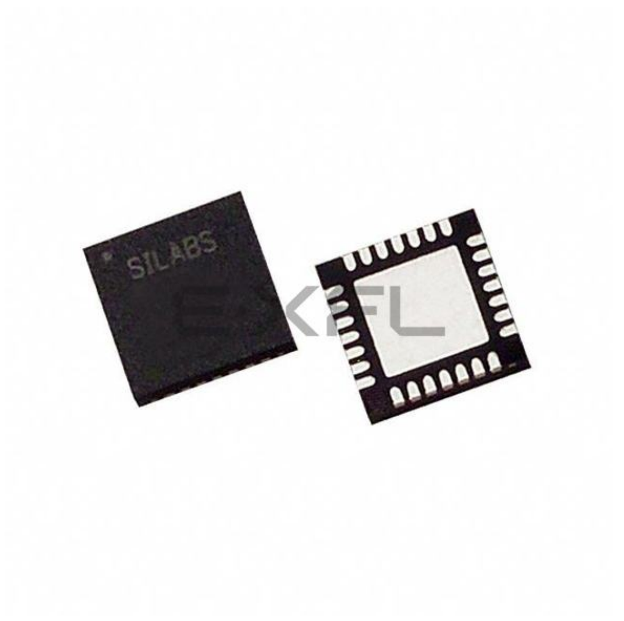

What is "Embedded - Microcontrollers"?

"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	Obsolete
Core Processor	8051
Core Size	8-Bit
Speed	25MHz
Connectivity	SMBus (2-Wire/I²C), SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	25
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.25K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 17x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VFQFN Exposed Pad
Supplier Device Package	28-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f311

1.5. Serial Ports

The C8051F31x Family includes an SMBus/I2C interface, a full-duplex UART with enhanced baud rate configuration, and an Enhanced SPI interface. Each of the serial buses is fully implemented in hardware and makes extensive use of the CIP-51's interrupts, thus requiring very little CPU intervention.

1.6. Programmable Counter Array

An on-chip Programmable Counter/Timer Array (PCA) is included in addition to the four 16-bit general purpose counter/timers. The PCA consists of a dedicated 16-bit counter/timer time base with five programmable capture/compare modules. The PCA clock is derived from one of six sources: the system clock divided by 12, the system clock divided by 4, Timer 0 overflows, an External Clock Input (ECI), the system clock, or the external oscillator clock source divided by 8. The external clock source selection is useful for real-time clock functionality, where the PCA is clocked by an external source while the internal oscillator drives the system clock.

Each capture/compare module can be configured to operate in one of six modes: Edge-Triggered Capture, Software Timer, High Speed Output, 8- or 16-bit Pulse Width Modulator, or Frequency Output. Additionally, Capture/Compare Module 4 offers watchdog timer (WDT) capabilities. Following a system reset, Module 4 is configured and enabled in WDT mode. The PCA Capture/Compare Module I/O and External Clock Input may be routed to Port I/O via the Digital Crossbar.

