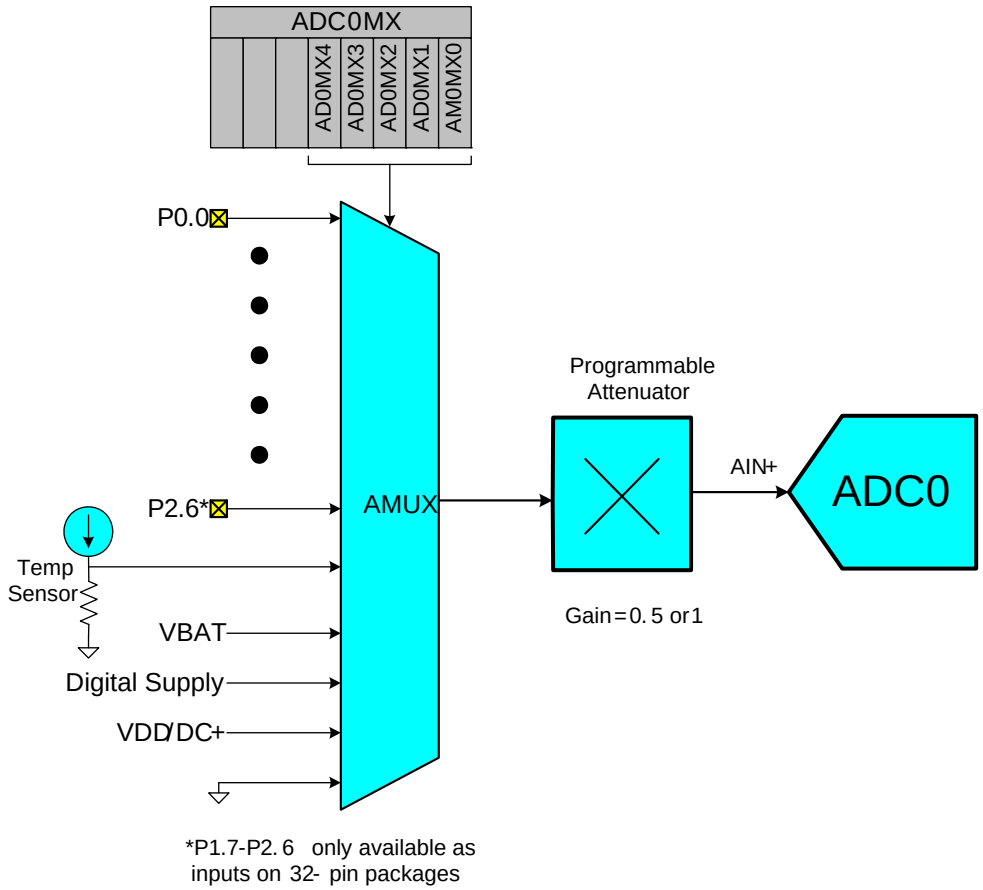


Figure 1.8. ADC0 Multiplexer Block Diagram

1.6. Programmable Current Reference (IREF0)

C8051F93x-C8051F92x devices include an on-chip programmable current reference (source or sink) with two output current settings: low power mode and high current mode. The maximum current output in low power mode is 63 µA (1 µA steps) and the maximum current output in high current mode is 504 µA (8 µA steps).

1.7. Comparators

C8051F93x-C8051F92x devices include two on-chip programmable voltage comparators: Comparator 0 (CPT0) which is shown in Figure 1.9; Comparator 1 (CPT1) which is shown in Figure 1.10. The two comparators operate identically but may differ in their ability to be used as reset or wake-up sources. See Section “18. Reset Sources” on page 184 and the Section “14. Power Management” on page 159 for details on reset sources and low power mode wake-up sources, respectively.

The Comparator offers programmable response time and hysteresis, an analog input multiplexer, and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0, CP1), or an asynchronous “raw” output (CP0A, CP1A). The asynchronous CP0A signal is available even when the system clock is not active. This allows the Comparator to operate and generate an output when the device is in some low power modes.

The comparator inputs may be connected to Port I/O pins or to other internal signals. Port pins may also be used to directly sense capacitive touch switches.

2. Ordering Information

Table 2.1. Product Selection Guide

Ordering Part Number	MIPS (Peak)	Flash Memory (kB)	RAM (bytes)	SmaRTClock Real Time Clock	SMBus/I^2C	UART	Enhanced SPI	Timers (16-bit)	Programmable Counter Array	Digital Port I/Os	10-bit 300ksps ADC	Programmable Current Reference	Internal Voltage Reference	Temperature Sensor	Analog Comparators	Lead-free (RoHS Compliant)	Package
C8051F930-G-GM	25	64	4352	✓	1	1	2	4	✓	24	✓	✓	✓	✓	2	✓	QFN-32
C8051F930-G-GQ	25	64	4352	✓	1	1	2	4	✓	24	✓	✓	✓	✓	2	✓	LQFP-32
C8051F931-G-GM	25	64	4352	✓	1	1	2	4	✓	16	✓	✓	✓	✓	2	✓	QFN-24
C8051F920-G-GM	25	32	4352	✓	1	1	2	4	✓	24	✓	✓	✓	✓	2	✓	QFN-32
C8051F920-G-GQ	25	32	4352	✓	1	1	2	4	✓	24	✓	✓	✓	✓	2	✓	LQFP-32
C8051F921-G-GM	25	32	4352	✓	1	1	2	4	✓	16	✓	✓	✓	✓	2	✓	QFN-24

***Note:** Starting with silicon revision F, the ordering part numbers have been updated to include the silicon revision and use this format: "C8051F930-F-GM". Package marking diagrams are included as Figure 3.4, Figure 3.5, and Figure 3.6 to help identify the silicon revision.

