

Welcome to [E-XFL.COM](https://www.e-xfl.com)

What is "[Embedded - Microcontrollers](#)"?

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

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Obsolete
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I ² C), SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	17
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x24b; D/A 2x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VFQFN Exposed Pad
Supplier Device Package	28-MLP (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f351r

Table 20.6. Timer Settings for Standard Baud Rates Using an External Oscillator	176
21. Serial Peripheral Interface (SPI0)	177
Figure 21.1. SPI Block Diagram	177
Figure 21.2. Multiple-Master Mode Connection Diagram	180
Figure 21.3. 3-Wire Single Master and Single Slave Mode Connection Diagram	180
Figure 21.4. 4-Wire Single Master and Slave Mode Connection Diagram	180
Figure 21.5. Data/Clock Timing Relationship	182
Figure 21.6. SPI0CFG: SPI0 Configuration Register	183
Figure 21.7. SPI0CN: SPI0 Control Register	184
Figure 21.8. SPI0CKR: SPI0 Clock Rate Register	185
Figure 21.9. SPI0DAT: SPI0 Data Register	186
Figure 21.10. SPI Master Timing (CKPHA = 0)	187
Figure 21.11. SPI Master Timing (CKPHA = 1)	187
Figure 21.12. SPI Slave Timing (CKPHA = 0)	188
Figure 21.13. SPI Slave Timing (CKPHA = 1)	188
Table 21.1. SPI Slave Timing Parameters	189
22. Timers	191
Figure 22.1. T0 Mode 0 Block Diagram	192
Figure 22.2. T0 Mode 2 Block Diagram	193
Figure 22.3. T0 Mode 3 Block Diagram	194
Figure 22.4. TCON: Timer Control Register	195
Figure 22.5. TMOD: Timer Mode Register	196
Figure 22.6. CKCON: Clock Control Register	197
Figure 22.7. TL0: Timer 0 Low Byte	198
Figure 22.8. TL1: Timer 1 Low Byte	198
Figure 22.9. TH0: Timer 0 High Byte	198
Figure 22.10. TH1: Timer 1 High Byte	198
Figure 22.11. Timer 2 16-Bit Mode Block Diagram	199
Figure 22.12. Timer 2 8-Bit Mode Block Diagram	200
Figure 22.13. TMR2CN: Timer 2 Control Register	201
Figure 22.14. TMR2RLL: Timer 2 Reload Register Low Byte	202
Figure 22.15. TMR2RLH: Timer 2 Reload Register High Byte	202
Figure 22.16. TMR2L: Timer 2 Low Byte	202
Figure 22.17. TMR2H: Timer 2 High Byte	202
Figure 22.18. Timer 3 16-Bit Mode Block Diagram	203
Figure 22.19. Timer 3 8-Bit Mode Block Diagram	204
Figure 22.20. TMR3CN: Timer 3 Control Register	205
Figure 22.21. TMR3RLL: Timer 3 Reload Register Low Byte	206
Figure 22.22. TMR3RLH: Timer 3 Reload Register High Byte	206
Figure 22.23. TMR3L: Timer 3 Low Byte	206
Figure 22.24. TMR3H: Timer 3 High Byte	206
23. Programmable Counter Array	207
Figure 23.1. PCA Block Diagram	207
Figure 23.2. PCA Counter/Timer Block Diagram	208
Table 23.1. PCA Timebase Input Options	208

Figure 23.3. PCA Interrupt Block Diagram	209
Table 23.2. PCA0CPM Register Settings for PCA Capture/Compare Modules	209
Figure 23.4. PCA Capture Mode Diagram.....	210
Figure 23.5. PCA Software Timer Mode Diagram	211
Figure 23.6. PCA High Speed Output Mode Diagram.....	212
Figure 23.7. PCA Frequency Output Mode	213
Figure 23.8. PCA 8-Bit PWM Mode Diagram	214
Figure 23.9. PCA 16-Bit PWM Mode.....	215
Figure 23.10. PCA Module 2 with Watchdog Timer Enabled	216
Table 23.3. Watchdog Timer Timeout Intervals†.....	217
Figure 23.11. PCA0CN: PCA Control Register	218
Figure 23.12. PCA0MD: PCA Mode Register.....	219
Figure 23.13. PCA0CPMn: PCA Capture/Compare Mode Registers.....	220
Figure 23.14. PCA0L: PCA Counter/Timer Low Byte.....	221
Figure 23.15. PCA0H: PCA Counter/Timer High Byte	221
Figure 23.16. PCA0CPLn: PCA Capture Module Low Byte	222
Figure 23.17. PCA0CPHn: PCA Capture Module High Byte.....	222
24. C2 Interface	223
Figure 24.1. C2ADD: C2 Address Register.....	223
Figure 24.2. DEVICEID: C2 Device ID Register.....	223
Figure 24.3. REVID: C2 Revision ID Register.....	224
Figure 24.4. FPCTL: C2 Flash Programming Control Register.....	224
Figure 24.5. FPDAT: C2 Flash Programming Data Register.....	224
Figure 24.6. Typical C2 Pin Sharing.....	225

1. System Overview

C8051F350/1/2/3 devices are fully integrated mixed-signal System-on-a-Chip MCUs. Highlighted features are listed below. Refer to Table 1.1 for specific product feature selection.