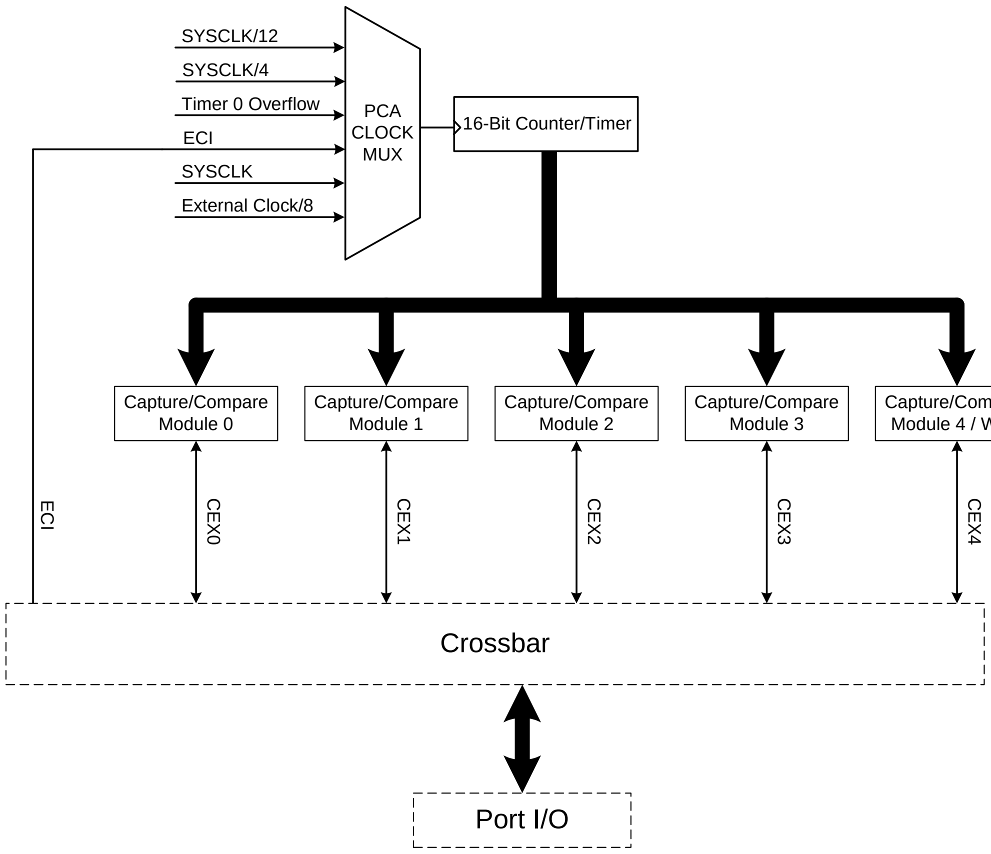

Figure 1.14. PCA Block Diagram

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.

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 re-programmable Flash can also be read and changed a single byte at a time by the application software using the MOVC and MOVX instructions. This feature allows program memory to be used for non-volatile data storage as well as updating program code under software control.

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 “20. C2 Interface” on page 223**.

The CIP-51 is supported by development tools from Silicon Labs and third party vendors. Silicon Labs provides an integrated development environment (IDE) including an editor, evaluation compiler, assembler, 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

Table 8.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
ORL A, direct	OR direct byte to A	2	2
ORL A, @Ri	OR indirect RAM to A	1	2
ORL A, #data	OR immediate to A	2	2
ORL direct, A	OR A to direct byte	2	2
ORL direct, #data	OR immediate to direct byte	3	3
XRL A, Rn	Exclusive-OR Register to A	1	1
XRL A, direct	Exclusive-OR direct byte to A	2	2
XRL A, @Ri	Exclusive-OR indirect RAM to A	1	2
XRL A, #data	Exclusive-OR immediate to A	2	2
XRL direct, A	Exclusive-OR A to direct byte	2	2
XRL direct, #data	Exclusive-OR immediate to direct byte	3	3
CLR A	Clear A	1	1
CPL A	Complement A	1	1
RL A	Rotate A left	1	1
RLC A	Rotate A left through Carry	1	1
RR A	Rotate A right	1	1
RRC A	Rotate A right through Carry	1	1
SWAP A	Swap nibbles of A	1	1
Data Transfer			
MOV A, Rn	Move Register to A	1	1
MOV A, direct	Move direct byte to A	2	2
MOV A, @Ri	Move indirect RAM to A	1	2
MOV A, #data	Move immediate to A	2	2
MOV Rn, A	Move A to Register	1	1
MOV Rn, direct	Move direct byte to Register	2	2
MOV Rn, #data	Move immediate to Register	2	2
MOV direct, A	Move A to direct byte	2	2
MOV direct, Rn	Move Register to direct byte	2	2
MOV direct, direct	Move direct byte to direct byte	3	3
MOV direct, @Ri	Move indirect RAM to direct byte	2	2
MOV direct, #data	Move immediate to direct byte	3	3
MOV @Ri, A	Move A to indirect RAM	1	2
MOV @Ri, direct	Move direct byte to indirect RAM	2	2
MOV @Ri, #data	Move immediate to indirect RAM	2	2
MOV DPTR, #data16	Load DPTR with 16-bit constant	3	3
MOVC A, @A+DPTR	Move code byte relative DPTR to A	1	3
MOVC A, @A+PC	Move code byte relative PC to A	1	3
MOVX A, @Ri	Move external data (8-bit address) to A	1	3
MOVX @Ri, A	Move A to external data (8-bit address)	1	3
MOVX A, @DPTR	Move external data (16-bit address) to A	1	3
MOVX @DPTR, A	Move A to external data (16-bit address)	1	3
PUSH direct	Push direct byte onto stack	2	2
POP direct	Pop direct byte from stack	2	2
XCH A, Rn	Exchange Register with A	1	1
XCH A, direct	Exchange direct byte with A	2	2

Table 8.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
XCH A, @Ri	Exchange indirect RAM with A	1	2
XCHD A, @Ri	Exchange low nibble of indirect RAM with A	1	2
Boolean Manipulation			
CLR C	Clear Carry	1	1
CLR bit	Clear direct bit	2	2
SETB C	Set Carry	1	1
SETB bit	Set direct bit	2	2
CPL C	Complement Carry	1	1
CPL bit	Complement direct bit	2	2
ANL C, bit	AND direct bit to Carry	2	2
ANL C, /bit	AND complement of direct bit to Carry	2	2
ORL C, bit	OR direct bit to carry	2	2
ORL C, /bit	OR complement of direct bit to Carry	2	2
MOV C, bit	Move direct bit to Carry	2	2
MOV bit, C	Move Carry to direct bit	2	2
JC rel	Jump if Carry is set	2	2/3
JNC rel	Jump if Carry is not set	2	2/3
JB bit, rel	Jump if direct bit is set	3	3/4
JNB bit, rel	Jump if direct bit is not set	3	3/4
JBC bit, rel	Jump if direct bit is set and clear bit	3	3/4
Program Branching			
ACALL addr11	Absolute subroutine call	2	3
LCALL addr16	Long subroutine call	3	4
RET	Return from subroutine	1	5
RETI	Return from interrupt	1	5
AJMP addr11	Absolute jump	2	3
LJMP addr16	Long jump	3	4
SJMP rel	Short jump (relative address)	2	3
JMP @A+DPTR	Jump indirect relative to DPTR	1	3
JZ rel	Jump if A equals zero	2	2/3
JNZ rel	Jump if A does not equal zero	2	2/3
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	3/4
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3/4
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3/4
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4/5
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2/3
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3/4
NOP	No operation	1	1