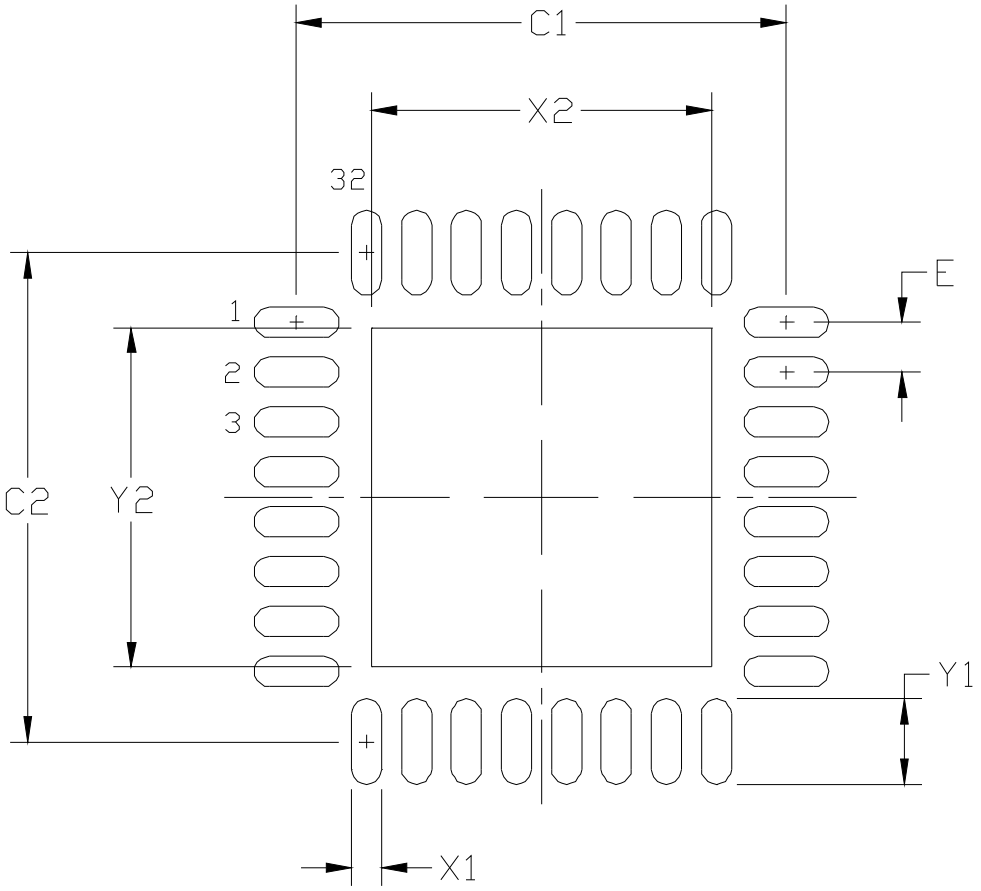


Figure 3.8. Typical QFN-32 Landing Diagram

5.4.1. Window Detector In Single-Ended Mode

Figure 5.5 shows two example window comparisons for right-justified data, with ADC0LTH:ADC0LTL = 0x0080 (128d) and ADC0GTH:ADC0GTL = 0x0040 (64d). The input voltage can range from 0 to VREF x (1023/1024) with respect to GND, and is represented by a 10-bit unsigned integer value. In the left example, an AD0WINT interrupt will be generated if the ADC0 conversion word (ADC0H:ADC0L) is within the range defined by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL (if 0x0040 < ADC0H:ADC0L < 0x0080). In the right example, and AD0WINT interrupt will be generated if the ADC0 conversion word is outside of the range defined by the ADC0GT and ADC0LT registers (if ADC0H:ADC0L < 0x0040 or ADC0H:ADC0L > 0x0080). Figure 5.6 shows an example using left-justified data with the same comparison values.