- High-speed pipelined 8051-compatible microcontroller core (up to 50 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- 24 or 16-bit single-ended/differential ADC with analog multiplexer
- Two 8-bit Current Output DACs
- Precision programmable 24.5 MHz internal oscillator
- 8 kB of on-chip Flash memory
- 768 bytes of on-chip RAM
- SMBus/I2C, Enhanced UART, and SPI serial interfaces implemented in hardware
- Four general-purpose 16-bit timers
- Programmable counter/timer array (PCA) with three capture/compare modules and watchdog timer function
- On-chip power-on reset, V_{DD} monitor, and temperature sensor
- On-chip voltage comparator
- 17 Port I/O (5 V tolerant)

With on-chip power-on reset, V_{DD} monitor, watchdog timer, and clock oscillator, the C8051F350/1/2/3 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 on-chip Silicon Labs 2-Wire (C2) Development Interface 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 and modification of memory and registers, setting breakpoints, single stepping, 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 2.7 V-to-3.6 V operation over the industrial temperature range (-45 to +85 °C). The Port I/O and /RST pins are tolerant of input signals up to 5 V. The C8051F350/1/2/3 are available in 28-pin MLP or 32-pin LQFP packaging, as shown in Figure 1.1 through Figure 1.4.

C8051F350/1/2/3

Table 1.1. Product Selection Guide

	MIPS (Peak)	Flash Memory	RAM	Calibrated Internal 24.5 MHz Oscillator	Clock Multiplier	SMBus/I2C	SPI	UART	Timers (16-bit)	Programmable Counter Array	Digital Port I/Os	24-bit ADC	16-bit ADC	Two 8-bit Current Output DACs	Internal Voltage Reference	Temperature Sensor	Analog Comparator	Package
C8051F350	50	8 kB	768	✓	✓	✓	✓	✓	4	✓	17	✓	-	✓	✓	✓	✓	LQFP-32
C8051F351	50	8 kB	768	✓	✓	✓	✓	✓	4	✓	17	✓	-	✓	✓	✓	✓	MLP-28
C8051F352	50	8 kB	768	✓	✓	✓	✓	✓	4	✓	17	-	✓	✓	✓	✓	✓	LQFP-32
C8051F353	50	8 kB	768	✓	✓	✓	✓	✓	4	✓	17	-	✓	✓	✓	✓	✓	MLP-28