8.2.6. Special Function Registers

The direct-access data memory locations from 0x80 to 0xFF constitute the special function registers (SFRs). The SFRs provide control and data exchange with the CIP-51's resources and peripherals. The CIP-51 duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the MCU. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 8.2 lists the SFRs implemented in the CIP-51 System Controller.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g. P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing these areas will have an indeterminate effect and should be avoided. Refer to the corresponding pages of the datasheet, as indicated in Table 8.3, for a detailed description of each register.

Table 8.2. Special Function Register (SFR) Memory Map

F8	SPI0CN	PCA0L	PCA0H	PCA0CPL0	PCA0CPH0	PCA0CPL4	PCA0CPH4	VDM0CN
F0	B	P0MDIN	P1MDIN	P2MDIN	P3MDIN		EIP1	
E8	ADC0CN	PCA0CPL1	PCA0CPH1	PCA0CPL2	PCA0CPH2	PCA0CPL3	PCA0CPH3	RSTSRC
E0	ACC	XBR0	XBR1		IT01CF		EIE1	
D8	PCA0CN	PCA0MD	PCA0CPM0	PCA0CPM1	PCA0CPM2	PCA0CPM3	PCA0CPM4	
D0	PSW	REF0CN			P0SKIP	P1SKIP	P2SKIP	
C8	TMR2CN		TMR2RLL	TMR2RLH	TMR2L	TMR2H		
C0	SMB0CN	SMB0CF	SMB0DAT	ADC0GTL	ADC0GTH	ADC0LTL	ADC0LTH	
B8	IP		AMX0N	AMX0P	ADC0CF	ADC0L	ADC0H	
B0	P3	OSCXCN	OSCICN	OSCICL			FLSCL	FLKEY
A8	IE	CLKSEL	EMI0CN					
A0	P2	SPI0CFG	SPI0CKR	SPI0DAT	P0MDOUT	P1MDOUT	P2MDOUT	P3MDOUT
98	SCON0	SBUF0	CPT1CN	CPT0CN	CPT1MD	CPT0MD	CPT1MX	CPT0MX
90	P1	TMR3CN	TMR3RLL	TMR3RLH	TMR3L	TMR3H		
88	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON	PSCTL
80	P0	SP	DPL	DPH				PCON
	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)

(bit addressable)

SFR Definition 8.6. B: B Register

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xF0

Bits7–0: B: B Register.
This register serves as a second accumulator for certain arithmetic operations.

8.3. Interrupt Handler

The CIP-51 includes an extended interrupt system supporting a total of 14 interrupt sources with two priority levels. The allocation of interrupt sources between on-chip peripherals and external inputs pins varies according to the specific version of the device. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

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

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

Note: Any instruction that clears the EA bit should be immediately followed by an instruction that has two or more opcode bytes. For example:

```
// in 'C':
EA = 0;    // clear EA bit
EA = 0;    // ... followed by another 2-byte opcode

; in assembly:
CLR  EA    ; clear EA bit
CLR  EA    ; ... followed by another 2-byte opcode
```

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

Some interrupt-pending flags are automatically cleared by the hardware when the CPU vectors to the ISR. However, most are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the return-from-interrupt (RETI)

8.4. Power Management Modes

The CIP-51 core has two software programmable power management modes: Idle and Stop. Idle mode halts the CPU while leaving the peripherals and clocks active. In Stop mode, the CPU is halted, all interrupts and timers (except the Missing Clock Detector) are inactive, and the internal oscillator is stopped (analog peripherals remain in their selected states; the external oscillator is not effected). Since clocks are running in Idle mode, power consumption is dependent upon the system clock frequency and the number of peripherals left in active mode before entering Idle. Stop mode consumes the least power. SFR Definition 8.12 describes the Power Control Register (PCON) used to control the CIP-51's power management modes.

Although the CIP-51 has Idle and Stop modes built in (as with any standard 8051 architecture), power management of the entire MCU is better accomplished by enabling/disabling individual peripherals as needed. Each analog peripheral can be disabled when not in use and placed in low power mode. Digital peripherals, such as timers or serial buses, draw little power when they are not in use. Turning off the oscillators lowers power consumption considerably; however, a reset is required to restart the MCU.