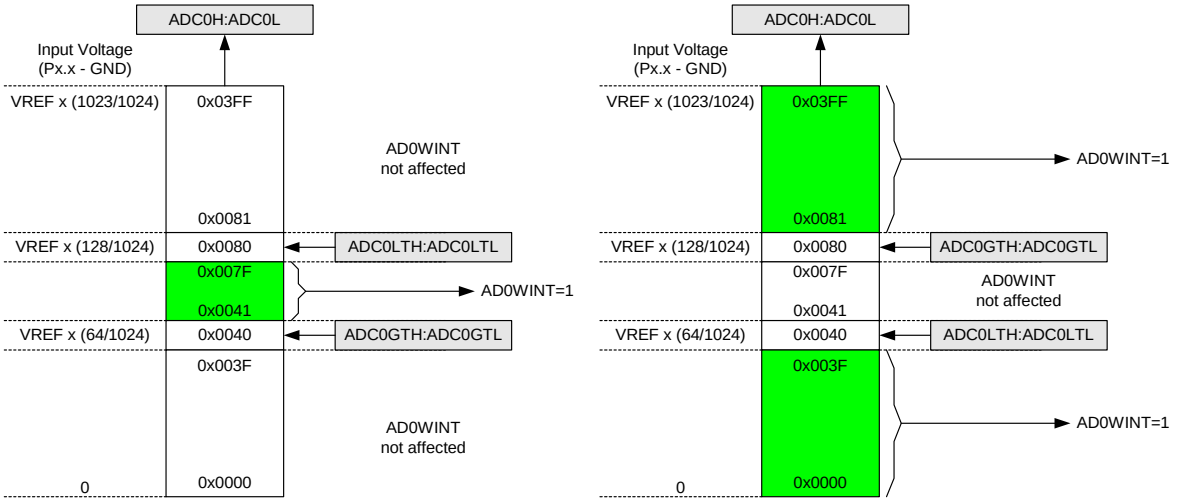


Figure 5.5. ADC Window Compare Example: Right-Justified Single-Ended Data

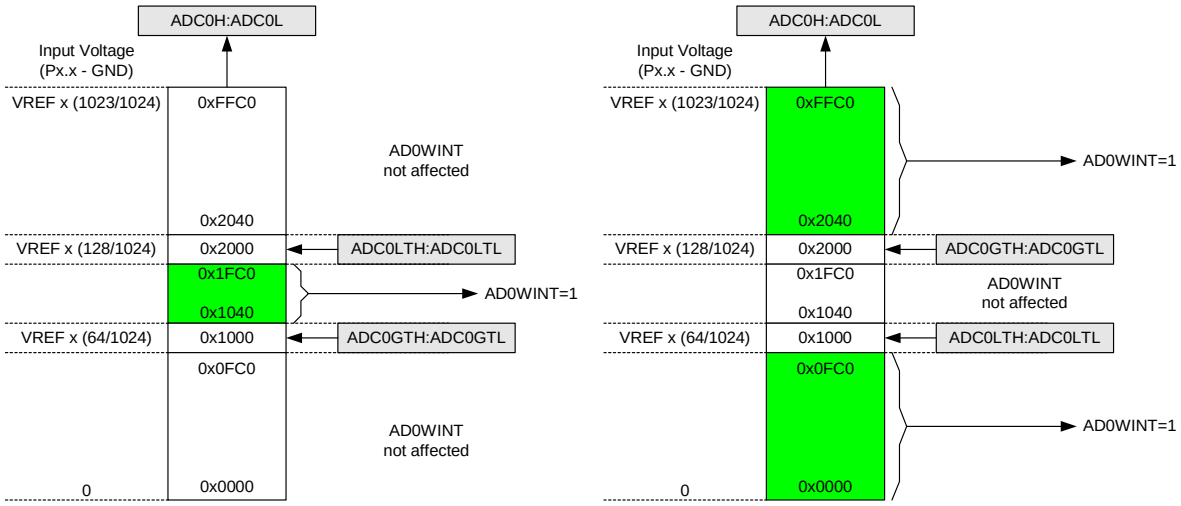


Figure 5.6. ADC Window Compare Example: Left-Justified Single-Ended Data

5.4.2. ADC0 Specifications

See “4. Electrical Characteristics” on page 45 for a detailed listing of ADC0 specifications.

SFR Definition 5.12. ADC0MX: ADC0 Input Channel Select

Bit	7	6	5	4	3	2	1	0
Name				AD0MX				
Type	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

SFR Page = 0x0; SFR Address = 0xBB

Bit	Name	Function
7:5	Unused	**Unused.** Read = 000b; Write = Don't Care.
4:0	AD0MX	**AMUX0 Positive Input Selection.** Selects the positive input channel for ADC0.

00000:	P0.0	10000:	P2.0 (C8051F920/30 Only)
00001:	P0.1	10001:	P2.1 (C8051F920/30 Only)
00010:	P0.2	10010:	P2.2 (C8051F920/30 Only)
00011:	P0.3	10011:	P2.3 (C8051F920/30 Only)
00100:	P0.4	10100:	P2.4 (C8051F920/30 Only)
00101:	P0.5	10101:	P2.5 (C8051F920/30 Only)
00110:	P0.6	10110:	P2.6 (C8051F920/30 Only)
00111:	P0.7	10111:	Reserved.
01000:	P1.0	11000:	Reserved.
01001:	P1.1	11001:	Reserved.
01010:	P1.2	11010:	Reserved.
01011:	P1.3	11011:	Temperature Sensor*
01100:	P1.4	11100:	VBAT Supply Voltage (0.9–1.8 V) or (1.8–3.6 V)
01101:	P1.5		
01110:	P1.6	11101:	Digital Supply Voltage (VREG0 Output, 1.7 V Typical)
01111:	P1.7 (C8051F920/30 Only)	11110:	VDD/DC+ Supply Voltage (1.8–3.6 V)
		11111:	Ground

***Note:** Before switching the ADC multiplexer from another channel to the temperature sensor, the ADC mux should select the 'Ground' channel as an intermediate step. The intermediate 'Ground' channel selection step will discharge any voltage on the ADC sampling capacitor from the previous channel selection. This will prevent the possibility of a high voltage (> 2V) being presented to the temperature sensor circuit, which can otherwise impact its long-term reliability.

7.2. Comparator Outputs

When a comparator is enabled, its output is a logic 1 if the voltage at the positive input is higher than the voltage at the negative input. When disabled, the comparator output is a logic 0. The comparator output is synchronized with the system clock as shown in Figure 7.2. The synchronous “latched” output (CP0, CP1) can be polled in software (CPnOUT bit), used as an interrupt source, or routed to a Port pin through the Crossbar.

The asynchronous “raw” comparator output (CP0A, CP1A) is used by the low power mode wakeup logic and reset decision logic. See the Power Options chapter and the Reset Sources chapter for more details on how the asynchronous comparator outputs are used to make wake-up and reset decisions. The asynchronous comparator output can also be routed directly to a Port pin through the Crossbar, and is available for use outside the device even if the system clock is stopped.

When using a Comparator as an interrupt source, Comparator interrupts can be generated on rising-edge and/or falling-edge comparator output transitions. Two independent interrupt flags (CPnRIF and CPnFIF) allow software to determine which edge caused the Comparator interrupt. The comparator rising-edge and falling-edge interrupt flags are set by hardware when a corresponding edge is detected regardless of the interrupt enable state. Once set, these bits remain set until cleared by software.

The rising-edge and falling-edge interrupts can be individually enabled using the CPnRIE and CPnFIE interrupt enable bits in the CPTnMD register. In order for the CPnRIF and/or CPnFIF interrupt flags to generate an interrupt request to the CPU, the Comparator must be enabled as an interrupt source and global interrupts must be enabled. See the Interrupt Handler chapter for additional information.