C8051F350/1/2/3

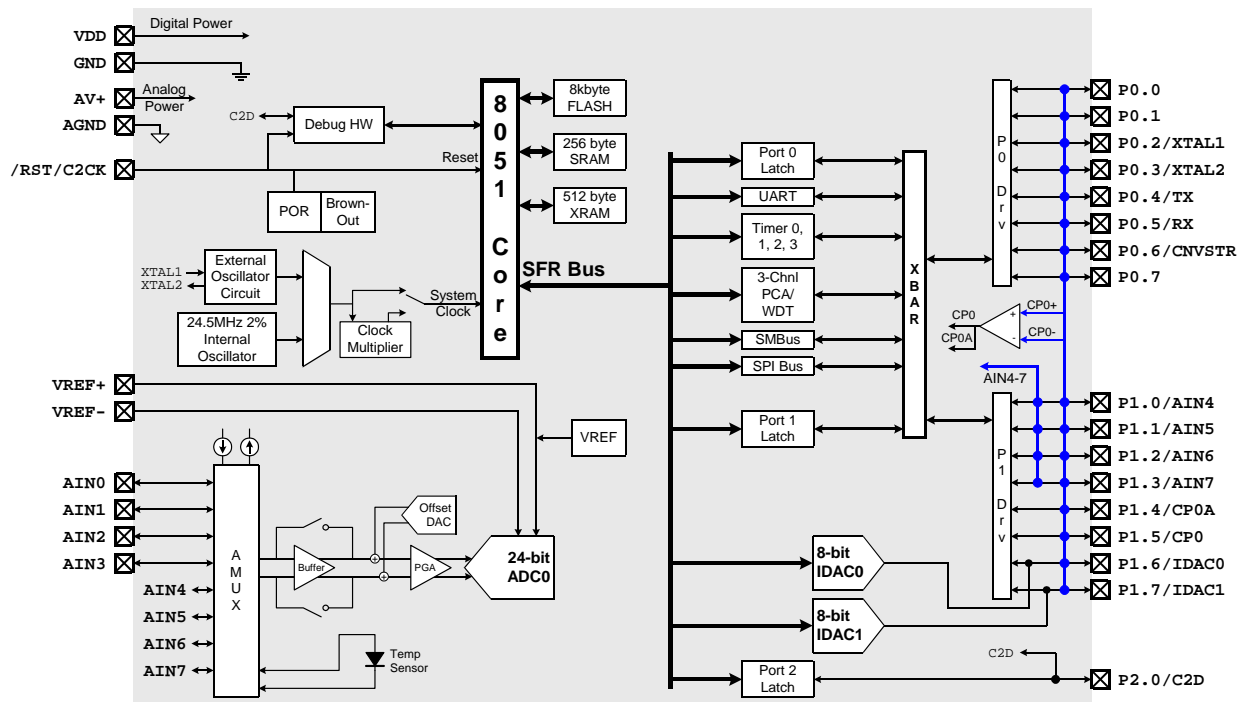


Figure 1.2. C8051F351 Block Diagram

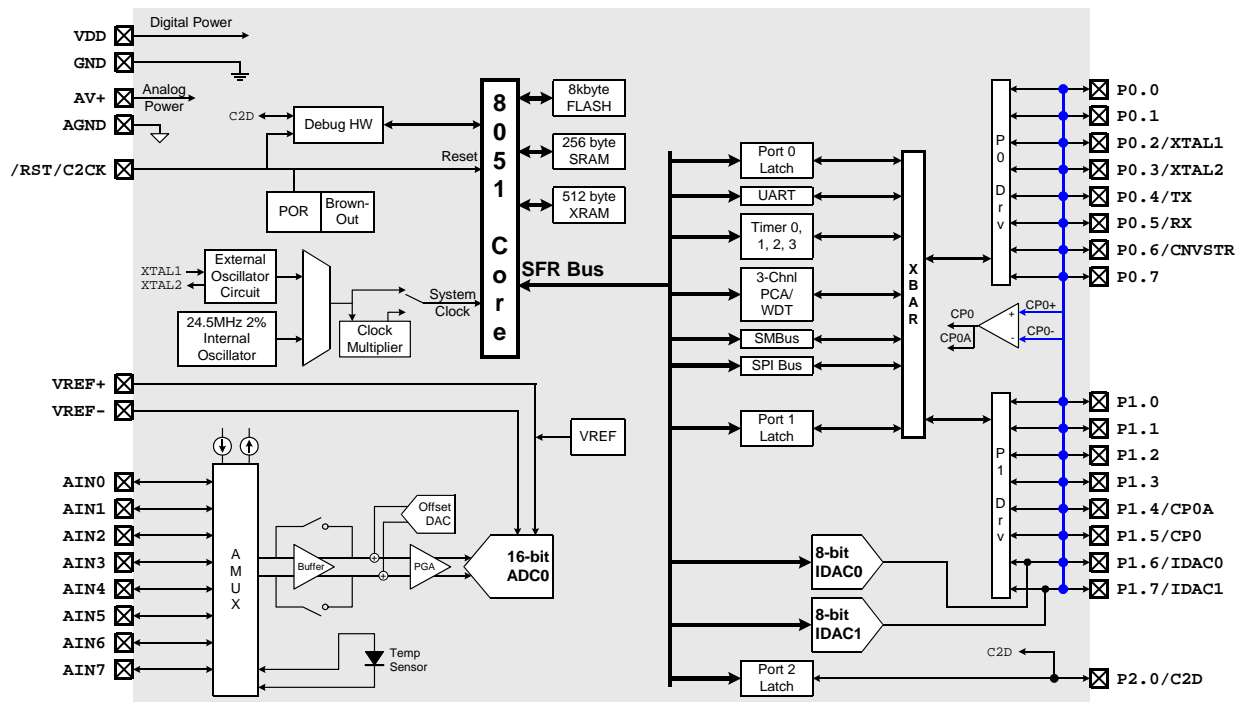


Figure 1.3. C8051F352 Block Diagram

1.1. CIP-51™ Microcontroller

1.1.1. Fully 8051 Compatible Instruction Set

The C8051F35x devices use Silicon Labs' proprietary CIP-51 microcontroller core. 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 C8051F35x family has a superset of all the peripherals included with a standard 8052.

1.1.2. Improved Throughput

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-to-24 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 system clock running at 50 MHz, it has a peak throughput of 50 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

1.1.3. Additional Features

The C8051F350/1/2/3 SoC family includes several key enhancements to the CIP-51 core and peripherals to improve performance and ease of use in end applications.

An extended interrupt handler allows the numerous analog and digital peripherals to operate independently of the controller core and interrupt the controller only when necessary. By requiring less intervention from the microcontroller core, an interrupt-driven system is more efficient and allows for easier implementation of multi-tasking, real-time systems.

Eight reset sources are available: power-on reset circuitry (POR), an on-chip V_{DD} monitor, a Watchdog Timer, a Missing Clock Detector, a voltage level detection from Comparator0, a forced software reset, an external reset pin, and an illegal Flash access protection circuit. Each reset source except for the POR, Reset Input Pin, or Flash error may be disabled by the user in software. The WDT may be permanently enabled in software after a power-on reset during MCU initialization.

The internal oscillator is factory calibrated to 24.5 MHz $\pm 2\%$. An external oscillator drive circuit is also included, allowing an external crystal, ceramic resonator, capacitor, RC, or CMOS clock source to generate the system clock. A clock multiplier allows for operation at up to 50 MHz. An external oscillator can also be extremely useful in low power applications, allowing the MCU to run from a slow (power saving) source, while periodically switching to the fast internal oscillator as needed.

C8051F350/1/2/3

1.6. Programmable Comparator

C8051F350/1/2/3 devices include a software-configurable voltage comparator with an input multiplexer. The Comparator offers programmable response time and hysteresis and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0), or an asynchronous “raw” output (CP0A). Comparator interrupts may be generated on rising, falling, or both edges. When in IDLE mode, these interrupts may be used as a “wake-up” source for the processor. Comparator0 may also be configured as a reset source. A block diagram of the Comparator is shown in Figure 1.9.