8.4.1. Idle Mode

Setting the Idle Mode Select bit (PCON.0) causes the CIP-51 to halt the CPU and enter Idle mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. All analog and digital peripherals can remain active during Idle mode.

Idle mode is terminated when an enabled interrupt is asserted or a reset occurs. The assertion of an enabled interrupt will cause the Idle Mode Selection bit (PCON.0) to be cleared and the CPU to resume operation. The pending interrupt will be serviced and the next instruction to be executed after the return from interrupt (RETI) will be the instruction immediately following the one that set the Idle Mode Select bit. If Idle mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

If enabled, the Watchdog Timer (WDT) will eventually cause an internal watchdog reset and thereby terminate the Idle mode. This feature protects the system from an unintended permanent shutdown in the event of an inadvertent write to the PCON register. If this behavior is not desired, the WDT may be disabled by software prior to entering the Idle mode if the WDT was initially configured to allow this operation. This provides the opportunity for additional power savings, allowing the system to remain in the Idle mode indefinitely, waiting for an external stimulus to wake up the system. Refer to **Section “9.6. PCA Watchdog Timer Reset” on page 108** for more information on the use and configuration of the WDT.

Note: Any instruction that sets the IDLE bit should be immediately followed by an instruction that has 2 or more opcode bytes. For example:

```
// in 'C':
PCON |= 0x01; // set IDLE bit
PCON  = PCON; // ... followed by a 3-cycle dummy instruction

; in assembly:
ORL PCON, #01h ; set IDLE bit
MOV PCON, PCON; ... followed by a 3-cycle dummy instruction
```

If the instruction following the write of the IDLE bit is a single-byte instruction and an interrupt occurs during the execution phase of the instruction that sets the IDLE bit, the CPU may not wake from IDLE mode when a future interrupt occurs.

10. Make certain that the Flash write and erase pointer variables are not located in XRAM. See your compiler documentation for instructions regarding how to explicitly locate variables in different memory areas.
11. Add address bounds checking to the routines that write or erase Flash memory to ensure that a routine called with an illegal address does not result in modification of the Flash.

10.4.3. System Clock

12. If operating from an external crystal, be advised that crystal performance is susceptible to electrical interference and is sensitive to layout and to changes in temperature. If the system is operating in an electrically noisy environment, use the internal oscillator or use an external CMOS clock.
13. If operating from the external oscillator, switch to the internal oscillator during Flash write or erase operations. The external oscillator can continue to run, and the CPU can switch back to the external oscillator after the Flash operation has completed.

Additional Flash recommendations and example code can be found in AN201, "Writing to Flash from Firmware", available from the Silicon Laboratories web site.

SFR Definition 10.1. PSCTL: Program Store R/W Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	-	-	-	-	PSEE	PSWE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x8F

Bits7–2: UNUSED: Read = 000000b, Write = don’t care.

Bit1: PSEE: Program Store Erase Enable
Setting this bit (in combination with PSWE) allows an entire page of Flash program memory to be erased. If this bit is logic 1 and Flash writes are enabled (PSWE is logic 1), a write to Flash memory using the MOVX instruction will erase the entire page that contains the location addressed by the MOVX instruction. The value of the data byte written does not matter.
0: Flash program memory erasure disabled.
1: Flash program memory erasure enabled.

Bit0: PSWE: Program Store Write Enable
Setting this bit allows writing a byte of data to the Flash program memory using the MOVX write instruction. The Flash location should be erased before writing data.
0: Writes to Flash program memory disabled.
1: Writes to Flash program memory enabled; the MOVX write instruction targets Flash memory.

NOTES:

SFR Definition 13.2. XBR1: Port I/O Crossbar Register 1

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
WEAKPUD	XBARE	T1E	T0E	ECIE	PCA0ME			00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xE2

Bit7:	WEAKPUD: Port I/O Weak Pullup Disable. 0: Weak Pullups enabled (except for Ports whose I/O are configured as analog input). 1: Weak Pullups disabled.
Bit6:	XBARE: Crossbar Enable. 0: Crossbar disabled. 1: Crossbar enabled.
Bit5:	T1E: T1 Enable 0: T1 unavailable at Port pin. 1: T1 routed to Port pin.
Bit4:	T0E: T0 Enable 0: T0 unavailable at Port pin. 1: T0 routed to Port pin.
Bit3:	ECIE: PCA0 External Counter Input Enable 0: ECI unavailable at Port pin. 1: ECI routed to Port pin.
Bits2–0:	PCA0ME: PCA Module I/O Enable Bits. 000: All PCA I/O unavailable at Port pins. 001: CEX0 routed to Port pin. 010: CEX0, CEX1 routed to Port pins. 011: CEX0, CEX1, CEX2 routed to Port pins. 100: CEX0, CEX1, CEX2, CEX3 routed to Port pins. 101: CEX0, CEX1, CEX2, CEX3, CEX4 routed to Port pins.