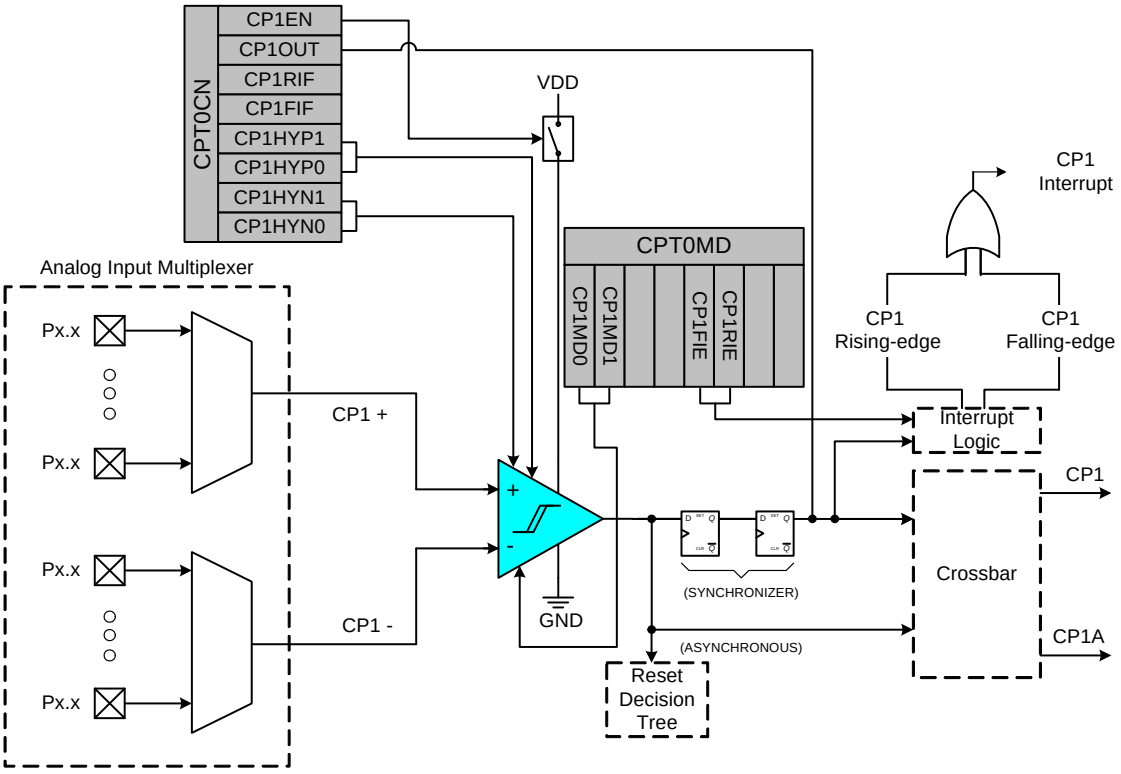


Figure 7.2. Comparator 1 Functional Block Diagram

8. 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 27), 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 8.1 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.

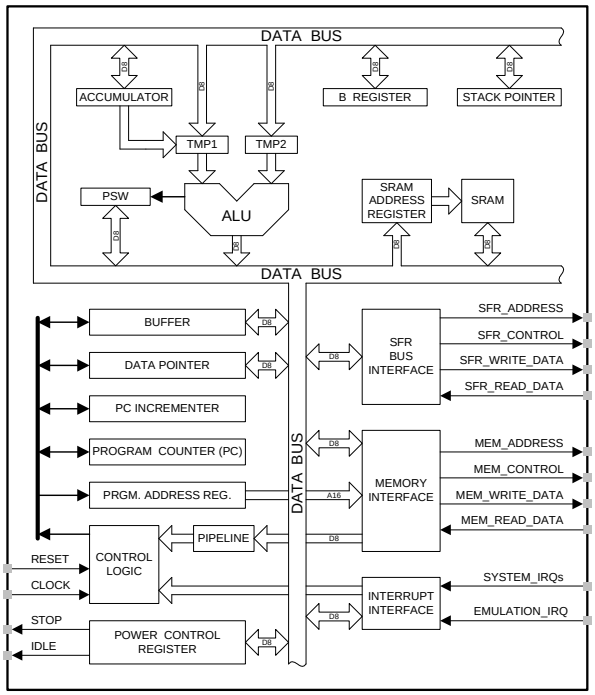


Figure 8.1. CIP-51 Block Diagram

With the CIP-51's maximum system clock at 25 MHz, it has a peak throughput of 25 MIPS. The CIP-51 has a total of 109 instructions. The table below shows the total number of instructions that require each execution time.

Clocks to Execute	1	2	2/3	3	3/4	4	4/5	5	8
Number of Instructions	26	50	5	14	7	3	1	2	1

Programming and Debugging Support

In-system programming of the Flash program memory and communication with on-chip debug support logic is accomplished via the Silicon Labs 2-Wire Development Interface (C2).

The on-chip debug support logic facilitates full speed in-circuit debugging, allowing the setting of hardware breakpoints, starting, stopping and single stepping through program execution (including interrupt service routines), examination of the program's call stack, and reading/writing the contents of registers and memory. This method of on-chip debugging is completely non-intrusive, requiring no RAM, Stack, timers, or other on-chip resources. C2 details can be found in Section “27. C2 Interface” on page 324.

The CIP-51 is supported by development tools from Silicon Labs and third party vendors. Silicon Labs provides an integrated development environment (IDE) including editor, debugger and programmer. The IDE's debugger and programmer interface to the CIP-51 via the C2 interface to provide fast and efficient in-system device programming and debugging. Third party macro assemblers and C compilers are also available.

8.1. Instruction Set

The instruction set of the CIP-51 System Controller is fully compatible with the standard MCS-51™ instruction set. Standard 8051 development tools can be used to develop software for the CIP-51. All CIP-51 instructions are the binary and functional equivalent of their MCS-51™ counterparts, including opcodes, addressing modes and effect on PSW flags. However, instruction timing is different than that of the standard 8051.