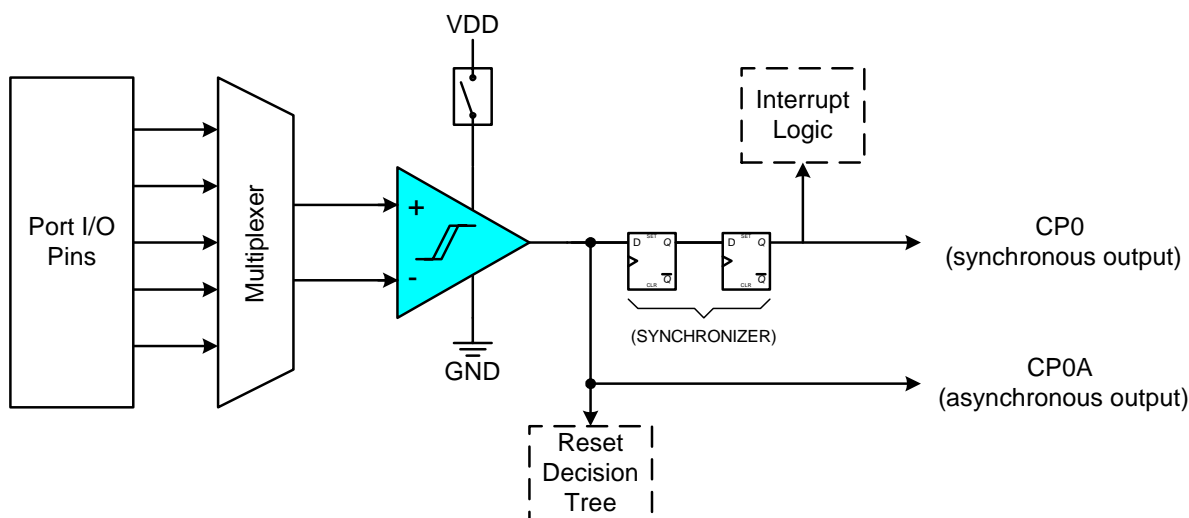


Figure 1.9. Comparator0 Block Diagram

1.7. Serial Ports

The C8051F350/1/2/3 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.

Top View

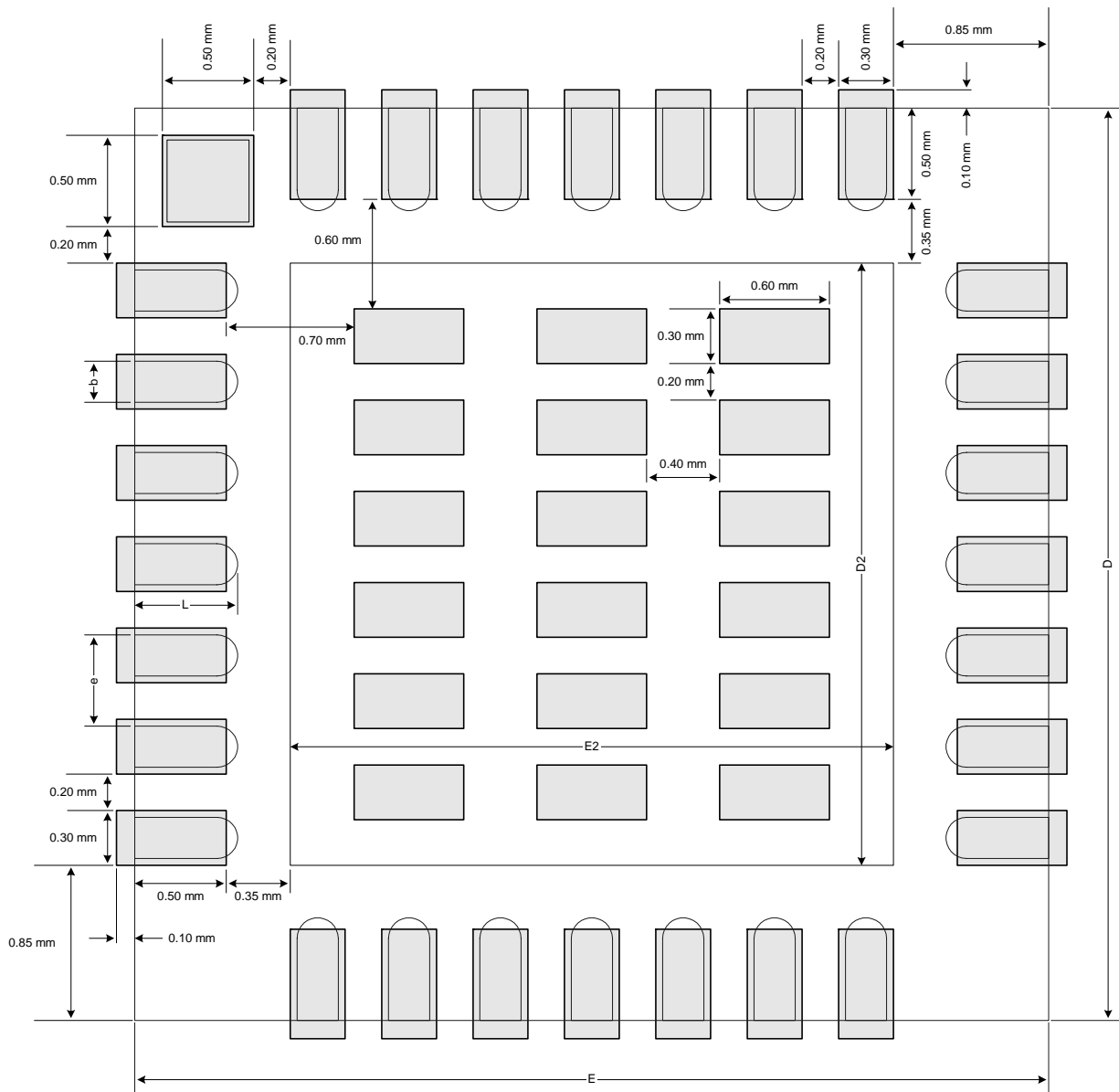


Figure 4.6. Typical MLP-28 Solder Paste Diagram

10.2. Register Descriptions

Following are descriptions of SFRs related to the operation of the CIP-51 System Controller. Reserved bits should not be set to logic 1. Future product versions may use these bits to implement new features in which case the reset value of the bit will be logic 0, selecting the feature's default state. Detailed descriptions of the remaining SFRs are included in the sections of the datasheet associated with their corresponding system function.

Figure 10.2. SP: Stack Pointer

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0x81

Bits7-0: SP: Stack Pointer.
The Stack Pointer holds the location of the top of the stack. The stack pointer is incremented before every PUSH operation. The SP register defaults to 0x07 after reset.

Figure 10.3. DPL: Data Pointer Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0x82

Bits7-0: DPL: Data Pointer Low.
The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

Figure 10.4. DPH: Data Pointer High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

SFR Address: 0x83

Bits7-0: DPH: Data Pointer High.
The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

Figure 10.6. ACC: Accumulator

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
ACC.7	ACC.6	ACC.5	ACC.4	ACC.3	ACC.2	ACC.1	ACC.0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Bit Addressable
								SFR Address: 0xE0

Bits7-0: ACC: Accumulator.
This register is the accumulator for arithmetic operations.

Figure 10.7. 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	Bit Addressable
								SFR Address: 0xF0

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

12. Interrupt Handler

The C8051F35x family includes an extended interrupt system supporting a total of 12 interrupt sources with two priority levels. The allocation of interrupt sources between on-chip peripherals and external input 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 a 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 the Interrupt Enable and Extended Interrupt Enable SFRs. 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 that interrupts which occur when the EA bit is set to logic 0 will be held in a pending state, and will not be serviced until the EA bit is set back to logic 1.

Some interrupt-pending flags are automatically cleared by the hardware when the CPU vectors to the ISR. However, most are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the return-from-interrupt (RETI) instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

12.1. MCU Interrupt Sources and Vectors

The MCUs support 12 interrupt sources. Software can simulate an interrupt by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag. MCU interrupt sources, associated vector addresses, priority order and control bits are summarized in Table 12.1 on page 100. Refer to the datasheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

12.2. Interrupt Priorities

Each interrupt source can be individually programmed to one of two priority levels: low or high. A low priority interrupt service routine can be preempted by a high priority interrupt. A high priority interrupt cannot be preempted. Each interrupt has an associated interrupt priority bit in an SFR (IP or IIP1) used to configure its priority level. Low priority is the default. If two interrupts are recognized simultaneously, the interrupt with the higher priority is serviced first. If both interrupts have the same priority level, a fixed priority order is used to arbitrate, given in Table 12.1.

12.3. Interrupt Latency

Interrupt response time depends on the state of the CPU when the interrupt occurs. Pending interrupts are sampled and priority decoded each system clock cycle. Therefore, the fastest possible response time is 5 system clock cycles: 1 clock cycle to detect the interrupt and 4 clock cycles to complete the LCALL to the ISR. If an interrupt is pending when a RETI is executed, a single instruction is executed before an LCALL

is made to service the pending interrupt. Therefore, the maximum response time for an interrupt (when no other interrupt is currently being serviced or the new interrupt is of greater priority) occurs when the CPU is performing an RETI instruction followed by a DIV as the next instruction. In this case, the response time is 18 system clock cycles: 1 clock cycle to detect the interrupt, 5 clock cycles to execute the RETI, 8 clock cycles to complete the DIV instruction and 4 clock cycles to execute the LCALL to the ISR. If the CPU is executing an ISR for an interrupt with equal or higher priority, the new interrupt will not be serviced until the current ISR completes, including the RETI and following instruction.

Table 12.1. Interrupt Summary

Interrupt Source	Interrupt Vector	Priority Order	Pending Flag	Bit addressable?	Cleared by HW?	Enable Flag	Priority Control
Reset	0x0000	Top	None	N/A	N/A	Always Enabled	Always Highest
External Interrupt 0 (/INT0)	0x0003	0	IE0 (TCON.1)	Y	Y	EX0 (IE.0)	PX0 (IP.0)
Timer 0 Overflow	0x000B	1	TF0 (TCON.5)	Y	Y	ET0 (IE.1)	PT0 (IP.1)
External Interrupt 1 (/INT1)	0x0013	2	IE1 (TCON.3)	Y	Y	EX1 (IE.2)	PX1 (IP.2)
Timer 1 Overflow	0x001B	3	TF1 (TCON.7)	Y	Y	ET1 (IE.3)	PT1 (IP.3)
UART0	0x0023	4	RI0 (SCON0.0) TI0 (SCON0.1)	Y	N	ES0 (IE.4)	PS0 (IP.4)
Timer 2 Overflow	0x002B	5	TF2H (TMR2CN.7) TF2L (TMR2CN.6)	Y	N	ET2 (IE.5)	PT2 (IP.5)
SPI0	0x0033	6	SPIF (SPI0CN.7) WCOL (SPI0CN.6) MODF (SPI0CN.5) RXOVRN (SPI0CN.4)	Y	N	ESPI0 (IE.6)	PSPI0 (IP.6)
SMB0	0x003B	7	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)	PSMB0 (EIP1.0)
RESERVED	0x0043	8	N/A	N/A	N/A	N/A	N/A
RESERVED	0x004B	9	N/A	N/A	N/A	N/A	N/A
ADC0	0x0053	10	AD0INT (ADC0STA.5)	Y	N	EADC0 (EIE1.3)	PADC0 (EIP1.3)
Programmable Counter Array	0x005B	11	CF (PCA0CN.7) CCFn (PCA0CN.n)	Y	N	EPCA0 (EIE1.4)	PPCA0 (EIP1.4)
Comparator0	0x0063	12	CP0FIF (CPT0CN.4) CP0RIF (CPT0CN.5)	N	N	ECP0 (EIE1.5)	PCP0 (EIP1.5)
RESERVED	0x006B	13	N/A	N/A	N/A	N/A	N/A
Timer 3 Overflow	0x0073	14	TF3H (TMR3CN.7) TF3L (TMR3CN.6)	N	N	ET3 (EIE1.7)	PT3 (EIP1.7)

Preliminary

15.2. Non-volatile Data Storage

The Flash memory can be used for non-volatile data storage as well as program code. This allows data such as calibration coefficients to be calculated and stored at run time. Data is written using the MOVX write instruction and read using the MOVC instruction. Note: MOVX read instructions always target XRAM.

15.3. Security Options

The CIP-51 provides security options to protect the Flash memory from inadvertent modification by software as well as to prevent the viewing of proprietary program code and constants. The Program Store Write Enable (bit PSWE in register PSCTL) and the Program Store Erase Enable (bit PSEE in register PSCTL) bits protect the Flash memory from accidental modification by software. PSWE must be explicitly set to '1' before software can modify the Flash memory; both PSWE and PSEE must be set to '1' before software can erase Flash memory. Additional security features prevent proprietary program code and data constants from being read or altered across the C2 interface.

A Security Lock Byte located at the last byte of Flash user space offers protection of the Flash program memory from access (reads, writes, or erases) by unprotected code or the C2 interface. The Flash security mechanism allows the user to lock n 512-byte Flash pages, starting at page 0 (addresses 0x0000 to 0x01FF), where n is the 1's complement number represented by the Security Lock Byte. Note that the page containing the Flash Security Lock Byte is locked when any other Flash pages are locked. See example below.

Security Lock Byte:	11111101b
1's Complement:	00000010b
Flash pages locked:	3 (First two Flash pages + Lock Byte Page)
Addresses locked:	0x0000 to 0x03FF (first two Flash pages) and 0x1C00 to 0x1DFF (Lock Byte Page)

Important Notes About the Flash Security:

1. Clearing any bit of the Lock Byte to '0' will lock the Flash page containing the Lock Byte (in addition to the selected pages).
2. Locked pages cannot be read, written, or erased via the C2 interface.
3. Locked pages cannot be read, written, or erased by user firmware executing from unlocked memory space.
4. User firmware executing in a locked page may read and write Flash memory in any locked or unlocked page excluding the reserved area.
5. User firmware executing in a locked page may erase Flash memory in any locked or unlocked page excluding the reserved area and the page containing the Lock Byte.
6. Locked pages can only be unlocked by a C2 Device Erase command.
7. If a user firmware Flash access attempt is denied (per restrictions #3, #4, and #5 above), a Flash Error system reset will be generated.

19.5.2. Master Receiver Mode

Serial data is received on SDA while the serial clock is output on SCL. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data. After each byte is received, ACKRQ is set to '1' and an interrupt is generated. Software must write the ACK bit (SMB0CN.1) to define the outgoing acknowledge value (Note: writing a '1' to the ACK bit generates an ACK; writing a '0' generates a NACK). Software should write a '0' to the ACK bit after the last byte is received, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. Note that the interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 19.9 shows a typical Master Receiver sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the 'data byte transferred' interrupts occur **before** the ACK cycle in this mode.

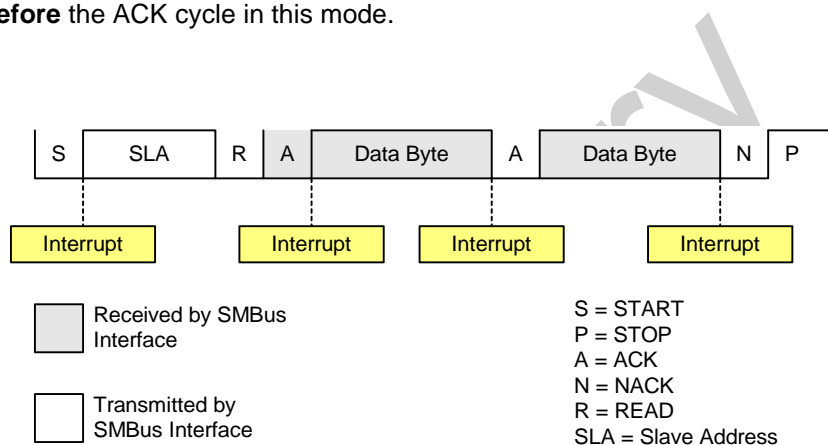


Figure 19.9. Typical Master Receiver Sequence

20.2. Operational Modes

UART0 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S0MODE bit (SCON0.7). Typical UART connection options are shown below.

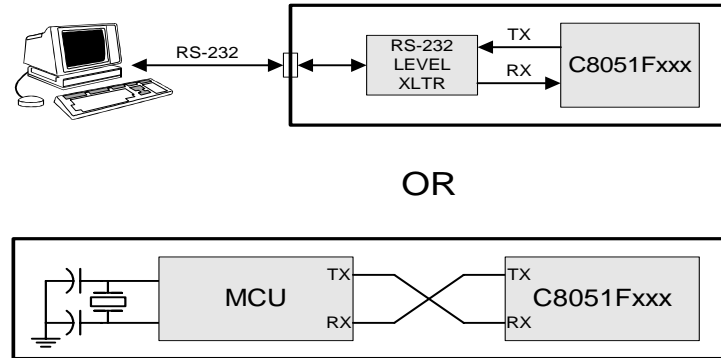


Figure 20.3. UART Interconnect Diagram

20.2.1. 8-Bit UART

8-Bit UART mode uses a total of 10 bits per data byte: one start bit, eight data bits (LSB first), and one stop bit. Data are transmitted LSB first from the TX0 pin and received at the RX0 pin. On receive, the eight data bits are stored in SBUF0 and the stop bit goes into RB80 (SCON0.2).

Data transmission begins when software writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to logic 1. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: RI0 must be logic 0, and if MCE0 is logic 1, the stop bit must be logic 1. In the event of a receive data overrun, the first received 8 bits are latched into the SBUF0 receive register and the following overrun data bits are lost.

If these conditions are met, the eight bits of data is stored in SBUF0, the stop bit is stored in RB80 and the RI0 flag is set. If these conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set. An interrupt will occur if enabled when either TI0 or RI0 is set.

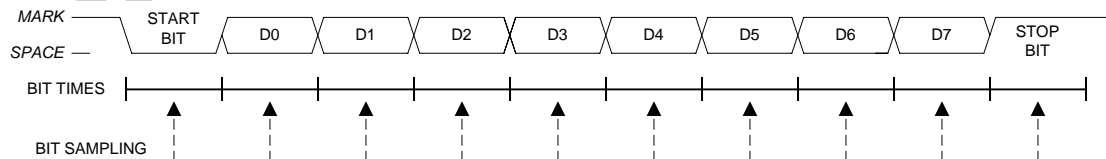


Figure 20.4. 8-Bit UART Timing Diagram

21.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

21.1.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to slave devices. It is used to serially transfer data from the master to the slave. This signal is an output when SPI0 is operating as a master and an input when SPI0 is operating as a slave. Data is transferred most-significant bit first. When configured as a master, MOSI is driven by the MSB of the shift register in both 3- and 4-wire mode.

21.1.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to the master device. It is used to serially transfer data from the slave to the master. This signal is an input when SPI0 is operating as a master and an output when SPI0 is operating as a slave. Data is transferred most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled and when the SPI operates in 4-wire mode as a slave that is not selected. When acting as a slave in 3-wire mode, MISO is always driven by the MSB of the shift register.

21.1.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. SPI0 generates this signal when operating as a master. The SCK signal is ignored by a SPI slave when the slave is not selected (NSS = 1) in 4-wire slave mode.

21.1.4. Slave Select (NSS)

The function of the slave-select (NSS) signal is dependent on the setting of the NSSMD1 and NSSMD0 bits in the SPI0CN register. There are three possible modes that can be selected with these bits:

1. NSSMD[1:0] = 00: 3-Wire Master or 3-Wire Slave Mode: SPI0 operates in 3-wire mode, and NSS is disabled. When operating as a slave device, SPI0 is always selected in 3-wire mode. Since no select signal is present, SPI0 must be the only slave on the bus in 3-wire mode. This is intended for point-to-point communication between a master and one slave.
2. NSSMD[1:0] = 01: 4-Wire Slave or Multi-Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an input. When operating as a slave, NSS selects the SPI0 device. When operating as a master, a 1-to-0 transition of the NSS signal disables the master function of SPI0 so that multiple master devices can be used on the same SPI bus.
3. NSSMD[1:0] = 1x: 4-Wire Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an output. The setting of NSSMD0 determines what logic level the NSS pin will output. This configuration should only be used when operating SPI0 as a master device.

See Figure 21.2, Figure 21.3, and Figure 21.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device. See Section “18. Port Input/Output” on page 133 for general purpose port I/O and crossbar information.

22.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload

Mode 2 configures Timer 0 and Timer 1 to operate as 8-bit counter/timers with automatic reload of the start value. TL0 holds the count and TH0 holds the reload value. When the counter in TL0 overflows from all ones to 0x00, the timer overflow flag TF0 (TCON.5) is set and the counter in TL0 is reloaded from TH0. If Timer 0 interrupts are enabled, an interrupt will occur when the TF0 flag is set. The reload value in TH0 is not changed. TL0 must be initialized to the desired value before enabling the timer for the first count to be correct. When in Mode 2, Timer 1 operates identically to Timer 0.

Both counter/timers are enabled and configured in Mode 2 in the same manner as Mode 0. Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or when the input signal /INT0 is active as defined by bit IN0PL in register IT01CF (see [Section “12.5. External Interrupts” on page 105](#) for details on the external input signals /INT0 and /INT1).

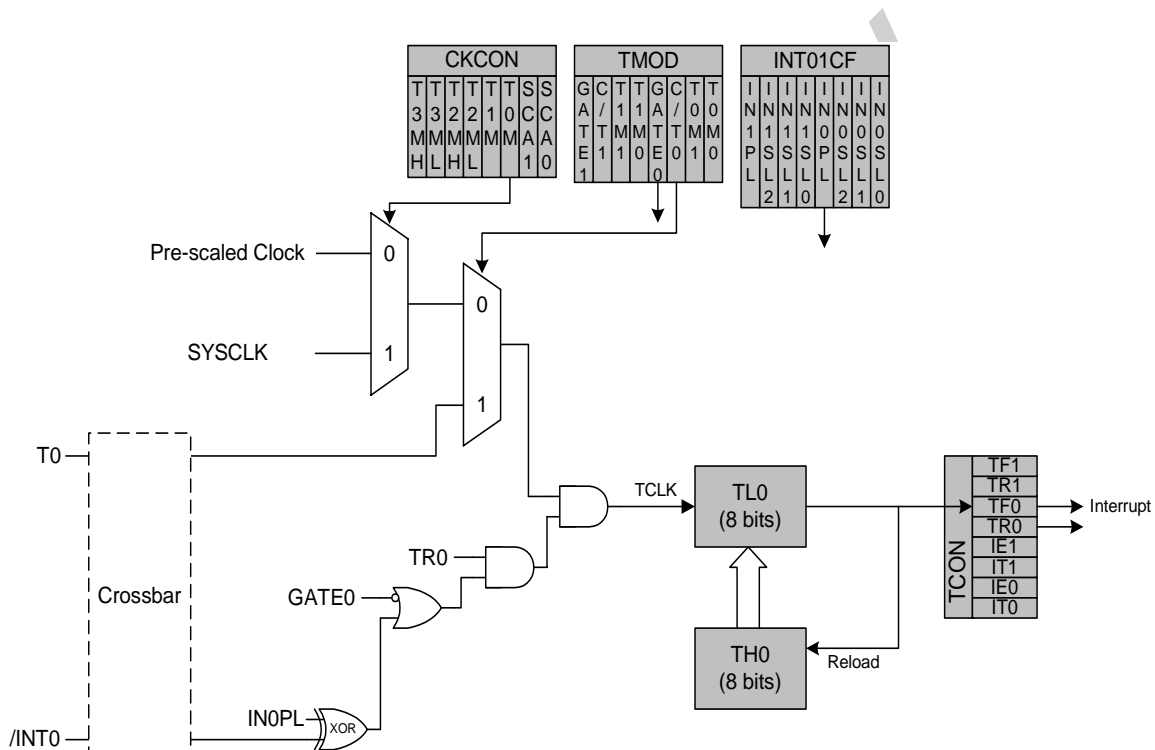


Figure 22.2. T0 Mode 2 Block Diagram

23.2.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEX_n 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 (PCA0CPL_n and PCA0CPH_n). The CAPP_n and CAPN_n bits in the PCA0CPM_n 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 (CCF_n) in PCA0CN is set to logic 1 and an interrupt request is generated if CCF interrupts are enabled. The CCF_n bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPP_n and CAPN_n bits are set to logic 1, then the state of the Port pin associated with CEX_n can be read directly to determine whether a rising-edge or falling-edge caused the capture.

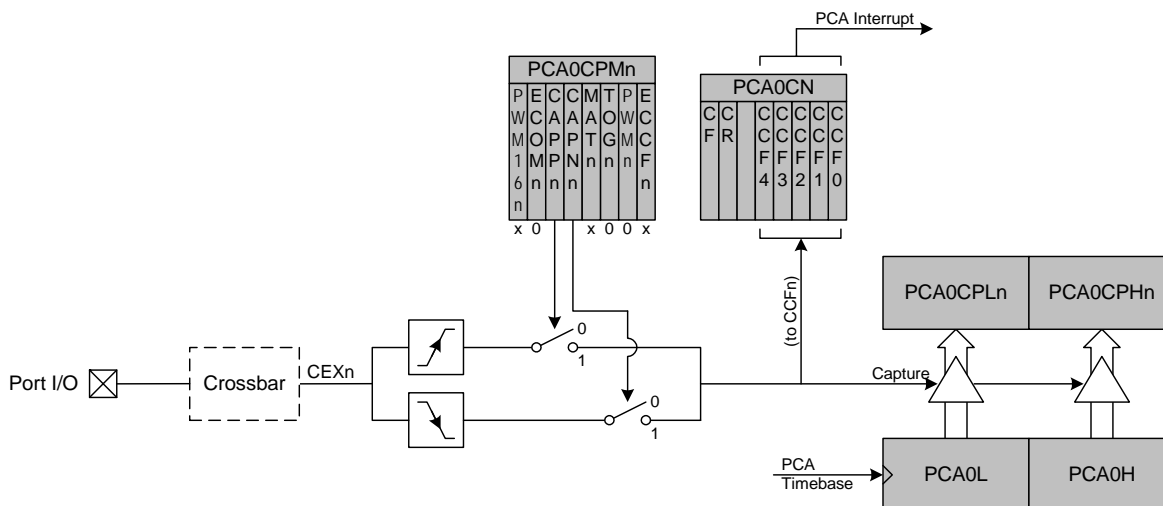


Figure 23.4. PCA Capture Mode Diagram

Note: The CEX_n input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.