13.3. General Purpose Port I/O

Port pins that remain unassigned by the Crossbar and are not used by analog peripherals can be used for general purpose I/O. Ports3-0 are accessed through corresponding special function registers (SFRs) that are both byte addressable and bit addressable. When writing to a Port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the Port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the Crossbar, the Port register can always read its corresponding Port I/O pin). The exception to this is the execution of the read-modify-write instructions. The read-modify-write instructions when operating on a Port SFR are the following: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ and MOV, CLR or SET, when the destination is an individual bit in a Port SFR. For these instructions, the value of the register (not the pin) is read, modified, and written back to the SFR.

SFR Definition 13.11. P2: Port2

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xA0

Bits7–0: P2.[7:0]
Write - Output appears on I/O pins per Crossbar Registers.
0: Logic Low Output.
1: Logic High Output (high impedance if corresponding P2MDOUT.n bit = 0).
Read - Always reads ‘1’ if selected as analog input in register P2MDIN. Directly reads Port pin when configured as digital input.
0: P2.n pin is logic low.
1: P2.n pin is logic high.

Note: Only P2.0–P2.5 are associated with Port pins on the C8051F316/7 devices.

SFR Definition 13.12. P2MDIN: Port2 Input Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xF3

Bits7–0: Analog Input Configuration Bits for P2.7–P2.0 (respectively).
Port pins configured as analog inputs have their weak pullup, digital driver, and digital receiver disabled.
0: Corresponding P2.n pin is configured as an analog input.
1: Corresponding P2.n pin is not configured as an analog input.

Note: Only P2.0–P2.5 are associated with Port pins on the C8051F316/7 devices.

SFR Definition 13.15. P3: Port3

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0	11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xB0

Bits7–0: P3.[7:0]
Write - Output appears on I/O pins.
0: Logic Low Output.
1: Logic High Output (high impedance if corresponding P3MDOUT.n bit = 0).
Read - Always reads ‘1’ if selected as analog input in register P3MDIN. Directly reads Port pin when configured as digital input.
0: P3.n pin is logic low.
1: P3.n pin is logic high.

Note: Only P3.0–P3.4 are associated with Port pins on C8051F310/2/4 devices; Only P3.0 is associated with a Port pin on C8051F311/3/5/6/7 devices.

SFR Definition 13.16. P3MDIN: Port3 Input Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	-						11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xF4

Bits7–5: UNUSED. Read = 000b; Write = don’t care.
Bits4–0: Input Configuration Bits for P3.4–P3.0 (respectively).
Port pins configured as analog inputs have their weak pullup, digital driver, and digital receiver disabled.
0: Corresponding P3.n pin is configured as an analog input.
1: Corresponding P3.n pin is not configured as an analog input.

Note: Only P3.0–P3.4 are associated with Port pins on C8051F310/2/4 devices; Only P3.0 is associated with a Port pin on C8051F311/3/5/6/7 devices.

Figure 14.4 shows the typical SCL generation described by Equation 14.2. Notice that T_{HIGH} is typically twice as large as T_{LOW}. The actual SCL output may vary due to other devices on the bus (SCL may be extended low by slower slave devices, or driven low by contending master devices). The bit rate when operating as a master will never exceed the limits defined by equation Equation 14.1.

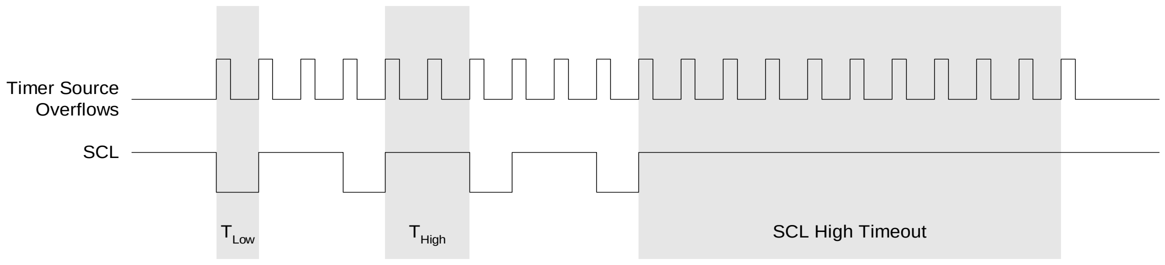

Figure 14.4. Typical SMBus SCL Generation

Setting the EXTHOLD bit extends the minimum setup and hold times for the SDA line. The minimum SDA setup time defines the absolute minimum time that SDA is stable before SCL transitions from low-to-high. The minimum SDA hold time defines the absolute minimum time that the current SDA value remains stable after SCL transitions from high-to-low. EXTHOLD should be set so that the minimum setup and hold times meet the SMBus Specification requirements of 250 ns and 300 ns, respectively. Table 14.2 shows the minimum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

Table 14.2. Minimum SDA Setup and Hold Times

EXTHOLD	Minimum SDA Setup Time	Minimum SDA Hold Time
0	T_{low} – 4 system clocks OR 1 system clock + s/w delay*	3 system clocks
1	11 system clocks	12 system clocks

***Note:** Setup Time for ACK bit transmissions and the MSB of all data transfers. The s/w delay occurs between the time SMB0DAT or ACK is written and when SI is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero.