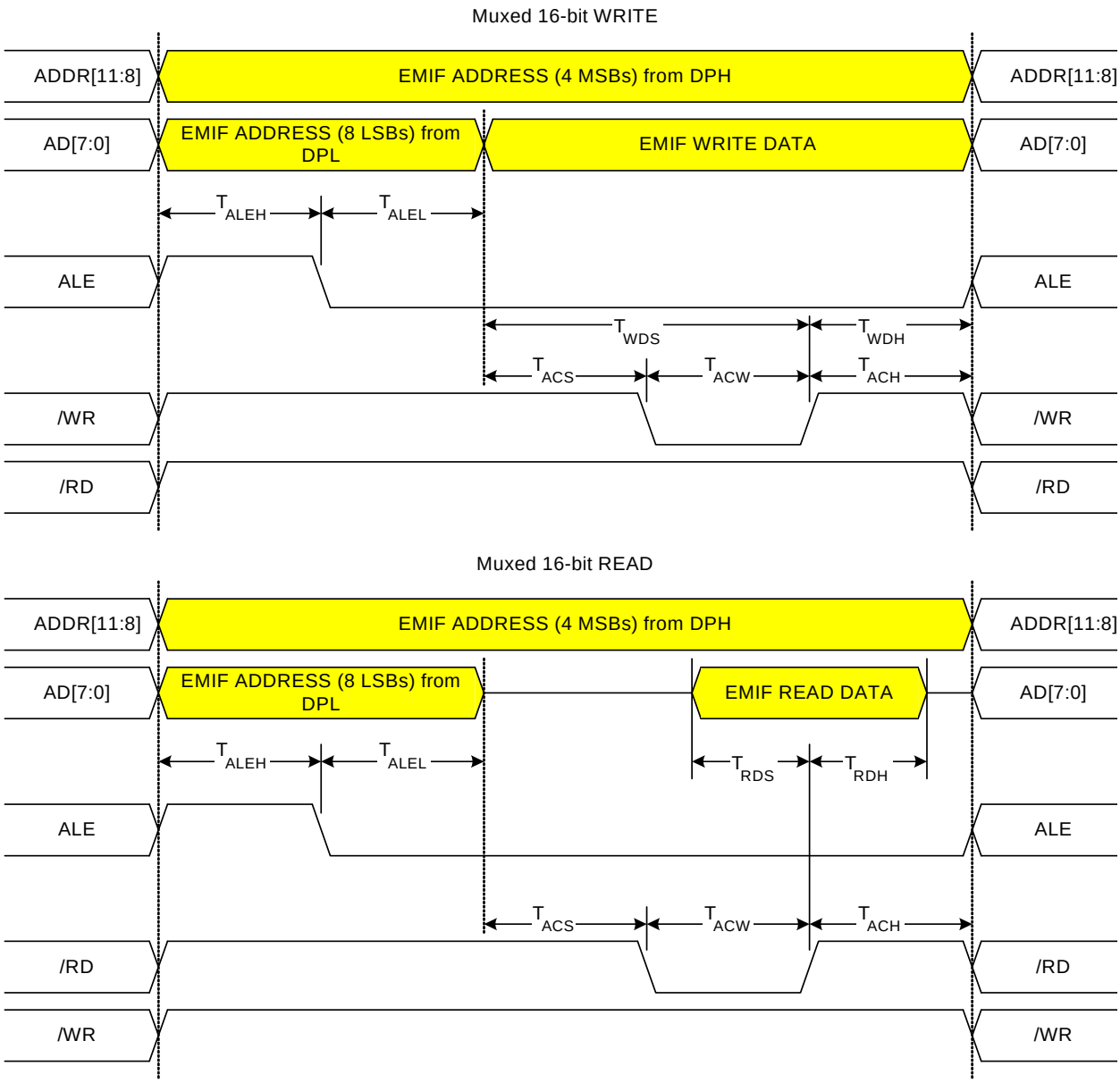
8.1.1. Instruction and CPU Timing

In many 8051 implementations, a distinction is made between machine cycles and clock cycles, with machine cycles varying from 2 to 12 clock cycles in length. However, the CIP-51 implementation is based solely on clock cycle timing. All instruction timings are specified in terms of clock cycles.

Due to the pipelined architecture of the CIP-51, most instructions execute in the same number of clock cycles as there are program bytes in the instruction. Conditional branch instructions take one less clock cycle to complete when the branch is not taken as opposed to when the branch is taken. Table 8.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

10.8. EMIF Timing Diagrams

10.8.1. Multiplexed 16-bit MOVX: EMI0CF[3:2] = 01, 10, or 11



Note: See the Port Input/Output chapter to determine which port pins are mapped to the ADDR[11:8], AD[7:0], ALE, /RD, and /WR signals.

Figure 10.4. Multiplexed 16-bit MOVX Timing

13. Flash Memory

On-chip, re-programmable Flash memory is included for program code and non-volatile data storage. The Flash memory can be programmed in-system through the C2 interface or by software using the MOVX write instruction. Once cleared to logic 0, a Flash bit must be erased to set it back to logic 1. Flash bytes would typically be erased (set to 0xFF) before being reprogrammed. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operations is not required. Code execution is stalled during Flash write/erase operations. Refer to Table 4.6 for complete Flash memory electrical characteristics.

13.1. Programming The Flash Memory

The simplest means of programming the Flash memory is through the C2 interface using programming tools provided by Silicon Laboratories or a third party vendor. This is the only means for programming a non-initialized device. For details on the C2 commands to program Flash memory, see Section “27. C2 Interface” on page 324.

The Flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before programming Flash memory using MOVX, Flash programming operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1 (this directs the MOVX writes to target Flash memory); and (2) Writing the Flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software. For detailed guidelines on programming Flash from firmware, please see Section “13.5. Flash Write and Erase Guidelines” on page 153.

To ensure the integrity of the Flash contents, the on-chip VDD Monitor must be enabled and enabled as a reset source in any system that includes code that writes and/or erases Flash memory from software. Furthermore, there should be no delay between enabling the V_{DD} Monitor and enabling the V_{DD} Monitor as a reset source. Any attempt to write or erase Flash memory while the V_{DD} Monitor is disabled, or not enabled as a reset source, will cause a Flash Error device reset.

13.1.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The Flash Lock and Key Register (FLKEY) must be written with the correct key codes, in sequence, before Flash operations may be performed. The key codes are: 0xA5, 0xF1. The timing does not matter, but the codes must be written in order. If the key codes are written out of order, or the wrong codes are written, Flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a Flash write or erase is attempted before the key codes have been written properly. The Flash lock resets after each write or erase; the key codes must be written again before a following Flash operation can be performed. The FLKEY register is detailed in SFR Definition 13.2.