With the SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts (see **Section “14.3.3. SCL Low Timeout” on page 148**). The SMBus interface will force Timer 3 to reload while SCL is high, and allow Timer 3 to count when SCL is low. The Timer 3 interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods (see Figure 14.4). When a Free Timeout is detected, the interface will respond as if a STOP was detected (an interrupt will be generated, and STO will be set).

15. UART0

UART0 is an asynchronous, full duplex serial port offering modes 1 and 3 of the standard 8051 UART. Enhanced baud rate support allows a wide range of clock sources to generate standard baud rates (details in **Section “15.1. Enhanced Baud Rate Generation” on page 164**). Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

UART0 has two associated SFRs: Serial Control Register 0 (SCON0) and Serial Data Buffer 0 (SBUF0). The single SBUF0 location provides access to both transmit and receive registers. **Writes to SBUF0 always access the Transmit register. Reads of SBUF0 always access the buffered Receive register; it is not possible to read data from the Transmit register.**

With UART0 interrupts enabled, an interrupt is generated each time a transmit is completed (TI0 is set in SCON0), or a data byte has been received (RI0 is set in SCON0). The UART0 interrupt flags are not cleared by hardware when the CPU vectors to the interrupt service routine. They must be cleared manually by software, allowing software to determine the cause of the UART0 interrupt (transmit complete or receive complete).

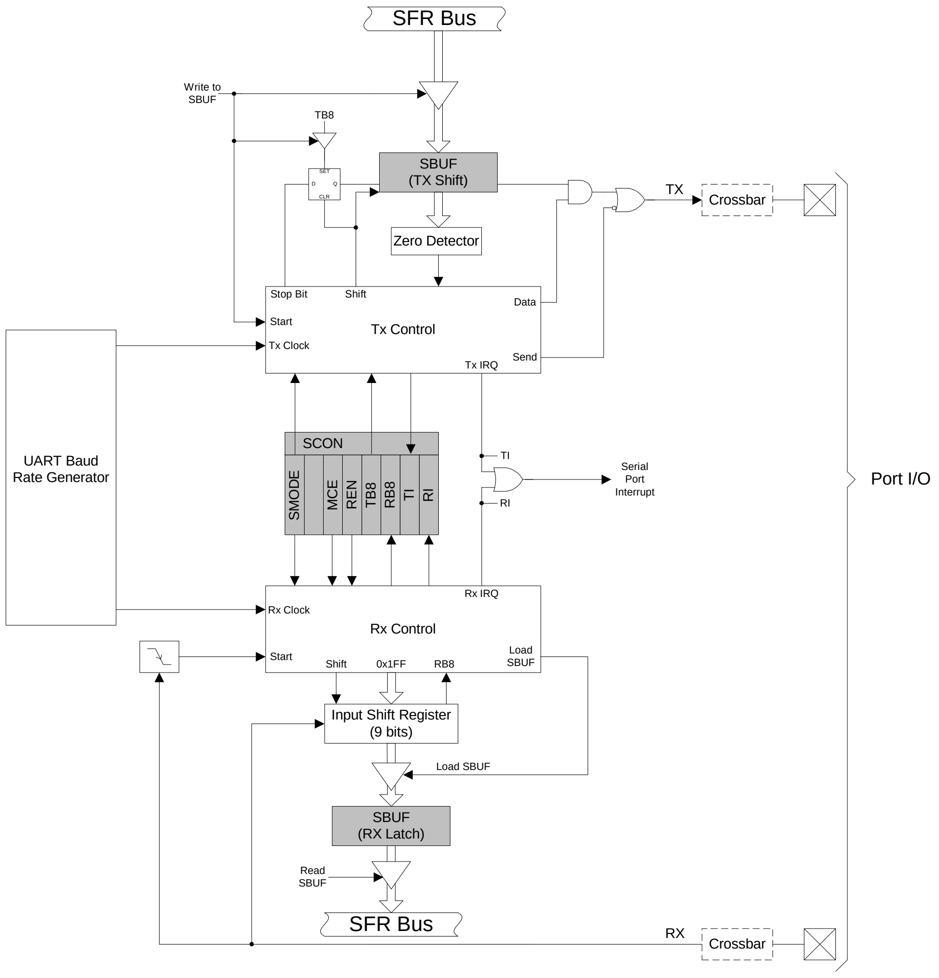

Figure 15.1. UART0 Block Diagram

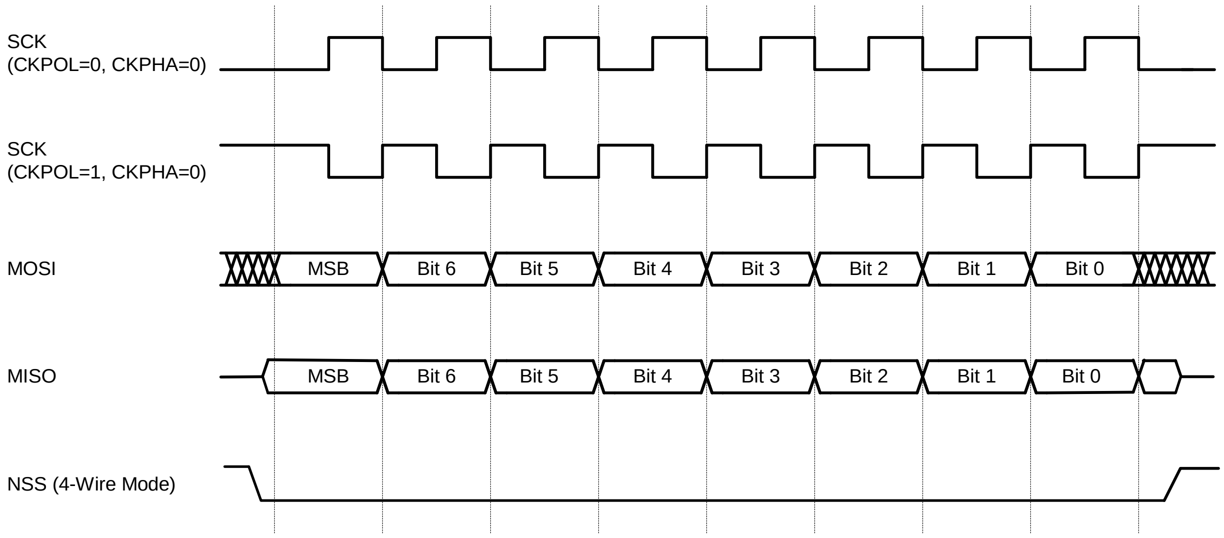

Figure 16.6. Slave Mode Data/Clock Timing (CKPHA = 0)

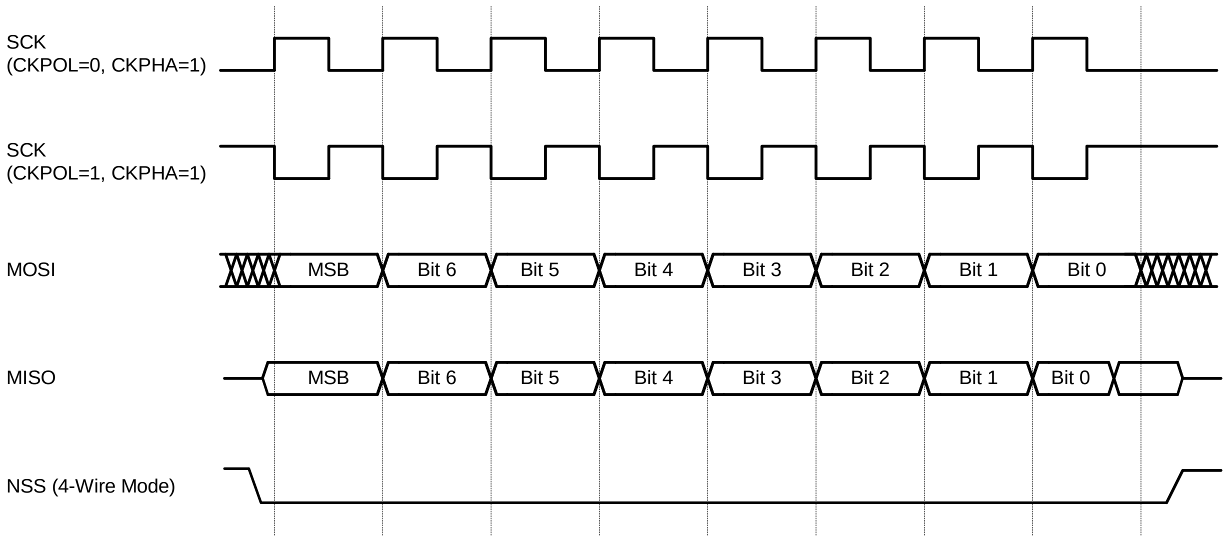

Figure 16.7. Slave Mode Data/Clock Timing (CKPHA = 1)

SFR Definition 17.2. TMOD: Timer Mode

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

Bit7: GATE1: Timer 1 Gate Control.
0: Timer 1 enabled when TR1 = 1 irrespective of /INT1 logic level.
1: Timer 1 enabled only when TR1 = 1 AND /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 8.11).