The level of Flash security depends on the Flash access method. The three Flash access methods that can be restricted are reads, writes, and erases from the C2 debug interface, user firmware executing on unlocked pages, and user firmware executing on locked pages. Table 13.1 summarizes the Flash security features of the C8051F93x-C8051F92x devices.

Table 13.1. Flash Security Summary

Action	C2 Debug Interface	User Firmware executing from:	
		an unlocked page	a locked page
Read, Write or Erase unlocked pages (except page with Lock Byte)	Permitted	Permitted	Permitted
Read, Write or Erase locked pages (except page with Lock Byte)	Not Permitted	FEDR	Permitted
Read or Write page containing Lock Byte (if no pages are locked)	Permitted	Permitted	Permitted
Read or Write page containing Lock Byte (if any page is locked)	Not Permitted	FEDR	Permitted
Read contents of Lock Byte (if no pages are locked)	Permitted	Permitted	Permitted
Read contents of Lock Byte (if any page is locked)	Not Permitted	FEDR	Permitted
Erase page containing Lock Byte (if no pages are locked)	Permitted	FEDR	FEDR
Erase page containing Lock Byte - Unlock all pages (if any page is locked)	Only by C2DE	FEDR	FEDR
Lock additional pages (change 1s to 0s in the Lock Byte)	Not Permitted	FEDR	FEDR
Unlock individual pages (change 0s to 1s in the Lock Byte)	Not Permitted	FEDR	FEDR
Read, Write or Erase Reserved Area	Not Permitted	FEDR	FEDR

C2DE—C2 Device Erase (Erases all Flash pages including the page containing the Lock Byte)
FEDR—Not permitted; Causes Flash Error Device Reset (FERROR bit in RSTSRC is '1' after reset)

- All prohibited operations that are performed via the C2 interface are ignored (do not cause device reset).
- Locking any Flash page also locks the page containing the Lock Byte.
- Once written to, the Lock Byte cannot be modified except by performing a C2 Device Erase.
- If user code writes to the Lock Byte, the Lock does not take effect until the next device reset.
- The scratchpad is locked when all other Flash pages are locked.
- The scratchpad is erased when a Flash Device Erase command is performed.

20.1.2. Using RTC0ADR and RTC0DAT to Access SmaRTClock Internal Registers

The SmaRTClock internal registers can be read and written using RTC0ADR and RTC0DAT. The RTC0ADR register selects the SmaRTClock internal register that will be targeted by subsequent reads or writes. Recommended instruction timing is provided in this section. If the recommended instruction timing is not followed, then BUSY (RTC0ADR.7) should be checked prior to each read or write operation to make sure the SmaRTClock Interface is not busy performing the previous read or write operation. A SmaRTClock Write operation is initiated by writing to the RTC0DAT register. Below is an example of writing to a SmaRTClock internal register.

1. Poll BUSY (RTC0ADR.7) until it returns 0 or follow recommended instruction timing.
2. Write 0x05 to RTC0ADR. This selects the internal RTC0CN register at SmaRTClock Address 0x05.
3. Write 0x00 to RTC0DAT. This operation writes 0x00 to the internal RTC0CN register.

A SmaRTClock Read operation is initiated by setting the SmaRTClock Interface Busy bit. This transfers the contents of the internal register selected by RTC0ADR to RTC0DAT. The transferred data will remain in RTC0DAT until the next read or write operation. Below is an example of reading a SmaRTClock internal register.

1. Poll BUSY (RTC0ADR.7) until it returns 0 or follow recommended instruction timing.
2. Write 0x05 to RTC0ADR. This selects the internal RTC0CN register at SmaRTClock Address 0x05.
3. Write 1 to BUSY. This initiates the transfer of data from RTC0CN to RTC0DAT.
4. Poll BUSY (RTC0ADR.7) until it returns 0 or follow recommend instruction timing.
5. Read data from RTC0DAT. This data is a copy of the RTC0CN register.

Note: The RTC0ADR and RTC0DAT registers will retain their state upon a device reset.

20.1.3. RTC0ADR Short Strobe Feature

Reads and writes to indirect SmaRTClock registers normally take 7 system clock cycles. To minimize the indirect register access time, the Short Strobe feature decreases the read and write access time to 6 system clocks. The Short Strobe feature is automatically enabled on reset and can be manually enabled/disabled using the SHORT (RTC0ADR.4) control bit.

Recommended Instruction Timing for a single register read with short strobe enabled:

```
mov RTC0ADR, #095h
nop
nop
nop
mov A, RTC0DAT
```

Recommended Instruction Timing for a single register write with short strobe enabled:

```
mov RTC0ADR, #095h
mov RTC0DAT, #000h
nop
```

Table 22.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X	0001
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X	0100
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X	0001
	0101	0	X	X	An illegal STOP or bus error was detected while a Slave Transmission was in progress.	Clear STO.	0	0	X	—
Slave Receiver	0010	0	0	X	A slave address + R/W was received; ACK sent.	If Write, Set ACK for first data byte.	0	0	1	0000
						If Read, Load SMB0DAT with data byte	0	0	X	0100
		0	1	X	Lost arbitration as master; slave address + R/W received; ACK sent.	If Write, Set ACK for first data byte.	0	0	1	0000
						If Read, Load SMB0DAT with data byte	0	0	X	0100
						Reschedule failed transfer	1	0	X	1110
	0001	0	0	X	A STOP was detected while addressed as a Slave Transmitter or Slave Receiver.	Clear STO.	0	0	X	—
		0	1	X	Lost arbitration while attempting a STOP.	No action required (transfer complete/aborted).	0	0	0	—
	0000	0	0	X	A slave byte was received.	Set ACK for next data byte; Read SMB0DAT.	0	0	1	0000
						Set NACK for next data byte; Read SMB0DAT.	0	0	0	0000
Bus Error Condition	0010	0	1	X	Lost arbitration while attempting a repeated START.	Abort failed transfer.	0	0	X	—
						Reschedule failed transfer.	1	0	X	1110
	0001	0	1	X	Lost arbitration due to a detected STOP.	Abort failed transfer.	0	0	X	—
						Reschedule failed transfer.	1	0	X	1110
	0000	0	1	X	Lost arbitration while transmitting a data byte as master.	Abort failed transfer.	0	0	X	—
						Reschedule failed transfer.	1	0	X	1110

SFR Definition 23.2. SBUF0: Serial (UART0) Port Data Buffer

Bit	7	6	5	4	3	2	1	0
Name	SBUF0[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 Page = 0x0; SFR Address = 0x99

Bit	Name	Function
7:0	SBUF0	**Serial Data Buffer Bits 7:0 (MSB–LSB).** This SFR accesses two registers; a transmit shift register and a receive latch register. When data is written to SBUF0, it goes to the transmit shift register and is held for serial transmission. Writing a byte to SBUF0 initiates the transmission. A read of SBUF0 returns the contents of the receive latch.

25.1. Timer 0 and Timer 1

Each timer is implemented as a 16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register (Section “12.5. Interrupt Register Descriptions” on page 139); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (Section “12.5. Interrupt Register Descriptions” on page 139). Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently. Each operating mode is described below.

25.1.1. Mode 0: 13-bit Counter/Timer

Timer 0 and Timer 1 operate as 13-bit counter/timers in Mode 0. The following describes the configuration and operation of Timer 0. However, both timers operate identically, and Timer 1 is configured in the same manner as described for Timer 0.

The TH0 register holds the eight MSBs of the 13-bit counter/timer. TL0 holds the five LSBs in bit positions TL0.4–TL0.0. The three upper bits of TL0 (TL0.7–TL0.5) are indeterminate and should be masked out or ignored when reading. As the 13-bit timer register increments and overflows from 0x1FFF (all ones) to 0x0000, the timer overflow flag TF0 (TCON.5) is set and an interrupt will occur if Timer 0 interrupts are enabled.

The C/T0 bit (TMOD.2) selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to Section “21.3. Priority Crossbar Decoder” on page 221 for information on selecting and configuring external I/O pins). Clearing C/T selects the clock defined by the T0M bit (CKCON.3). When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON (see SFR Definition 25.1).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or the input signal $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 12.7). Setting GATE0 to 1 allows the timer to be controlled by the external input signal $\overline{\text{INT0}}$ (see Section “12.5. Interrupt Register Descriptions” on page 139), facilitating pulse width measurements

Table 25.1. Timer 0 Running Modes

TR0	GATE0	$\overline{\text{INT0}}$	Counter/Timer
0	X	X	Disabled
1	0	X	Enabled
1	1	0	Disabled
1	1	1	Enabled
Note: X = Don't Care			

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal $\overline{\text{INT1}}$ is used with Timer 1; the $\overline{\text{INT1}}$ polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 12.7).

SFR Definition 25.8. TMR2CN: Timer 2 Control

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

SFR Page = 0x0; SFR Address = 0xC8; Bit-Addressable

Bit	Name	Function
7	TF2H	**Timer 2 High Byte Overflow Flag.** Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF2L	**Timer 2 Low Byte Overflow Flag.** Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. TF2L will be set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware.
5	TF2LEN	**Timer 2 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 2 Low Byte interrupts. If Timer 2 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 2 overflows.
4	TF2CEN	**Timer 2 Capture Enable.** When set to 1, this bit enables Timer 2 Capture Mode.
3	T2SPLIT	**Timer 2 Split Mode Enable.** When set to 1, Timer 2 operates as two 8-bit timers with auto-reload. Otherwise, Timer 2 operates in 16-bit auto-reload mode.
2	TR2	**Timer 2 Run Control.** Timer 2 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in split mode.
1:0	T2XCLK[1:0]	**Timer 2 External Clock Select.** This bit selects the “external” and “capture trigger” clock sources for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the “external” clock source for both timer bytes. Timer 2 Clock Select bits (T2MH and T2ML in register CKCON) may still be used to select between the “external” clock and the system clock for either timer. Note: External clock sources are synchronized with the system clock. 00: External Clock is SYSCLK/12. Capture trigger is SmaRTClock/8. 01: External Clock is Comparator 0. Capture trigger is SmaRTClock/8. 10: External Clock is SYSCLK/12. Capture trigger is Comparator 0. 11: External Clock is SmaRTClock/8. Capture trigger is Comparator 0.

26. Programmable Counter Array

The Programmable Counter Array (PCA0) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. The PCA consists of a dedicated 16-bit counter/timer and six 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section “26.3. Capture/Compare Modules” on page 308). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 26.1

Important Note: The PCA Module 5 may be used as a watchdog timer (WDT), and is enabled in this mode following a system reset. **Access to certain PCA registers is restricted while WDT mode is enabled**. See Section 26.4 for details.

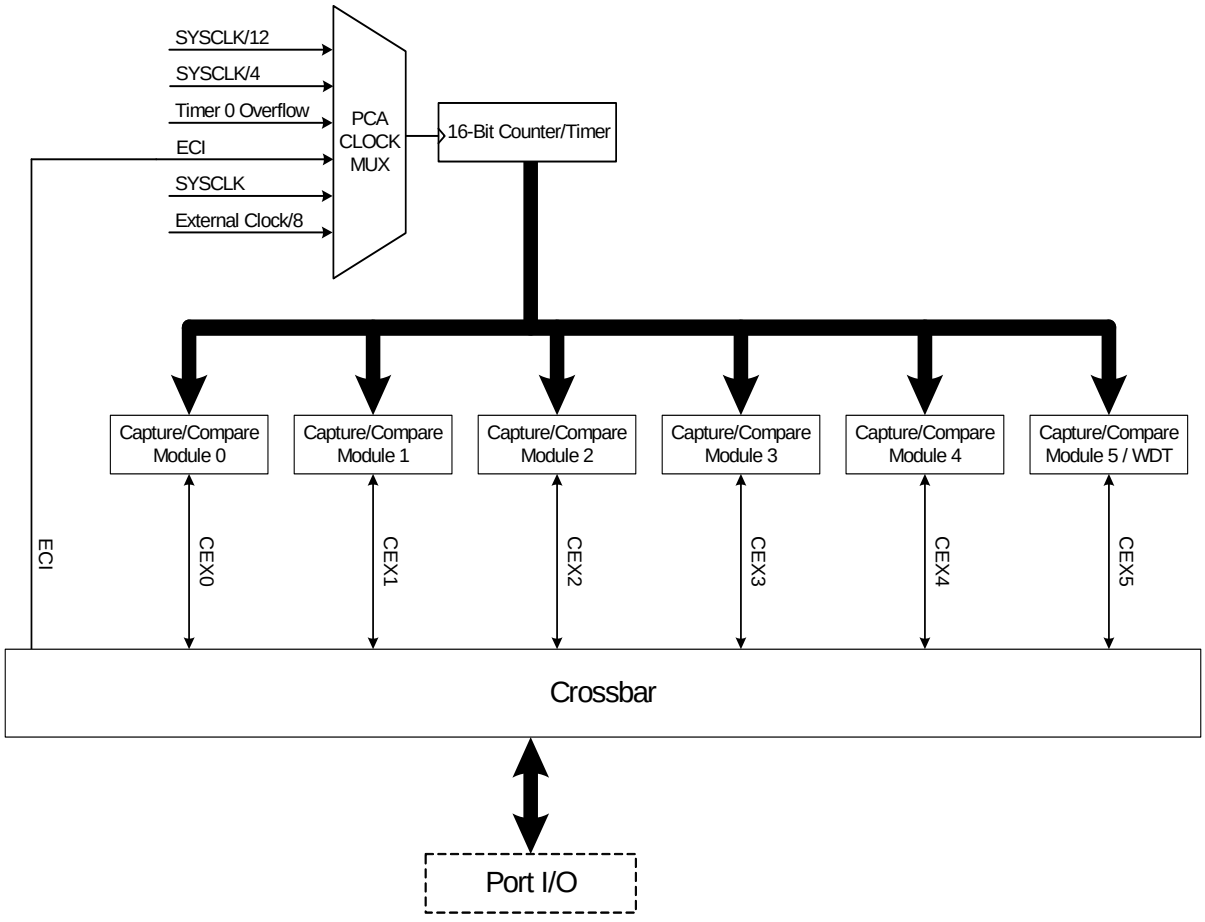


Figure 26.1. PCA Block Diagram

27. C2 Interface

C8051F93x-C8051F92x devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow Flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

27.1. C2 Interface Registers

The following describes the C2 registers necessary to perform Flash programming through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

C2 Register Definition 27.1. C2ADD: C2 Address

Bit	7	6	5	4	3	2	1	0
Name	C2ADD[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function	
7:0	C2ADD[7:0]	**C2 Address.** The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands.	
		Address	**Description**
		0x00	Selects the Device ID register for Data Read instructions
		0x01	Selects the Revision ID register for Data Read instructions
		0x02	Selects the C2 Flash Programming Control register for Data Read/Write instructions
		0xB4	Selects the C2 Flash Programming Data register for Data Read/Write instructions

Revision 1.2 to Revision 1.3

- Added labels to indicate center pad as “GND (optional)” to pinout diagrams in Figure 3.1 and Figure 3.2.
- Added package marking diagrams as Figure 3.4, Figure 3.5, and Figure 3.6 to help identify the silicon revision.
- Clarified conditions that apply to ‘VBAT Ramp Time for Power On’ for one-cell mode vs two-cell mode in Table 4.4, “Reset Electrical Characteristics,” on page 59.
- Updated Section “5.2.3. Burst Mode” on page 71 and Figure 5.3 to show difference in behavior between internal convert start signals and external CNVSTR signal.
- Added note about the need to ground the ADC mux before switching to the temperature sensor in Section “5.6. Temperature Sensor” on page 86 and in SFR Definition 5.12 “ADC0MX”.
- Updated Figure 7.4, “CPn Multiplexer Block Diagram,” to show $\overline{\text{CPnOUT}}$ pull-up voltage (inverted)and to correct the locations of VDD/DC+, VBAT, Digital Supply, and GND multiplexer inputs.
- Updated Table 8.1 to correct number of clock cycles for ‘CJNE A, direct, rel’.
- Corrected VDD ramp time reference in item 2 of Section “13.5.1. VDD Maintenance and the VDD Monitor” on page 153.
- Updated CPT0WK bit description in SFR Definition 14.1, “PMU0CF”.
- Added Section “15.2. 32-bit CRC Algorithm” on page 169 to illustrate the 32-bit CRC algorithm.
- Updated Section “21.1.3. Interfacing Port I/O to 5 V and 3.3 V Logic” on page 218 to include notes about sizing external pull-up resistors and other related information when using multi-voltage interfaces.
- Corrected clock sources associated with T3XCLK settings in Section “25.3.2. 8-bit Timers with Auto-Reload” on page 300, Figure 25.7, Figure 25.8, and Figure 25.9 to match the description in SFR Definition 25.13.
- Removed ‘SmaRTClock divided by 8’ from list of possible clock sources in text description in Section “26. Programmable Counter Array” on page 305.
- Replaced incorrect PCA channel references from PCA0CPH2 to PCA0CPH5 in Section “26.4. Watchdog Timer Mode” on page 316 and Figure 26.11.

Revision 1.3 to Revision 1.4

- Updated part numbers to Revision G in “Product Selection Guide” on page 26.
- Updated Figure 7.4, “CPn Multiplexer Block Diagram,” to remove the bar over the CPnOUT signals.
- Updated the “Reset Sources” on page 184 chapter to reflect the correct state of the $\overline{\text{RST}}$ pin during a power-on reset.