Bit6: C/T1: Counter/Timer 1 Select.
0: Timer Function: Timer 1 incremented by clock defined by T1M bit (CKCON.4).
1: Counter Function: Timer 1 incremented by high-to-low transitions on external input pin (T1).

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

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

Bit3: GATE0: Timer 0 Gate Control.
0: Timer 0 enabled when TR0 = 1 irrespective of /INT0 logic level.
1: Timer 0 enabled only when TR0 = 1 AND /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 8.11).

Bit2: C/T0: Counter/Timer Select.
0: Timer Function: Timer 0 incremented by clock defined by T0M bit (CKCON.3).
1: Counter Function: Timer 0 incremented by high-to-low transitions on external input pin (T0).

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

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

18.2.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1 and an interrupt request is generated if CCF interrupts are 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. If both CAPPn and CAPNn bits are set to logic 1, then the state of the Port pin associated with CEXn can be read directly to determine whether a rising-edge or falling-edge caused the capture.

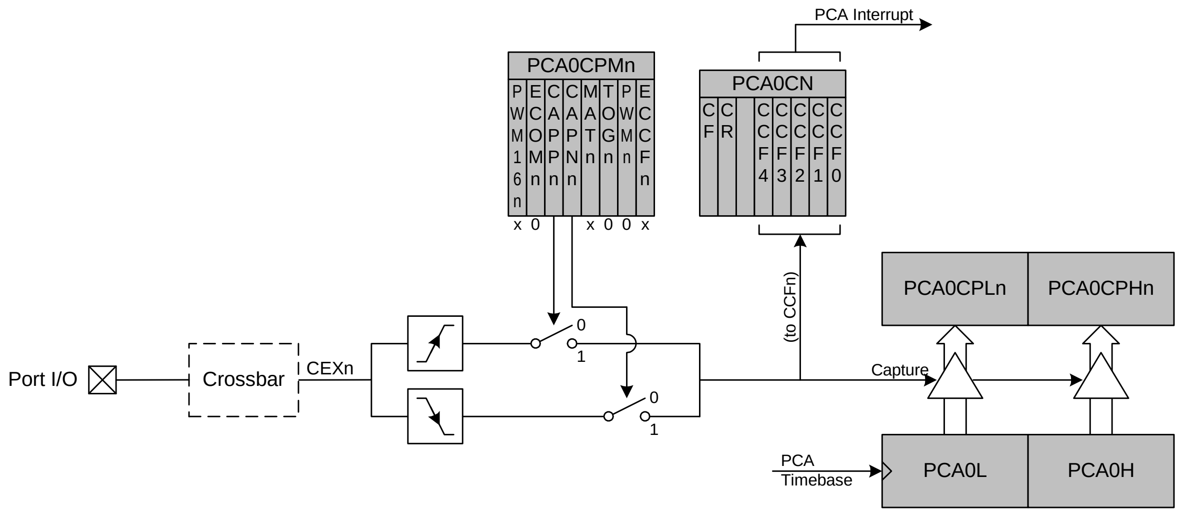

Figure 18.4. PCA Capture Mode Diagram

Note: The CEXn input signal must remain high or low for at least 2 system clock cycles in order to be valid.

SFR Definition 18.2. PCA0MD: PCA Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CIDL	WDTE	WDLCK		CPS2	CPS1	CPS0	ECF	01000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xD9

Bit7: CIDL: PCA Counter/Timer Idle Control.
Specifies PCA behavior when CPU is in Idle Mode.
0: PCA continues to function normally while the system controller is in Idle Mode.
1: PCA operation is suspended while the system controller is in Idle Mode.

Bit6: WDTE: Watchdog Timer Enable
If this bit is set, PCA Module 4 is used as the watchdog timer.
0: Watchdog Timer disabled.
1: PCA Module 4 enabled as Watchdog Timer.

Bit5: WDLCK: Watchdog Timer Lock
This bit locks/unlocks the Watchdog Timer Enable. When WDLCK is set, the Watchdog Timer may not be disabled until the next system reset.
0: Watchdog Timer Enable unlocked.
1: Watchdog Timer Enable locked.

Bit4: UNUSED. Read = 0b, Write = don't care.

Bits3–1: CPS2–CPS0: PCA Counter/Timer Pulse Select.
These bits select the timebase source for the PCA counter.

CPS2	CPS1	CPS0	Timebase
0	0	0	System clock divided by 12
0	0	1	System clock divided by 4
0	1	0	Timer 0 overflow
0	1	1	High-to-low transitions on ECI (max rate = system clock divided by 4)
1	0	0	System clock
1	0	1	External clock divided by 8*
1	1	0	Reserved
1	1	1	Reserved

***Note:** External oscillator source divided by 8 is synchronized with the system clock.

Bit0: ECF: PCA Counter/Timer Overflow Interrupt Enable.
This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.
0: Disable the CF interrupt.
1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set.

Note: When the WDTE bit is set to '1', the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled.