C8051T600/1/2/3/4/5

Mixed Signal OTP EPROM MCU Family

Analog Peripherals

- **10-Bit Analog to Digital Converter**
  - Up to 500 ksps
  - Up to 8 external inputs
  - V_{REF} from external pin, V_{DD}, or internal regulator
  - Built-in temperature sensor
  - External conversion start input option
- **Comparator**
  - Programmable hysteresis and response time
  - Configurable as interrupt or reset source
  - Low current (< 0.5 µA)

Memory

- 256 bytes internal data RAM
- 8, 4 or 2 kB one time programmable code memory

On-Chip Debug

- C8051F300 can be used as in-system code development platform; complete development kit available
- On-chip debug circuitry facilitates full speed, non-intrusive in-system debug

Supply Voltage 1.8 to 3.6 V

- On-chip LDO regulator for core supply
- Typical operating current: 5.0 mA @ 25 MHz
- Typical stop mode current (regulator off): <0.1 µA
- Built-in brown-out detector

High Speed 8051 µC Core

- Pipe-lined instruction architecture; executes 70% of instructions in 1 or 2 system clocks
- 25 MIPS peak throughput with 25 MHz clock
- Expanded interrupt handler

Digital Peripherals

- 8 port I/O; All 5 V tolerant with high sink current
- Hardware enhanced UART and SMBus™ serial ports
- Three general purpose 16-bit counter/timers
- 16-Bit programmable counter array (PCA) with three capture/compare modules
  - 8 or 16-bit PWM
  - Rising / falling edge capture
  - Frequency output
  - Software timer

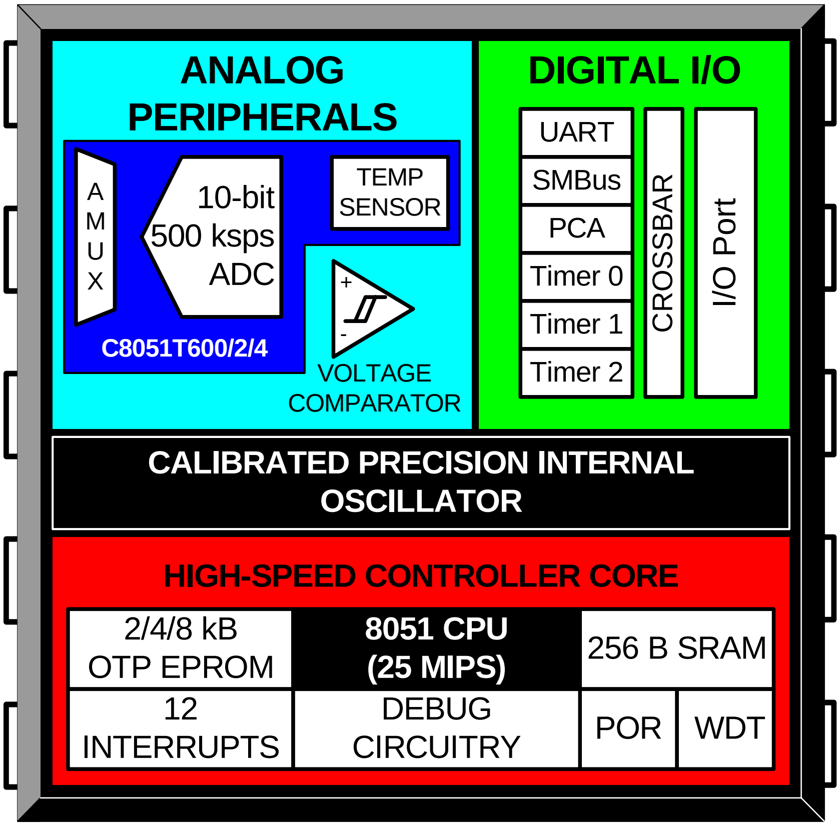
Clock Sources

- Internal oscillator: 24.5 MHz with ±2% accuracy supports UART operation
- External oscillator: RC, Single Capacitor, or CMOS Clock Modes
- Can switch between clock sources on-the-fly; useful in power saving modes

Temperature Range: –40 to +85 °C

11-Pin QFN or 14-Pin SOIC Package

- QFN Size = 3 x 3 mm



This information applies to a product under development. Its characteristics and specifications are subject to change without notice.

Table of Contents

NOTES:

List of Figures

1. **System Overview**

2. **Absolute Maximum Ratings**
3. **Global Electrical Characteristics**
4. **Pinout and Package Definitions**

5. **ADC0 - 10-Bit SAR ADC (C8051T600/2/4 Only)**

6. **Voltage Reference Options**

7. **Comparator0**

8. **Voltage Regulator (REG0)**
9. **CIP-51 Microcontroller**

10. **Reset Sources**

11. **One-Time Programmable Read-Only Memory**

12. **Oscillators**

13. **Port Input/Output**

14. SMBus

15. UART0

16. Timers

17. Programmable Counter Array

18. Revision Specific Behavior

19. C2 Interface

List of Tables

1. System Overview

2. Absolute Maximum Ratings

3. Global Electrical Characteristics

4. Pinout and Package Definitions

5. ADC0 - 10-Bit SAR ADC (C8051T600/2/4 Only)

6. Voltage Reference Options

7. Comparator0

8. Voltage Regulator (REG0)

9. CIP-51 Microcontroller

10. Reset Sources

11. One-Time Programmable Read-Only Memory

12. Oscillators

13. Port Input/Output

14. SMBus

15. UART0

16. Timers

17. Programmable Counter Array

18. Revision Specific Behavior

19. C2 Interface

List of Registers

1. System Overview

C8051T600/1/2/3/4/5 devices are fully integrated mixed-signal System-on-a-Chip MCUs. Highlighted features are listed below. Refer to Table 1.1 on page 14 for specific product feature selection.

- High-speed pipelined 8051-compatible microcontroller core (up to 25 MIPS)
- In-system, full-speed, non-intrusive debug interface
- C8051F300 ISP Flash device is available for quick in-system code development
- 10-bit 500 ksps ADC with programmable analog multiplexer and integrated temperature sensor
- Precision calibrated 24.5 MHz internal oscillator
- 2/4/8 kB of on-chip One-Time Programmable (OTP) EPROM
- 256 bytes of on-chip RAM
- SMBus/I^{2}C and Enhanced UART serial interfaces implemented in hardware
- Three general-purpose 16-bit timers
- Programmable Counter/Timer Array (PCA) with three capture/compare modules and Watchdog Timer function
- On-chip Power-On Reset and Supply Monitor
- On-chip Voltage Comparator
- Byte-wide I/O Port (5 V tolerant)

With on-chip Power-On Reset, Supply Monitor, Watchdog Timer, and clock oscillator, the C8051T600/1/2/3/4/5 devices are truly stand-alone System-on-a-Chip solutions. User software has complete control of all peripherals, and may individually shut down any or all peripherals for power savings.

Code written for the C8051T600/1/2/3/4/5 family of processors will run on the C8051F300 Mixed-signal ISP Flash microcontroller, providing a quick, cost-effective way to develop code without requiring special emulator circuitry. The C8051T600/1/2/3/4/5 processors include Silicon Laboratories' 2-Wire C2 Debug and Programming interface, which allows non-intrusive (uses no on-chip resources), full speed, in-circuit debugging using the production MCU installed in the final application. This debug logic supports inspection of memory, viewing and modification of special function registers, setting breakpoints, single stepping, run and halt commands. All analog and digital peripherals are fully functional while debugging using C2. The two C2 interface pins can be shared with user functions, allowing in-system debugging without occupying package pins.

Each device is specified for 1.8-to-3.6 V operation over the industrial temperature range (–45 to +85 °C) . An internal LDO is used to supply the processor core voltage at 1.8 V. The Port I/O and $\overline{\text{RST}}$ pins are tolerant of input signals up to 5 V. The C8051T600/1/2/3/4/5 are available in 3 x 3 mm 11-pin QFN or 14-pin SOIC packaging.

Table 1.1. Product Selection Guide

Part Number	MIPS (Peak)	OTP EPROM (Bytes)	RAM (Bytes)	Calibrated Internal Oscillator	SMBus/I^{2}C	UART	Timers (16-bit)	Programmable Counter Array	Digital Port I/Os	10-bit 500ksps ADC	Temperature Sensor	Analog Comparators	Lead-Free (ROHS Compliant)	Package
C8051T600-GM	25	8k*	256	✓	✓	✓	3	✓	8	✓	✓	1	✓	QFN-11
C8051T600-GS	25	8k*	256	✓	✓	✓	3	✓	8	✓	✓	1	✓	SOIC-14
C8051T601-GM	25	8k*	256	✓	✓	✓	3	✓	8	—	—	1	✓	QFN-11
C8051T601-GS	25	8k*	256	✓	✓	✓	3	✓	8	—	—	1	✓	SOIC-14
C8051T602-GM	25	4k	256	✓	✓	✓	3	✓	8	✓	✓	1	✓	QFN-11
C8051T602-GS	25	4k	256	✓	✓	✓	3	✓	8	✓	✓	1	✓	SOIC-14
C8051T603-GM	25	4k	256	✓	✓	✓	3	✓	8	—	—	1	✓	QFN-11
C8051T603-GS	25	4k	256	✓	✓	✓	3	✓	8	—	—	1	✓	SOIC-14
C8051T604-GM	25	2k	256	✓	✓	✓	3	✓	8	✓	✓	1	✓	QFN-11
C8051T604-GS	25	2k	256	✓	✓	✓	3	✓	8	✓	✓	1	✓	SOIC-14
C8051T605-GM	25	2k	256	✓	✓	✓	3	✓	8	-	-	1	✓	QFN-11
C8051T605-GS	25	2k	256	✓	✓	✓	3	✓	8	-	-	1	✓	SOIC-14

*512 Bytes Reserved for Factory Use

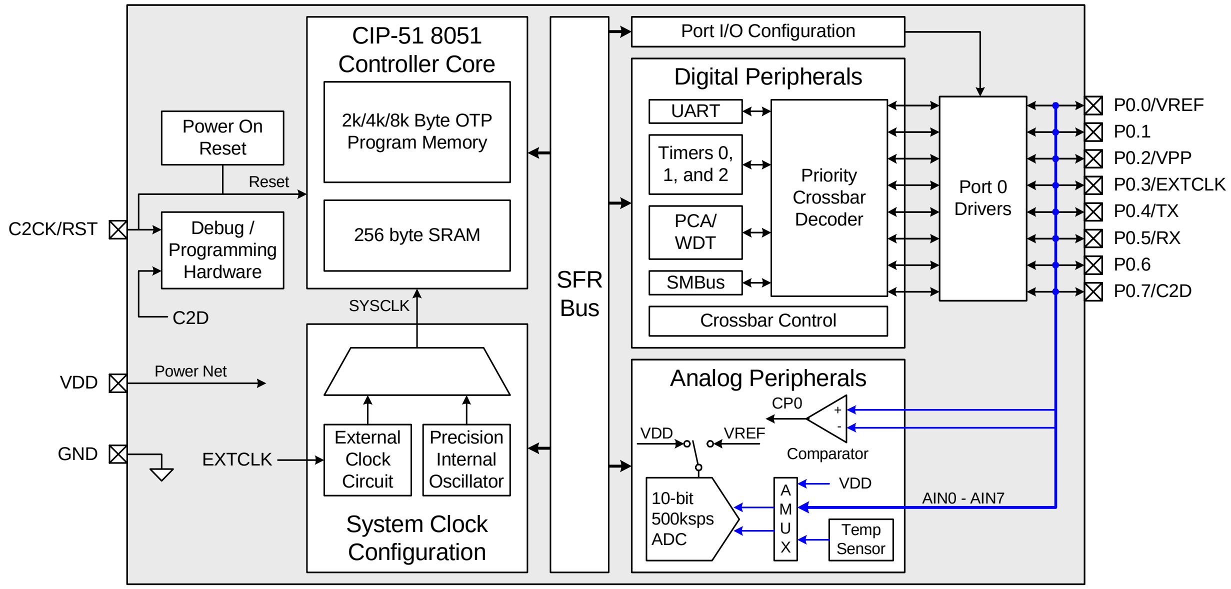


Figure 1.1. C8051T600/2/4 Block Diagram

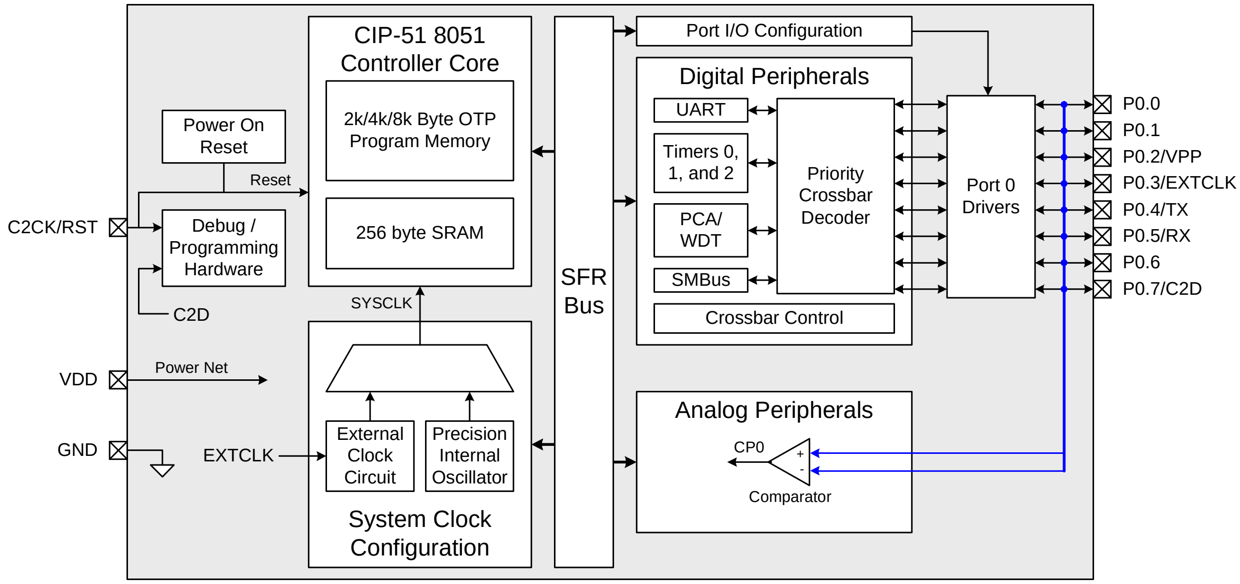


Figure 1.2. C8051T601/3/5 Block Diagram

1.1. CIP-51™ Microcontroller Core

1.1.1. Fully 8051 Compatible

The C8051T600/1/2/3/4/5 family utilizes 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 CIP-51 core offers all the peripherals included with a standard 8052, including two standard 16-bit counter/timers, one enhanced 16-bit counter/timer with external clock input, a full-duplex UART with extended baud rate configuration, 256 bytes of internal RAM, 128 byte Special Function Register (SFR) address space, and a byte-wide I/O Port.

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 with 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 only four instructions taking more than four 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

1.1.3. Additional Features

The C8051T600/1/2/3/4/5 SoC family includes several key enhancements to the CIP-51 core and peripherals to improve performance and ease of use in end applications.

The extended interrupt handler provides 12 interrupt sources into the CIP-51, allowing numerous analog and digital peripherals to interrupt the controller. An interrupt driven system requires less intervention by the MCU, giving it more effective throughput. The extra interrupt sources are very useful when building multi-tasking, real-time systems.

Several reset sources are available: power-on reset circuitry (POR), a Supply 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 OTP read/write detection. Each reset source except for the POR, Reset Input Pin, and OTP protection 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, and is accurate to ±2% over the entire operating supply and temperature range. The internal oscillator can be directly divided by factors of 2, 4, or 8 to provide slower internal clock options. An external oscillator input is also included, allowing an external CMOS clock, external capacitor, or external RC circuit to generate the system clock. If desired, the system clock source may be switched on-the-fly to the external oscillator circuit. An external oscillator can be extremely useful in low power applications, allowing the MCU to run from a slower (power saving) external clock source, while periodically switching to the internal oscillator when faster operation is required.

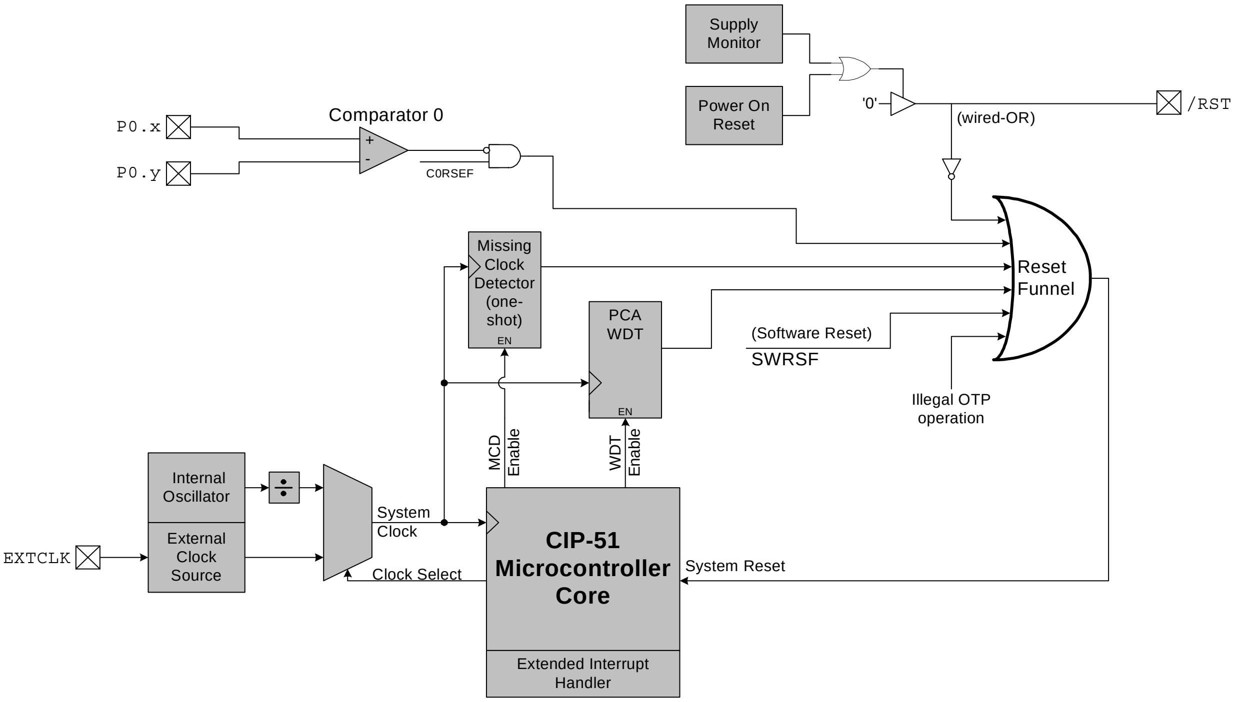


Figure 1.3. On-Chip Clock and Reset

1.2. On-Chip Memory

The CIP-51 has a standard 8051 program and data address configuration. It includes 256 bytes of data RAM, with the upper 128 bytes dual-mapped. Indirect addressing accesses the upper 128 bytes of general purpose RAM, and direct addressing accesses the 128 byte SFR address space. The lower 128 bytes of RAM are accessible via direct and indirect addressing. The first 32 bytes are addressable as four banks of general purpose registers, and the next 16 bytes can be byte addressable or bit addressable.

The C8051T600/1 include 8 kB of EPROM program memory, the C8051T602/3 include 4 kB, and the C8051T604/5 include 2 kB. See Figure 1.4 for the C8051T600/1 system memory map.

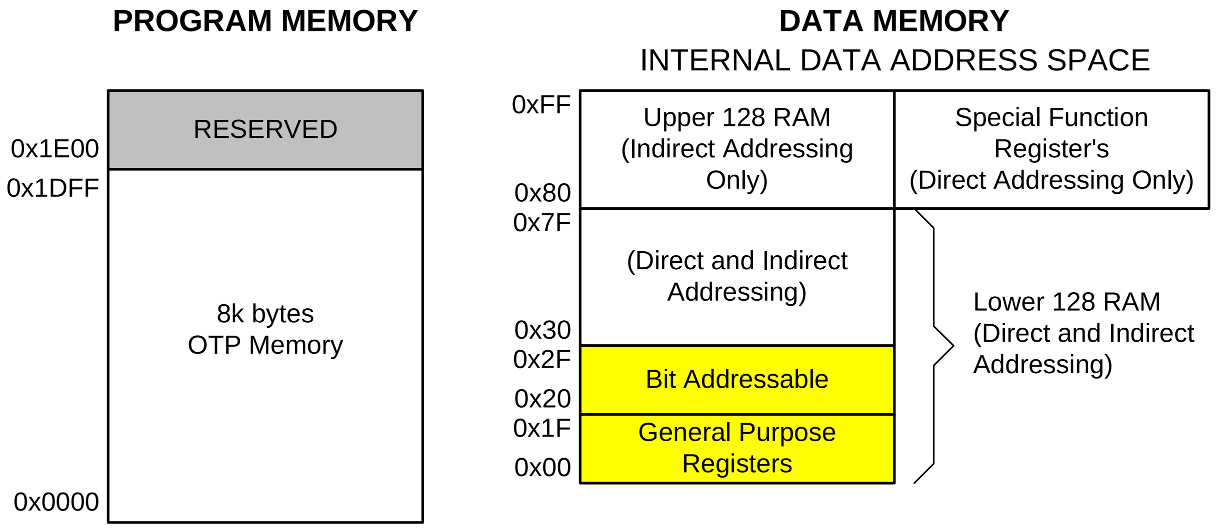


Figure 1.4. On-chip Memory Map (C8051T600/1 shown)

1.3. On-Chip Debug Circuitry and Code Development Options

The C8051T600/1/2/3/4/5 devices include on-chip Silicon Labs 2-Wire (C2) debug circuitry that provides non-intrusive, full speed, in-circuit debugging of the production part *installed in the end application.*

Silicon Labs' debugging system supports inspection and modification of memory and registers, breakpoints, and single stepping. No additional target RAM, program memory, timers, or communications channels are required. All the digital and analog peripherals are functional and work correctly while debugging. All the peripherals (except for the ADC and SMBus) are stalled when the MCU is halted, during single stepping, or at a breakpoint in order to keep them synchronized.

The C8051F300 MCU can be used to quickly develop code for a system using a device in the C8051T600/1/2/3/4/5 family. The C8051F300 is an In-System Programmable, Flash-based device that uses the same pinout as the C8051T600/1/2/3/4/5 devices, and can run code written for the C8051T600/1/2/3/4/5. The C8051T600DK development kit provides all the hardware and software necessary to develop application code and perform in-circuit debugging for the C8051T600/1/2/3/4/5 MCUs. The kit includes software with a developer's studio and debugger, an assembler/linker and evaluation 'C' compiler, and the necessary cables for connection to the target board or the end-system. The development kit includes an SOIC Socket Daughter Card for programming SOIC devices, samples of the C8051T600-GS, and a C8051F300 Emulation Daughter Card for rapid code development. An AC to DC wall adapter is supplied for powering the board.

1.4. Programmable Digital I/O and Crossbar

C8051T600/1/2/3/4/5 devices include a byte-wide I/O Port that behaves like a typical 8051 Port with a few enhancements. Each Port pin may be configured as an analog input or a digital I/O pin. Pins selected as digital I/Os may additionally be configured for push-pull or open-drain output. The “weak pullups” that are fixed on typical 8051 devices may be globally disabled, providing additional power savings.

Perhaps the most unique Port I/O enhancement is the Digital Crossbar. This is a digital switching network that allows mapping of internal digital system resources to Port I/O pins (See Figure 1.5). On-chip counter/timers, serial buses, HW interrupts, comparator output, and other digital signals in the controller can be configured to appear on the Port I/O pins using the Crossbar Control registers. This allows the user to select the exact mix of general purpose Port I/O and digital resources needed for the application.

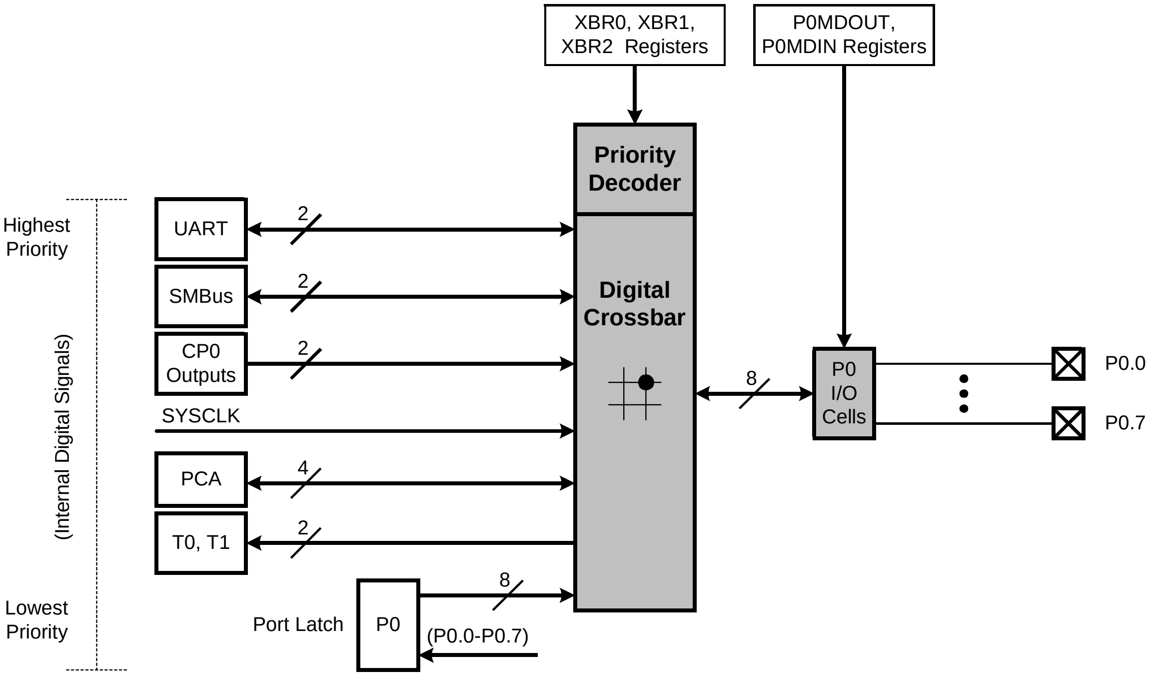


Figure 1.5. Digital Crossbar Diagram

1.5. Serial Ports

The C8051T600/1/2/3/4/5 Family includes an SMBus/I^{2}C interface and a full-duplex UART with enhanced baud rate configuration. 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. Both serial buses can be used at the same time.

1.6. Programmable Counter Array

An on-chip Programmable Counter/Timer Array (PCA) is included in addition to the three 16-bit general purpose counter/timers. The PCA consists of a dedicated 16-bit counter/timer time base with three 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 2 offers watchdog timer (WDT) capabilities. Following a system reset, Module 2 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.

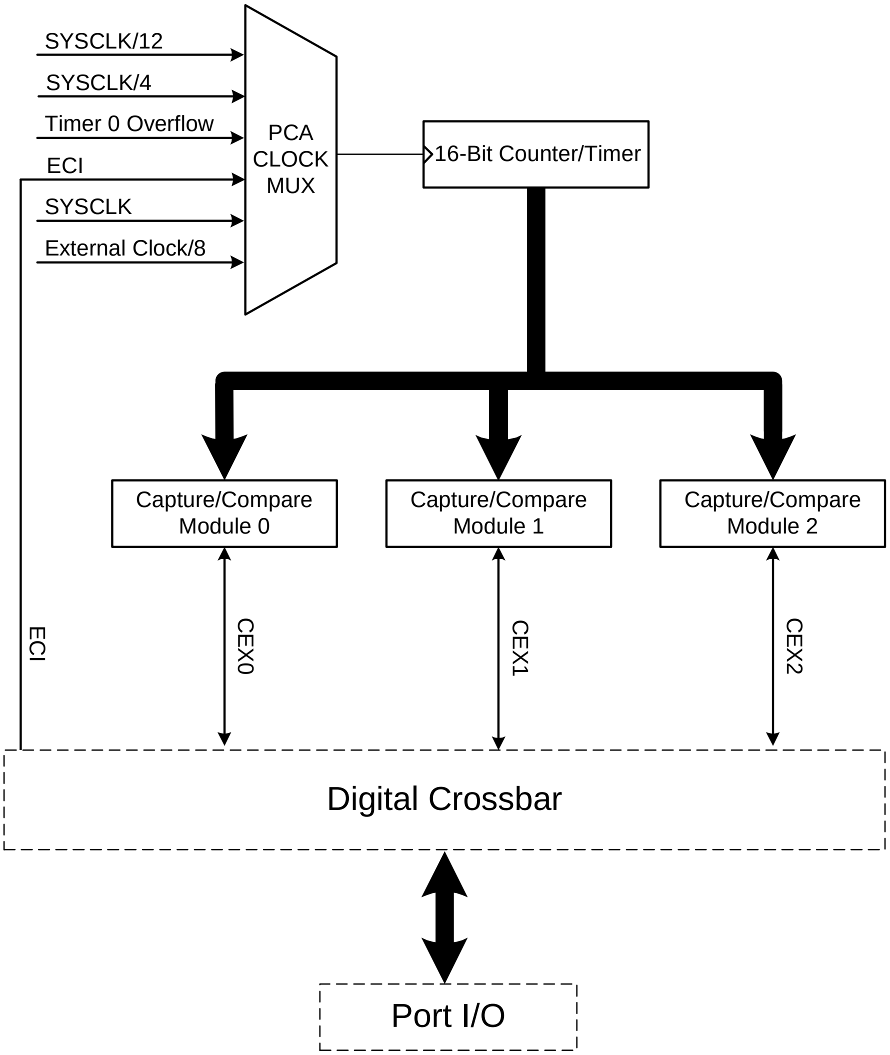


Figure 1.7. PCA Block Diagram

1.7. 10-Bit Analog to Digital Converter (C8051T600/2/4)

The C8051T600/2/4 include an on-chip 10-bit SAR ADC with a 10-channel input multiplexer. With a maximum throughput of 500 ksps, the ADC offers true 10-bit accuracy with an INL of ±1LSB. The ADC system includes a configurable analog multiplexer that selects the ADC input, and gain settings of 1x or 0.5x. Each Port pin is available as an ADC input; additionally, the on-chip Temperature Sensor output and the power supply voltage (V_{DD}) are available as ADC inputs. User firmware may shut down the ADC when it is not in use to save power.

Conversions can be started in five ways: a software command, an overflow of Timer 0, 1, or 2, or an external convert start signal. This flexibility allows the start of conversion to be triggered by software events, a periodic signal (timer overflows), or external HW signals. Conversion completions are indicated by a status bit and an interrupt (if enabled). The resulting 10-bit data word is latched into two SFRs upon completion of a conversion.

An 8-bit compatibility mode is included in the device to facilitate development and backwards-compatibility with the C8051F300.

Window compare registers for the ADC data can be configured to interrupt the controller when ADC data is either within or outside of a specified range. The ADC can monitor a key voltage continuously in background mode, but not interrupt the controller unless the converted data is within or outside of the specified range.

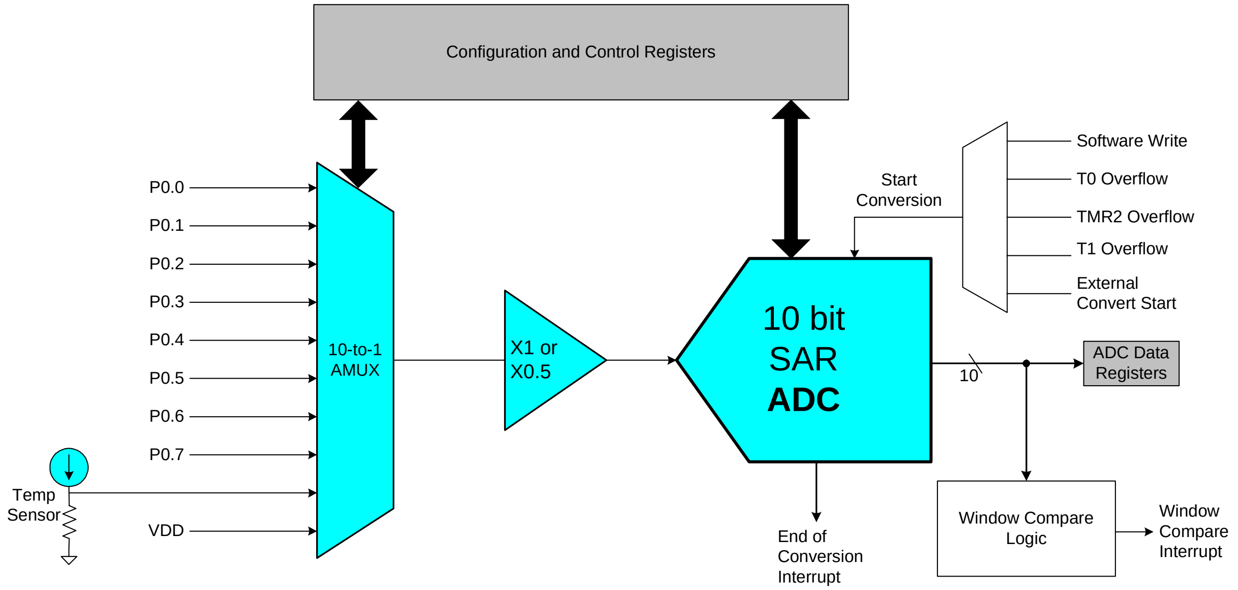


Figure 1.8. 10-Bit ADC Block Diagram

1.8. Comparator

C8051T600/1/2/3/4/5 devices include an on-chip voltage comparator that is enabled/disabled and configured via user software. All Port I/O pins may be configured as comparator inputs. Two comparator outputs may be routed to a Port pin if desired: a latched output and/or an unlatched (asynchronous) output. Comparator response time is programmable, allowing the user to select between high-speed and low-power modes. Positive and negative hysteresis is also configurable.

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. The comparator may also be configured as a reset source.

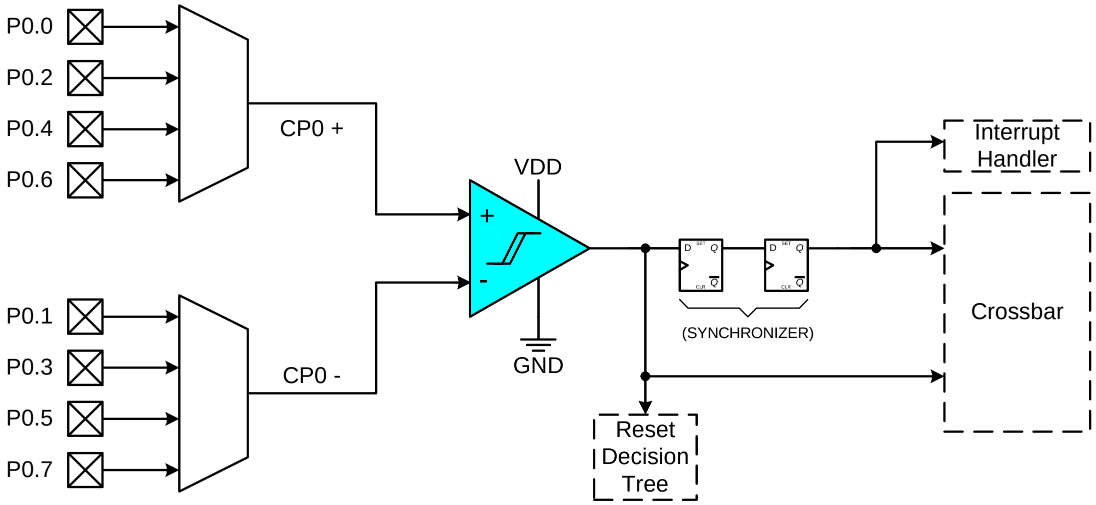


Figure 1.9. Comparator Block Diagram

2. Absolute Maximum Ratings

Table 2.1. Absolute Maximum Ratings*

Parameter	Conditions	Min	Typ	Max	Units
Ambient temperature under bias		–55	—	125	°C
Storage Temperature		–65	—	150	℃
Voltage on $\overline{RST}$ or any Port I/O Pin (except V_{PP} during programming) with respect to GND	$V_{DD} \geq 2.2$ V $V_{DD} < 2.2$ V	–0.3 –0.3	— —	5.8 $V_{DD} + 3.6$	V V
Voltage on V_{PP} with respect to GND during a programming operation	$V_{DD} \geq 2.4$ V	–0.3	—	7.0	V
Duration of High-voltage on V_{PP} pin (cumulative)	$V_{PP} > (V_{DD} + 3.6\text{ V})$			10	s
Voltage on V_{DD} with respect to GND	Regulator in Normal Mode Regulator in Bypass Mode	–0.3 –0.3	— —	4.2 1.98	V V
Maximum Total current through V_{DD} and GND		—	—	500	mA
Maximum output current sunk by $\overline{RST}$ or any Port pin		—	—	100	mA

***Note:** Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the devices at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

3. Global Electrical Characteristics

Table 3.1. Global DC Electrical Characteristics

–40 to +85 °C, 25 MHz System Clock unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Supply Voltage[1]	Regulator in Normal Mode Regulator in Bypass Mode	1.8 1.7	3.0 1.8	3.6 1.9	V V
Digital Supply Current with CPU Active	V_{DD} = 1.8 V, Clock = 25 MHz V_{DD} = 1.8 V, Clock = 1 MHz[2] V_{DD} = 3.0 V, Clock = 25 MHz V_{DD} = 3.0 V, Clock = 1 MHz[2]	—	4.3 TBD 5.0 TBD	—	mA mA mA mA
Digital Supply Current with CPU Inactive (not accessing EPROM)	V_{DD} = 1.8 V, Clock = 25 MHz[2] V_{DD} = 1.8 V, Clock = 1 MHz[2] V_{DD} = 3.0 V, Clock = 25 MHz[2] V_{DD} = 3.0 V, Clock = 1 MHz[2]	—	TBD TBD TBD TBD	—	mA mA mA mA
Digital Supply Current (shutdown)	Oscillator not running, Internal Regulator Off	—	TBD	—	µA
Digital Supply RAM Data Retention Voltage		—	TBD	—	V
Specified Operating Temperature Range		–40	—	+85	°C
SYSCLK (system clock frequency)[3]		0	—	25	MHz
SYSCLK Duty Cycle		40	—	60	%

Notes:

1. Analog performance is degraded when V_{DD} is below 1.8 V.
2. Specifications below 2 MHz or with CPU Inactive assume memory power controller is enabled.
3. SYSCLK must be at least 32 kHz to enable debugging.

Other electrical characteristics tables are found in the data sheet section corresponding to the associated peripherals. For more information on electrical characteristics for a specific peripheral, refer to the page indicated in Table 3.2.

Table 3.2. Index to Electrical Characteristics Tables

4. Pinout and Package Definitions

Table 4.1. Pin Definitions for the C8051T600/1/2/3/4/5

Pin Number QFN-11	Pin Number SOIC-14	Name	Type	Description
1	5	VREF / P0.0	A In D I/O or A In	External Voltage Reference Input. Port 0.0. See **Section 13** for complete description.
2	6	P0.1	D I/O or A In	Port 0.1. See **Section 13** for complete description.
3	7	V_{DD}		Power Supply Voltage.
4	8	V_{PP} / P0.2	A In D I/O or A In	V_{PP} Programming Supply Voltage Port 0.2. See **Section 13** for complete description.
5	10	EXTCLK/ P0.3	D I/O or A I/O D I/O or A In	This pin can be used as the external clock input for CMOS, RC, or Capacitor configurations. See **Section 12.2**. Port 0.3. See **Section 13** for complete description.
6	12	P0.4	D I/O or A In	Port 0.4. See **Section 13** for complete description.
7	13	P0.5	D I/O or A In	Port 0.5. See **Section 13** for complete description.
8	14	C2CK / $\overline{\text{RST}}$	D I/O D I/O	Clock signal for the C2 Development Interface. Device Reset. Open-drain output of internal POR or V_{DD} monitor. An external source can initiate a system reset by driving this pin low for at least 10 µs.
9	1	P0.6	D I/O or A In	Port 0.6. See **Section 13** for complete description.
10	2	C2D / P0.7	D I/O D I/O or A In	Data signal for the C2 Development Interface. Port 0.7. See **Section 13** for complete description.
11	3	GND		Ground.
—	4, 9, 11	NC		No Connection

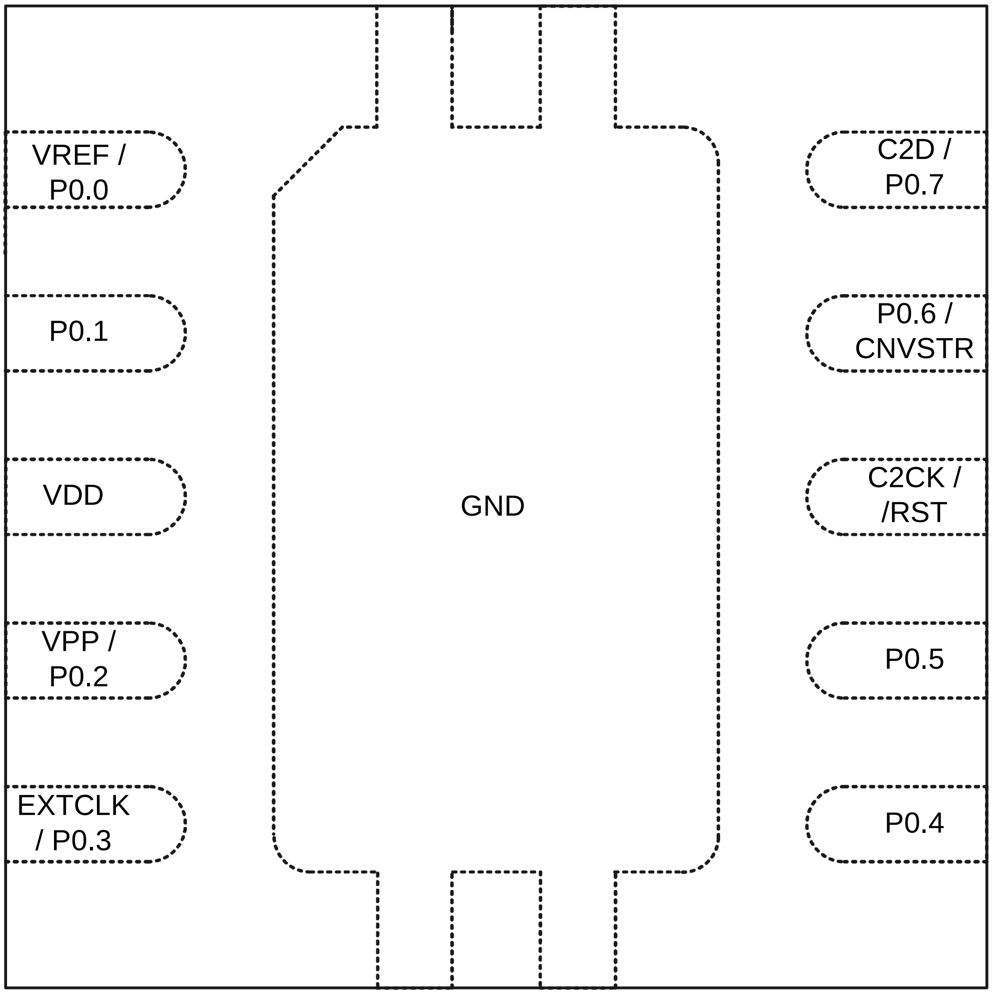


Figure 4.1. QFN-11 Pinout Diagram (Top View)

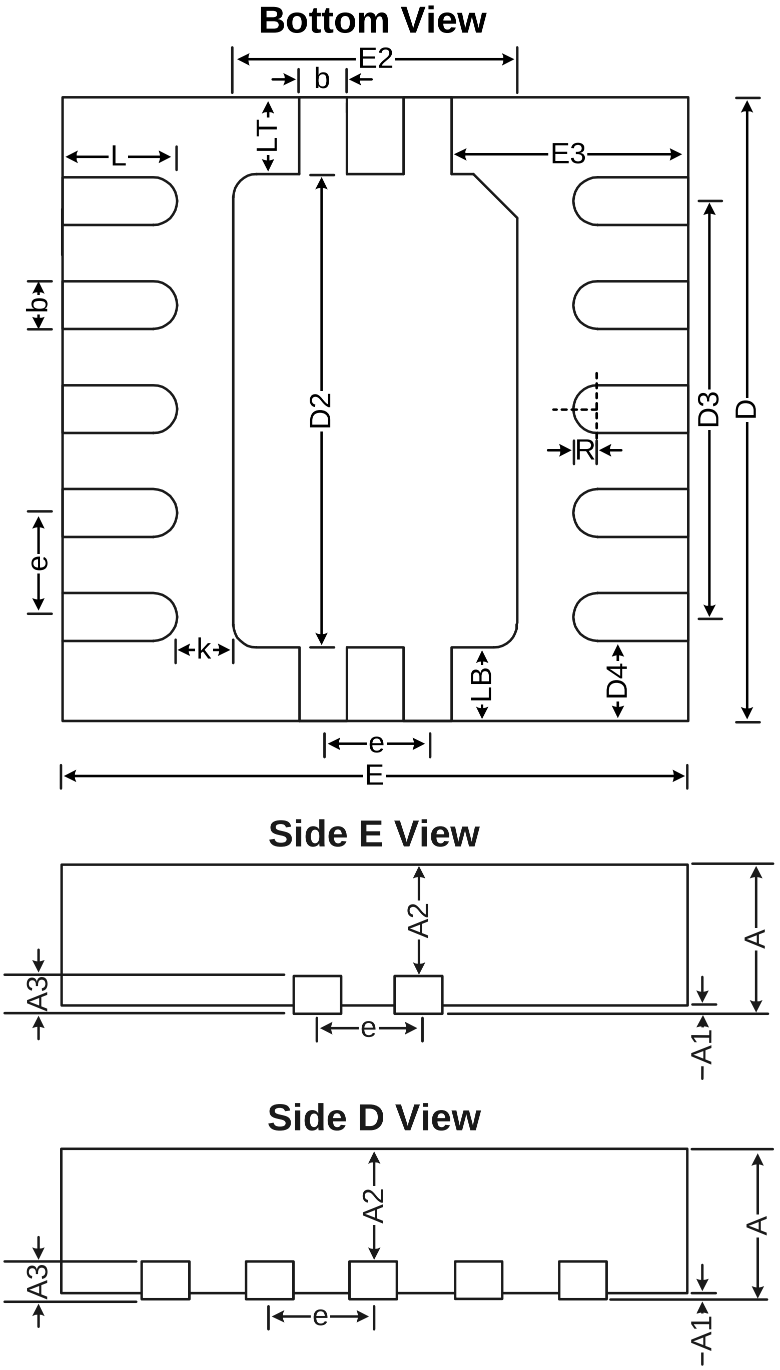


Figure 4.2. QFN-11 Package Drawing

Table 4.2. QFN-11 Package Dimensions

	MM		
	Min	**Typ**	**Max**
A	0.80	0.90	1.00
A1	0	0.02	0.05
A2	0	0.65	1.00
A3	—	0.25	—
b	0.18	0.23	0.30
D	—	3.00	—
D2	—	2.20	2.25
D3	—	2.00	—
D4	—	0.386	—
E	—	3.00	—
E2	—	1.36	—
E3	—	1.135	—
e	—	0.5	—
k	—	0.27	—
L	0.45	0.55	0.65
LB	—	0.36	—
LT	—	0.37	—
R	0.09	—	—

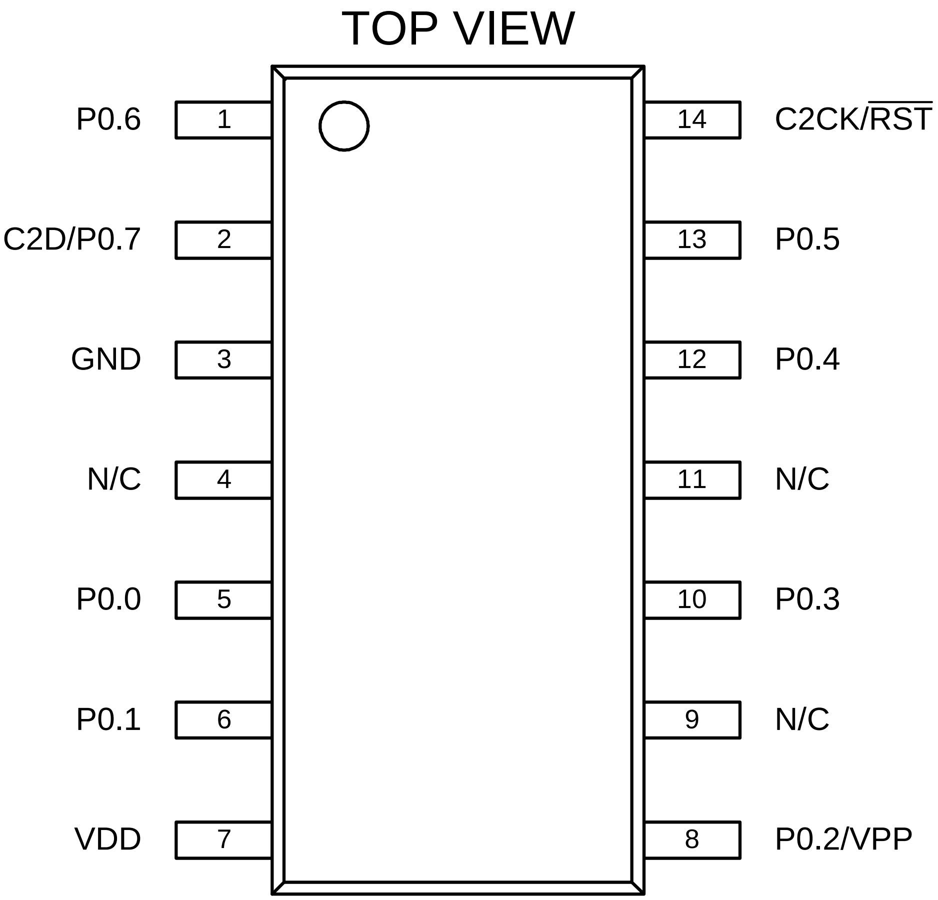


Figure 4.3. SOIC-14 Pinout Diagram (Top View)

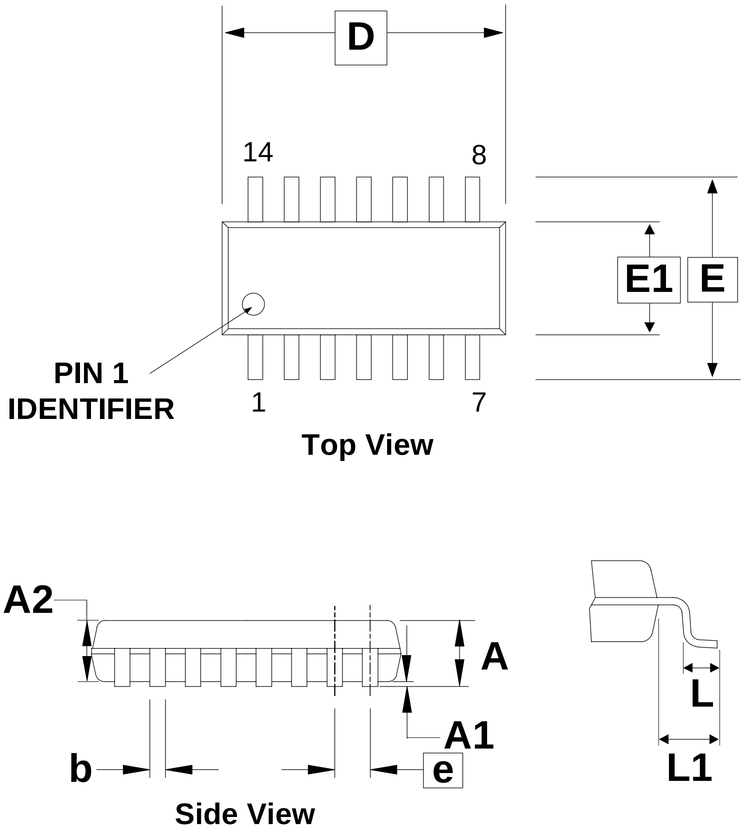


Table 4.3. SOIC-14 Package Dimensions

	MM		
	Min	Typ	Max
A	1.35	—	1.75
A1	0.10	—	0.25
A2	1.25	—	1.65
b	0.31	—	0.51
D	8.65 BSC		
E	6.00 BSC		
E1	3.90 BSC		
e	—	1.27	—
L	0.40	—	1.27
L1	1.04 REF		

Figure 4.4. SOIC-14 Package Drawing

5. ADC0 - 10-Bit SAR ADC (C8051T600/2/4 Only)

The ADC0 subsystem for the C8051T600/2/4 devices consists an analog multiplexer (referred to as AMUX0) with 10 input selection options, a gain stage programmable to 1x or 0.5x, and a 500 ksps, 10-bit successive-approximation-register (SAR) ADC with integrated track-and-hold and programmable window detector (see block diagram in Figure 5.1). The multiplexer, data conversion modes and window detector are all configurable under software control via the Special Function Registers shown in Figure 5.1. ADC0 may be configured to measure any Port pin, the Temperature Sensor output, or VDD with respect to GND. The ADC0 subsystem is enabled only when the AD0EN bit in the ADC0 Control register (ADC0CN) is set to logic 1. The ADC0 subsystem remains in a low power shutdown state when this bit is logic 0. A special 8-bit mode is also provided for backwards-compatibility with the C8051F300 development platform.

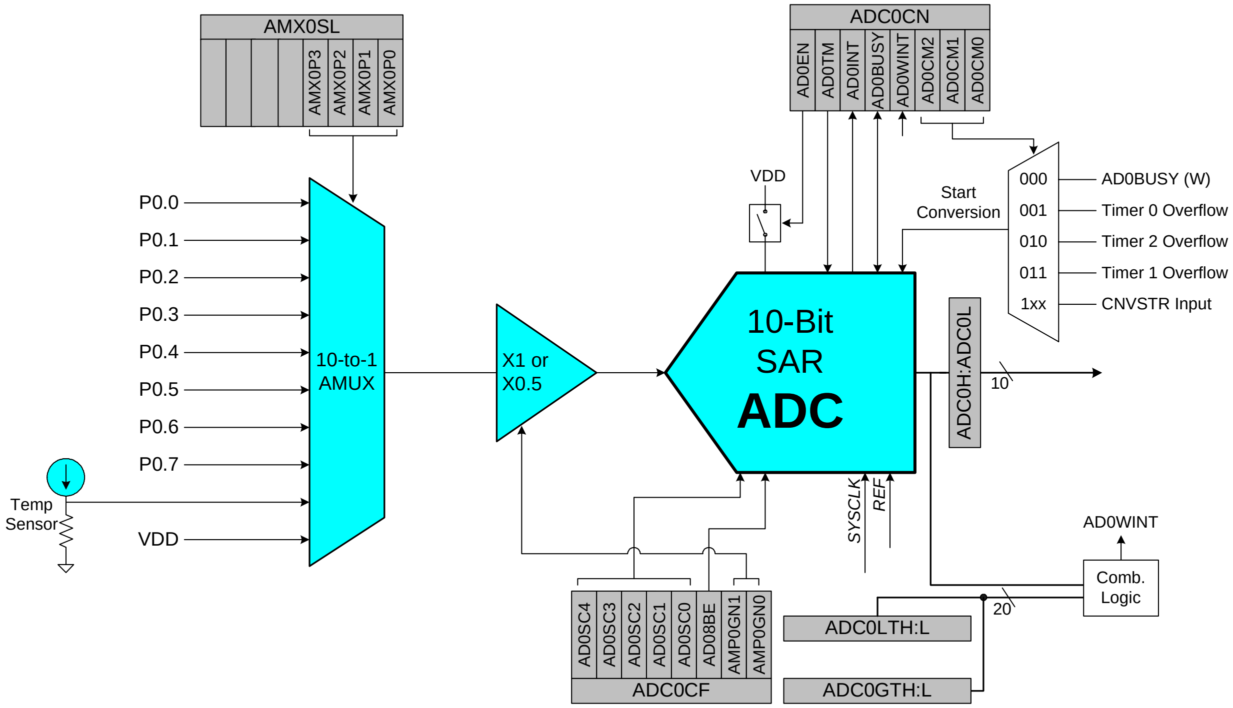


Figure 5.1. ADC0 Functional Block Diagram

5.1. Analog Multiplexer

The analog multiplexer (AMUX0) selects the positive input to the ADC, allowing any Port pin to be measured relative to GND. Additionally, the on-chip temperature sensor or the positive power supply (V_{DD}) may be selected as the positive ADC input. The ADC0 input channel is selected in the AMX0SL register as described in Figure 5.3. When an external Voltage Reference is supplied to P0.0 or the internal regulator is used as VREF, the V_{DD} Voltage supply can be determined by taking a measurement of V_{DD} with the gain setting at 0.5x.

Important Note About ADC0 Input Configuration: Port pins selected as ADC0 inputs should be configured as analog inputs and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, clear the corresponding bit in register P0MDIN to ‘0’. To force the Crossbar to skip a Port pin, set the corresponding bit in register XBR0 to ‘1’. See **Section “13. Port Input/Output” on page 97** for more Port I/O configuration details.

5.2. Gain Setting

The ADC has gain settings of 1x and 0.5x. In 1x mode, the full scale reading of the ADC is determined directly by V_{REF}. In 0.5x mode, the full-scale reading of the ADC occurs when the input voltage is V_{REF} x 2. The 0.5x gain setting can be useful to obtain a higher input Voltage range when using a small V_{REF} voltage, or to measure input voltages that are between V_{REF} and V_{DD}. Gain settings for the ADC are controlled by the AMP0GN1–0 bits in register ADC0CF.

5.3. Output Coding

The conversion code format for the ADC is shown below. Conversion codes are represented as 10-bit unsigned integers. Inputs are measured from ‘0’ to V_{REF} x 1023/1024. All conversions are left-justified in the ADC0H and ADC0L registers (ADC0H holds the 8 most significant bits, and the two least significant bits are stored in ADC0L). Example codes are shown below.

Input Voltage (AIN – GND), Gain = 1	10-bit Output (Conversion Code)	ADC0H:ADC0L Register Coding
V_{REF} x 1023/1024	0x3FF	0xFF : 0xC0
$V_{REF}/2$	0x200	0x80 : 0x00
$V_{REF}/4$	0x100	0x40 : 0x00
0	0x00	0x00 : 0x00

5.4. 8-Bit Compatibility Mode

Setting the ADC08BE bit in register ADC0CF to ‘1’ will put the ADC in 8-bit compatibility mode. This mode allows backward compatibility with the C8051F300 device family. In 8-bit compatibility mode, only the 8 MSBs of data are converted. The two LSBs of a conversion are always ‘00’ in this mode, and the ADC0L register will always read back 0x00. 8-bit conversions take two fewer SAR clock cycles than 10-bit conversions, so the conversion is completed faster, and a 500 ksps sampling rate can be achieved with a slower SAR clock.

5.5. Temperature Sensor

The temperature sensor transfer function is shown in Figure 5.2. The output voltage (V_{TEMP}) is the ADC input when the temperature sensor is selected by bits AMX0P2–0 in register AMX0SL. Values for the Offset and Slope parameters can be found in Table 5.1.

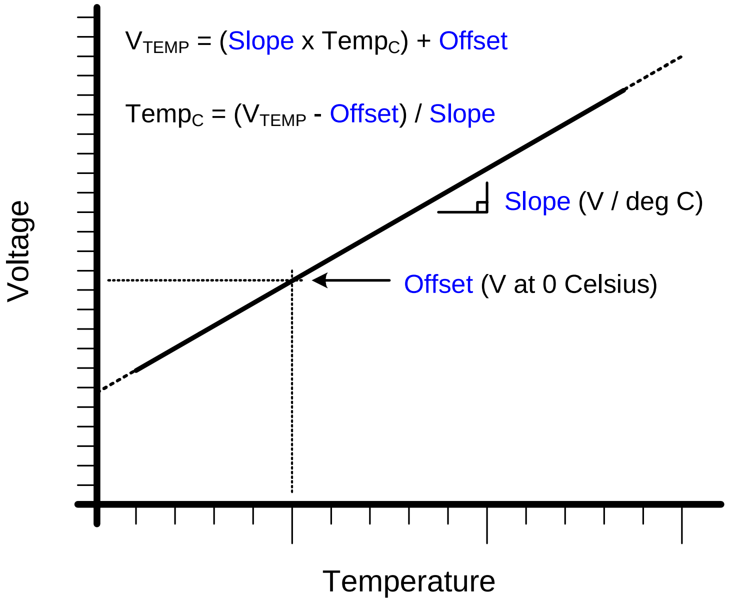


Figure 5.2. Temperature Sensor Transfer Function

5.5.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements (see Table 5.1 for linearity specifications). For absolute temperature measurements, offset and/or gain calibration is recommended. Typically a 1-point (offset) calibration includes the following steps:

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

Figure 5.3 shows the typical temperature sensor error assuming a 1-point calibration at 25 °C. **Parameters that affect ADC measurement, in particular the voltage reference value, will also affect temperature measurement.**

A single-point offset measurement of the temperature sensor is performed on each device during production test. The measurement is performed at 25 ℃ ±TBD ℃, using the ADC with the internal regulator selected as the Voltage Reference. The direct ADC result of the measurement is stored in the SFR registers TOFFH and TOFFL, shown in SFR Definition 5.1 and SFR Definition 5.2.

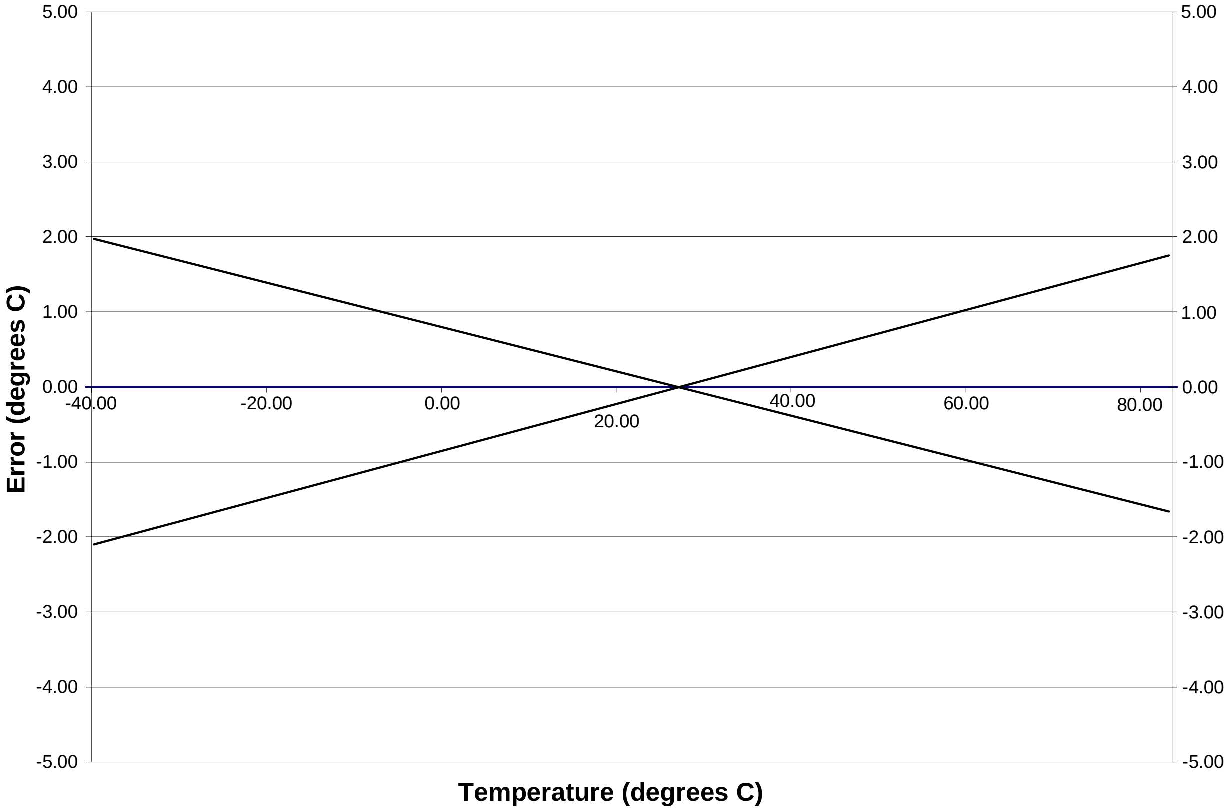


Figure 5.3. Temperature Sensor Error with 1-Point Calibration (V_{REF} = 2.4 V)

SFR Definition 5.1. TOFFH: Temperature Offset Measurement High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
TOFF9	TOFF8	TOFF7	TOFF6	TOFF5	TOFF4	TOFF3	TOFF2	Varies
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xA3

Bits7–0: Bits 9–2 of temperature sensor measurement.

The temperature sensor offset measurement is taken during production test of the device. The measurement is intended to be used as an offset correction for the temperature sensor. It is taken under the conditions V_{REF} = VREG; T_{AMB} = 25 ℃ ± TBD °C. One LSB of the temperature sensor offset measurement is equivalent to one LSB of the ADC output under the measurement conditions.

SFR Definition 5.2. TOFFL: Temperature Offset Measurement Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
TOFF1	TOFF0	—	—	—	—	—	—	Varies
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xA2

Bits7–6: Bits 1-0 of temperature sensor offset measurement.
Bits5–0: Read: 000000b, Write = Don't Care

The temperature sensor offset measurement is taken during production test of the device. The measurement is intended to be used as an offset correction for the temperature sensor. It is taken under the conditions V_{REF} = VREG; T_{AMB} = 25 °C ± TBD °C. One LSB of the temperature sensor offset measurement is equivalent to one LSB of the ADC output under the measurement conditions.

5.6. Modes of Operation

ADC0 has a maximum sampling rate of 500 ksps. The ADC0 SAR clock is a divided version of the system clock, determined by the AD0SC bits in the ADC0CF register (system clock divided by (AD0SC + 1) for $0 \leq \text{AD0SC} \leq 31$).

5.6.1. Starting a Conversion

A conversion can be initiated in one of five ways, depending on the programmed states of the ADC0 Start of Conversion Mode bits (AD0CM2–0) in register ADC0CN. Conversions may be initiated by one of the following:

1. Writing a ‘1’ to the AD0BUSY bit of register ADC0CN
2. A Timer 0 overflow (i.e., timed continuous conversions)
3. A Timer 2 overflow
4. A Timer 1 overflow
5. A rising edge on the CNVSTR input signal (pin P0.6)

Writing a ‘1’ to AD0BUSY provides software control of ADC0 whereby conversions are performed "on-demand". During conversion, the AD0BUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the ADC0 interrupt flag (AD0INT).

Note: When polling for ADC conversion completions, the ADC0 interrupt flag (AD0INT) should be used. Converted data is available in the ADC0 data register, ADC0, when bit AD0INT is logic 1. Note that when Timer 2 overflows are used as the conversion source, Timer 2 Low Byte overflows are used if Timer 2 is in 8-bit mode; Timer 2 High byte overflows are used if Timer 2 is in 16-bit mode. See **Section “16. Timers” on page 131** for timer configuration.

Important Note About Using CNVSTR: The CNVSTR input pin also functions as Port pin P0.6. When the CNVSTR input is used as the ADC0 conversion source, Port pin P0.6 should be skipped by the Digital Crossbar. To configure the Crossbar to skip P0.6, set to ‘1’ Bit6 in register XBR0. See **Section “13. Port Input/Output” on page 97** for details on Port I/O configuration.

5.6.2. Tracking Modes

The AD0TM bit in register ADC0CN enables "delayed conversions", and will delay the actual conversion start by three SAR clock cycles, during which time the ADC will continue to track the input. If AD0TM is left at logic 0, a conversion will begin immediately, without the extra tracking time. For internal start-of-conversion sources, the ADC will track anytime it is not performing a conversion. When the CNVSTR signal is used to initiate conversions, ADC0 will track either when AD0TM is logic 1, or when AD0TM is logic 0 and CNVSTR is held low. See Figure 5.4 for track and convert timing details. Delayed conversion mode is useful when AMUX settings are frequently changed, due to the settling time requirements described in **Section “5.6.3. Settling Time Requirements” on page 38**.

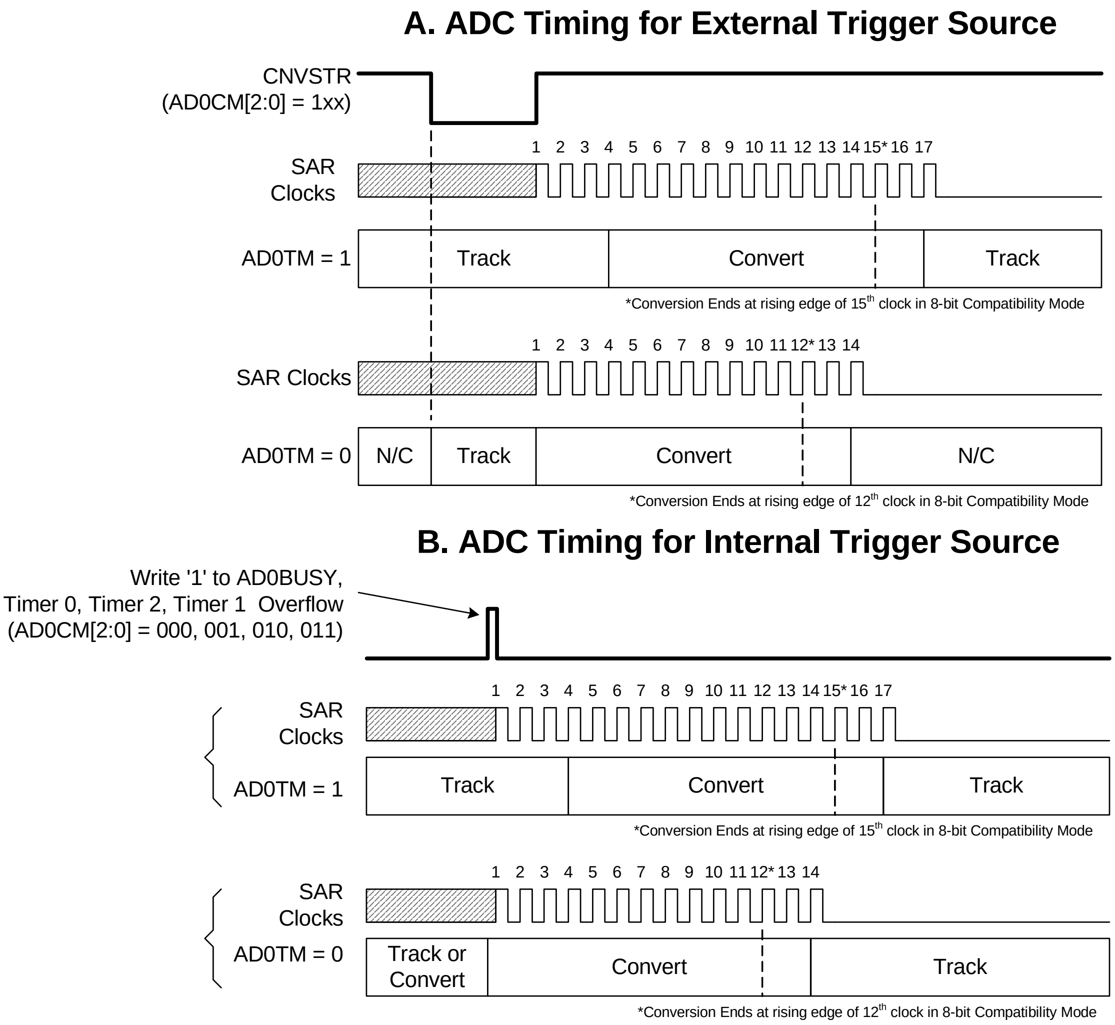


Figure 5.4. ADC Tracking and Conversion Timing

5.6.3. Settling Time Requirements

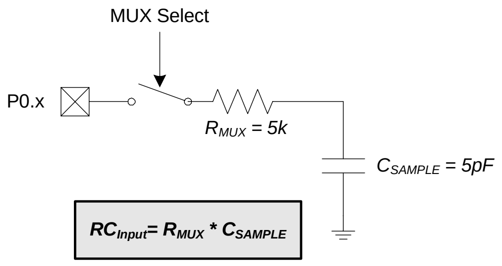
A minimum amount of tracking time is required before each conversion can be performed, to allow the sampling capacitor voltage to settle. This tracking time is determined by the AMUX0 resistance, the ADC0 sampling capacitance, any external source resistance, and the accuracy required for the conversion.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

Figure 5.5 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 5.1. When measuring the Temperature Sensor output or V_{DD} with respect to GND, R_{TOTAL} reduces to R_{MUX}. See Table 5.1 for ADC0 minimum settling time (track/hold time) requirements.

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

Equation 5.1. ADC0 Settling Time Requirements

Where:
SA is the settling accuracy, given as a fraction of an LSB (for example, 0.25 to settle within 1/4 LSB)
t is the required settling time in seconds
R_{TOTAL} is the sum of the AMUX0 resistance and any external source resistance.
n is the ADC resolution in bits (10).



Note: When the PGA gain is set to 0.5, C_{SAMPLE} = 3pF

Figure 5.5. ADC0 Equivalent Input Circuits

SFR Definition 5.3. AMX0SL: AMUX0 Channel Select

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	-	-	AMX0P3	AMX0P2	AMX0P1	AMX0P0	10000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xBB

Bits7–4: Unused.

Bits3–0: AMX0P3–0: AMUX0 Positive Input Selection.
0000–1001b: ADC0 Positive Input selected per the chart below.
1010–1111b: RESERVED.

AMX0P3–0	ADC0 Positive Input
0000	P0.0
0001	P0.1
0010	P0.2
0011	P0.3
0100	P0.4
0101	P0.5
0110	P0.6
0111	P0.7
1000	Temperature Sensor
1001	V_{DD}

SFR Definition 5.4. ADC0CF: ADC0 Configuration

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
AD0SC4	AD0SC3	AD0SC2	AD0SC1	AD0SC0	AD08BE	AMP0GN1	AMP0GN0	11111000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xBC

Bits7–3: AD0SC4–0: ADC0 SAR Conversion Clock Period Bits.
SAR Conversion clock is derived from system clock by the following equation, where *AD0SC* refers to the 5-bit value held in bits AD0SC4–0. SAR Conversion clock requirements are given in Table 5.1.
Note: if the OTP Power Controller is enabled (OTPPCE = '1'), AD0SC must be set to at least "00001" for proper ADC operation.

$$AD0SC = \frac{SYSCLK}{CLK_{SAR}} - 1$$

Bit2: AD08BE: 8-Bit Mode Enable.
0: ADC operates in 10-bit mode (normal).
1: ADC operates in 8-bit mode.

Bits1–0: AMP0GN1–0: ADC Gain Control Bits.
00: Gain = 0.5
01: Gain = 1

SFR Definition 5.5. ADC0H: ADC0 Data Word High Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
D9	D8	D7	D6	D5	D4	D3	D2	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xBE

Bits7–0: ADC0 Data Word bits 9–2.
ADC0H holds the upper 8 bits of output data from the most recently completed ADC0 conversion. In 8-bit compatibility mode, the ADC0H register holds all 8 bits of the conversion data word.

SFR Definition 5.6. ADC0L: ADC0 Data Word Low Byte

R/W	R/W	R	R	R	R	R	R	Reset Value
D1	D0	-	-	-	-	-	-	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xBD

Bits7–6: ADC0 Data Word bits 1–0.
Bits5–0: Read: 000000b, Write = Don’t Care
ADC0L holds the lowest 2 bits of output data from the most recently completed ADC0 conversion. In 8-bit compatibility mode, the ADC0L register always returns 0x00.

SFR Definition 5.7. ADC0CN: ADC0 Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
AD0EN	AD0TM	AD0INT	AD0BUSY	AD0WINT	AD0CM2	AD0CM1	AD0CM0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							Bit Addressable	0xE8

Bit7: AD0EN: ADC0 Enable Bit.
0: ADC0 Disabled. ADC0 is in low-power shutdown.
1: ADC0 Enabled. ADC0 is active and ready for data conversions.

Bit6: AD0TM: ADC0 Track Mode Bit.
0: Normal Track Mode: When ADC0 is enabled, a start-of-conversion signal begins the conversion. For internal conversion start sources, tracking is continuous whenever a conversion is not in progress. For external CNVSTR signal, tracking of the input occurs when CNVSTR is held low.
1: Delayed Track Mode: When ADC0 is enabled, input is tracked when a conversion is not in progress. A start-of-conversion signal initiates three SAR clocks of additional tracking, and then begins the conversion.

Bit5: AD0INT: ADC0 Conversion Complete Interrupt Flag.
0: ADC0 has not completed a data conversion since the last time AD0INT was cleared.
1: ADC0 has completed a data conversion.

Bit4: AD0BUSY: ADC0 Busy Bit.
Read: Unused.
Write:
0: No Effect.
1: Initiates ADC0 Conversion if AD0CM2–0 = 000b

Bit3: AD0WINT: ADC0 Window Compare Interrupt Flag.
0: ADC0 Window Comparison Data match has not occurred since this flag was last cleared.
1: ADC0 Window Comparison Data match has occurred.

Bits2–0: AD0CM2–0: ADC0 Start of Conversion Mode Select.
000: ADC0 conversion initiated on every write of ‘1’ to AD0BUSY.
001: ADC0 conversion initiated on overflow of Timer 0.
010: ADC0 conversion initiated on overflow of Timer 2.
011: ADC0 conversion initiated on overflow of Timer 1.
1xx: ADC0 conversion initiated on rising edge of external CNVSTR.
Note: Start of conversion is delayed by three SAR clock cycles when AD0TM = 1.

5.7. Programmable Window Detector

The ADC Programmable Window Detector continuously compares the ADC0 output to user-programmed limits, and notifies the system when a desired condition is detected. This is especially effective in an interrupt-driven system, saving code space and CPU bandwidth while delivering faster system response times. The window detector interrupt flag (AD0WINT in register ADC0CN) can also be used in polled mode. The ADC0 Greater-Than (ADC0GTH, ADC0GTL) and Less-Than (ADC0LTH, ADC0LTL) registers hold the comparison values. Example comparisons are shown in Figure 5.6. Notice that the window detector flag can be programmed to indicate when measured data is inside or outside of the user-programmed limits depending on the contents of the ADC0LTH:L and ADC0GTH:L registers.

5.7.1. Window Detector Example

Figure 5.6 shows two example window comparisons, using the ADC in 10-bit 1x gain mode. The ADC output codes represent input voltages (AIN – GND) from 0 V to V_{REF} x (1023/1024) and are represented as 10-bit unsigned integers. Note that the hexadecimal numbers shown are left-justified, 10-bit values. In the example on the left-hand side, an AD0WINT interrupt will be generated if the ADC0 conversion word (ADC0H:L) is within the range defined by ADC0GTH:L and ADC0LTH:L (if 0x1000 < ADC0H:L < 0x2000). In the example on the right-hand side, an AD0WINT interrupt will be generated if ADC0 is outside of the range defined by ADC0GTH:L and ADC0LTH:L (if ADC0H:L < 0x1000 or ADC0H:L > 0x2000).

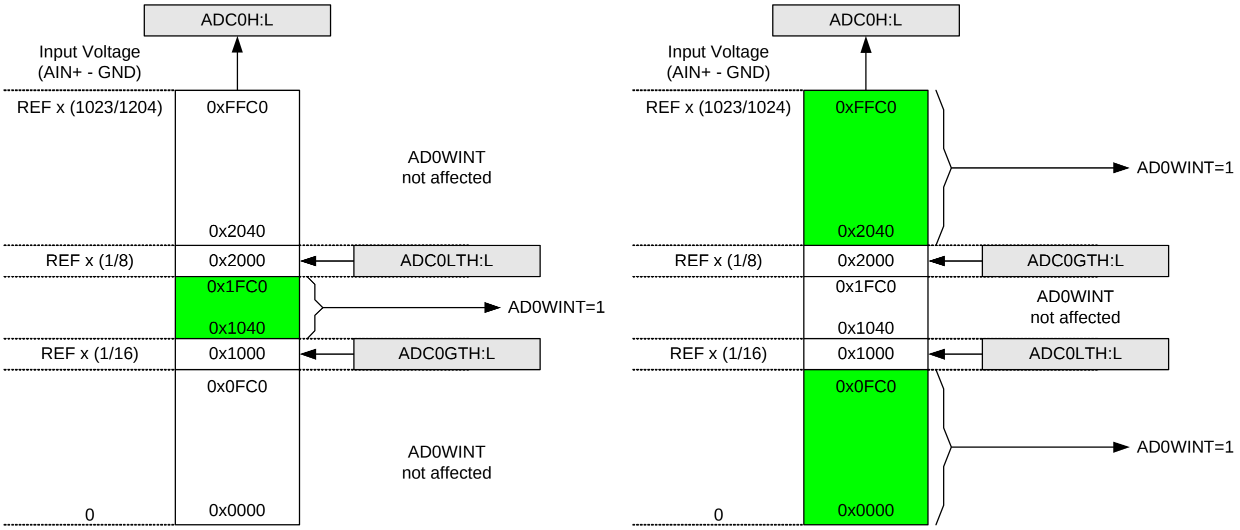


Figure 5.6. ADC Window Compare Examples

SFR Definition 5.8. ADC0GTH: ADC0 Greater-Than High Byte

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:
								0xC4

Bits7–0: ADC0 Greater-Than High Byte.

SFR Definition 5.9. ADC0GTL: ADC0 Greater-Than Low Byte

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
								11000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xC3

Bits7–0: ADC0 Greater-Than Low Byte. In 8-bit compatibility mode, this register should be set to 0x00.

SFR Definition 5.10. ADC0LTH: ADC0 Less-Than 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:
								0xC6

Bits7–0: ADC0 Less-Than High Byte.

SFR Definition 5.11. ADC0LTL: ADC0 Less-Than 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:
								0xC5

Bits7–0: ADC0 Less-Than Low Byte. In 8-bit compatibility mode, this register should be set to 0x00.

Table 5.1. ADC0 Electrical Characteristics

V_{DD} = 3.0 V, V_{REF} = 2.50 V (REFSL = 0), PGA Gain = 1, –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
DC Accuracy					
Resolution		10			bits
Integral Nonlinearity		—	±0.5	±1	LSB
Differential Nonlinearity	Guaranteed Monotonic	—	±0.5	±1	LSB
Offset Error[1]		—	TBD	TBD	LSB
Full Scale Error[1]		—	TBD	TBD	LSB
Dynamic Performance (10 kHz sine-wave input, 1 dB below Full Scale, 500 ksps)					
Signal-to-Noise Plus Distortion		TBD	TBD	—	dB
Total Harmonic Distortion	Up to the 5th harmonic	—	TBD	—	dB
Spurious-Free Dynamic Range		—	TBD	—	dB
Conversion Rate					
SAR Conversion Clock[2,3]		—	—	8.33	MHz
Conversion Time in SAR Clocks	10-bit Mode 8-bit Mode	13 11	—	—	clocks clocks
Track/Hold Acquisition Time	$V_{DD} \geq 2.0$ V	300	—	—	ns
	$V_{DD} < 2.0$ V	2.0	0.3	—	µs
Throughput Rate[3]		—	—	500	ksps
Analog Inputs					
Absolute Voltage on External ADC Input		GND – 0.3	—	V_{DD} + 0.3	V
Input Voltage Range (Gain = 1x)	AIN – GND	0	—	V_{REF}	V
SAR Sampling Capacitor	1x Gain 0.5x Gain	— —	5 3	— —	pF pF
Temperature Sensor		—		—	
Linearity[1,4]		—	TBD	—	°C
Slope[4]		—	3.2	—	mV/°C
Slope Error[1,4]		—	±80	—	µV/C
Offset[4]	Temp = 0° C	—	903	—	mV
Offset Error[1,4]	Temp = 0° C	—	±10	—	mV
Power Specifications					
Power Supply Current (V_{DD} supplied to ADC0)	Operating Mode, 500 ksps	—	400	900	µA
Power Supply Rejection		—	TBD	—	mV/V

Notes:

1. Represents mean ± one standard deviation.
2. When using the C8051F300 for code development, SAR clock should be limited to 6 MHz.
3. See **Section "18. Revision Specific Behavior" on page 159**.
4. Includes ADC offset, gain, and linearity variations.

6. Voltage Reference Options

The voltage reference MUX is configurable to use an externally connected voltage reference, the unregulated power supply voltage (V_{DD}), or the regulated 1.8 V internal supply (see Figure 6.1). The REFSL bit in the Reference Control register (REF0CN) selects the reference source. For an external source, REFSL should be set to ‘0’; For V_{DD} as the reference source, REFSL should be set to ‘1’. To override this selection, and use the internal regulator as the reference source, the REGOVR bit can be set to ‘1’. See Figure 6.1 for REF0CN register details. The electrical specifications for the voltage reference circuit are given in Table 6.1.

Important Note About the V_{REF} Input: Port pin P0.0 is used as the external V_{REF} input. When using an external voltage reference, P0.0 should be configured as analog input and skipped by the Digital Crossbar. To configure P0.0 as analog input, set to ‘1’ Bit0 in register P0MDIN. To configure the Crossbar to skip P0.0, set to ‘1’ Bit0 in register XBR0. Refer to **Section “13. Port Input/Output” on page 97** for complete Port I/O configuration details. The external reference voltage must be within the range $0 \leq V_{REF} \leq V_{DD}$.

On C8051T600/2 devices, the temperature sensor connects to the highest order input of the ADC0 positive input multiplexer (see **Section “5.1. Analog Multiplexer” on page 32** for details). The TEMPE bit in register REF0CN enables/disables the temperature sensor. While disabled, the temperature sensor defaults to a high impedance state and any ADC0 measurements performed on the sensor result in meaningless data.

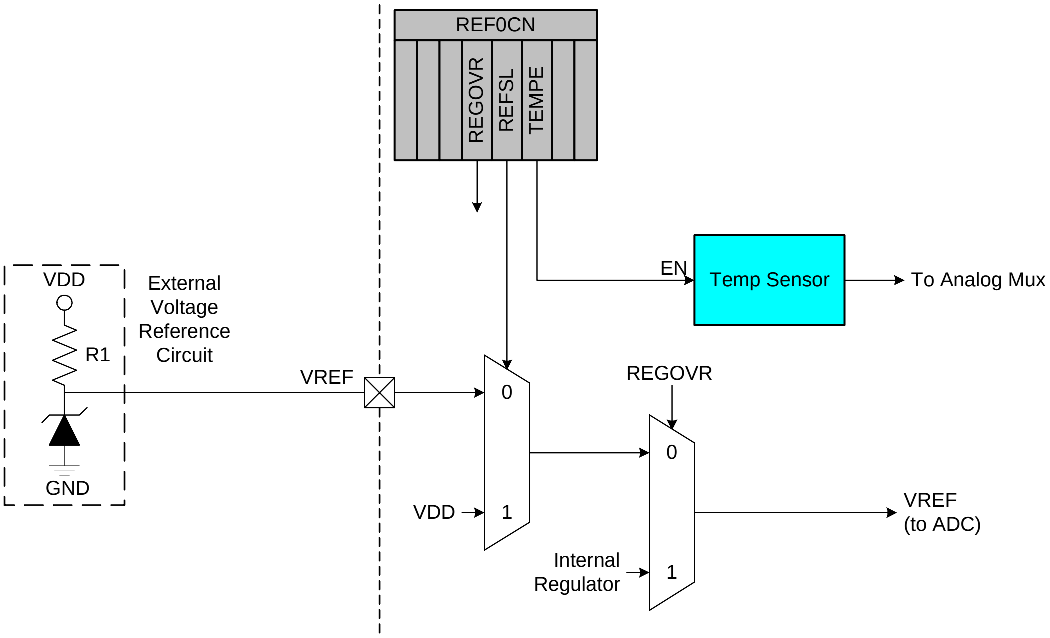


Figure 6.1. Voltage Reference Functional Block Diagram

SFR Definition 6.1. REF0CN: Reference Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
—	—	—	REGOVR	REFSL	TEMPE	—	—	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xD1

Bits7–5: UNUSED. Read = 000b; Write = don't care.

Bit4: REGOVR: Regulator Reference Override.
This bit "overrides" the REFSL bit, and allows the internal regulator to be used as a reference source.
0: The voltage reference source is selected by the REFSL bit.
1: The internal regulator is used as the voltage reference, regardless of the REFSL setting.

Bit3: REFSL: Voltage Reference Select.
This bit selects the source for the internal voltage reference when REGOVR is set to '0'.
0: V_{REF} input pin used as voltage reference.
1: V_{DD} used as voltage reference.

Bit2: TEMPE: Temperature Sensor Enable Bit.
0: Internal Temperature Sensor off.
1: Internal Temperature Sensor on.

Bits1-0: UNUSED. Read = 00b. Write = don't care.

Table 6.1. External Voltage Reference Circuit Electrical Characteristics

V_{DD} = 3.0 V; –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Input Voltage Range		0	—	V_{DD}	V
Input Current	Sample Rate = 500 ksps; V_{REF} = 2.5 V	—	12	—	µA

7. Comparator0

C8051T600/1/2/3/4/5 devices include an on-chip programmable voltage comparator, which is shown in Figure 7.1. Comparator0 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), or an asynchronous “raw” output (CP0A). The asynchronous CP0A signal is available even when the system clock is not active. This allows Comparator0 to operate and generate an output with the device in STOP mode (when the internal regulator is still active). When assigned to a Port pin, the Comparator0 output may be configured as open drain or push-pull (see **Section “13.2. Port I/O Initialization” on page 99**). Comparator0 may also be used as a reset source (see **Section “10.5. Comparator0 Reset” on page 83**).

The inputs for Comparator0 are selected in the CPT0MX register (SFR Definition 7.2). The CMX0P1–CMX0P0 bits select the Comparator0 positive input; the CMX0N1–CMX0N0 bits select the Comparator0 negative input.

Important Note About Comparator Inputs: The Port pins selected as comparator inputs should be configured as analog inputs in their associated Port configuration register, and configured to be skipped by the Crossbar (for details on Port configuration, see **Section “13.3. General Purpose Port I/O” on page 102**).

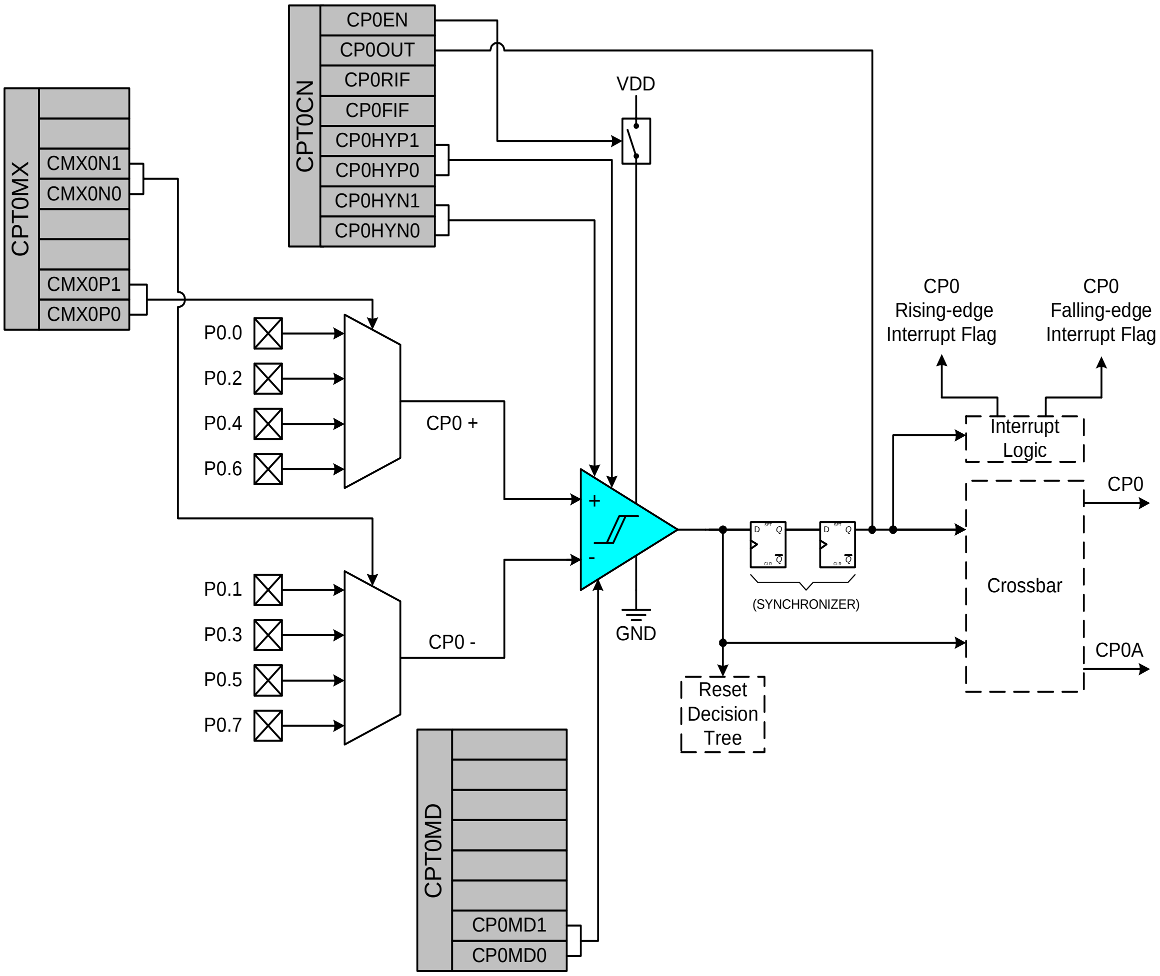


Figure 7.1. Comparator0 Functional Block Diagram

The output of Comparator0 can be polled in software, used as an interrupt source, and/or routed to a Port pin. When routed to a Port pin, the Comparator0 output is available asynchronous or synchronous to the system clock; the asynchronous output is available even in STOP mode (with no system clock active). When disabled, the Comparator0 output (if assigned to a Port I/O pin via the Crossbar) defaults to the logic low state. See **Section “13.1. Priority Crossbar Decoder” on page 98** for details on configuring the Comparator0 output via the digital Crossbar. Comparator0 inputs can be externally driven from –0.25 V to (V_{DD} + 0.25 V) without damage or upset. The complete electrical specifications for Comparator0 are given in Table 7.1.

The Comparator0 response time may be configured in software via the CP0MD1–0 bits in register CPT0MD (see SFR Definition 7.3). Selecting a longer response time reduces the amount of power consumed by Comparator0. See Table 7.1 for complete timing and power consumption specifications.

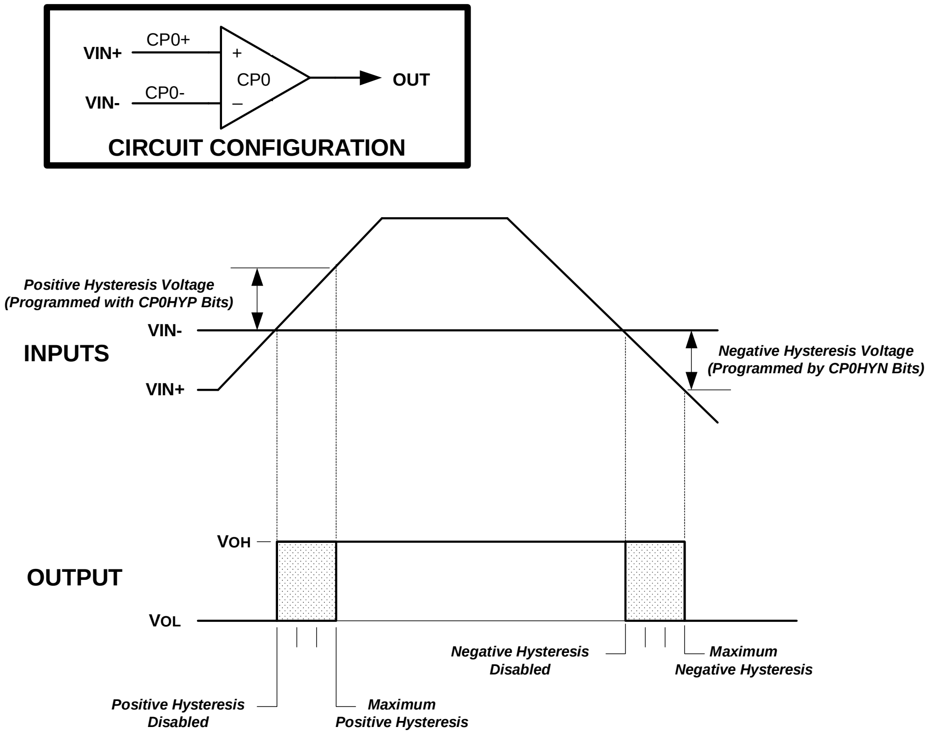


Figure 7.2. Comparator Hysteresis Plot

The hysteresis of Comparator0 is software-programmable via its Comparator0 Control register (CPT0CN). The user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage.

The Comparator0 hysteresis is programmed using Bits3–0 in the Comparator0 Control Register CPT0CN (shown in SFR Definition 7.1). The amount of negative hysteresis voltage is determined by the settings of the CP0HYN bits. The amount of positive hysteresis can be programmed using the CP0HYP bits.

Comparator0 interrupts can be generated on both rising-edge and falling-edge output transitions. (For Interrupt enable and priority control, see **Section “9.3. Interrupt Handler” on page 70**). The CP0FIF flag is set to logic 1 upon a Comparator0 falling-edge interrupt, and the CP0RIF flag is set to logic 1 upon the

Comparator0 rising-edge interrupt. Once set, these bits remain set until cleared by software. The output state of Comparator0 can be obtained at any time by reading the CP0OUT bit. Comparator0 is enabled by setting the CP0EN bit to logic 1, and is disabled by clearing this bit to logic 0.

SFR Definition 7.1. CPT0CN: Comparator0 Control

R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CP0EN	CP0OUT	CP0RIF	CP0FIF	CP0HYP1	CP0HYP0	CP0HYN1	CP0HYN0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xF8

Bit7: CP0EN: Comparator0 Enable Bit.
0: Comparator0 Disabled.
1: Comparator0 Enabled.

Bit6: CP0OUT: Comparator0 Output State Flag.
0: Voltage on CP0+ < CP0–.
1: Voltage on CP0+ > CP0–.

Bit5: CP0RIF: Comparator0 Rising-Edge Interrupt Flag.
0: No Comparator0 Rising Edge Interrupt has occurred since this flag was last cleared.
1: Comparator0 Rising Edge Interrupt has occurred.

Bit4: CP0FIF: Comparator0 Falling-Edge Interrupt Flag.
0: No Comparator0 Falling-Edge Interrupt has occurred since this flag was last cleared.
1: Comparator0 Falling-Edge Interrupt has occurred.

Bits3–2: CP0HYP1–0: Comparator0 Positive Hysteresis Control Bits.
00: Positive Hysteresis Disabled.
01: Positive Hysteresis = 5 mV.
10: Positive Hysteresis = 10 mV.
11: Positive Hysteresis = 20 mV.

Bits1–0: CP0HYN1–0: Comparator0 Negative Hysteresis Control Bits.
00: Negative Hysteresis Disabled.
01: Negative Hysteresis = 5 mV.
10: Negative Hysteresis = 10 mV.
11: Negative Hysteresis = 20 mV.

SFR Definition 7.2. CPT0MX: Comparator0 MUX Selection

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
—	—	CMX0N1	CMX0N0	—	—	CMX0P1	CMX0P0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x9F

Bits7–6: UNUSED. Read = 00b, Write = don’t care.

Bits6–4: CMX0N1–CMX0N0: Comparator0 Negative Input MUX Select.
These bits select which Port pin is used as the Comparator0 negative input.

CMX0N1	CMX0N0	Negative Input
0	0	P0.1
0	1	P0.3
1	0	P0.5
1	1	P0.7

Bits3–2: UNUSED. Read = 00b, Write = don’t care.

Bits1–0: CMX0P1–CMX0P0: Comparator0 Positive Input MUX Select.
These bits select which Port pin is used as the Comparator0 positive input.

CMX0P1	CMX0P0	Positive Input
0	0	P0.0
0	1	P0.2
1	0	P0.4
1	1	P0.6

SFR Definition 7.3. CPT0MD: Comparator0 Mode Selection

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
—	—	—	—	—	—	CP0MD1	CP0MD0	00000010
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x9D

Bits7–2: UNUSED. Read = 000000b, Write = don’t care.
Bits1–0: CP0MD1–CP0MD0: Comparator0 Mode Select.
These bits select the response time and power consumption for Comparator0.

Mode	CP0MD1	CP0MD0	CP0 Response Time / Power Setting*
0	0	0	Fastest Response Time
1	0	1	—
2	1	0	—
3	1	1	Lowest Power Consumption
***Note:** See Table 7.1 for response time and power consumption values.			

Table 7.1. Comparator0 Electrical Characteristics

V_{DD} = 3.0 V, –40 to +85 °Cu nless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Response Time: Mode 0, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	100	—	ns
	CP0+ – CP0– = –100 mV	—	250	—	ns
Response Time: Mode 1, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	175	—	ns
	CP0+ – CP0– = –100 mV	—	500	—	ns
Response Time: Mode 2, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	320	—	ns
	CP0+ – CP0– = –100 mV	—	1100	—	ns
Response Time: Mode 3, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	1050	—	ns
	CP0+ – CP0– = –100 mV	—	5200	—	ns
Common-Mode Rejection Ratio		—	1.5	TBD	mV/V
Positive Hysteresis 1	CP0HYP1–0 = 00	—	0	TBD	mV
Positive Hysteresis 2	CP0HYP1–0 = 01	TBD	5	TBD	mV
Positive Hysteresis 3	CP0HYP1–0 = 10	TBD	10	TBD	mV
Positive Hysteresis 4	CP0HYP1–0 = 11	TBD	20	TBD	mV
Negative Hysteresis 1	CP0HYN1–0 = 00	—	0	TBD	mV
Negative Hysteresis 2	CP0HYN1–0 = 01	TBD	5	TBD	mV
Negative Hysteresis 3	CP0HYN1–0 = 10	TBD	10	TBD	mV
Negative Hysteresis 4	CP0HYN1–0 = 11	TBD	20	TBD	mV
Inverting or Non-Inverting Input Voltage Range		–0.25	—	V_{DD} + 0.25	V
Input Capacitance		—	7	—	pF
Input Bias Current		–5	0.001	TBD	nA
Input Offset Voltage		–5	—	TBD	mV
Power Supply					
Power Supply Rejection		—	0.1	TBD	mV/V
Powerup Time		—	10	—	µs
Supply Current at DC	Mode 0	—	20	—	µA
	Mode 1	—	8.3	—	µA
	Mode 2	—	2.6	—	µA
	Mode 3	—	0.5	—	µA

***Note:** Vcm is the common-mode voltage on CP0+ and CP0–.

8. Voltage Regulator (REG0)

C8051T600/1/2/3/4/5 devices include an internal voltage regulator (REG0) to regulate the internal core supply to 1.8 V from a V_{DD} supply of 1.8 to 3.6 V. Two power-saving modes are built into the regulator to help reduce current consumption in low-power applications. These modes are accessed through the REG0CN register (SFR Definition 8.1). Electrical characteristics for the on-chip regulator are specified in Table 8.1

If an external regulator is used to power the device, the internal regulator may be put into bypass mode using the BYPASS bit. **The internal regulator should never be placed in bypass mode unless an external 1.8 V regulator is used to supply V_{DD}. Doing so could cause permanent damage to the device.**

Under default conditions, when the device enters STOP mode the internal regulator will remain on. This allows any enabled reset source to generate a reset for the device and bring the device out of STOP mode. For additional power savings, the STOPCF bit can be used to shut down the regulator and the internal power network of the device when the part enters STOP mode. When STOPCF is set to ‘1’, the $\overline{\text{RST}}$ pin and a full power cycle of the device are the only methods of generating a reset.

SFR Definition 8.1. REG0CN: Voltage Regulator Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
STOPCF	BYPASS	—	—	—	—	—	OTPPCE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xC7

Bit 7	STOPCF: Stop Mode Configuration. This bit configures the regulator’s behavior when the device enters STOP mode. 0: Regulator is still active in STOP mode. Any enabled reset source will reset the device. 1: Regulator is shut down in STOP mode. Only the $\overline{\text{RST}}$ pin, Supply Monitor or power cycle can reset the device.
Bit 6	BYPASS: Bypass Internal Regulator. This bit places the regulator in bypass mode, turning off the regulator, and allowing the core to run directly from the V_{DD} supply pin. 0: Normal Mode - Regulator is on. 1: Bypass Mode - Regulator is off, and the microcontroller core operates directly from the V_{DD} supply voltage. **IMPORTANT: Bypass mode is for use with an external regulator as the supply voltage only. Never place the regulator in bypass mode when the V_{DD} supply voltage is greater than the specifications given in Table 2.1 on page 23. Doing so may cause permanent damage to the device.**
Bits 5–1	RESERVED. Must Always Write 00000b.
Bit 0	OTPPCE: OTP Power Controller Enable. This bit can help the system save power at slower system clock frequencies (about 2 MHz or less) by automatically shutting down the OTP memory between clocks when information is not being fetched from the OTP memory. 0: Normal Mode - OTP power controller disabled (OTP memory is always on). 1: Low Power Mode - OTP power controller enabled (OTP memory turns on/off as needed). **Note**: If an external clock source is used with the OTP Power Controller enabled, and the clock frequency changes from slow (<2 MHz) to fast (>2 MHz), the OTP power will turn on, and up to 20 clocks may be "skipped" to ensure that the OTP power is stable before reading memory.

Table 8.1. Internal Voltage Regulator Electrical Characteristics

–40 to +85 ℃ unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Input Voltage Range		1.8	—	3.6	V
Bias Current	Normal Mode	—	TBD	—	µA

9. 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. Included are three 16-bit counter/timers (see description in **Section 16**), an enhanced full-duplex UART (see description in **Section 15**), 256 bytes of internal RAM, 128 byte Special Function Register (SFR) address space (**Section 9.2.6**), and one byte-wide I/O Port (see description in **Section 13**). The CIP-51 also includes on-chip debug hardware (see description in **Section 19**), 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 9.1 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
- 256 Bytes of Internal RAM
- Byte-Wide I/O Port
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

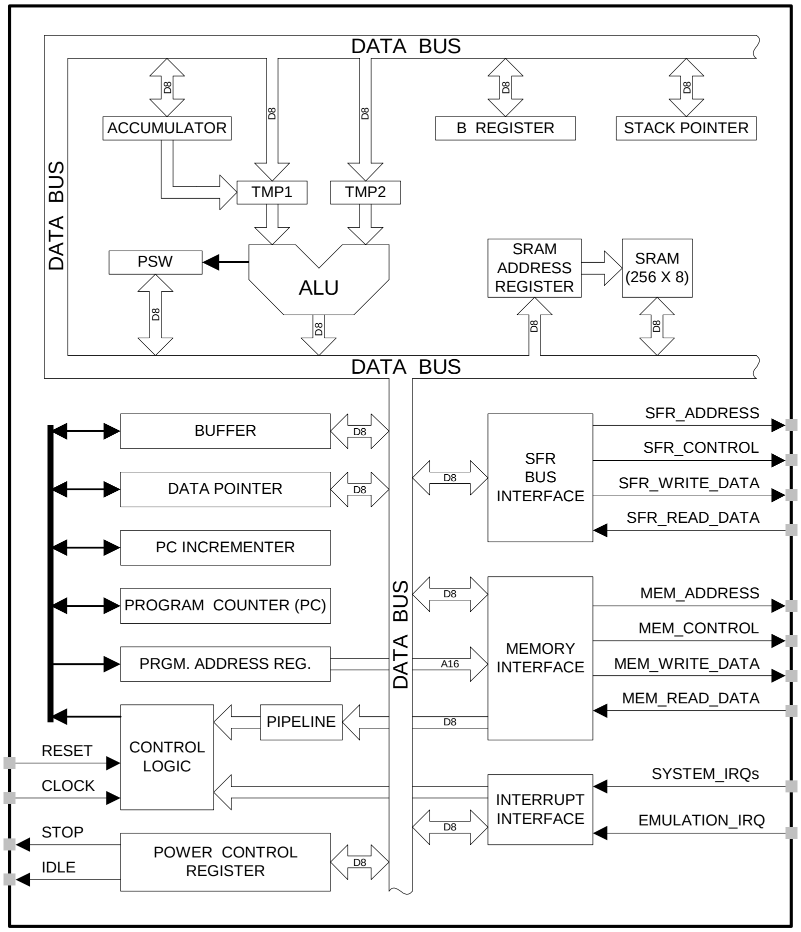


Figure 9.1. CIP-51 Block Diagram

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 for 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 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. 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 “19. C2 Interface” on page 161**.

The C8051F300 can be used as a code development platform. The C8051F300 utilizes the same pinout, and can operate with the same firmware, but contains re-programmable Flash memory, allowing for quick development of code.

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.

9.1. 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.

9.1.1. 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 9.1 is the **CIP-51 Instruction Set Summary**, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

Table 9.1. CIP-51 Instruction Set Summary

Mnemonic	Description	Bytes	Clock Cycles
Arithmetic Operations			
ADD A, Rn	Add register to A	1	1
ADD A, direct	Add direct byte to A	2	2
ADD A, @Ri	Add indirect RAM to A	1	2
ADD A, #data	Add immediate to A	2	2
ADDC A, Rn	Add register to A with carry	1	1
ADDC A, direct	Add direct byte to A with carry	2	2
ADDC A, @Ri	Add indirect RAM to A with carry	1	2
ADDC A, #data	Add immediate to A with carry	2	2
SUBB A, Rn	Subtract register from A with borrow	1	1
SUBB A, direct	Subtract direct byte from A with borrow	2	2
SUBB A, @Ri	Subtract indirect RAM from A with borrow	1	2
SUBB A, #data	Subtract immediate from A with borrow	2	2
INC A	Increment A	1	1
INC Rn	Increment register	1	1
INC direct	Increment direct byte	2	2
INC @Ri	Increment indirect RAM	1	2
DEC A	Decrement A	1	1
DEC Rn	Decrement register	1	1
DEC direct	Decrement direct byte	2	2
DEC @Ri	Decrement indirect RAM	1	2
INC DPTR	Increment Data Pointer	1	1
MUL AB	Multiply A and B	1	4
DIV AB	Divide A by B	1	8
DA A	Decimal adjust A	1	1

Table 9.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
Logical Operations			
ANL A, Rn	AND Register to A	1	1
ANL A, direct	AND direct byte to A	2	2
ANL A, @Ri	AND indirect RAM to A	1	2
ANL A, #data	AND immediate to A	2	2
ANL direct, A	AND A to direct byte	2	2
ANL direct, #data	AND immediate to direct byte	3	3
ORL A, Rn	OR Register to A	1	1
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

Table 9.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
*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
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
***Note:** MOVX instructions are implemented within the CIP-51 core, though no XRAM space is available on these devices.			

Notes on Registers, Operands and Addressing Modes:
Rn—Register R0–R7 of the currently selected register bank. @Ri—Data RAM location addressed indirectly through R0 or R1. **rel**—8-bit, signed (two’s complement) offset relative to the first byte of the following instruction. Used by SJMP and all conditional jumps. **direct**—8-bit internal data location’s address. This could be a direct-access Data RAM location (0x00–0x7F) or an SFR (0x80–0xFF). **#data**—8-bit constant **#data16**—16-bit constant **bit**—Direct-accessed bit in Data RAM or SFR **addr11**—11-bit destination address used by ACALL and AJMP. The destination must be within the same 2 kB page of program memory as the first byte of the following instruction. **addr16**—16-bit destination address used by LCALL and LJMP. The destination may be anywhere within the 8 kB program memory space. There is one unused opcode (0xA5) that performs the same function as NOP. All mnemonics copyrighted © Intel Corporation 1980.

9.2. Memory Organization

The memory organization of the CIP-51 System Controller is similar to that of a standard 8051. There are two separate memory spaces: program memory and data memory. Program and data memory share the same address space but are accessed via different instruction types. The CIP-51 memory organization is shown in Figure 9.2 and Figure 9.3.

9.2.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051T600/1 implements 8192 bytes of this program memory space as in-system, OTP EPROM, organized in a contiguous block from addresses 0x0000 to 0x1FFF. Note: 512 bytes (0x1E00 – 0x1FFF) of this memory are reserved for factory use and are not available for user program storage. The C8051T602/3 implements 4096 bytes of OTP EPROM program memory space; the C8051T604/5 implements 2048 bytes of OTP EPROM program memory space. Figure 9.2 shows the program memory maps for C8051T600/1/2/3/4/5 devices.

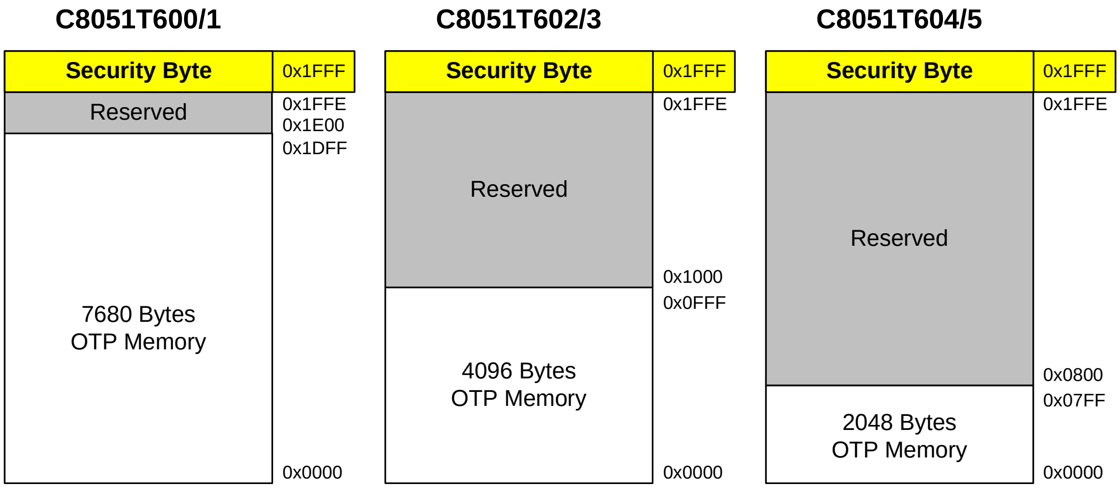


Figure 9.2. Program Memory Maps

Program memory is read-only from within firmware. Individual program memory bytes can be read using the MOVC instruction. This facilitates the use of OTP EPROM space for constant storage.

9.2.2. Data Memory

The CIP-51 includes 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight byte-wide registers. The next 16 bytes, locations 0x20 through 0x2F, may either be addressed as bytes or as 128 bit locations accessible with the direct addressing mode.

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFRs. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the upper 128 bytes of data memory. Figure 9.3 illustrates the data memory organization of the CIP-51.

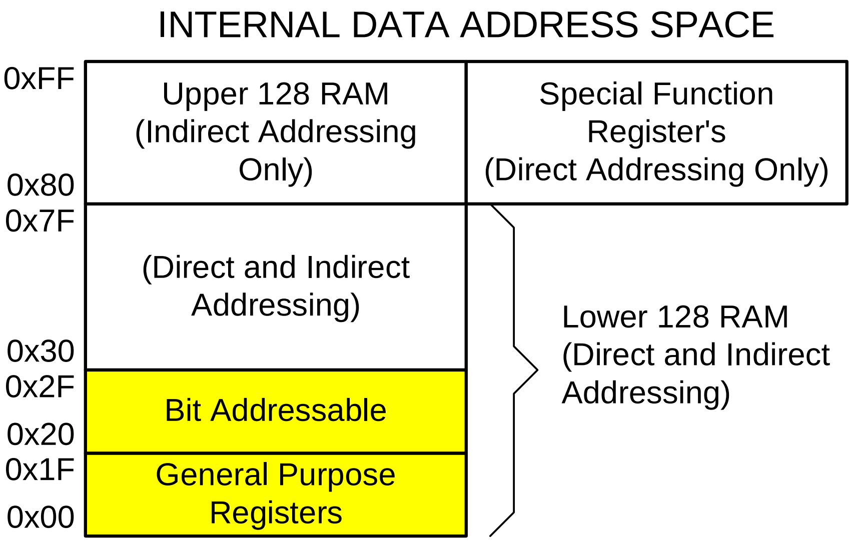


Figure 9.3. Data Memory Map

9.2.3. General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in Figure 9.4). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

9.2.4. Bit Addressable Locations

In addition to direct access to data memory organized as bytes, the sixteen data memory locations at 0x20 through 0x2F are also accessible as 128 individually addressable bits. Each bit has a bit address from 0x00 to 0x7F. Bit 0 of the byte at 0x20 has bit address 0x00 while bit 7 of the byte at 0x20 has bit address 0x07. Bit 7 of the byte at 0x2F has bit address 0x7F. A bit access is distinguished from a full byte access by the type of instruction used (bit source or destination operands as opposed to a byte source or destination).

The MCS-51™ assembly language allows an alternate notation for bit addressing of the form XX.B where XX is the byte address and B is the bit position within the byte. For example, the instruction:

```
MOV    C, 22.3h
```

moves the Boolean value at 0x13 (bit 3 of the byte at location 0x22) into the Carry flag.

9.2.5. Stack

A programmer's stack can be located anywhere in the 256-byte data memory. The stack area is designated using the Stack Pointer (SP, 0x81) SFR. The SP will point to the last location used. The next value pushed on the stack is placed at SP+1 and then SP is incremented. A reset initializes the stack pointer to location 0x07. Therefore, the first value pushed on the stack is placed at location 0x08, which is also the first register (R0) of register bank 1. Thus, if more than one register bank is to be used, the SP should be initialized to a location in the data memory not being used for data storage. The stack depth can extend up to 256 bytes.

9.2.6. 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 9.2 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 data sheet, as indicated in Table 9.3, for a detailed description of each register.

Table 9.2. Special Function Register (SFR) Memory Map

F8	CPT0CN	PCA0L	PCA0H	PCA0CPL0	PCA0CPH0			
F0	B	P0MDIN					EIP1	
E8	ADC0CN	PCA0CPL1	PCA0CPH1	PCA0CPL2	PCA0CPH2			RSTSRC
E0	ACC	XBR0	XBR1	XBR2	IT01CF		EIE1	
D8	PCA0CN	PCA0MD	PCA0CPM0	PCA0CPM1	PCA0CPM2			
D0	PSW	REF0CN						
C8	TMR2CN		TMR2RLL	TMR2RLH	TMR2L	TMR2H		
C0	SMB0CN	SMB0CF	SMB0DAT	ADC0GTL	ADC0GTH	ADC0LTL	ADC0LTH	REG0CN
B8	IP			AMX0SL	ADC0CF	ADC0L	ADC0H	
B0		OSCXCN	OSCICN	OSCICL				
A8	IE							
A0			TOFFL	TOFFH	P0MDOUT			
98	SCON0	SBUF0				CPT0MD		CPT0MX
90								
88	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON	
80	P0	SP	DPL	DPH				PCON
	0(8) (bit addressable)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)

Table 9.3. Special Function Registers

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

Register	Address	Description	Page #
ACC	0xE0	Accumulator	69
ADC0CF	0xBC	ADC0 Configuration	40
ADC0CN	0xE8	ADC0 Control	41
ADC0GTH	0xC4	ADC0 Greater-Than Compare High Byte	43
ADC0GTL	0xC3	ADC0 Greater-Than Compare Low Byte	43
ADC0LTH	0xC6	ADC0 Less-Than Compare High Byte	43
ADC0LTL	0xC5	ADC0 Less-Than Compare Low Byte	43
ADC0H	0xBE	ADC0 Data Word High Byte	40
ADC0L	0xBD	ADC0 Data Word Low Byte	40
AMX0SL	0xBB	ADC0 Multiplexer Channel Select	39
B	0xF0	B Register	69
CKCON	0x8E	Clock Control	137
CPT0CN	0xF8	Comparator0 Control	49
CPT0MD	0x9D	Comparator0 Mode Selection	51
CPT0MX	0x9F	Comparator0 MUX Selection	50
DPH	0x83	Data Pointer High	67
DPL	0x82	Data Pointer Low	67
EIE1	0xE6	Extended Interrupt Enable 1	75
EIP1	0xF6	External Interrupt Priority 1	76
IE	0xA8	Interrupt Enable	73
IP	0xB8	Interrupt Priority	74
IT01CF	0xE4	INT0/INT1 Configuration Register	77
OSCICL	0xB3	Internal Oscillator Calibration	92
OSCICN	0xB2	Internal Oscillator Control	92
OSCXCN	0xB1	External Oscillator Control	94
P0	0x80	Port 0 Latch	103
P0MDIN	0xF1	Port 0 Input Mode Configuration	103
P0MDOUT	0xA4	Port 0 Output Mode Configuration	103
PCA0CN	0xD8	PCA Control	154
PCA0MD	0xD9	PCA Mode	155
PCA0CPH0	0xFC	PCA Capture 0 High	158
PCA0CPH1	0xEA	PCA Capture 1 High	158
PCA0CPH2	0xEC	PCA Capture 2 High	158
PCA0CPL0	0xFB	PCA Capture 0 Low	158
PCA0CPL1	0xE9	PCA Capture 1 Low	158
PCA0CPL2	0xEB	PCA Capture 2 Low	158
PCA0CPM0	0xDA	PCA Module 0 Mode Register	156
PCA0CPM1	0xDB	PCA Module 1 Mode Register	156

Table 9.3. Special Function Registers (Continued)

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

Register	Address	Description	Page #
PCA0CPM2	0xDC	PCA Module 2 Mode Register	156
PCA0H	0xFA	PCA Counter High	157
PCA0L	0xF9	PCA Counter Low	157
PCON	0x87	Power Control	79
PSW	0xD0	Program Status Word	68
REF0CN	0xD1	Voltage Reference Control	46
REG0CN	0xC7	Voltage Regulator Control	54
RSTSRC	0xEF	Reset Source Configuration/Status	85
SBUF0	0x99	UART 0 Data Buffer	129
SCON0	0x98	UART 0 Control	128
SMB0CF	0xC1	SMBus Configuration	112
SMB0CN	0xC0	SMBus Control	114
SMB0DAT	0xC2	SMBus Data	116
SP	0x81	Stack Pointer	67
TMR2CN	0xC8	Timer/Counter 2 Control	141
TCON	0x88	Timer/Counter Control	135
TH0	0x8C	Timer/Counter 0 High	138
TH1	0x8D	Timer/Counter 1 High	138
TL0	0x8A	Timer/Counter 0 Low	138
TL1	0x8B	Timer/Counter 1 Low	138
TMOD	0x89	Timer/Counter Mode	136
TMR2RLH	0xCB	Timer/Counter 2 Reload High	142
TMR2RLL	0xCA	Timer/Counter 2 Reload Low	142
TMR2H	0xCD	Timer/Counter 2 High	142
TMR2L	0xCC	Timer/Counter 2 Low	142
TOFFH	0xA3	Temperature Sensor Offset Measurement High	34
TOFFL	0xA2	Temperature Sensor Offset Measurement Low	35
XBR0	0xE1	Port I/O Crossbar Control 0	100
XBR1	0xE2	Port I/O Crossbar Control 1	101
XBR2	0xE3	Port I/O Crossbar Control 2	102
All other SFR locations		Reserved	

9.2.7. Register Descriptions

Following are descriptions of SFRs related to the operation of the CIP-51 System Controller. Reserved bits should only be set to the value indicated in the register description. Future product versions may use these bits to implement new features in which case the reset value of the bit will select the feature's default state. Detailed descriptions of the remaining SFRs are included in the sections of the data sheet associated with their corresponding system function.

SFR Definition 9.1. 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.

SFR Definition 9.2. 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.

SFR Definition 9.3. 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.

SFR Definition 9.4. PSW: Program Status Word

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	Reset Value
CY	AC	F0	RS1	RS0	OV	F1	PARITY	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xD0

Bit7: CY: Carry Flag.
This bit is set when the last arithmetic operation resulted in a carry (addition) or a borrow (subtraction). It is cleared to logic 0 by all other arithmetic operations.

Bit6: AC: Auxiliary Carry Flag
This bit is set when the last arithmetic operation resulted in a carry into (addition) or a borrow from (subtraction) the high order nibble. It is cleared to logic 0 by all other arithmetic operations.

Bit5: F0: User Flag 0.
This is a bit-addressable, general purpose flag for use under software control.

Bits4–3: RS1-RS0: Register Bank Select.
These bits select which register bank is used during register accesses.

RS1	RS0	Register Bank	Address
0	0	0	0x00–0x07
0	1	1	0x08–0x0F
1	0	2	0x10–0x17
1	1	3	0x18–0x1F

Bit2: OV: Overflow Flag.
The OV flag is set to 1 under the following circumstances:
- An ADD, ADDC, or SUBB instruction causes a sign-change overflow.
- A MUL instruction results in an overflow (result is greater than 255).
- A DIV instruction causes a divide-by-zero condition.

The OV flag is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases.

Bit1: F1: User Flag 1.
This is a bit-addressable, general purpose flag for use under software control.

Bit0: PARITY: Parity Flag.
This bit is set to logic 1 if the sum of the eight bits in the accumulator is odd and cleared if the sum is even.

SFR Definition 9.5. 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	SFR Address:
							(bit addressable)	0xE0

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

SFR Definition 9.6. 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.

9.3. Interrupt Handler

The CIP-51 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 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) instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

9.3.1. 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 9.4 on page 72. Refer to the data sheet 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).

9.3.2. External Interrupts

The /INT0 and /INT1 external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL (/INT0 Polarity) and IN1PL (/INT1 Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (**Section “16.1. Timer 0 and Timer 1” on page 131**) select level or edge sensitive. The table below lists the possible configurations.

IT0	IN0PL	/INT0 Interrupt
1	0	Active low, edge sensitive
1	1	Active high, edge sensitive
0	0	Active low, level sensitive
0	1	Active high, level sensitive

IT1	IN1PL	/INT1 Interrupt
1	0	Active low, edge sensitive
1	1	Active high, edge sensitive
0	0	Active low, level sensitive
0	1	Active high, level sensitive

/INT0 and /INT1 are assigned to Port pins as defined in the IT01CF register (see SFR Definition 9.11). Note that /INT0 and /INT1 Port pin assignments are independent of any Crossbar assignments. /INT0 and /INT1 will monitor their assigned Port pins without disturbing the peripheral that was assigned the Port pin via the Crossbar. To assign a Port pin only to /INT0 and/or /INT1, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register XBR0 (see **Section “13.1. Priority Crossbar Decoder” on page 98** for complete details on configuring the Crossbar).

IE0 (TCON.1) and IE1 (TCON.3) serve as the interrupt-pending flags for the /INT0 and /INT1 external interrupts, respectively. If an /INT0 or /INT1 external interrupt is configured as edge-sensitive, the corresponding interrupt-pending flag is automatically cleared by the hardware when the CPU vectors to the ISR. When configured as level sensitive, the interrupt-pending flag remains logic 1 while the input is active as defined by the corresponding polarity bit (IN0PL or IN1PL); the flag remains logic 0 while the input is inactive. The external interrupt source must hold the input active until the interrupt request is recognized. It must then deactivate the interrupt request before execution of the ISR completes or another interrupt request will be generated.

9.3.3. 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 EIP1) 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 9.4.

9.3.4. 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 9.4. 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)
SMBus Interface	0x0033	6	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)	PSMB0 (EIP1.0)
ADC0 Window Compare	0x003B	7	AD0WINT (ADC0CN.3)	Y	N	EWADC0 (EIE1.1)	PWADC0 (EIP1.1)
ADC0 Conversion Complete	0x0043	8	AD0INT (ADC0CN.5)	Y	N	EADC0C (EIE1.2)	PADC0C (EIP1.2)
Programmable Counter Array	0x004B	9	CF (PCA0CN.7) CCFn (PCA0CN.n)	Y	N	EPCA0 (EIE1.3)	PPCA0 (EIP1.3)
Comparator0 Falling Edge	0x0053	10	CP0FIF (CPT0CN.4)	N	N	ECP0F (EIE1.4)	PCP0F (EIP1.4)
Comparator0 Rising Edge	0x005B	11	CP0RIF (CPT0CN.5)	N	N	ECP0R (EIE1.5)	PCP0R (EIP1.5)

9.3.5. Interrupt Register Descriptions

The SFRs used to enable the interrupt sources and set their priority level are described below. Refer to the data sheet 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).

SFR Definition 9.7. IE: Interrupt Enable

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
EA	IEGF0	ET2	ES0	ET1	EX1	ET0	EX0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
						(bit addressable)		0xA8

Bit7: EA: Enable All Interrupts.
This bit globally enables/disables all interrupts. It overrides the individual interrupt mask settings.
0: Disable all interrupt sources.
1: Enable each interrupt according to its individual mask setting.

Bit6: IEGF0: General Purpose Flag 0.
This is a general purpose flag for use under software control.

Bit5: ET2: Enable Timer 2 Interrupt.
This bit sets the masking of the Timer 2 interrupt.
0: Disable Timer 2 interrupt.
1: Enable interrupt requests generated by the TF2L or TF2H flags.

Bit4: ES0: Enable UART0 Interrupt.
This bit sets the masking of the UART0 interrupt.
0: Disable UART0 interrupt.
1: Enable UART0 interrupt.

Bit3: ET1: Enable Timer 1 Interrupt.
This bit sets the masking of the Timer 1 interrupt.
0: Disable all Timer 1 interrupt.
1: Enable interrupt requests generated by the TF1 flag.

Bit2: EX1: Enable External Interrupt 1.
This bit sets the masking of external interrupt 1.
0: Disable external interrupt 1.
1: Enable interrupt requests generated by the /INT1 input.

Bit1: ET0: Enable Timer 0 Interrupt.
This bit sets the masking of the Timer 0 interrupt.
0: Disable all Timer 0 interrupt.
1: Enable interrupt requests generated by the TF0 flag.

Bit0: EX0: Enable External Interrupt 0.
This bit sets the masking of external interrupt 0.
0: Disable external interrupt 0.
1: Enable interrupt requests generated by the /INT0 input.

SFR Definition 9.8. IP: Interrupt Priority

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	PT2	PS0	PT1	PX1	PT0	PX0	11000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xB8

Bits7–6: UNUSED. Read = 11b, Write = don't care.

Bit5: PT2: Timer 2 Interrupt Priority Control.
This bit sets the priority of the Timer 2 interrupt.
0: Timer 2 interrupt set to low priority level.
1: Timer 2 interrupt set to high priority level.

Bit4: PS0: UART0 Interrupt Priority Control.
This bit sets the priority of the UART0 interrupt.
0: UART0 interrupt set to low priority level.
1: UART0 interrupt set to high priority level.

Bit3: PT1: Timer 1 Interrupt Priority Control.
This bit sets the priority of the Timer 1 interrupt.
0: Timer 1 interrupt set to low priority level.
1: Timer 1 interrupt set to high priority level.

Bit2: PX1: External Interrupt 1 Priority Control.
This bit sets the priority of the External Interrupt 1 interrupt.
0: External Interrupt 1 set to low priority level.
1: External Interrupt 1 set to high priority level.

Bit1: PT0: Timer 0 Interrupt Priority Control.
This bit sets the priority of the Timer 0 interrupt.
0: Timer 0 interrupt set to low priority level.
1: Timer 0 interrupt set to high priority level.

Bit0: PX0: External Interrupt 0 Priority Control.
This bit sets the priority of the External Interrupt 0 interrupt.
0: External Interrupt 0 set to low priority level.
1: External Interrupt 0 set to high priority level.

SFR Definition 9.9. EIE1: Extended Interrupt Enable 1

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	ECP0R	ECP0F	EPCA0	EADC0C	EWADC0	ESMB0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xE6

Bits7–6:	UNUSED. Read = 00b. Write = don’t care.
Bit5:	ECP0R: Enable Comparator0 (CP0) Rising Edge Interrupt. This bit sets the masking of the CP0 Rising Edge interrupt. 0: Disable CP0 Rising Edge interrupt. 1: Enable interrupt requests generated by the CP0RIF flag.
Bit4:	ECP0F: Enable Comparator0 (CP0) Falling Edge Interrupt. This bit sets the masking of the CP0 Falling Edge interrupt. 0: Disable CP0 Falling Edge interrupt. 1: Enable interrupt requests generated by the CP0FIF flag.
Bit3:	EPCA0: Enable Programmable Counter Array (PCA0) Interrupt. This bit sets the masking of the PCA0 interrupts. 0: Disable all PCA0 interrupts. 1: Enable interrupt requests generated by PCA0.
Bit2:	EADC0C: Enable ADC0 Conversion Complete Interrupt. This bit sets the masking of the ADC0 Conversion Complete interrupt. 0: Disable ADC0 Conversion Complete interrupt. 1: Enable interrupt requests generated by the AD0INT flag.
Bit1:	EWADC0: Enable Window Comparison ADC0 Interrupt. This bit sets the masking of ADC0 Window Comparison interrupt. 0: Disable ADC0 Window Comparison interrupt. 1: Enable interrupt requests generated by ADC0 Window Compare flag.
Bit0:	ESMB0: Enable SMBus Interrupt. This bit sets the masking of the SMBus interrupt. 0: Disable all SMBus interrupts. 1: Enable interrupt requests generated by the SI flag.

SFR Definition 9.10. EIP1: Extended Interrupt Priority 1

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	-	PCP0R	PCP0F	PPCA0	PADC0C	PWADC0	PSMB0	11000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xF6

Bits7–6: UNUSED. Read = 11b. Write = don't care.

Bit5: PCP0R: Comparator0 (CP0) Rising Interrupt Priority Control.
This bit sets the priority of the CP0 rising-edge interrupt.
0: CP0 rising interrupt set to low priority level.
1: CP0 rising interrupt set to high priority level.

Bit4: PCP0F: Comparator0 (CP0) Falling Interrupt Priority Control.
This bit sets the priority of the CP0 falling-edge interrupt.
0: CP0 falling interrupt set to low priority level.
1: CP0 falling interrupt set to high priority level.

Bit3: PPCA0: Programmable Counter Array (PCA0) Interrupt Priority Control.
This bit sets the priority of the PCA0 interrupt.
0: PCA0 interrupt set to low priority level.
1: PCA0 interrupt set to high priority level.

Bit2: PADC0C ADC0 Conversion Complete Interrupt Priority Control
This bit sets the priority of the ADC0 Conversion Complete interrupt.
0: ADC0 Conversion Complete interrupt set to low priority level.
1: ADC0 Conversion Complete interrupt set to high priority level.

Bit1: PWADC0: ADC0 Window Comparator Interrupt Priority Control.
This bit sets the priority of the ADC0 Window interrupt.
0: ADC0 Window interrupt set to low priority level.
1: ADC0 Window interrupt set to high priority level.

Bit0: PSMB0: SMBus Interrupt Priority Control.
This bit sets the priority of the SMBus interrupt.
0: SMBus interrupt set to low priority level.
1: SMBus interrupt set to high priority level.

SFR Definition 9.11. IT01CF: INT0/INT1 Configuration

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
IN1PL	IN1SL2	IN1SL1	IN1SL0	IN0PL	IN0SL2	IN0SL1	IN0SL0	00000001
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xE4

Note: Refer to SFR Definition 16.1 for INT0/1 edge- or level-sensitive interrupt selection.

Bit7: IN1PL: /INT1 Polarity
0: /INT1 input is active low.
1: /INT1 input is active high.

Bits6–4: IN1SL2–0: /INT1 Port Pin Selection Bits
These bits select which Port pin is assigned to /INT1. Note that this pin assignment is independent of the Crossbar; /INT1 will monitor the assigned Port pin without disturbing the peripheral that has been assigned the Port pin via the Crossbar. The Crossbar will not assign the Port pin to a peripheral if it is configured to skip the selected pin (accomplished by setting to ‘1’ the corresponding bit in register XBR0).

IN1SL2–0	/INT1 Port Pin
000	P0.0
001	P0.1
010	P0.2
011	P0.3
100	P0.4
101	P0.5
110	P0.6
111	P0.7

Bit3: IN0PL: /INT0 Polarity
0: /INT0 interrupt is active low.
1: /INT0 interrupt is active high.

Bits2–0: INT0SL2–0: /INT0 Port Pin Selection Bits
These bits select which Port pin is assigned to /INT0. Note that this pin assignment is independent of the Crossbar. /INT0 will monitor the assigned Port pin without disturbing the peripheral that has been assigned the Port pin via the Crossbar. The Crossbar will not assign the Port pin to a peripheral if it is configured to skip the selected pin (accomplished by setting to ‘1’ the corresponding bit in register XBR0).

IN0SL2–0	/INT0 Port Pin
000	P0.0
001	P0.1
010	P0.2
011	P0.3
100	P0.4
101	P0.5
110	P0.6
111	P0.7

9.4. 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 system clock is stopped (analog peripherals remain in their selected states). 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 9.12 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.

9.4.1. 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 "17.3. Watchdog Timer Mode" on page 151** 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.

9.4.2. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the CIP-51 to enter Stop mode as soon as the instruction that sets the bit completes execution. In Stop mode the internal oscillator, CPU, and all digital peripherals are stopped; the state of the external oscillator circuit is not affected. Each analog peripheral (including the external oscillator circuit) may be shut down individually prior to entering Stop Mode. Stop mode can only be terminated by an internal or external reset. On reset, the CIP-51 performs the normal reset sequence and begins program execution at address 0x0000.

If enabled, the Missing Clock Detector will cause an internal reset and thereby terminate the Stop mode. The Missing Clock Detector should be disabled if the CPU is to be put to in STOP mode for longer than the MCD timeout.

By default, when in Stop Mode the internal regulator is still active. However, the regulator can be configured to shut down while in Stop Mode to save power. To shut down the regulator in Stop Mode, the STOPCF bit in register REG0CN should be set to '1' prior to setting the STOP bit (see SFR Definition 8.1). If the regulator is shut down using the STOPCF bit, only the $\overline{\text{RST}}$ pin or a full power cycle are capable of resetting the device.

Note: It is important to follow the instruction to enter Stop mode with one that does not access any SFRs or RAM (such as a NOP). This will prevent additional supply current in Stop mode.

SFR Definition 9.12. PCON: Power Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
GF5	GF4	GF3	GF2	GF1	GF0	STOP	IDLE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x87

Bits7–2: GF5–GF0: General Purpose Flags 5–0.
These are general purpose flags for use under software control.

Bit1: STOP: Stop Mode Select.
Setting this bit will place the CIP-51 in Stop mode. This bit will always be read as 0.
1: CPU goes into Stop mode (turns off internal oscillator).

Bit0: IDLE: Idle Mode Select.
Setting this bit will place the CIP-51 in Idle mode. This bit will always be read as 0.
1: CPU goes into Idle mode (shuts off clock to CPU, but clock to Timers, Interrupts, Serial Ports, and Analog Peripherals are still active).

NOTES:

10. Reset Sources

Reset circuitry allows the controller to be easily placed in a predefined default condition. On entry to this reset state, the following occur:

- CIP-51 halts program execution
- Special Function Registers (SFRs) are initialized to their defined reset values
- External Port pins are forced to a known state
- Interrupts and timers are disabled.

All SFRs are reset to the predefined values noted in the SFR detailed descriptions. The contents of internal data memory are unaffected during a reset; any previously stored data is preserved. However, since the stack pointer SFR is reset, the stack is effectively lost even though the data on the stack is not altered.

The Port I/O latches are reset to 0xFF (all logic ones) in open-drain mode. Weak pullups are enabled during and after the reset. For V_{DD} Monitor and power-on resets, the $\overline{\text{RST}}$ pin is driven low until the device exits the reset state.

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to the internal oscillator. Refer to **Section “12. Oscillators” on page 91** for information on selecting and configuring the system clock source. The Watchdog Timer is enabled with the system clock divided by 12 as its clock source (**Section “17.3. Watchdog Timer Mode” on page 151** details the use of the Watchdog Timer). Once the system clock source is stable, program execution begins at location 0x0000.

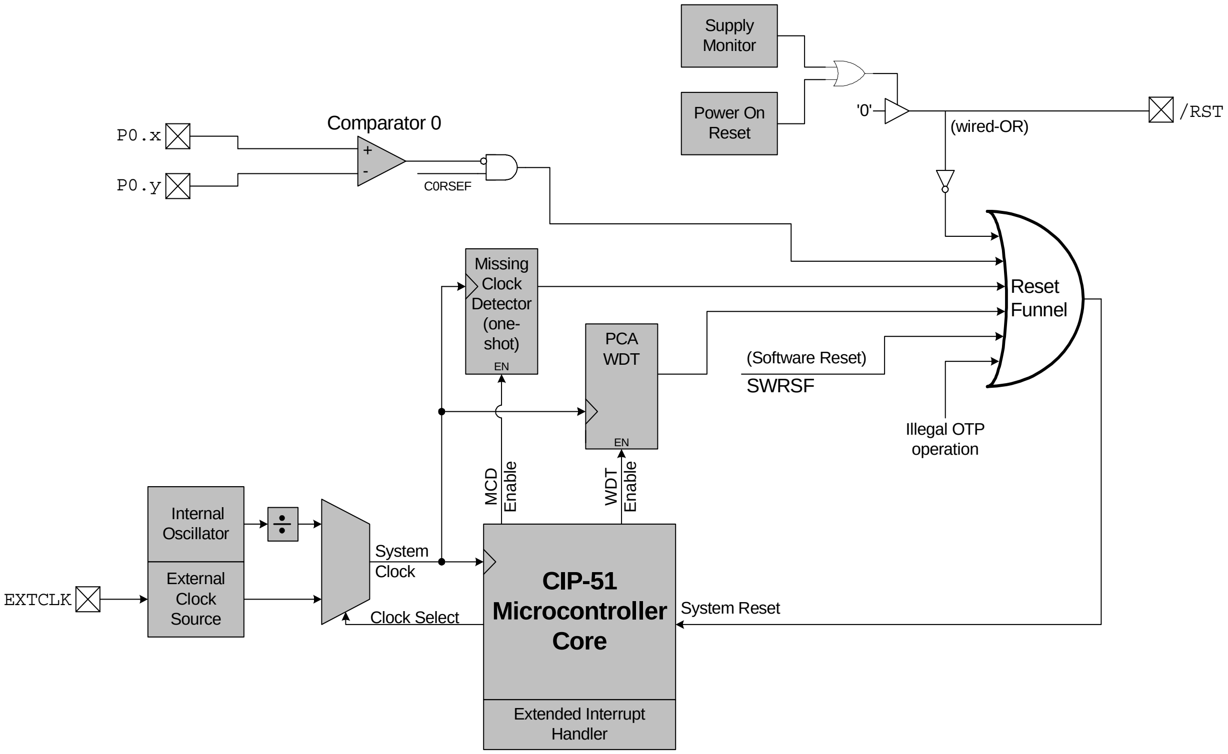


Figure 10.1. Reset Sources

10.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{\text{RST}}$ pin is driven low. An additional delay occurs before the device is released from reset; the delay decreases as the V_{DD} ramp time increases (V_{DD} ramp time is defined as how fast V_{DD} ramps from 0 V to 1.8 V). Figure 10.2. plots the power-on and V_{DD} monitor reset timing. For valid ramp times (less than 1 ms), the power-on reset delay ($T_{PORDelay}$) is typically less than 0.3 ms. The maximum V_{DD} ramp time is 1 ms; slower ramp times may cause the device to be released from reset before V_{DD} reaches the V_{RST} level. If the V_{DD} ramp time in an application will exceed 1 ms, external circuitry should be used to hold $\overline{\text{RST}}$ low until the V_{DD} supply is within the valid supply range for the device.

On exit from a power-on reset, the PORSF flag (RSTSRC.1) is set by hardware to logic 1. When PORSF is set, all of the other reset flags in the RSTSRC Register are indeterminate (PORSF is cleared by all other resets). Since all resets cause program execution to begin at the same location (0x0000) software can read the PORSF flag to determine if a power-up was the cause of reset. The content of internal data memory is undefined after a power-on reset. The V_{DD} monitor is disabled following a power-on reset.

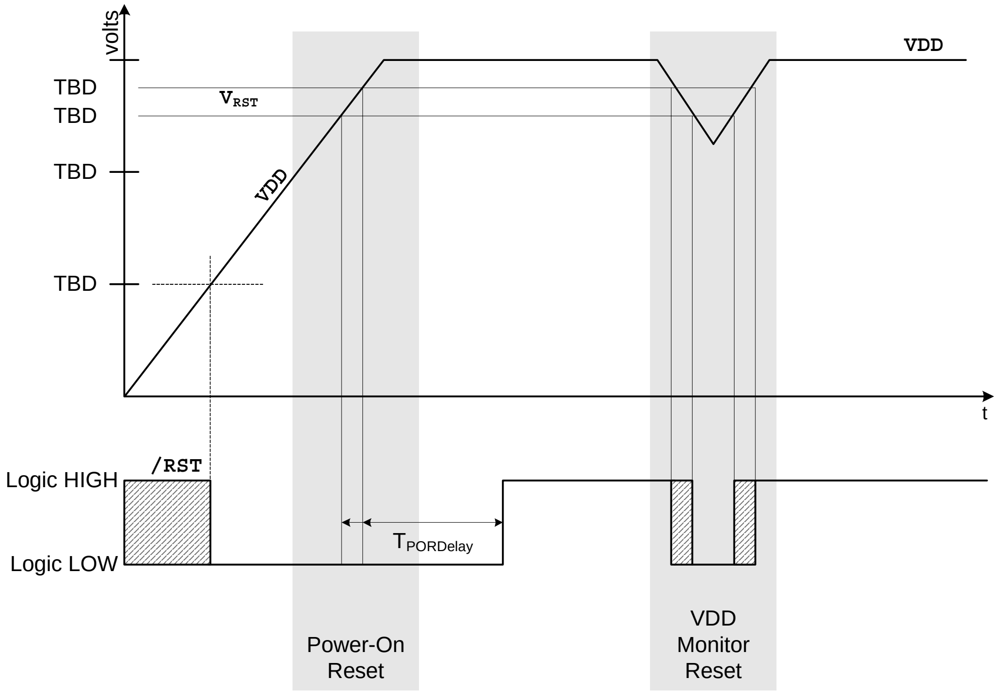


Figure 10.2. Power-On and V_{DD} Monitor Reset Timing

10.2. Power-Fail Reset/V_{DD} Monitor

If the power supply monitor is enabled, when a power-down transition or power irregularity causes V_{DD} to drop below V_{RST}, the power supply monitor will drive the $\overline{RST}$ pin low and hold the CIP-51 in a reset state (see Figure 10.2). When V_{DD} returns to a level above V_{RST}, the CIP-51 will be released from the reset state. Note that even though internal data memory contents are not altered by the power-fail reset, it is impossible to determine if V_{DD} dropped below the level required for data retention. If the PORSF flag reads '1', the data may no longer be valid. The V_{DD} monitor is disabled after power-on resets; however its defined state (enabled/disabled) is not altered by any other reset source. For example, if the V_{DD} monitor is enabled and a software reset is performed, the V_{DD} monitor will still be enabled after the reset. The V_{DD} monitor is enabled by writing a '1' to the PORSF bit in register RSTSRC. See Figure 10.2 for V_{DD} monitor timing; note that the reset delay is not incurred after a V_{DD} monitor reset. See Table 10.2 for electrical characteristics of the V_{DD} monitor.

Important Note: Enabling the V_{DD} monitor when it is not already enabled will immediately generate a system reset. The device will then return from the reset state with the V_{DD} monitor enabled. **Writing a logic '1' to the PORSF flag when the V_{DD} monitor is already enabled does not cause a system reset.**

10.3. External Reset

The external $\overline{RST}$ pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the $\overline{RST}$ pin generates a reset; an external pullup and/or decoupling of the $\overline{RST}$ pin may be necessary to avoid erroneous noise-induced resets. See Table 10.2 for complete $\overline{RST}$ pin specifications. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

10.4. Missing Clock Detector Reset

The Missing Clock Detector (MCD) is a one-shot circuit that is triggered by the system clock. If the system clock remains high or low for more than 100 µs, the one-shot will time out and generate a reset. After a MCD reset, the MCDRSF flag (RSTSRC.2) will read '1', signifying the MCD as the reset source; otherwise, this bit reads '0'. Writing a '1' to the MCDRSF bit enables the Missing Clock Detector; writing a '0' disables it. The state of the $\overline{RST}$ pin is unaffected by a missing clock detector reset.

10.5. Comparator0 Reset

Comparator0 can be configured as a reset source by writing a '1' to the C0RSEF flag (RSTSRC.5). Comparator0 should be enabled and allowed to settle prior to writing to C0RSEF to prevent any turn-on chatter on the output from generating an unwanted reset. The Comparator0 reset is active-low: if the non-inverting input voltage (on CP0+) is less than the inverting input voltage (on CP0–), the device is put into the reset state. After a Comparator0 reset, the C0RSEF flag (RSTSRC.5) will read '1' signifying Comparator0 as the reset source; otherwise, this bit reads '0'. The state of the $\overline{RST}$ pin is unaffected by a Comparator0 reset.

10.6. PCA Watchdog Timer Reset

The programmable Watchdog Timer (WDT) function of the Programmable Counter Array (PCA) can be used to prevent software from running out of control during a system malfunction. The PCA WDT function can be enabled or disabled by software as described in **Section "17.3. Watchdog Timer Mode" on page 151**; the WDT is enabled and clocked by SYSCLK / 12 following any reset. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit (RSTSRC.5) is set to '1'. The state of the $\overline{RST}$ pin is unaffected by a WDT reset.

10.7. OTP Error Reset

If an OTP program read or a write procedure targets an illegal address, a system reset is generated. This may occur due to any of the following:

- Programming hardware attempts to write or read an OTP location which is above the user code space address limit.
- An OTP read from firmware is attempted above user code space. This occurs when a MOVC operation is attempted above the user code space address limit.
- A Program read is attempted above user code space. This occurs when user code attempts to branch to an address above the user code space address limit.

Table 10.1. User Code Space Address Limits

Device	User Code Space Address Limit
C8051T600/1	0x1DFF
C8051T602/3	0x0FFF
C8051T604/5	0x07FF

The OTPERR bit (RSTSRC.6) is set following any OTP error reset. The state of the $\overline{\text{RST}}$ pin is unaffected by an OTP error reset.

10.8. Software Reset

Software may force a reset by writing a ‘1’ to the SWRSF bit (RSTSRC.4). The SWRSF bit will read ‘1’ following a software forced reset. The state of the $\overline{\text{RST}}$ pin is unaffected by a software reset.

SFR Definition 10.1. RSTSRC: Reset Source

R	R	R/W	R/W	R	R/W	R/W	R	Reset Value
-	OTPERR	C0RSEF	SWRSF	WDTRSF	MCDRSF	PORSF	PINRSF	Variable
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xEF

Note: For bits that act as both reset source enables (on a write) and reset indicator flags (on a read), read-modify-write instructions read and modify the source enable only.

Bit7: UNUSED. Read = 0. Write = don’t care.
Bit6: OTPERR: OTP Error Indicator.
0: Source of last reset was not an OTP read error.
1: Source of last reset was an OTP read error.
Bit5: C0RSEF: Comparator0 Reset Enable and Flag.
Write
0: Comparator0 is not a reset source.
1: Comparator0 is a reset source (active-low).
Read
0: Source of last reset was not Comparator0.
1: Source of last reset was Comparator0.
Bit4: SWRSF: Software Reset Force and Flag.
Write
0: No Effect.
1: Forces a system reset.
Read
0: Source of last reset was not a write to the SWRSF bit.
1: Source of last was a write to the SWRSF bit.
Bit3: WDTRSF: Watchdog Timer Reset Flag.
0: Source of last reset was not a WDT timeout.
1: Source of last reset was a WDT timeout.
Bit2: MCDRSF: Missing Clock Detector Flag.
Write:
0: Missing Clock Detector disabled.
1: Missing Clock Detector enabled; triggers a reset if a missing clock condition is detected.
Read:
0: Source of last reset was not a Missing Clock Detector timeout.
1: Source of last reset was a Missing Clock Detector timeout.
Bit1: PORSF: Power-On Reset Force and Flag.
This bit is set anytime a power-on reset occurs. This may be due to a true power-on reset or a V_{DD} monitor reset. In either case, data memory should be considered indeterminate following the reset. Writing this bit enables/disables the V_{DD} monitor.
Write:
0: V_{DD} monitor disabled.
1: V_{DD} monitor enabled.
Read:
0: Last reset was not a power-on or V_{DD} monitor reset.
1: Last reset was a power-on or V_{DD} monitor reset; all other reset flags indeterminate.
Bit0: PINRSF: HW Pin Reset Flag.
0: Source of last reset was not $\overline{\text{RST}}$ pin.
1: Source of last reset was $\overline{\text{RST}}$ pin.

Table 10.2. Reset Electrical Characteristics

–40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
$\overline{\text{RST}}$ Output Low Voltage	I_{OL} = 8.5 mA, V_{DD} = 1.8 to 3.6 V	—	—	0.6	V
$\overline{\text{RST}}$ Input High Voltage		0.7 x V_{DD}	—		V
$\overline{\text{RST}}$ Input Low Voltage		—	—	0.3 x V_{DD}	
$\overline{\text{RST}}$ Input Leakage Current	$\overline{\text{RST}}$ = 0.0 V	—	25	40	µA
V_{DD} Ramp Time for POR	Ramp from 0 to 1.8 V	—	—	1	ms
V_{DD} Monitor Threshold (V_{RST})		TBD	TBD	TBD	V
Missing Clock Detector Timeout	Time from last system clock rising edge to reset initiation	300	450	600	µs
Reset Time Delay	Delay between release of any reset source and code execution at location 0x0000	—	—	60	µs
Minimum $\overline{\text{RST}}$ Low Time to Generate a System Reset		15	—	—	µs

11. One-Time Programmable Read-Only Memory

C8051T600/1/2/3/4/5 devices include 8 kB (C8051T600/1), 4 kB (C8051T602/3), or 2 kB (C8051T604/5) of on-chip One Time Programmable (OTP) EPROM for program code storage. The EPROM memory can be programmed via the C2 debug and programming interface when a special programming voltage is applied to the V_{PP} pin. Table 11.1 shows the EPROM specifications.

Table 11.1. EPROM Electrical Characteristics

Parameter	Conditions	Min	Typ	Max	Units
OTP EPROM Size	C8051T600/1	8192*	—	—	Bytes
OTP EPROM Size	C8051T602/3	4096	—	—	Bytes
OTP EPROM Size	C8051T604/5	2048	—	—	Bytes
Write Cycle Time (per Byte)		—	100	—	µs
Programming Voltage (V_{PP})		6.25	6.5	6.75	V
***Note:** 512 bytes at location 0x1E00 to 0x1FFF are not available for program storage					

11.1. Programming the EPROM Memory

Programming of the OTP EPROM memory is accomplished through the C2 programming and debug interface. When creating hardware to program the EPROM, it is necessary to follow the programming steps listed below. Please refer to the “C2 Interface Specification” available at http://www.silabs.com for details on communicating via the C2 interface. **Section “19. C2 Interface” on page 161** has information about C2 register addresses for the C8051T600/1/2/3/4/5.

1. **Reset the device using the /RST pin**.
2. **Wait at least 20 µs** before sending the first C2 command.
3. Place the device in core reset: **Write 0x04 to the DEVCTL register**.
4. Set the device to program mode (1st step): **Write 0x40 to the EPCTL register**.
5. Set the device to program mode (2nd step): **Write 0x58 to the EPCTL register**.
6. **Apply the V_{PP} programming Voltage**.
7. **Write the first EPROM address for programming to EPADDRH and EPADDRL**.
8. **Write a data byte to EPDAT**. EPADDRH:L will increment by 1 after this write.
9. **Poll the OTPBusy bit** using a C2 Address Read command. Note: If OTPError is set at this time, the write operation failed.
10. If programming is not finished, return to Step 8 to write the next address in sequence, or return to Step 7 to program a new address.
11. Remove program mode (1st step): **Write 0x40 to the EPCTL register**.
12. **Remove the V_{PP} programming Voltage**.
13. Remove program mode (2nd step): **Write 0x00 to the EPCTL register**.
14. Reset the device: **Write 0x02 and then 0x00 to the DEVCTL register**.

Important Note: There is a finite amount of time which V_{PP} can be applied without damaging the device, which is cumulative over the life of the device. Refer to Table 2.1 on page 23 for the V_{PP} timing specification.

11.2. Security Options

The C8051T600/1/2/3/4/5 devices provide security options to prevent unauthorized viewing of proprietary program code and constants. A security byte stored at location 0x1FFF in EPROM address space can be used to lock the program memory from being read or written across the C2 interface. The lock byte can always be read regardless of the security settings. Table 11.2 shows the security byte decoding. See Figure 11.1 for the program memory map and security byte locations for each device.

Important Note: Once the security byte has been written, there are no means of unlocking the device. Locking memory from write access should be performed only after all other code has been successfully programmed to memory.

Table 11.2. Security Byte Decoding

Bits	Description
7–4	**Write Lock:** Clearing any of these bits to logic 0 prevents all code memory from being written across the C2 interface.
3–0	**Read Lock:** Clearing any of these bits to logic 0 prevents all code memory from being read across the C2 interface.

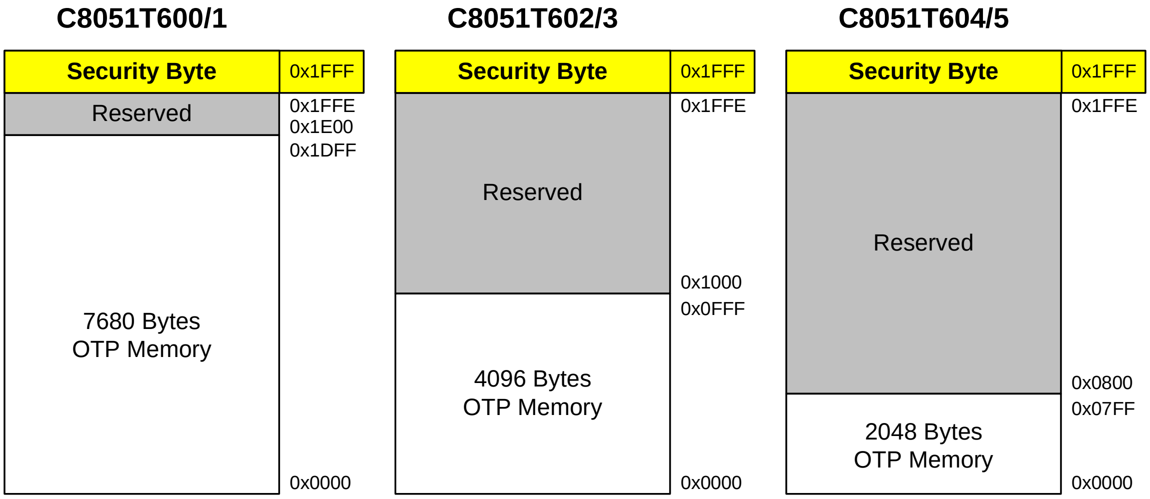


Figure 11.1. OTP EPROM Program Memory Map

11.3. Program Memory CRC

A CRC engine is included on-chip which provides a means of verifying EPROM contents once the device has been programmed. The CRC engine is available for EPROM verification even if the device is fully read and write locked, allowing for verification of code contents at any time.

The CRC engine is operated through the C2 debug and programming interface, and performs 16-bit CRCs on individual 256-Byte blocks of program memory, or a 32-bit CRC the entire memory space. To prevent hacking and extrapolation of security-locked source code, the CRC engine will only allow CRCs to be performed on contiguous 256-Byte blocks beginning on 256-Byte boundaries (lowest 8-bits of address are 0x00). For example, the CRC engine can perform a CRC for locations 0x0400 through 0x04FF, but it cannot perform a CRC for locations 0x0401 through 0x0500, or on block sizes smaller or larger than 256 Bytes.

11.3.1. Performing 32-bit CRCs on Full EPROM Content

A 32-bit CRC on the enter EPROM space is initiated by writing to the CRC1 byte over the C2 interface. The CRC calculation begins at address 0x0000, and ends at the end of user EPROM space. The OTPBusy bit in register C2ADD will be set during the CRC operation, and cleared once the operation is complete. The 32-bit results will be available in the CRC3-0 registers. CRC3 is the MSB, and CRC0 is the LSB. The polynomial used for the 32-bit CRC calculation is 0x04C11DB7. Note: If a 16-bit CRC has been performed since the last device reset, a device reset should be initiated before performing a 32-bit CRC operation.

11.3.2. Performing 16-bit CRCs on 256-Byte EPROM Blocks

A 16-bit CRC of individual 256-byte blocks of EPROM can be initiated by writing to the CRC0 byte over the C2 interface. The value written to CRC0 is the high byte of the beginning address for the CRC. For example, if CRC0 is written to 0x02, the CRC will be performed on the 256-bytes beginning at address 0x0200, and ending at address 0x2FF. The OTPBusy bit in register C2ADD will be set during the CRC operation, and cleared once the operation is complete. The 16-bit results will be available in the CRC1-0 registers. CRC1 is the MSB, and CRC0 is the LSB. The polynomial for the 16-bit CRC calculation is 0x1021.

NOTES:

12. Oscillators

All C8051T600/1/2/3/4/5 devices include a calibrated, precision internal oscillator and an external clock input circuit. The external clock input circuitry can be configured to operate from a CMOS clock, an external capacitor, or an external RC circuit. The internal oscillator can be enabled/disabled and adjusted using the OSCICN and OSCICL registers, as shown in Figure 12.1. The system clock can be sourced by the external clock input circuit, the internal oscillator, or a scaled version of the internal oscillator. The internal oscillator's electrical specifications are given in Table 12.1 on page 92.

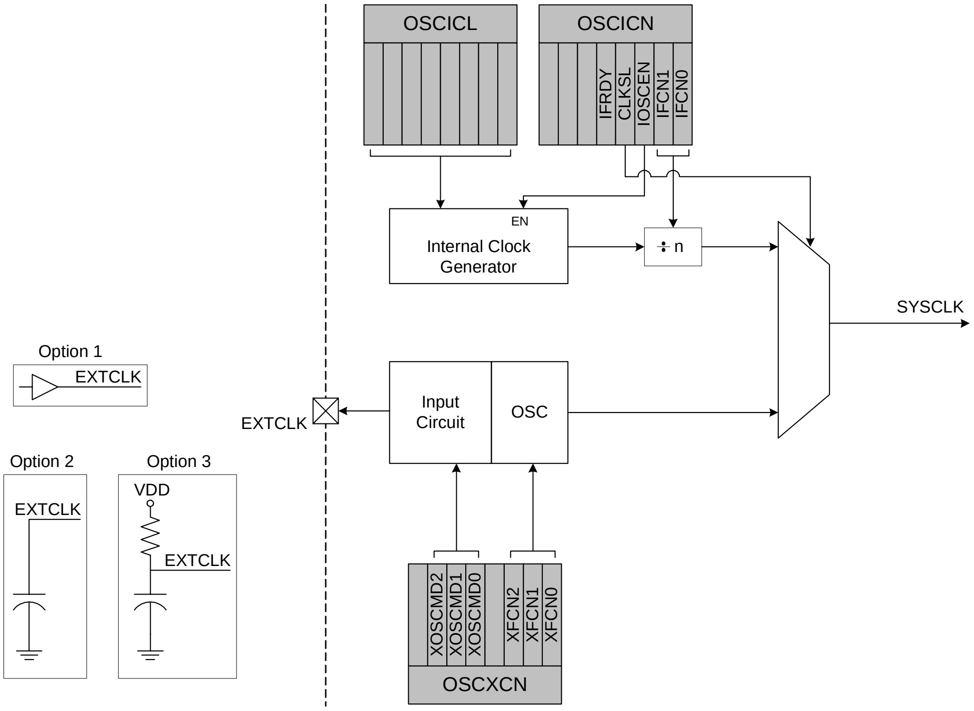


Figure 12.1. Oscillator Diagram

12.1. Calibrated Internal Oscillator

All C8051T600/1/2/3/4/5 devices include a calibrated internal oscillator that defaults as the system clock after a system reset. The oscillator is factory calibrated to obtain a 24.5 MHz frequency at room temperature, using the OSCICL register.

The controller’s core clock (SYSCLK) may be derived from the 24.5 MHz internal oscillator divided by 1, 2, 4, or 8, as defined by the IFCN bits in register OSCICN. The divide value defaults to 8 following a reset, producing a 3.0625 MHz system clock.

SFR Definition 12.1. OSCICL: Internal Oscillator Calibration

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-								Variable
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xB3

Bit7: UNUSED. Read = 0. Write = don’t care.

Bits6–0: OSCICL: Internal Oscillator Calibration Register.
This register adjusts the internal oscillator period. The reset value for OSCICL defines the internal oscillator base frequency. The reset value of this register is factory calibrated to generate an internal oscillator frequency of 24.5 MHz.

SFR Definition 12.2. OSCICN: Internal Oscillator Control

R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	Reset Value
-	-	-	IFRDY	CLKSL	IOSCEN	IFCN1	IFCN0	00010100
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
								0xB2

Bits7–5: UNUSED. Read = 000b, Write = don't care.

Bit4: IFRDY: Internal Oscillator Frequency Ready Flag.
0: Internal Oscillator is not running at programmed frequency.
1: Internal Oscillator is running at programmed frequency.

Bit3: CLKSL: System Clock Source Select Bit.
0: SYSCLK derived from the Internal Oscillator, and scaled as per the IFCN bits.
1: SYSCLK derived from the External Clock circuit.

Bit2: IOSCEN: Internal Oscillator Enable Bit.
0: Internal Oscillator Disabled.
1: Internal Oscillator Enabled.

Bits1–0: IFCN1–0: Internal Oscillator Frequency Control Bits.
00: SYSCLK derived from Internal Oscillator divided by 8.
01: SYSCLK derived from Internal Oscillator divided by 4.
10: SYSCLK derived from Internal Oscillator divided by 2.
11: SYSCLK derived from Internal Oscillator divided by 1.

Table 12.1. Internal Oscillator Electrical Characteristics

–40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Calibrated Internal Oscillator Frequency		TBD	24.5	TBD	MHz
Internal Oscillator Supply Current (from V_{DD})	OSCICN.2 = 1	—	TBD	—	µA

12.2. External Oscillator Circuit

The external oscillator circuit can accept an external CMOS clock, or operate from an external capacitor or RC circuit applied to the EXTCLK pin (P0.3). To operate in external CMOS mode, the EXTCLK pin should be configured as a digital input. For capacitor or RC mode, the EXTCLK pin should be configured as an analog input. See **Section "13.2. Port I/O Initialization" on page 99** for details on Port input mode selection. Whenever an external clock option is used, the Port I/O Crossbar should be configured to skip the EXTCLK pin. See **Section "13.1. Priority Crossbar Decoder" on page 98** for Crossbar configuration details. Note that the external oscillator control settings should not be changed while running the processor from an external oscillator source.

12.3. System Clock Selection

The CLKSL bit in register OSCICN selects which oscillator is used as the system clock. CLKSL must be set to '1' for the system clock to run from an external clock source; however, the external clock may still clock peripherals (timers, PCA) when the internal oscillator is selected as the system clock. The system clock may be switched on-the-fly between the internal and external clock sources, so long as the selected oscillator is enabled and has settled. The internal oscillator requires very little start-up time and may be enabled and selected as the system clock in the same write to OSCICN. When switching between internal and external clock sources, the hand-off to the other clock source lasts two clock cycles of the slower clock.

SFR Definition 12.3. OSCXCN: External Oscillator Control

R	R/W	R/W	R/W	R	R/W	R/W	R/W	Reset Value
-	XOSCMD2	XOSCMD1	XOSCMD0	-	XFCN2	XFCN1	XFCN0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xB1

Bit7: Unused. Read = 0, Write = don’t care.
Bits6–4: XOSCMD2–0: External Oscillator Mode Bits.
00x: External Oscillator circuit off.
010: External CMOS Clock Mode.
011: External CMOS Clock Mode with divide by 2 stage.
100 RC Oscillator Mode with divide by 2 stage.
101: Capacitor Oscillator Mode with divide by 2 stage.
11x: Reserved.
Bit3: Unused. Read = 0, Write = don't care.
Bits2–0: XFCN2–0: External Oscillator Frequency Control Bits.
000–111: See table below:

XFCN	K Factor (Capacitor Mode)	Approximate Frequency Range
000	0.87	f < 25 kHz
001	2.6	25 kHz < f < 50 kHz
010	7.7	50 kHz < f < 100 kHz
011	22	100 kHz < f < 200 kHz
100	65	200 kHz < f < 400 kHz
101	180	400 kHz < f < 800 kHz
110	664	800 kHz < f < 1.6 MHz
111	1590	1.6 MHz < f < 3.2 MHz

Capacitor Mode (Circuit from Figure 12.1, Option 2, XOSCMD = 101)
Choose K Factor (KF) matching the oscillation frequency desired:

$$f = \frac{\mathrm{KF}}{\mathrm{C} \times \mathrm{V_{DD}}}$$

where
f = frequency of oscillation in MHz
C = capacitor value on the EXTCLK pin in pF
V_{DD} = Power Supply on MCU in volts

RC Mode (Circuit from Figure 12.1, Option 3, XOSCMD = 100)
Choose XFCN value to match desired frequency range:

$$f = \frac{1.23 \times 10^{23}}{\mathrm{R} \times \mathrm{C}}$$

where
f = frequency of oscillation in MHz
C = capacitor value on the EXTCLK pin in pF
R = pullup resistor value in kΩ

12.4. External Capacitor Example

If a capacitor is used as an external oscillator for the MCU, the circuit should be configured as shown in Figure 12.1, Option 2. The capacitor should be no greater than 100 pF; however, for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, select the capacitor to be used and find the frequency of oscillation from the equations below. Assume V_{DD} = 3.0 V and f = 150 kHz:

$$f = \frac{KF}{C \times V_{DD}}$$

$$0.150\ \text{MHz} = \frac{KF}{C \times 3.0}$$

Since a frequency of roughly 150 kHz is desired, select the K Factor from SFR Definition 12.3 as KF = 22:

$$0.150\ \text{MHz} = \frac{22}{C \times 3.0\ \text{V}}$$

$$C = \frac{22}{0.150\ \text{MHz} \times 3.0\ \text{V}}$$

$$C = 48.8\ \text{pF}$$

Therefore, the XFCN value to use in this example is 011b and C is approximately 50 pF.

12.5. External RC Example

If an RC network is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 12.1, Option 3. The capacitor should be no greater than 100 pF; however for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, first select the RC network value to produce the desired frequency of oscillation. If the frequency desired is 100 kHz, let R = 246 k and C = 50 pF:

$$f = \frac{1.23 \times 10^{3}}{R \times C} = \frac{1.23 \times 10^{3}}{246 \times 50} = 100\ \text{kHz}$$

Referencing the table in SFR Definition 12.3, the required XFCN setting is 010b.

NOTES:

13. Port Input/Output

Digital and analog resources are available through a byte-wide digital I/O Port, Port0. Each of the Port pins can be defined as general-purpose I/O (GPIO), analog input, or assigned to one of the internal digital resources as shown in Figure 13.3. The designer has complete control over which functions are assigned, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a Priority Crossbar Decoder. The state of a Port I/O pin can always be read in the corresponding Port latch, regardless of the Crossbar settings.

The Crossbar assigns the selected internal digital resources to the I/O pins based on the Priority Decoder (Figure 13.3 and Figure 13.4). The registers XBR0, XBR1, and XBR2, defined in SFR Definition 13.1, SFR Definition 13.2, and SFR Definition 13.3 are used to select internal digital functions.

All Port I/Os are 5 V tolerant (refer to Figure 13.2 for the Port cell circuit). The Port I/O cells are configured as either push-pull or open-drain in the Port0 Output Mode register (P0MDOUT). Complete Electrical Specifications for Port I/O are given in Table 13.1 on page 104.

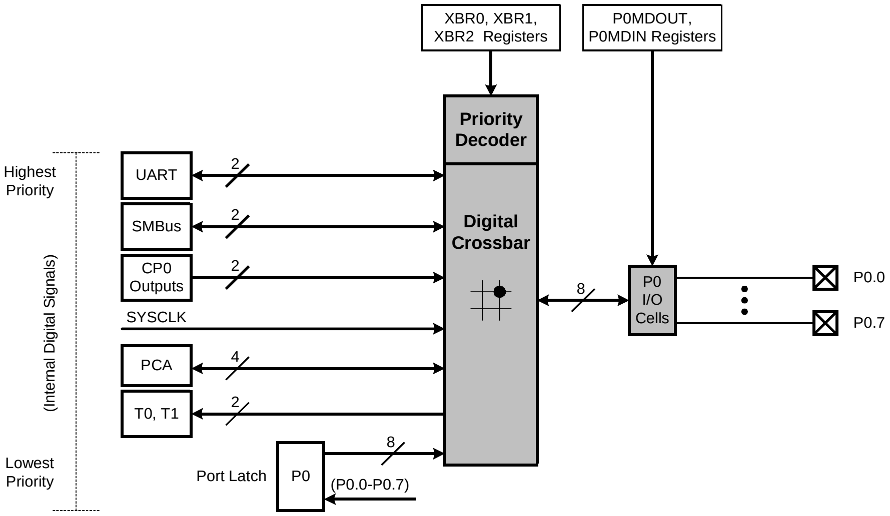


Figure 13.1. Port I/O Functional Block Diagram

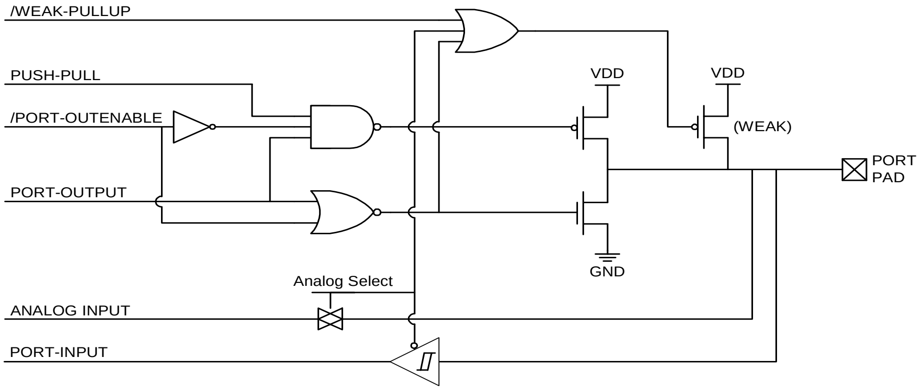
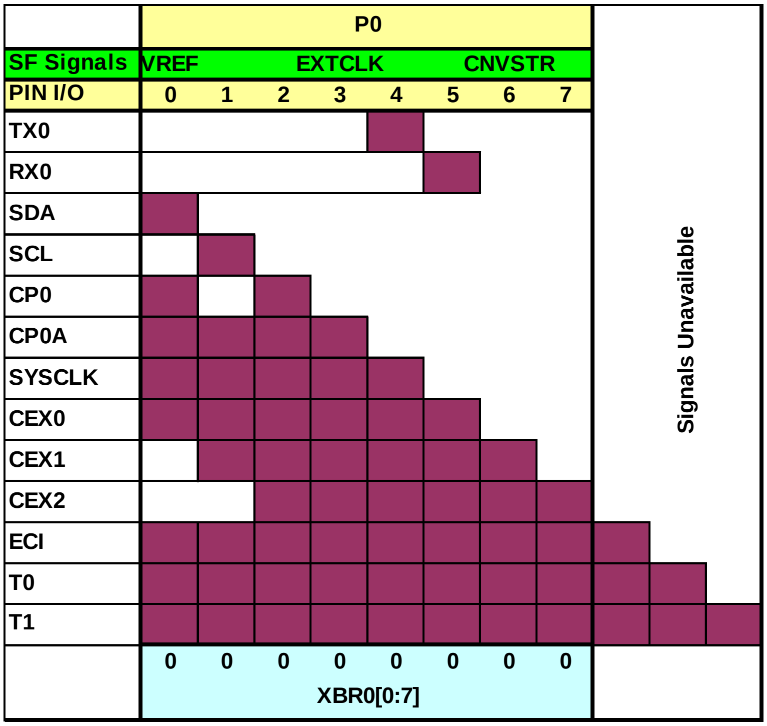


Figure 13.2. Port I/O Cell Block Diagram

13.1. Priority Crossbar Decoder

The Priority Crossbar Decoder (Figure 13.3) assigns a priority to each I/O function, starting at the top with UART0. When a digital resource is selected, the least-significant unassigned Port pin is assigned to that resource (excluding UART0, which is always at pins 4 and 5). If a Port pin is assigned, the Crossbar skips that pin when assigning the next selected resource. Additionally, the Crossbar will skip Port pins whose associated bits in the XBR0 register are set. The XBR0 register allows software to skip Port pins that are to be used for analog input or GPIO.

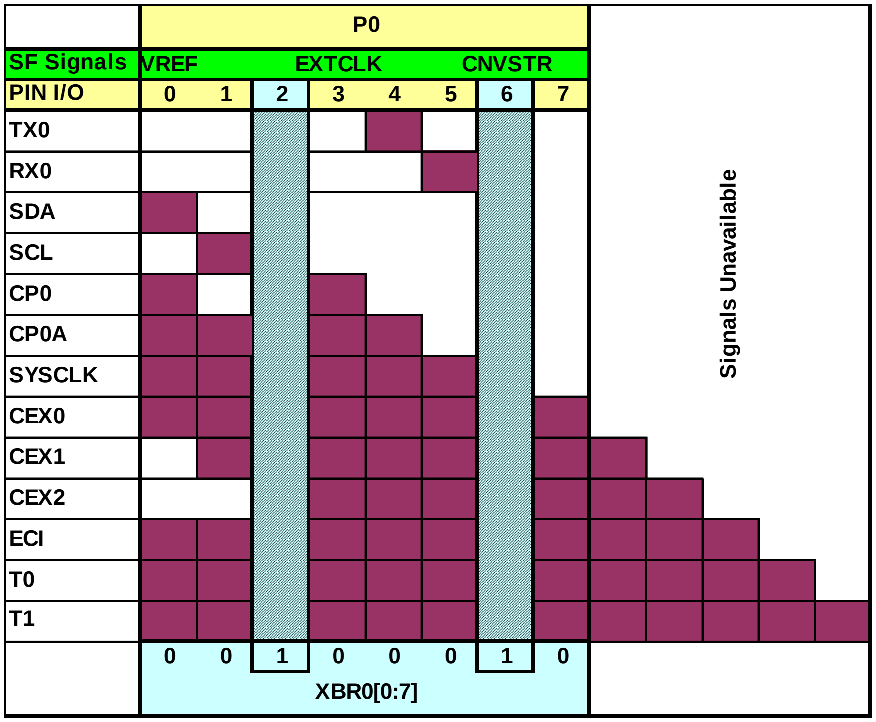
Important Note on Crossbar Configuration: If a Port pin is claimed by a peripheral without use of the Crossbar, its corresponding XBR0 bit should be set. This applies to P0.0 if V_{REF} is enabled, P0.3 if the external oscillator circuit is enabled, P0.6 if the ADC is configured to use the external conversion start signal (CNVSTR), and any selected ADC or Comparator inputs. The Crossbar skips selected pins as if they were already assigned, and moves to the next unassigned pin. Figure 13.3 shows the Crossbar Decoder priority with no Port pins skipped (XBR0 = 0x00); Figure 13.4 shows the Crossbar Decoder priority with pins 6 and 2 skipped (XBR0 = 0x44).



	P0								Signals Unavailable		
SF Signals	VREF			EXTCLK			CNVSTR				
PIN I/O	0	1	2	3	4	5	6	7			
TX0					▨						
RX0						▨					
SDA	▨										
SCL		▨									
CP0	▨		▨								
CP0A	▨	▨	▨	▨							
SYSCLK	▨	▨	▨	▨	▨						
CEX0	▨	▨	▨	▨	▨	▨					
CEX1		▨	▨	▨	▨	▨	▨				
CEX2			▨	▨	▨	▨	▨	▨			
ECI	▨	▨	▨	▨	▨	▨	▨	▨	▨		
T0	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	
T1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	0	0	0	0	0	0	0	0			
	XBR0[0:7]										

▨	Port pin potentially available to peripheral
SF Signals	Special Function Signals are not assigned by the crossbar. When these signals are enabled, the CrossBar must be manually configured to skip their corresponding port pins.

Figure 13.3. Crossbar Priority Decoder with XBR0 = 0x00



	P0												
SF Signals	VREF			EXTCLK			CNVSTR						
PIN I/O	0	1	2	3	4	5	6	7					
TX0			▨		■		▨						
RX0			▨			■	▨						
SDA	■		▨				▨						
SCL		■	▨				▨						
CP0	■		▨	■			▨						
CP0A	■	■	▨	■	■		▨						
SYSCLK	■	■	▨	■	■	■	▨						
CEX0	■	■	▨	■	■	■	▨	■					
CEX1		■	▨	■	■	■	▨	■	■				
CEX2			▨	■	■	■	▨	■	■	■			
ECI	■	■	▨	■	■	■	▨	■	■	■	■		
T0	■	■	▨	■	■	■	▨	■	■	■	■	■	
T1	■	■	▨	■	■	■	▨	■	■	■	■	■	■
	0	0	1	0	0	0	1	0					
	XBR0[0:7]												

■ Port pin potentially available to peripheral

▨ Port pin skipped by CrossBar

SF Signals Special Function Signals are not assigned by the crossbar. When these signals are enabled, the CrossBar must be manually configured to skip their corresponding port pins.

Figure 13.4. Crossbar Priority Decoder with XBR0 = 0x44

Registers XBR1 and XBR2 are used to assign the digital I/O resources to the physical I/O Port pins. Note that when the SMBus is selected, the Crossbar assigns both pins associated with the SMBus (SDA and SCL). Either or both of the UART signals may be selected by the Crossbar. UART0 pin assignments are fixed for bootloading purposes: when UART TX0 is selected, it is always assigned to P0.4; when UART RX0 is selected, it is always assigned to P0.5. Standard Port I/Os appear contiguously after the prioritized functions have been assigned. For example, if assigned functions that take the first 3 Port I/O (P0.[2:0]), 5 Port I/O are left for analog or GPIO use.

13.2. Port I/O Initialization

Port I/O initialization consists of the following steps:

Step 1. Select the input mode (analog or digital) for all Port pins, using the Port0 Input Mode register (P0MDIN).
Step 2. Select the output mode (open-drain or push-pull) for all Port pins, using the Port0 Output Mode register (P0MDOUT).
Step 3. Set XBR0 to skip any pins selected as analog inputs or special functions.
Step 4. Assign Port pins to desired peripherals.
Step 5. Enable the Crossbar.

All Port pins must be configured as either analog or digital inputs. Any pins to be used as Comparator or ADC inputs should be configured as an analog inputs. When a pin is configured as an analog input, its weak pull-up, digital driver, and digital receiver is disabled. This process saves power and reduces noise on the analog input. Pins configured as digital inputs may still be used by analog peripherals; however this practice is not recommended.

Additionally, all analog input pins should be configured to be skipped by the Crossbar (accomplished by setting the associated bits in XBR0). Port input mode is set in the P0MDIN register, where a ‘1’ indicates a digital input, and a ‘0’ indicates an analog input. All pins default to digital inputs on reset. See SFR Definition 13.5 for the P0MDIN register details.

The output driver characteristics of the I/O pins are defined using the Port0 Output Mode register P0MDOUT (see SFR Definition 13.6). Each Port Output driver can be configured as either open drain or push-pull. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is the SMBus (SDA, SCL) pins, which are configured as open-drain regardless of the P0MDOUT settings. When the WEAKPUD bit in XBR2 is ‘0’, a weak pull-up is enabled for all Port I/O configured as open-drain. WEAKPUD does not affect the push-pull Port I/O. Furthermore, the weak pull-up is turned off on an open-drain output that is driving a ‘0’ to avoid unnecessary power dissipation.

Registers XBR0, XBR1 and XBR2 must be loaded with the appropriate values to select the digital I/O functions required by the design. Setting the XBARE bit in XBR2 to ‘1’ enables the Crossbar. Until the Crossbar is enabled, the external pins remain as standard digital inputs (output drivers disabled) regardless of the XBRn Register settings. For given XBRn Register settings, one can determine the I/O pin-out using the Priority Decode Table; as an alternative, the Configuration Wizard utility of the Silicon Labs IDE software will determine the Port I/O pin-assignments based on the XBRn Register settings.

SFR Definition 13.1. XBR0: Port I/O Crossbar Register 0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	XSKP6	XSKP5	XSKP4	XSKP3	XSKP2	XSKP1	XSKP0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xE1

Bit7: UNUSED. Read = 0b; Write = don’t care.

Bits6–0: XSKP[6:0]: Crossbar Skip Enable Bits
These bits select Port pins to be skipped by the Crossbar Decoder. Port pins used as analog inputs (for ADC or Comparator) or used as special functions (V_{REF} input, external oscillator circuit, CNVSTR input) should be skipped by the Crossbar.
0: Corresponding P0.n pin is not skipped by the Crossbar.
1: Corresponding P0.n pin is skipped by the Crossbar.

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
PCA0ME		CP0AOEN	CP0OEN	SYSCKE	SMB0OEN	URX0EN	UTX0EN	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xE2

Bits7–6: PCA0ME: PCA Module I/0 Enable Bits
00: All PCA I/O unavailable at Port pins.
01: CEX0 routed to Port pin.
10: CEX0, CEX1 routed to Port pins.
11: CEX0, CEX1, CEX2 routed to Port pins.
Bit5: CP0AOEN: Comparator0 Asynchronous Output Enable
0: Asynchronous CP0 unavailable at Port pin.
1: Asynchronous CP0 routed to Port pin.
Bit4: CP0OEN: Comparator0 Output Enable
0: CP0 unavailable at Port pin.
1: CP0 routed to Port pin.
Bit3: SYSCKE: /SYSCLK Output Enable
0: /SYSCLK unavailable at Port pin.
1: /SYSCLK output routed to Port pin.
Bit2: SMB0OEN: SMBus I/O Enable
0: SMBus I/O unavailable at Port pins.
1: SDA, SCL routed to Port pins.
Bit1: URX0EN: UART RX Enable
0: UART RX0 unavailable at Port pin.
1: UART RX0 routed to Port pin P0.5.
Bit0: UTX0EN: UART TX Output Enable
0: UART TX0 unavailable at Port pin.
1: UART TX0 routed to Port pin P0.4.

SFR Definition 13.3. XBR2: Port I/O Crossbar Register 2

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
WEAKPUD	XBARE	-	-	-	T1E	T0E	ECIE	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xE3

Bit7: WEAKPUD: Port I/O Weak Pull-up Disable.
0: Weak Pull-ups enabled (except for Ports whose I/O are configured as push-pull).
1: Weak Pull-ups disabled.
Bit6: XBARE: Crossbar Enable.
0: Crossbar disabled.
1: Crossbar enabled.
Bits5–3: UNUSED: Read=000b. Write = don’t care.
Bit2: T1E: T1 Enable.
0: T1 unavailable at Port pin.
1: T1 routed to Port pin.
Bit1: T0E: T0 Enable.
0: T0 unavailable at Port pin.
1: T0 routed to Port pin.
Bit0: ECIE: PCA0 Counter Input Enable.
0: ECI unavailable at Port pin.
1: ECI routed to Port pin.

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. Port0 is accessed through a corresponding special function register (SFR) that is 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 13.4. P0: Port0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	11111111
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0x80

Bits7–0: P0.[7:0]
Write - Output appears on I/O pins per XBR0, XBR1, and XBR2 Registers
0: Logic Low Output.
1: Logic High Output (open-drain if corresponding P0MDOUT.n bit = 0)
Read - Always reads ‘1’ if selected as analog input in register P0MDIN. Directly reads Port pin when configured as digital input.
0: P0.n pin is logic low.
1: P0.n pin is logic high.

SFR Definition 13.5. P0MDIN: Port0 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:
								0xF1

Bits7–0: Input Configuration Bits for P0.7–P0.0 (respectively)
Port pins configured as analog inputs have their weak pull-up, digital driver, and digital receiver disabled.
0: Corresponding P0.n pin is configured as an analog input.
1: Corresponding P0.n pin is configured as a digital input.

SFR Definition 13.6. P0MDOUT: Port0 Output Mode

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:
								0xA4

Bits7–0: Output Configuration Bits for P0.7–P0.0 (respectively): ignored if corresponding bit in register P0MDIN is logic 0.
0: Corresponding P0.n Output is open-drain.
1: Corresponding P0.n Output is push-pull.

(Note: When SDA and SCL appear on any of the Port I/O, each are open-drain regardless of the value of P0MDOUT).

Table 13.1. Port I/O DC Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

Parameters	Conditions	Min	Typ	Max	Units
Output High Voltage	I_{OH} = –3 mA, Port I/O push-pull I_{OH} = –10 µA, Port I/O push-pull I_{OH} = –10 mA, Port I/O push-pull	0.8 x V_{DD} V_{DD} – 0.1 —	— — 0.7 x V_{DD}	— — —	V
Output Low Voltage	I_{OL} = 8.5 mA I_{OL} = 10 µA I_{OL} = 25 mA	— — —	— — 0.4 x V_{DD}	0.6 0.1 —	V
Input High Voltage		0.7 x V_{DD}	—	—	V
Input Low Voltage		—	—	0.3 x V_{DD}	V
Input Leakage Current	Weak Pull-up Off Weak Pull-up On, V_{IN} = 0 V	— —	— 25	±1 40	µA

14. SMBus

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I^2C serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/10th of the system clock operating as master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. Three SFRs are associated with the SMBus: SMB0CF configures the SMBus; SMB0CN controls the status of the SMBus; and SMB0DAT is the data register, used for both transmitting and receiving SMBus data and slave addresses.

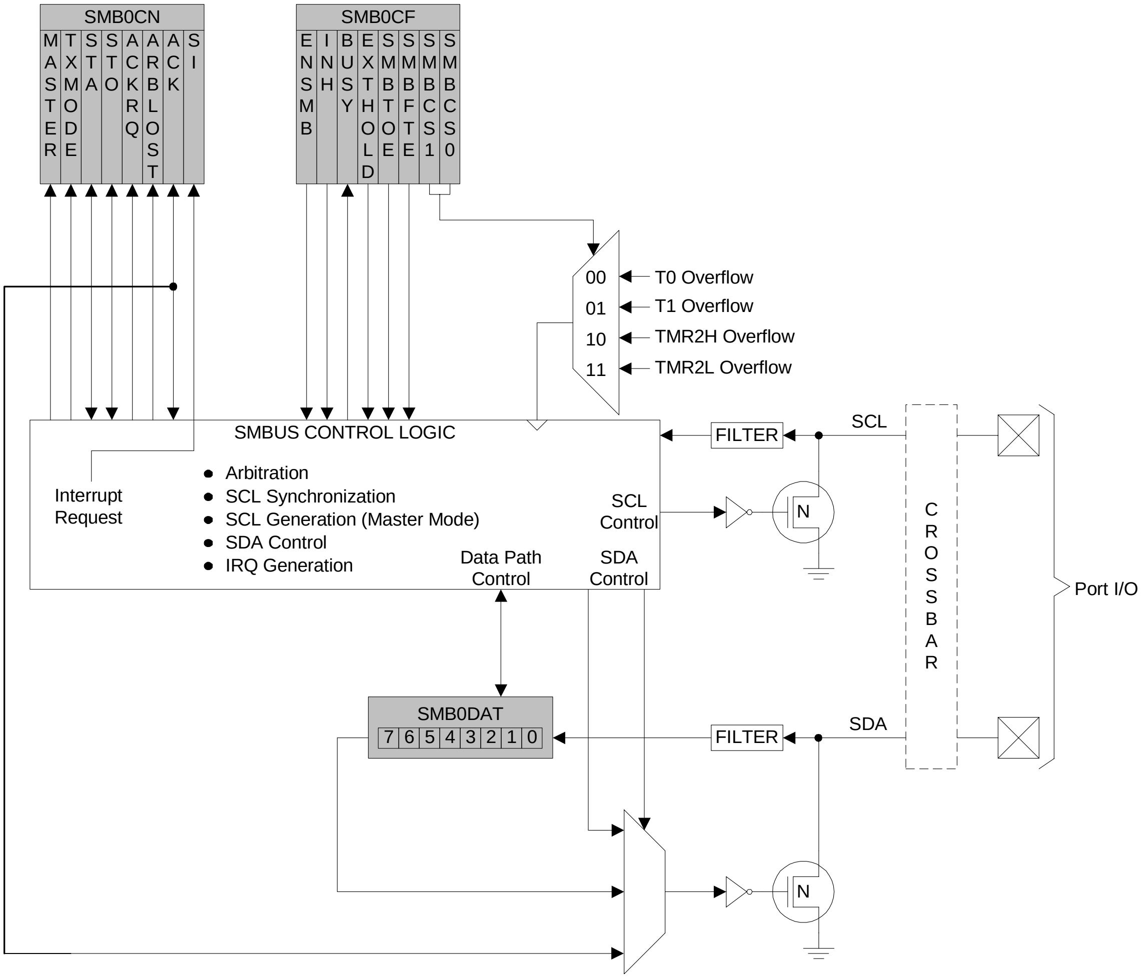


Figure 14.1. SMBus Block Diagram

14.1. Supporting Documents

It is assumed the reader is familiar with or has access to the following supporting documents:

1. The I^{2}C-Bus and How to Use It (including specifications), Philips Semiconductor.
2. The I^{2}C-Bus Specification—Version 2.0, Philips Semiconductor.
3. System Management Bus Specification—Version 1.1, SBS Implementers Forum.

14.2. SMBus Configuration

Figure 14.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 V and 5.0 V; different devices on the bus may operate at different voltage levels. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

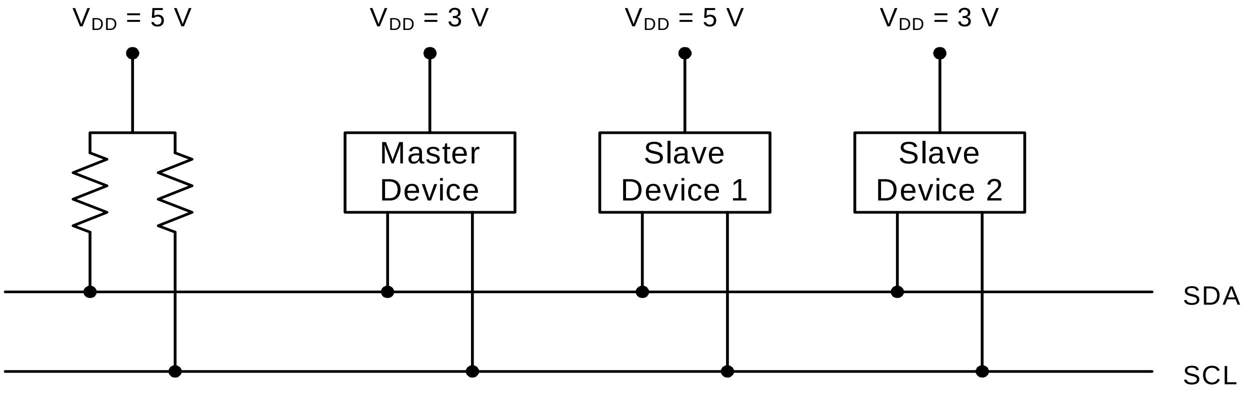


Figure 14.2. Typical SMBus Configuration

14.3. SMBus Operation

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. Note that it is not necessary to specify one device as the Master in a system; any device that transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Each byte that is received (by a master or slave) must be acknowledged (ACK) with a low SDA during a high SCL (see Figure 14.3). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address byte. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation.

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 14.3 illustrates a typical SMBus transaction.

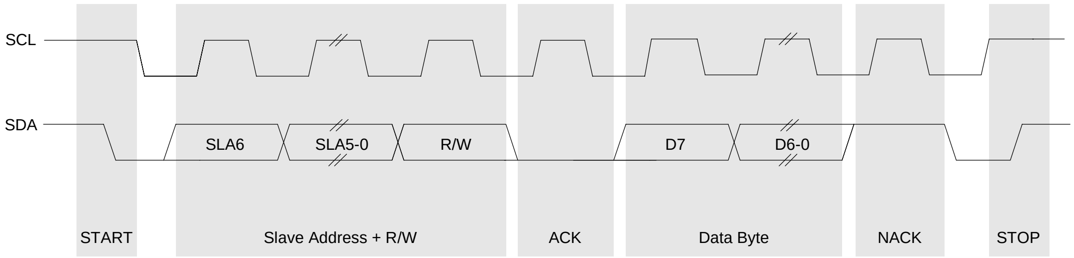


Figure 14.3. SMBus Transaction

14.3.1. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see **Section “14.3.4. SCL High (SMBus Free) Timeout” on page 108**). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

14.3.2. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I^2C, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

14.3.3. SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

When the SMBTOE bit in SMB0CF is set, Timer 2 is used to detect SCL low timeouts. Timer 2 is forced to reload when SCL is high, and allowed to count when SCL is low. With Timer 2 enabled and configured to overflow after 25 ms (and SMBTOE set), the Timer 2 interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout. Timer 2 configuration details can be found in **Section “16.2. Timer 2” on page 139**.

14.3.4. SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMBFTE bit in SMB0CF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods. If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. Note that a clock source is required for free timeout detection, even in a slave-only implementation.

14.4. Using the SMBus

The SMBus can operate in both Master and Slave modes. The interface provides timing and shifting control for serial transfers; higher level protocol is determined by user software. The SMBus interface provides the following application-independent features:

- Byte-wise serial data transfers
- Clock signal generation on SCL (Master Mode only) and SDA data synchronization
- Timeout/bus error recognition, as defined by the SMB0CF configuration register
- START/STOP timing, detection, and generation
- Bus arbitration
- Interrupt generation
- Status information

SMBus interrupts are generated for each data byte or slave address that is transferred. When transmitting, this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data, this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. See **Section “14.5. SMBus Transfer Modes” on page 117** for more details on transmission sequences.

Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN (SMBus Control register) to find the cause of the SMBus interrupt. The SMB0CN register is described in **Section "14.4.2. SMB0CN Control Register" on page 113**; Table 14.4 provides a quick SMB0CN decoding reference.

SMBus configuration options include:

- Timeout detection (SCL Low Timeout and/or Bus Free Timeout)
- SDA setup and hold time extensions
- Slave event enable/disable
- Clock source selection

These options are selected in the SMB0CF register, as described in **Section "14.4.1. SMBus Configuration Register" on page 110**.

14.4.1. SMBus Configuration Register

The SMBus Configuration register (SMB0CF) is used to enable the SMBus Master and/or Slave modes, select the SMBus clock source, and select the SMBus timing and timeout options. When the ENSMB bit is set, the SMBus is enabled for all master and slave events. Slave events may be disabled by setting the INH bit. With slave events inhibited, the SMBus interface will still monitor the SCL and SDA pins; however, the interface will NACK all received addresses and will not generate any slave interrupts. When the INH bit is set, all slave events will be inhibited following the next START (interrupts will continue for the duration of the current transfer).

Table 14.1. SMBus Clock Source Selection

SMBCS1	SMBCS0	SMBus Clock Source
0	0	Timer 0 Overflow
0	1	Timer 1 Overflow
1	0	Timer 2 High Byte Overflow
1	1	Timer 2 Low Byte Overflow

The SMBCS1–0 bits select the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 14.1. Note that the selected clock source may be shared by other peripherals so long as the timer is left running at all times. For example, Timer 1 overflows may generate the SMBus and UART baud rates simultaneously. Timer configuration is covered in **Section “16. Timers” on page 131**.

$$T_{HighMin} = T_{LowMin} = \frac{1}{f_{ClockSourceOverflow}}$$

Equation 14.1. Minimum SCL High and Low Times

The selected clock source should be configured to establish the minimum SCL High and Low times as per Equation 14.1. When the interface is operating as a master (and SCL is not driven or extended by any other devices on the bus), the typical SMBus bit rate is approximated by Equation 14.2.

$$BitRate = \frac{f_{ClockSourceOverflow}}{3}$$

Equation 14.2. Typical SMBus Bit Rate

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.

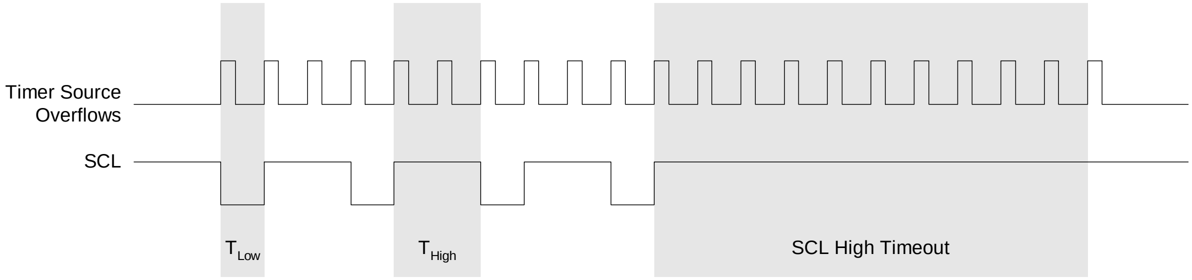


Figure 14.4. Typical SMBus SCL Generation

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 2 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 108**). The SMBus interface will force Timer 2 to reload while SCL is high, and allow Timer 2 to count when SCL is low. The Timer 2 interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus. Timer 2 configuration is described in **Section “16.2. Timer 2” on page 139**.

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).

SFR Definition 14.1. SMB0CF: SMBus Clock/Configuration

R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	Reset Value
ENSMB	INH	BUSY	EXTHOLD	SMBTOE	SMBFTE	SMBCS1	SMBCS0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xC1

Bit7: ENSMB: SMBus Enable.
This bit enables/disables the SMBus interface. When enabled, the interface constantly monitors the SDA and SCL pins.
0: SMBus interface disabled.
1: SMBus interface enabled.

Bit6: INH: SMBus Slave Inhibit.
When this bit is set to logic 1, the SMBus does not generate an interrupt when slave events occur. This effectively removes the SMBus slave from the bus. Master Mode interrupts are not affected.
0: SMBus Slave Mode enabled.
1: SMBus Slave Mode inhibited.

Bit5: BUSY: SMBus Busy Indicator.
This bit is set to logic 1 by hardware when a transfer is in progress. It is cleared to logic 0 when a STOP or free-timeout is sensed.

Bit4: EXTHOLD: SMBus Setup and Hold Time Extension Enable.
This bit controls the SDA setup and hold times according to Table 14.2.
0: SDA Extended Setup and Hold Times disabled.
1: SDA Extended Setup and Hold Times enabled.

Bit3: SMBTOE: SMBus SCL Timeout Detection Enable.
This bit enables SCL low timeout detection. If set to logic 1, the SMBus forces Timer 2 to reload while SCL is high and allows Timer 2 to count when SCL goes low. If Timer 2 is configured in split mode (T2SPLIT is set), only the high byte of Timer 2 is held in reload while SCL is high. Timer 2 should be programmed to generate interrupts at 25 ms, and the Timer 2 interrupt service routine should reset SMBus communication.

Bit2: SMBFTE: SMBus Free Timeout Detection Enable.
When this bit is set to logic 1, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods.

Bits1–0: SMBCS1–SMBCS0: SMBus Clock Source Selection.
These two bits select the SMBus clock source, which is used to generate the SMBus bit rate. The selected device should be configured according to Equation 14.1.

SMBCS1	SMBCS0	SMBus Clock Source
0	0	Timer 0 Overflow
0	1	Timer 1 Overflow
1	0	Timer 2 High Byte Overflow
1	1	Timer 2 Low Byte Overflow

14.4.2. SMB0CN Control Register

SMB0CN is used to control the interface and to provide status information (see SFR Definition 14.2). The higher four bits of SMB0CN (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER and TXMODE indicate the master/slave state and transmit/receive modes, respectively.

The STA bit indicates that a START has been detected or generated since the last SMBus interrupt. When set to ‘1’, the STA bit will cause the SMBus to enter Master mode and generate a START when the bus becomes free. STA is not cleared by hardware after the START is generated; it must be cleared by software.

As a master, writing the STO bit will cause the hardware to generate a STOP condition and end the current transfer after the next ACK cycle. STO is cleared by hardware after the STOP condition is generated. As a slave, STO indicates that a STOP condition has been detected since the last SMBus interrupt. STO is also used in slave mode to manage the transition from slave receiver to slave transmitter; see **Section 14.5.4** for details on this procedure.

If STO and STA are both set to ‘1’ (while in Master Mode), a STOP followed by a START will be generated.

As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. ACKRQ is set each time a byte is received, indicating that an outgoing ACK value is needed. When ACKRQ is set, software should write the desired outgoing value to the ACK bit before clearing SI. A NACK will be generated if software does not write the ACK bit before clearing SI. SDA will reflect the defined ACK value immediately following a write to the ACK bit; however SCL will remain low until SI is cleared. If a received slave address is not acknowledged, further slave events will be ignored until the next START is detected.

The ARBLOST bit indicates that the interface has lost an arbitration. This may occur anytime the interface is transmitting (master or slave). A lost arbitration while operating as a slave indicates a bus error condition. ARBLOST is cleared by hardware each time SI is cleared.

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost; see Table 14.3 for more details.
Important Note About the SI Bit: The SMBus interface is stalled while SI is set; thus SCL is held low, and the bus is stalled until software clears SI.

Table 14.3 lists all sources for hardware changes to the SMB0CN bits. Refer to Table 14.4 for SMBus status decoding using the SMB0CN register.

SFR Definition 14.2. SMB0CN: SMBus Control

R	R	R/W	R/W	R	R	R/W	R/W	Reset Value
MASTER	TXMODE	STA	STO	ACKRQ	ARBLOST	ACK	SI	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
						(bit addressable)		0xC0

Bit7: MASTER: SMBus Master/Slave Indicator.
This read-only bit indicates when the SMBus is operating as a master.
0: SMBus operating in Slave Mode.
1: SMBus operating in Master Mode.

Bit6: TXMODE: SMBus Transmit Mode Indicator.
This read-only bit indicates when the SMBus is operating as a transmitter.
0: SMBus in Receiver Mode.
1: SMBus in Transmitter Mode.

Bit5: STA: SMBus Start Flag.
Write:
0: No Start generated.
1: When operating as a master, a START condition is transmitted if the bus is free (If the bus is not free, the START is transmitted after a STOP is received or a free timeout is detected). If STA is set by software as an active Master, a repeated START will be generated after the next ACK cycle.
Read:
0: No Start or repeated Start detected.
1: Start or repeated Start detected.

Bit4: STO: SMBus Stop Flag.
Write:
As a master, setting this bit to ‘1’ causes a STOP condition to be transmitted after the next ACK cycle. STO is cleared to ‘0’ by hardware when the STOP is generated.
As a slave, software manages this bit when switching from Slave Receiver to Slave Transmitter mode. See **Section 14.5.4** for details.
Read:
0: No Stop condition detected.
1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).

Bit3: ACKRQ: SMBus Acknowledge Request.
This read-only bit is set to logic 1 when the SMBus has received a byte and needs the ACK bit to be written with the correct ACK response value.

Bit2: ARBLOST: SMBus Arbitration Lost Indicator.
This read-only bit is set to logic 1 when the SMBus loses arbitration while operating as a transmitter. A lost arbitration while a slave indicates a bus error condition.

Bit1: ACK: SMBus Acknowledge Flag.
This bit defines the out-going ACK level and records incoming ACK levels. It should be written each time a byte is received (when ACKRQ=1), or read after each byte is transmitted.
0: A "not acknowledge" has been received (if in Transmitter Mode) OR will be transmitted (if in Receiver Mode).
1: An "acknowledge" has been received (if in Transmitter Mode) OR will be transmitted (if in Receiver Mode).

Bit0: SI: SMBus Interrupt Flag.
This bit is set by hardware under the conditions listed in Table 14.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled.

Table 14.3. Sources for Hardware Changes to SMB0CN

Bit	Set by Hardware When:	Cleared by Hardware When:
MASTER	• A START is generated.	• A STOP is generated. • Arbitration is lost.
TXMODE	• START is generated. • The SMBus interface enters transmitter mode (after SMB0DAT is written before the start of an SMBus frame).	• A START is detected. • Arbitration is lost. • SMB0DAT is not written before the start of an SMBus frame.
STA	• A START followed by an address byte is received.	• Must be cleared by software.
STO	• A STOP is detected while addressed as a slave. • Arbitration is lost due to a detected STOP.	• A pending STOP is generated.
ACKRQ	• A byte has been received and an ACK response value is needed.	• After each ACK cycle.
ARBLOST	• A repeated START is detected as a MASTER when STA is low (unwanted repeated START). • SCL is sensed low while attempting to generate a STOP or repeated START condition. • SDA is sensed low while transmitting a ‘1’ (excluding ACK bits).	• Each time SI is cleared.
ACK	• The incoming ACK value is low (ACKNOWLEDGE).	• The incoming ACK value is high (NOT ACKNOWLEDGE).
SI	• A START has been generated. • Lost arbitration. • A byte has been transmitted and an ACK/NACK received. • A byte has been received. • A START or repeated START followed by a slave address + R/W has been received. • A STOP has been received.	• Must be cleared by software.

14.4.3. Data Register

The SMBus Data register SMB0DAT holds a byte of serial data to be transmitted or one that has just been received. Software may safely read or write to the data register when the SI flag is set. Software should not attempt to access the SMB0DAT register when the SMBus is enabled and the SI flag is cleared to logic 0, as the interface may be in the process of shifting a byte of data into or out of the register.

Data in SMB0DAT is always shifted out MSB first. After a byte has been received, the first bit of received data is located at the MSB of SMB0DAT. While data is being shifted out, data on the bus is simultaneously being shifted in. SMB0DAT always contains the last data byte present on the bus. In the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data or address in SMB0DAT.

SFR Definition 14.3. SMB0DAT: SMBus Data

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:
								0xC2

Bits7–0: SMB0DAT: SMBus Data.
The SMB0DAT register contains a byte of data to be transmitted on the SMBus serial interface or a byte that has just been received on the SMBus serial interface. The CPU can read from or write to this register whenever the SI serial interrupt flag (SMB0CN.0) is set to logic one. The serial data in the register remains stable as long as the SI flag is set. When the SI flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register.

14.5. SMBus Transfer Modes

The SMBus interface may be configured to operate as master and/or slave. At any particular time, it will be operating in one of the following four modes: Master Transmitter, Master Receiver, Slave Transmitter, or Slave Receiver. The SMBus interface enters Master Mode any time a START is generated, and remains in Master Mode until it loses arbitration or generates a STOP. An SMBus interrupt is generated at the end of all SMBus byte frames; however, note that the interrupt is generated before the ACK cycle when operating as a receiver, and after the ACK cycle when operating as a transmitter.

14.5.1. Master Transmitter Mode

Serial data is transmitted 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 0 (WRITE). The master then transmits one or more bytes of serial data. After each byte is transmitted, an acknowledge bit is generated by the slave. The transfer is ended when the STO bit is set and a STOP is generated. Note that the interface will switch to Master Receiver Mode if SMB0DAT is not written following a Master Transmitter interrupt. Figure 14.5 shows a typical Master Transmitter sequence. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that the 'data byte transferred' interrupts occur **after** the ACK cycle in this mode.

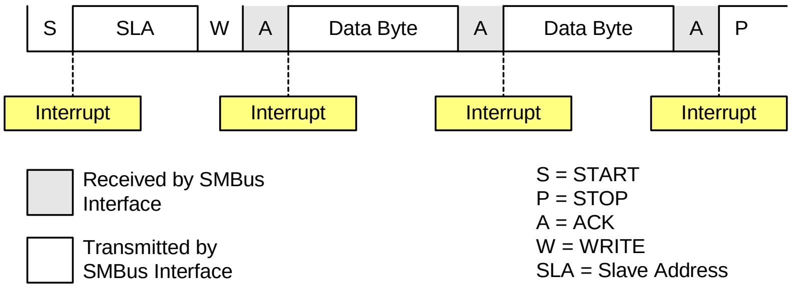


Figure 14.5. Typical Master Transmitter Sequence

14.5.2. 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 14.6 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.

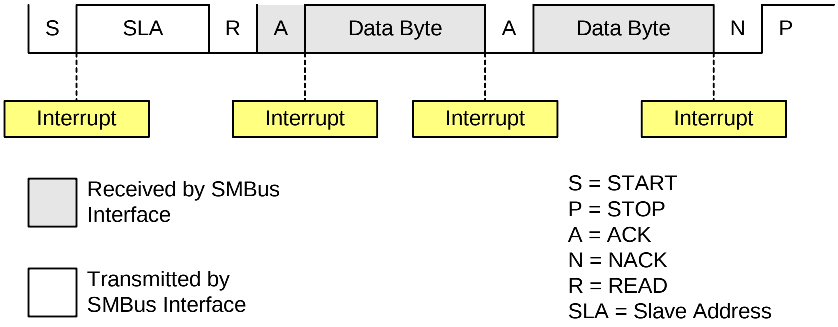


Figure 14.6. Typical Master Receiver Sequence

14.5.3. Slave Receiver Mode

Serial data is received on SDA and the clock is received on SCL. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. Upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. Software responds to the received slave address with an ACK, or ignores the received slave address with a NACK. If the received slave address is ignored, slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received. Software must write the ACK bit after each received byte to ACK or NACK the received byte. The interface exits Slave Receiver Mode after receiving a STOP. Note that the interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver; see **Section 14.5.4** for details on this procedure. Figure 14.7 shows a typical Slave 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.

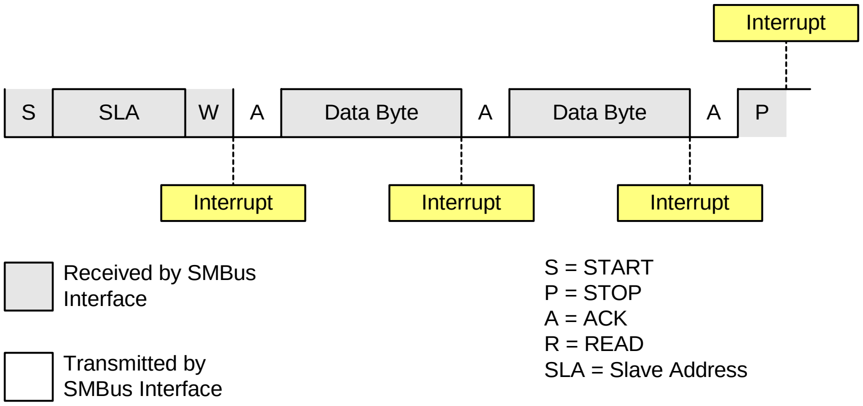


Figure 14.7. Typical Slave Receiver Sequence

14.5.4. Slave Transmitter Mode

Serial data is transmitted on SDA and the clock is received on SCL. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. Software responds to the received slave address with an ACK, or ignores the received slave address with a NACK. If the received address is ignored, slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, software should write data to SMB0DAT to force the SMBus into Slave Transmitter Mode. The switch from Slave Receiver to Slave Transmitter requires software management. Software should perform the steps outlined below only when a valid slave address is received (indicated by the label “RX-to-TX Steps” in Figure 14.8).

Step 1. Set ACK to ‘1’.
Step 2. Write outgoing data to SMB0DAT.
Step 3. Check SMB0DAT.7; if ‘1’, do not perform steps 4, 6 or 7.
Step 4. Set STO to ‘1’.
Step 5. Clear SI to ‘0’.
Step 6. Poll for TXMODE => ‘1’.
Step 7. Clear STO to ‘0’ (must be done before the next ACK cycle).

The interface enters Slave Transmitter Mode and transmits one or more bytes of data (the above steps are only required before the first byte of the transfer). After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (Note: an error condition may be generated if SMB0DAT is written following a received NACK while in Slave Transmitter Mode). The interface exits Slave Transmitter Mode after receiving a STOP. Note that the interface will switch to Slave Receiver Mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 14.8 shows a typical Slave Transmitter sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode.

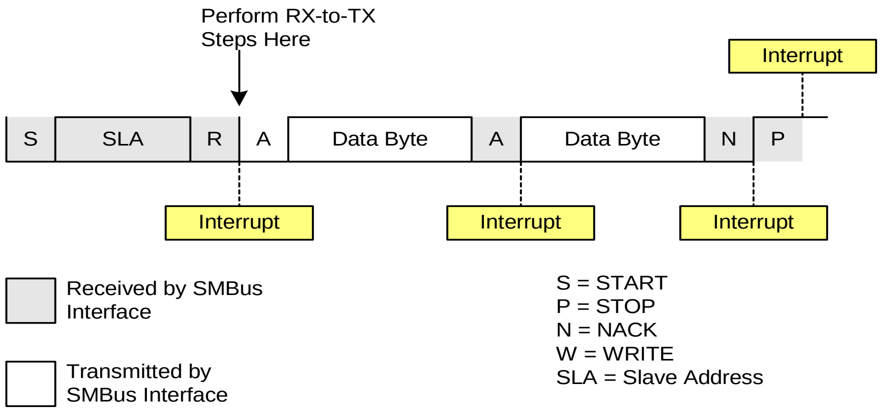


Figure 14.8. Typical Slave Transmitter Sequence

14.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. In the table below, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. Note that the shown response options are only the typical responses; application-specific procedures are allowed as long as they conform with the SMBus specification. Highlighted responses are allowed but do not conform to the SMBus specification.

Table 14.4. SMBus Status Decoding

Mode	Values Read				Current SMbus State	Typical Response Options	Values Written		
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STo	ACK
Master Transmitter	1110	0	0	X	A master START was generated.	Load slave address + R/W into SMB0DAT.	0	0	X
	1100	0	0	0	A master data or address byte was transmitted; NACK received.	Set STA to restart transfer.	1	0	X
						Abort transfer.	0	1	X
		0	0	1	A master data or address byte was transmitted; ACK received.	Load next data byte into SMB0DAT	0	0	X
						End transfer with STOP	0	1	X
						End transfer with STOP and start another transfer.	1	1	X
						Send repeated START	1	0	X
						Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT).	0	0	X
Master Receiver	1000	1	0	X	A master data byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1
						Send NACK to indicate last byte, and send STOP.	0	1	0
						Send NACK to indicate last byte, and send STOP followed by START.	1	1	0
						Send ACK followed by repeated START.	1	0	1
						Send NACK to indicate last byte, and send repeated START.	1	0	0
						Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	**0**	**0**	**1**
						Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	**0**	**0**	**0**
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X
	0101	0	X	X	A STOP was detected while an addressed Slave Transmitter.	No action required (transfer complete).	0	0	X

Table 14.4. SMBus Status Decoding (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values Written		
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STo	ACK
Slave Receiver	0010	1	0	X	A slave address was received; ACK requested.	Acknowledge received address (received slave address match, R/W bit = READ).	0	0	1
						Do not acknowledge received address.	0	0	0
						Acknowledge received address, and switch to transmitter mode (received slave address match, R/W bit = WRITE); see **Section 14.5.4** for procedure.	0	0	1
		1	1	X	Lost arbitration as master; slave address received; ACK requested.	Acknowledge received address (received slave address match, R/W bit = READ).	0	0	1
						Do not acknowledge received address.	0	0	0
						Acknowledge received address, and switch to transmitter mode (received slave address match, R/W bit = WRITE); see **Section 14.5.4** for procedure.	0	0	1
						Reschedule failed transfer; do not acknowledge received address	1	0	0
	0010	0	1	X	Lost arbitration while attempting a repeated START.	Abort failed transfer.	0	0	X
						Reschedule failed transfer.	1	0	X
	0001	1	1	X	Lost arbitration while attempting a STOP.	No action required (transfer complete/aborted).	0	0	0
		0	0	X	A STOP was detected while an addressed slave receiver.	No action required (transfer complete).	0	0	X
		0	1	X	Lost arbitration due to a detected STOP.	Abort transfer.	0	0	X
						Reschedule failed transfer.	1	0	X
	0000	1	0	X	A slave byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1
						Do not acknowledge received byte.	0	0	0
		1	1	X	Lost arbitration while transmitting a data byte as master.	Abort failed transfer.	0	0	0
						Reschedule failed transfer.	1	0	0

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 124**). 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. Reading SBUF0 accesses the buffered Receive register; writing SBUF0 accesses 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).

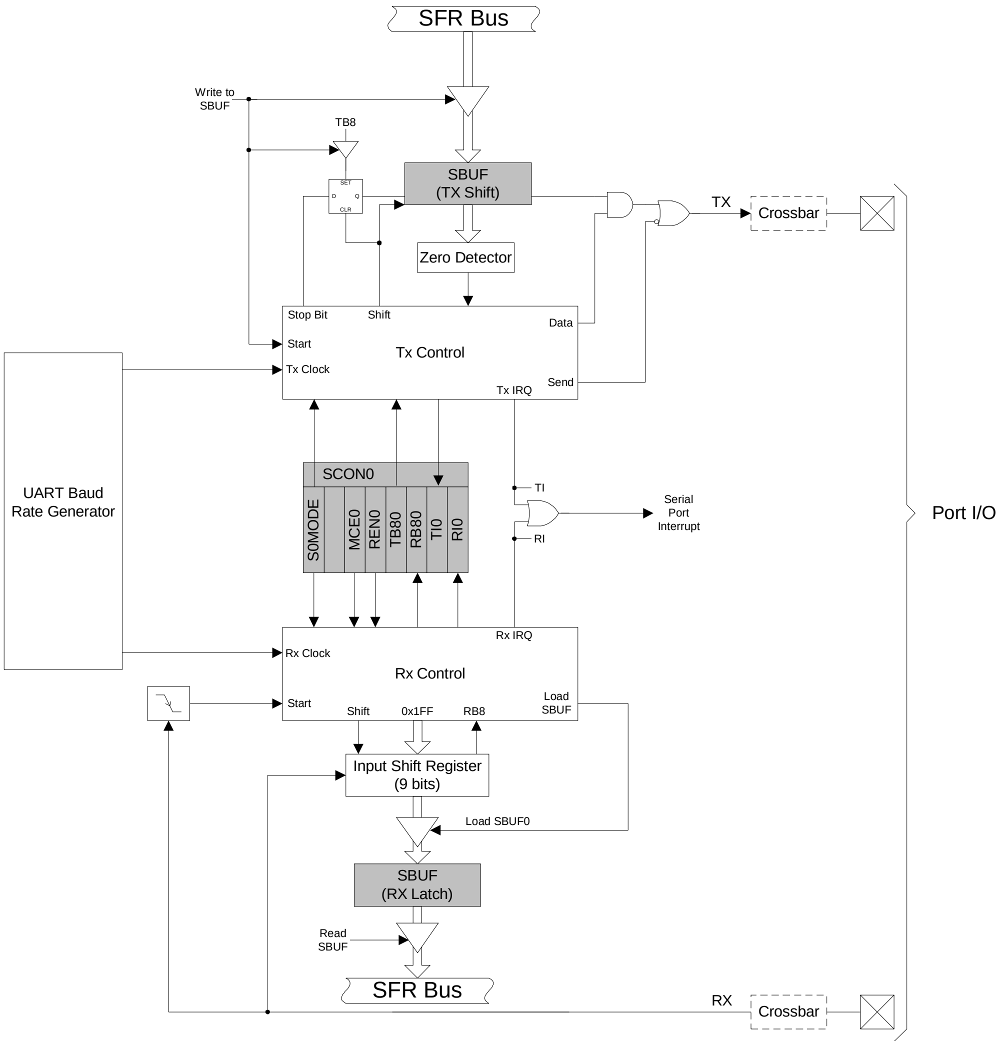


Figure 15.1. UART0 Block Diagram

15.1. Enhanced Baud Rate Generation

The UART0 baud rate is generated by Timer 1 in 8-bit auto-reload mode. The TX clock is generated by TL1; the RX clock is generated by a copy of TL1 (shown as RX Timer in Figure 15.2), which is not user-accessible. Both TX and RX Timer overflows are divided by two to generate the TX and RX baud rates. The RX Timer runs when Timer 1 is enabled, and uses the same reload value (TH1). However, an RX Timer reload is forced when a START condition is detected on the RX pin. This allows a receive to begin any time a START is detected, independent of the TX Timer state.

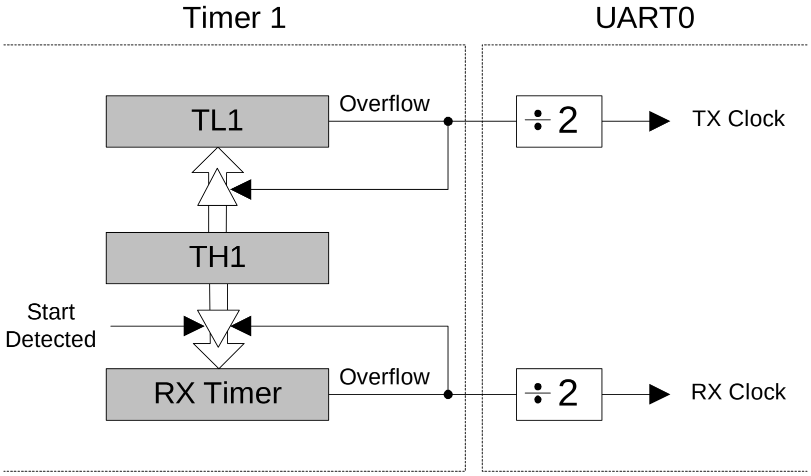


Figure 15.2. UART0 Baud Rate Logic

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see **Section “16.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload” on page 133**). The Timer 1 reload value should be set so that overflows will occur at two times the desired UART baud rate frequency. Note that Timer 1 may be clocked by one of five sources: SYSCLK, SYSCLK / 4, SYSCLK / 12, SYSCLK / 48, or the external oscillator clock / 8. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 15.1.

$$UartBaudRate = \frac{T1_{CLK}}{(256 - T1H)} \times \frac{1}{2}$$

Equation 15.1. UART0 Baud Rate

Where $T1_{CLK}$ is the frequency of the clock supplied to Timer 1, and $T1H$ is the high byte of Timer 1 (reload value). Timer 1 clock frequency is selected as described in **Section “16.2. Timer 2” on page 139**. A quick reference for typical baud rates and system clock frequencies is given in Tables 14.1 through 14.6. Note that the internal oscillator may still generate the system clock when the external oscillator is driving Timer 1 (see **Section “16.1. Timer 0 and Timer 1” on page 131** for more details).

15.2. 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 in Figure 15.3.

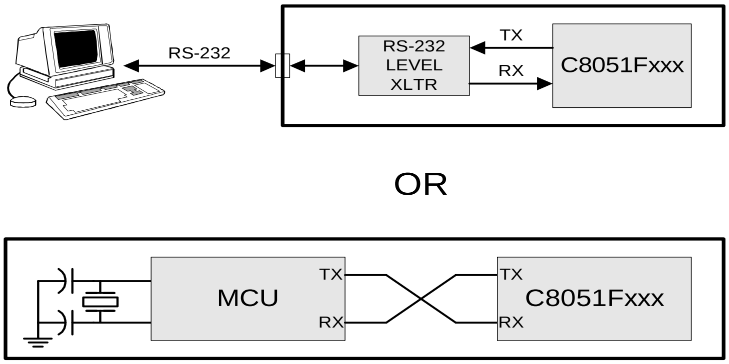


Figure 15.3. UART Interconnect Diagram

15.2.1. 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 TX pin and received at the RX 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.

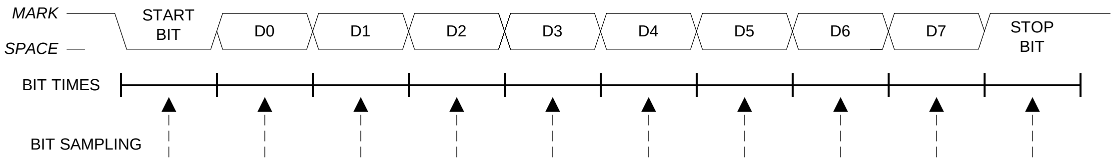


Figure 15.4. 8-Bit UART Timing Diagram

15.2.2. 9-Bit UART

9-bit UART mode uses a total of eleven bits per data byte: a start bit, 8 data bits (LSB first), a programmable ninth data bit, and a stop bit. The state of the ninth transmit data bit is determined by the value in TB80 (SCON0.3), which is assigned by user software. It can be assigned the value of the parity flag (bit P in register PSW) for error detection, or used in multiprocessor communications. On receive, the ninth data bit goes into RB80 (SCON0.2) and the stop bit is ignored.

Data transmission begins when an instruction 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 ‘1’. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: (1) RI0 must be logic 0, and (2) if MCE0 is logic 1, the 9th bit must be logic 1 (when MCE0 is logic 0, the state of the ninth data bit is unimportant). If these conditions are met, the eight bits of data are stored in SBUF0, the ninth bit is stored in RB80, and the RI0 flag is set to ‘1’. If the above conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set to ‘1’. A UART0 interrupt will occur if enabled when either TI0 or RI0 is set to ‘1’.

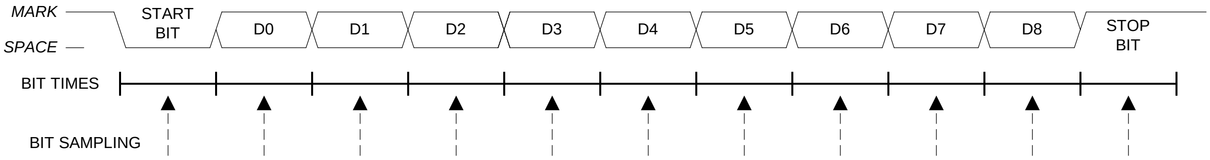


Figure 15.5. 9-Bit UART Timing Diagram

15.3. Multiprocessor Communications

9-Bit UART mode supports multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its ninth bit is logic 1; in a data byte, the ninth bit is always set to logic 0.

Setting the MCE0 bit (SCON.5) of a slave processor configures its UART such that when a stop bit is received, the UART will generate an interrupt only if the ninth bit is logic one (RB80 = 1) signifying an address byte has been received. In the UART interrupt handler, software will compare the received address with the slave's own assigned 8-bit address. If the addresses match, the slave will clear its MCE0 bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE0 bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its MCE0 bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

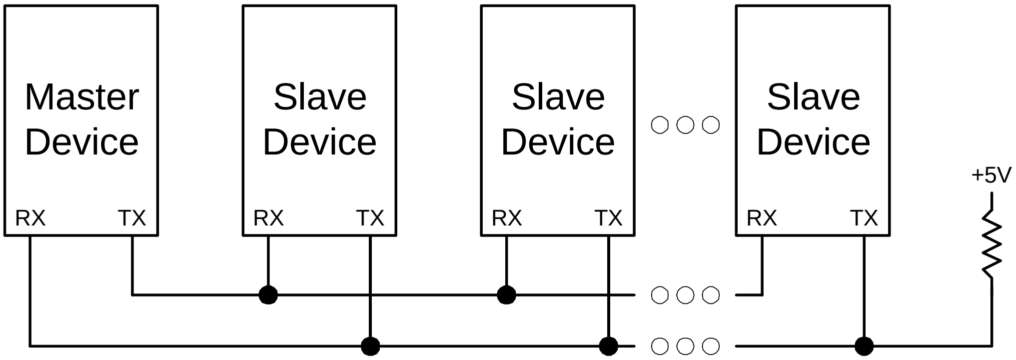


Figure 15.6. UART Multi-Processor Mode Interconnect Diagram

SFR Definition 15.1. SCON0: Serial Port 0 Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
S0MODE	-	MCE0	REN0	TB80	RB80	TI0	RI0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0x98

Bit7:	S0MODE: Serial Port 0 Operation Mode. This bit selects the UART0 Operation Mode. 0: Mode 0: 8-bit UART with Variable Baud Rate 1: Mode 1: 9-bit UART with Variable Baud Rate
Bit6:	UNUSED. Read = 1b. Write = don't care.
Bit5:	MCE0: Multiprocessor Communication Enable. The function of this bit is dependent on the Serial Port 0 Operation Mode. Mode 0: Checks for valid stop bit. 0: Logic level of stop bit is ignored. 1: RI0 will only be activated if stop bit is logic level 1. Mode 1: Multiprocessor Communications Enable. 0: Logic level of ninth bit is ignored. 1: RI0 is set and an interrupt is generated only when the ninth bit is logic 1.
Bit4:	REN0: Receive Enable. This bit enables/disables the UART receiver. 0: UART0 reception disabled. 1: UART0 reception enabled.
Bit3:	TB80: Ninth Transmission Bit. The logic level of this bit will be assigned to the ninth transmission bit in 9-bit UART Mode. It is not used in 8-bit UART Mode. Set or cleared by software as required.
Bit2:	RB80: Ninth Receive Bit. RB80 is assigned the value of the STOP bit in Mode 0; it is assigned the value of the 9th data bit in Mode 1.
Bit1:	TI0: Transmit Interrupt Flag. Set by hardware when a byte of data has been transmitted by UART0 (after the 8th bit in 8-bit UART Mode, or at the beginning of the STOP bit in 9-bit UART Mode). When the UART0 interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software
Bit0:	RI0: Receive Interrupt Flag. Set to '1' by hardware when a byte of data has been received by UART0 (set at the STOP bit sampling time). When the UART0 interrupt is enabled, setting this bit to '1' causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software.

SFR Definition 15.2. SBUF0: Serial (UART0) Port Data Buffer

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: 0x99

Bits7–0: SBUF0[7:0]: 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 is what initiates the transmission. A read of SBUF0 returns the contents of the receive latch.

Table 15.1. Timer Settings for Standard Baud Rates Using the Internal 24.5 MHz Oscillator

	Frequency: 24.5 MHz						
	Target Baud Rate (bps)	**Baud Rate % Error**	**Oscillator Divide Factor**	**Timer Clock Source**	**SCA1–SCA0 (pre-scale select)***	**T1M***	**Timer 1 Reload Value (hex)**
SYSCLK from Internal Osc.	230400	–0.32%	106	SYSCLK	XX	1	0xCB
	115200	–0.32%	212	SYSCLK	XX	1	0x96
	57600	0.15%	426	SYSCLK	XX	1	0x2B
	28800	–0.32%	848	SYSCLK/4	01	0	0x96
	14400	0.15%	1704	SYSCLK/12	00	0	0xB9
	9600	–0.32%	2544	SYSCLK/12	00	0	0x96
	2400	–0.32%	10176	SYSCLK/48	10	0	0x96
	1200	0.15%	20448	SYSCLK/48	10	0	0x2B

X = Don’t care

***Note:** SCA1–SCA0 and T1M bit definitions can be found in **Section 16.1**.

16. Timers

Each MCU includes 3 counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and one is a 16-bit auto-reload timer for use with the ADC, SMBus, or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2 offers 16-bit and split 8-bit timer functionality with auto-reload.

Timer 0 and Timer 1 Modes:	Timer 2 Modes:
13-bit counter/timer	16-bit timer with auto-reload
16-bit counter/timer	
8-bit counter/timer with auto-reload	Two 8-bit timers with auto-reload
Two 8-bit counter/timers (Timer 0 only)	

Timers 0 and 1 may be clocked by one of five sources, determined by the Timer Mode Select bits (T1M–T0M) and the Clock Scale bits (SCA1–SCA0). The Clock Scale bits define a pre-scaled clock from which Timer 0 and/or Timer 1 may be clocked (See SFR Definition 16.3 for pre-scaled clock selection).

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 may be clocked by the system clock, the system clock divided by 12, or the external oscillator clock source divided by 8.

Timer 0 and Timer 1 may also be operated as counters. When functioning as a counter, a counter/timer register is incremented on each high-to-low transition at the selected input pin. Events with a frequency of up to one-fourth the system clock's frequency can be counted. The input signal need not be periodic, but it should be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

16.1. Timer 0 and Timer 1

Each timer is implemented as 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 their status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register (**Section “9.3.5. Interrupt Register Descriptions” on page 73**); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (**Section 9.3.5**). 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.

16.1.1. 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. (See **Section “13.1. Priority**

Crossbar Decoder” on page 98 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 16.3).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or the input signal /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 9.11). Setting GATE0 to ‘1’ allows the timer to be controlled by the external input signal /INT0 (see **Section “9.3.5. Interrupt Register Descriptions” on page 73**), facilitating pulse width measurements.

TR0	GATE0	/INT0	Counter/Timer
0	X	X	Disabled
1	0	X	Enabled
1	1	0	Disabled
1	1	1	Enabled
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 /INT1 is used with Timer 1; the /INT1 polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 9.11).

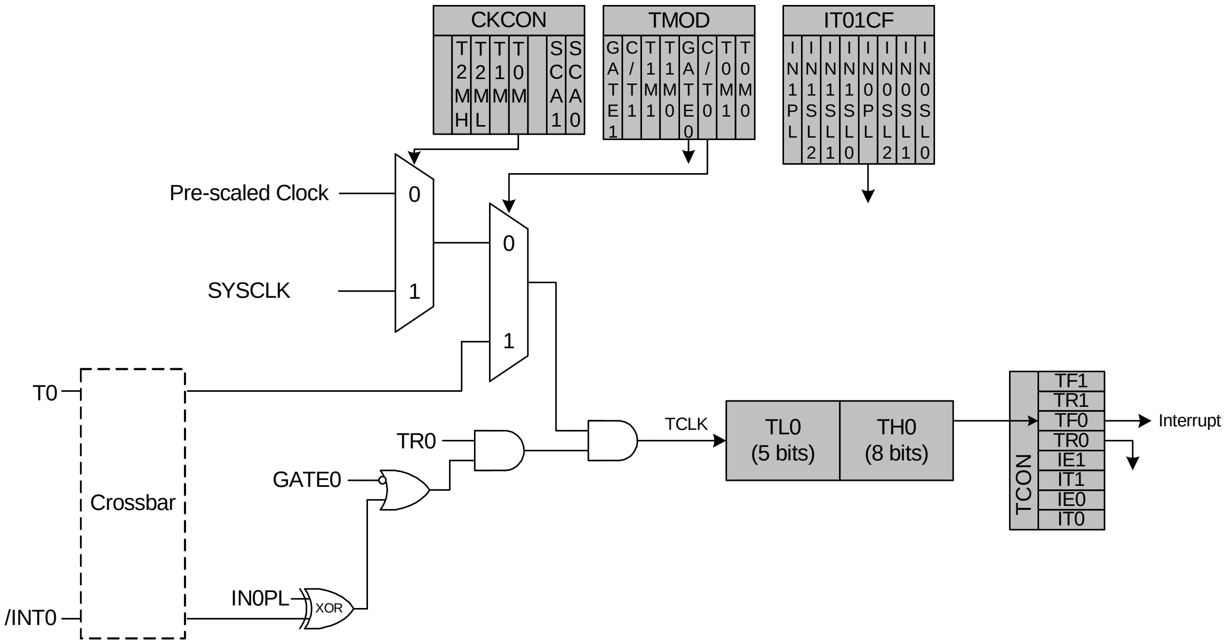


Figure 16.1. T0 Mode 0 Block Diagram

16.1.2. Mode 1: 16-bit Counter/Timer

Mode 1 operation is the same as Mode 0, except that the counter/timer registers use all 16 bits. The counter/timers are enabled and configured in Mode 1 in the same manner as for Mode 0.

16.1.3. 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 “9.3.2. External Interrupts” on page 71** for details on the external input signals /INT0 and /INT1).

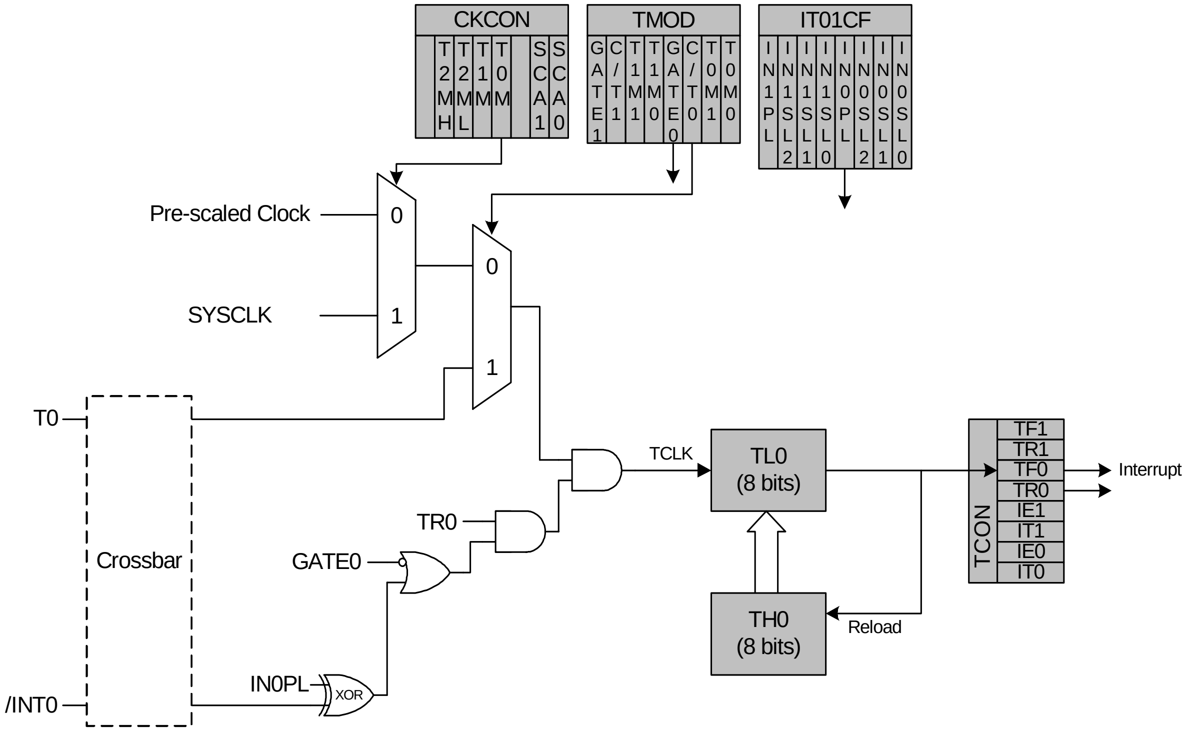


Figure 16.2. T0 Mode 2 Block Diagram

16.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, C/T0, GATE0 and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

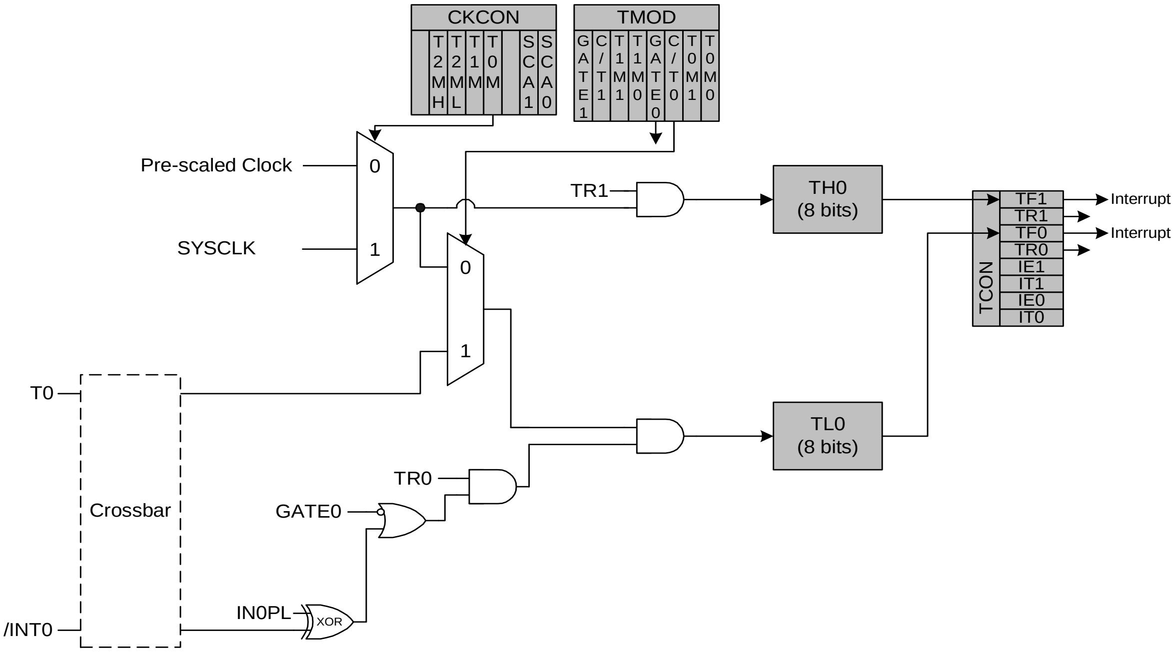


Figure 16.3. T0 Mode 3 Block Diagram

SFR Definition 16.1. TCON: Timer Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
						(bit addressable)		0x88

Bit7: TF1: Timer 1 Overflow Flag.
Set by hardware when Timer 1 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine.
0: No Timer 1 overflow detected.
1: Timer 1 has overflowed.

Bit6: TR1: Timer 1 Run Control.
0: Timer 1 disabled.
1: Timer 1 enabled.

Bit5: TF0: Timer 0 Overflow Flag.
Set by hardware when Timer 0 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine.
0: No Timer 0 overflow detected.
1: Timer 0 has overflowed.

Bit4: TR0: Timer 0 Run Control.
0: Timer 0 disabled.
1: Timer 0 enabled.

Bit3: IE1: External Interrupt 1.
This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine if IT1 = 1. When IT1 = 0, this flag is set to ‘1’ when /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 9.11).

Bit2: IT1: Interrupt 1 Type Select.
This bit selects whether the configured /INT1 interrupt will be edge or level sensitive. /INT1 is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 9.11).
0: /INT1 is level triggered.
1: /INT1 is edge triggered.

Bit1: IE0: External Interrupt 0.
This flag is set by hardware when an edge/level of type defined by IT0 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine if IT0 = 1. When IT0 = 0, this flag is set to ‘1’ when /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 9.11).

Bit0: IT0: Interrupt 0 Type Select.
This bit selects whether the configured /INT0 interrupt will be edge or level sensitive. /INT0 is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 9.11).
0: /INT0 is level triggered.
1: /INT0 is edge triggered.

SFR Definition 16.2. 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 9.11).

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 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 9.11).

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 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

SFR Definition 16.3. CKCON: Clock Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
-	T2MH	T2ML	T1M	T0M	-	SCA1	SCA0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0x8E

Bit7: UNUSED. Read = 0b, Write = don't care.

Bit6: T2MH: Timer 2 High Byte Clock Select
This bit selects the clock supplied to the Timer 2 high byte if Timer 2 is configured in split 8-bit timer mode. T2MH is ignored if Timer 2 is in any other mode.
0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN.
1: Timer 2 high byte uses the system clock.

Bit5: T2ML: Timer 2 Low Byte Clock Select
This bit selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.
0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN.
1: Timer 2 low byte uses the system clock.

Bit4: T1M: Timer 1 Clock Select.
This select the clock source supplied to Timer 1. T1M is ignored when C/T1 is set to logic 1.
0: Timer 1 uses the clock defined by the prescale bits, SCA1–SCA0.
1: Timer 1 uses the system clock.

Bit3: T0M: Timer 0 Clock Select.
This bit selects the clock source supplied to Timer 0. T0M is ignored when C/T0 is set to logic 1.
0: Counter/Timer 0 uses the clock defined by the prescale bits, SCA1–SCA0.
1: Counter/Timer 0 uses the system clock.

Bit2: UNUSED. Read = 0b, Write = don't care.

Bits1–0: SCA1–SCA0: Timer 0/1 Prescale Bits
These bits control the division of the clock supplied to Timer 0 and/or Timer 1 if configured to use prescaled clock inputs.

SCA1	SCA0	Prescaled Clock
0	0	System clock divided by 12
0	1	System clock divided by 4
1	0	System clock divided by 48
1	1	External clock divided by 8

Note: External clock divided by 8 is synchronized with the system clock, and the external clock must be less than or equal to the system clock to operate in this mode.

SFR Definition 16.4. TL0: Timer 0 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:
								0x8A

Bits 7–0: TL0: Timer 0 Low Byte.
The TL0 register is the low byte of the 16-bit Timer 0

SFR Definition 16.5. TL1: Timer 1 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:
								0x8B

Bits 7–0: TL1: Timer 1 Low Byte.
The TL1 register is the low byte of the 16-bit Timer 1.

SFR Definition 16.6. TH0: Timer 0 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:
								0x8C

Bits 7–0: TH0: Timer 0 High Byte.
The TH0 register is the high byte of the 16-bit Timer 0.

SFR Definition 16.7. TH1: Timer 1 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:
								0x8D

Bits 7–0: TH1: Timer 1 High Byte.
The TH1 register is the high byte of the 16-bit Timer 1.

16.2. Timer 2

Timer 2 is a 16-bit timer formed by two 8-bit SFRs: TMR2L (low byte) and TMR2H (high byte). Timer 2 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T2SPLIT bit (TMR2CN.3) defines the Timer 2 operation mode.

Timer 2 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. The external clock mode is ideal for real-time clock (RTC) functionality, where the internal oscillator drives the system clock while Timer 2 (and/or the PCA) is clocked by an external precision oscillator. Note that the external oscillator source divided by 8 is synchronized with the system clock.

16.2.1. 16-bit Timer with Auto-Reload

When T2SPLIT (TMR2CN.3) is zero, Timer 2 operates as a 16-bit timer with auto-reload. Timer 2 can be clocked by SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 2 reload registers (TMR2RLH and TMR2RLL) is loaded into the Timer 2 register as shown in Figure 16.4, and the Timer 2 High Byte Overflow Flag (TMR2CN.7) is set. If Timer 2 interrupts are enabled (if IE.5 is set), an interrupt will be generated on each Timer 2 overflow. Additionally, if Timer 2 interrupts are enabled and the TF2LEN bit is set (TMR2CN.5), an interrupt will be generated each time the lower 8 bits (TMR2L) overflow from 0xFF to 0x00.

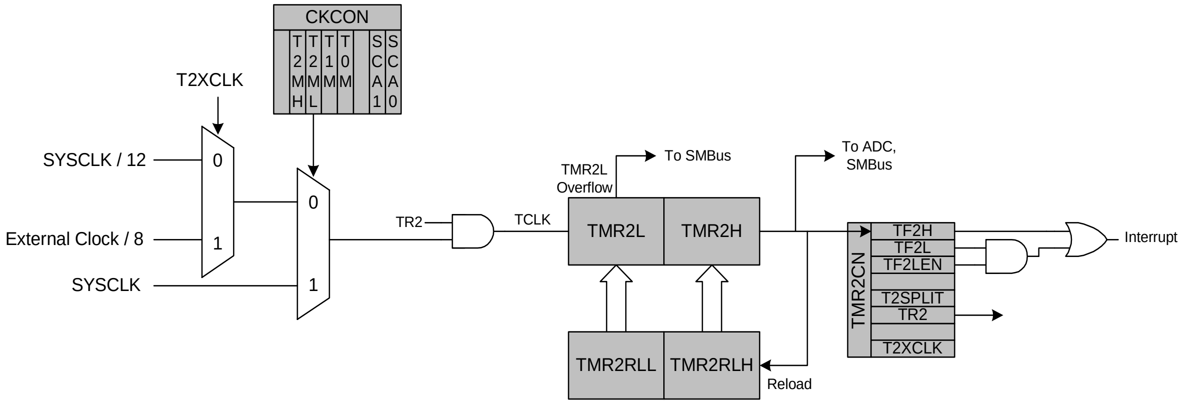


Figure 16.4. Timer 2 16-Bit Mode Block Diagram

16.2.2. 8-bit Timers with Auto-Reload

When T2SPLIT is set, Timer 2 operates as two 8-bit timers (TMR2H and TMR2L). Both 8-bit timers operate in auto-reload mode as shown in Figure 16.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. The Timer 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bit (T2XCLK in TMR2CN), as follows:

T2MH	T2XCLK	TMR2H Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

T2ML	T2XCLK	TMR2L Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

Note: External clock divided by 8 is synchronized with the system clock, and the external clock must be less than or equal to the system clock to operate in this mode.

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LEN (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LEN is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

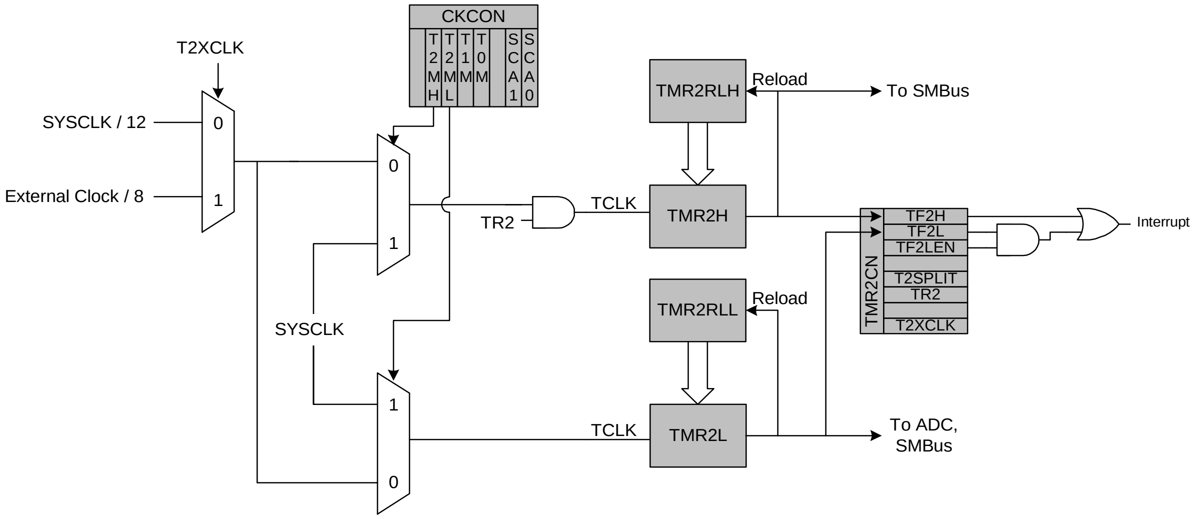


Figure 16.5. Timer 2 8-Bit Mode Block Diagram

SFR Definition 16.8. TMR2CN: Timer 2 Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
TF2H	TF2L	TF2LEN	-	T2SPLIT	TR2	-	T2XCLK	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xC8

Bit7: 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. TF2H is not automatically cleared by hardware and must be cleared by software.

Bit6: TF2L: Timer 2 Low Byte Overflow Flag
Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. When this bit is set, an interrupt will be generated if TF2LEN is set and Timer 2 interrupts are enabled. TF2L will set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware.

Bit5: TF2LEN: Timer 2 Low Byte Interrupt Enable.
This bit enables/disables Timer 2 Low Byte interrupts. If TF2LEN is set and Timer 2 interrupts are enabled, an interrupt will be generated when the low byte of Timer 2 overflows. This bit should be cleared when operating Timer 2 in 16-bit mode.
0: Timer 2 Low Byte interrupts disabled.
1: Timer 2 Low Byte interrupts enabled.

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

Bit3: T2SPLIT: Timer 2 Split Mode Enable
When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload.
0: Timer 2 operates in 16-bit auto-reload mode.
1: Timer 2 operates as two 8-bit auto-reload timers.

Bit2: TR2: Timer 2 Run Control.
This bit enables/disables Timer 2. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in this mode.
0: Timer 2 disabled.
1: Timer 2 enabled.

Bit1: UNUSED. Read = 0b. Write = don’t care.

Bit0: T2XCLK: Timer 2 External Clock Select
This bit selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the 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.
0: Timer 2 external clock selection is the system clock divided by 12.
1: Timer 2 external clock selection is the external clock divided by 8. Note that the external oscillator source divided by 8 is synchronized with the system clock.

SFR Definition 16.9. TMR2RLL: Timer 2 Reload Register 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: 0xCA

Bits 7–0: TMR2RLL: Timer 2 Reload Register Low Byte.
TMR2RLL holds the low byte of the reload value for Timer 2.

SFR Definition 16.10. TMR2RLH: Timer 2 Reload Register 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: 0xCB

Bits 7–0: TMR2RLH: Timer 2 Reload Register High Byte.
The TMR2RLH holds the high byte of the reload value for Timer 2.

SFR Definition 16.11. TMR2L: Timer 2 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: 0xCC

Bits 7–0: TMR2L: Timer 2 Low Byte.
In 16-bit mode, the TMR2L register contains the low byte of the 16-bit Timer 2. In 8-bit mode, TMR2L contains the 8-bit low byte timer value.

SFR Definition 16.12. TMR2H Timer 2 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: 0xCD

Bits 7–0: TMR2H: Timer 2 High Byte.
In 16-bit mode, the TMR2H register contains the high byte of the 16-bit Timer 2. In 8-bit mode, TMR2H contains the 8-bit high byte timer value.

17. 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 three 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 (See **Section "13.1. Priority Crossbar Decoder" on page 98** for details on configuring the Crossbar). The counter/timer is driven by a programmable timebase that can select between six sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0 overflow, 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-Bit PWM, or 16-Bit PWM (each mode is described in **Section "17.2. Capture/Compare Modules" on page 145**). 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 basic PCA block diagram is shown in Figure 17.1.

Important Note: The PCA Module 2 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 17.3** for details.

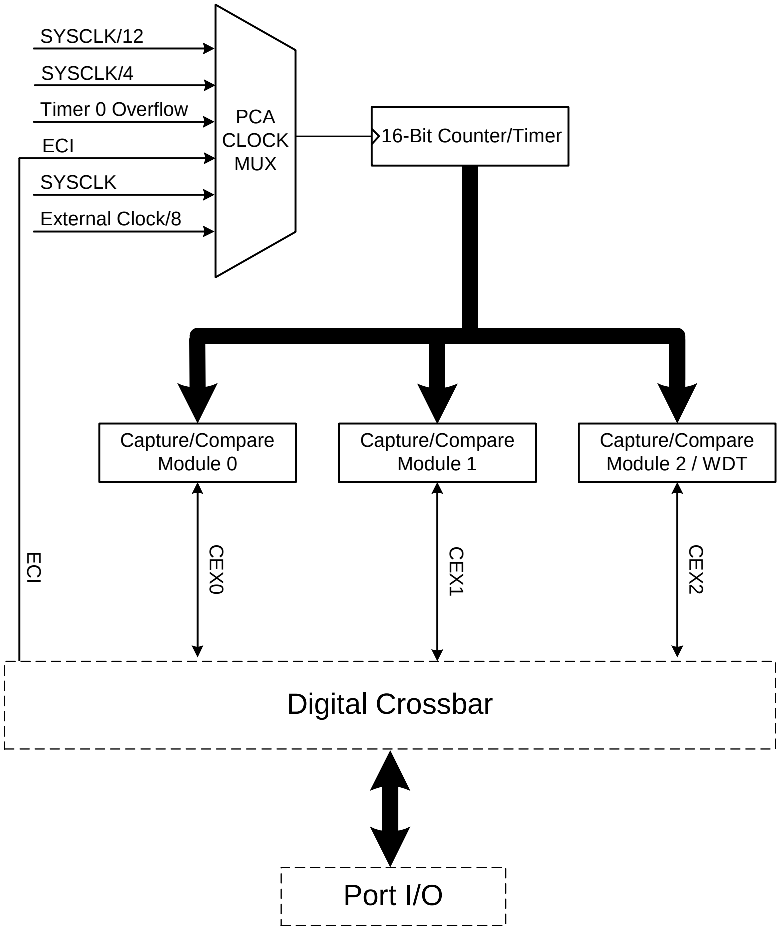


Figure 17.1. PCA Block Diagram

17.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte (MSB) of the 16-bit counter/timer and PCA0L is the low byte (LSB). Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register. **Reading the PCA0L Register first guarantees an accurate reading of the entire 16-bit PCA0 counter**. Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer as shown in Table 17.1. **Note that in ‘External oscillator source divided by 8’ mode, the external oscillator source is synchronized with the system clock, and must have a frequency less than or equal to the system clock.**

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software (Note: PCA0 interrupts must be globally enabled before CF interrupts are recognized. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1). Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

Table 17.1. PCA Timebase Input Options

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

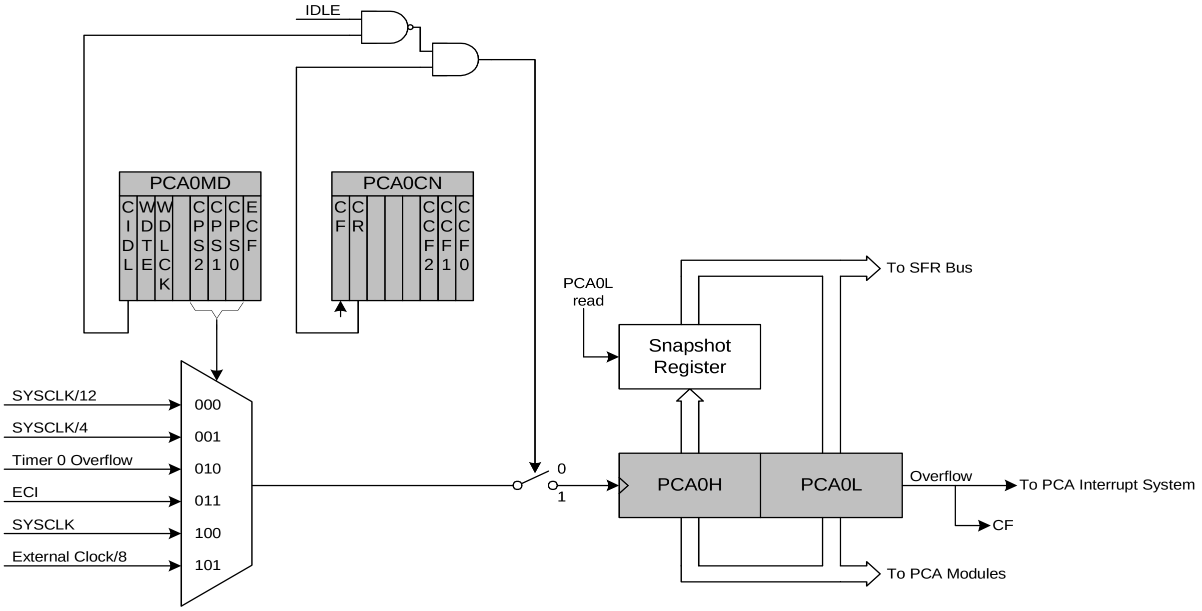


Figure 17.2. PCA Counter/Timer Block Diagram

17.2. Capture/Compare Modules

Each module can be configured to operate independently in one of six operation modes: Edge-triggered Capture, Software Timer, High Speed Output, Frequency Output, 8-Bit Pulse Width Modulator, or 16-Bit Pulse Width Modulator. Each module has Special Function Registers (SFRs) associated with it in the CIP-51 system controller. These registers are used to exchange data with a module and configure the module's mode of operation.

Table 17.2 summarizes the bit settings in the PCA0CPMn registers used to select the PCA capture/compare module's operating modes. Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt. Note: PCA0 interrupts must be globally enabled before individual CCFn interrupts are recognized. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1. See Figure 17.3 for details on the PCA interrupt configuration.

Table 17.2. PCA0CPM Register Settings for PCA Capture/Compare Modules

PWM16	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	Operation Mode
X	X	1	0	0	0	0	X	Capture triggered by positive edge on CEXn
X	X	0	1	0	0	0	X	Capture triggered by negative edge on CEXn
X	X	1	1	0	0	0	X	Capture triggered by transition on CEXn
X	1	0	0	1	0	0	X	Software Timer
X	1	0	0	1	1	0	X	High Speed Output
X	1	0	0	X	1	1	X	Frequency Output
0	1	0	0	X	0	1	X	8-Bit Pulse Width Modulator
1	1	0	0	X	0	1	X	16-Bit Pulse Width Modulator
X = Don't Care								

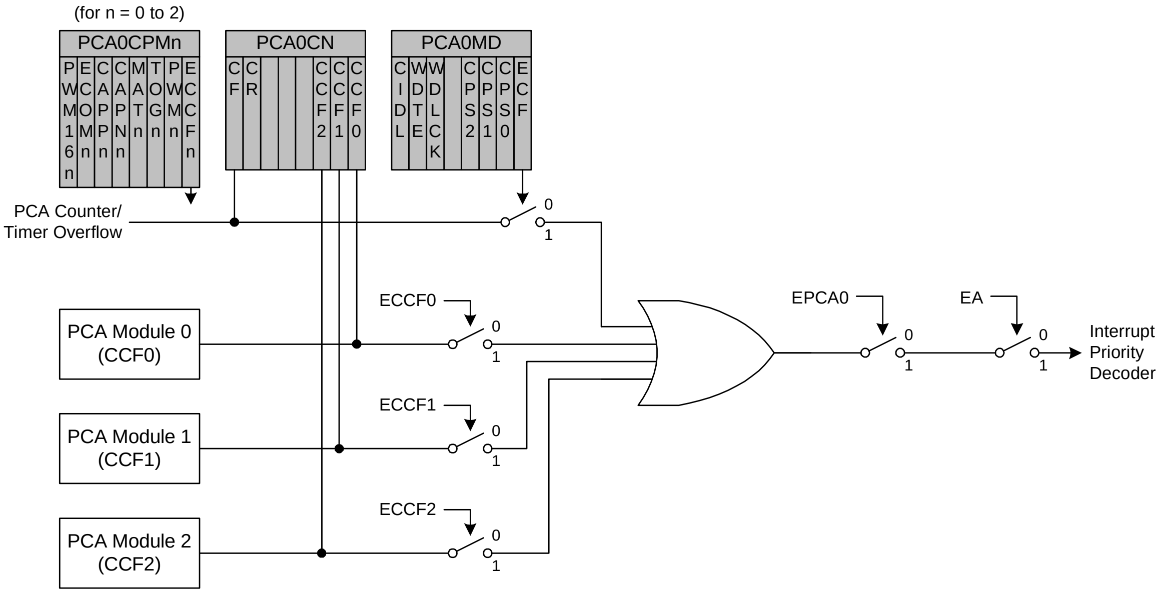


Figure 17.3. PCA Interrupt Block Diagram

17.2.1. 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 copy 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.

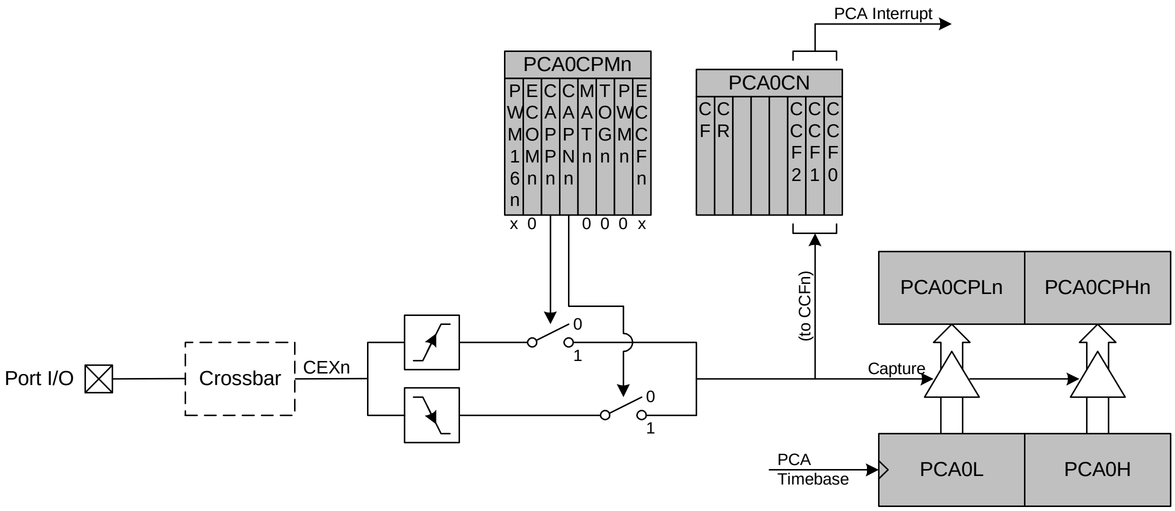


Figure 17.4. PCA Capture Mode Diagram

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

17.2.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1 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. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to ‘0’; writing to PCA0CPHn sets ECOMn to ‘1’.

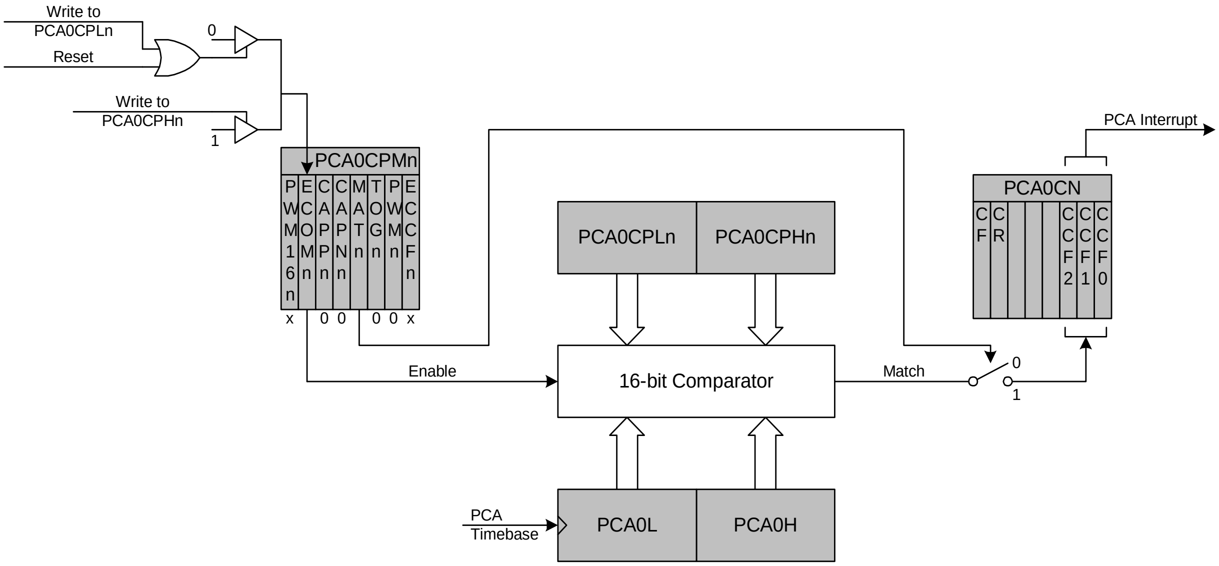


Figure 17.5. PCA Software Timer Mode Diagram

17.2.3. High Speed Output Mode

In High Speed Output mode, a module’s associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn) Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to ‘0’; writing to PCA0CPHn sets ECOMn to ‘1’.

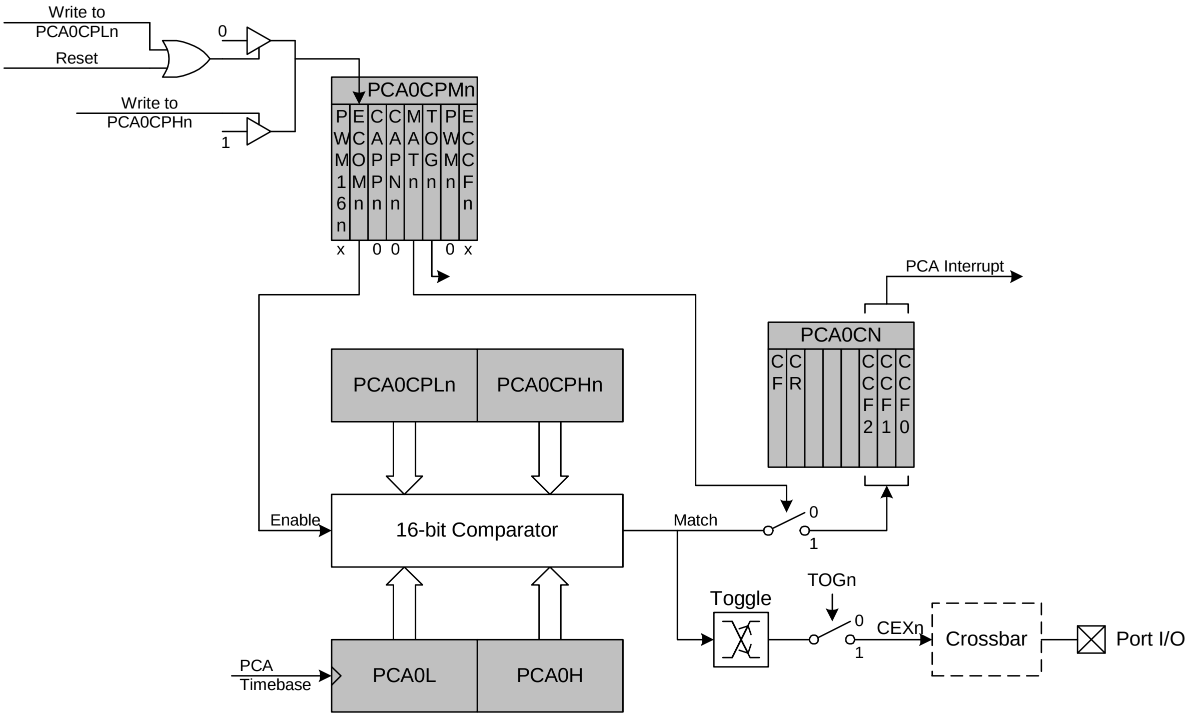


Figure 17.6. PCA High Speed Output Mode Diagram

17.2.4. Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module’s associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined by Equation 17.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

Equation 17.1. Square Wave Frequency Output

Where F_{PCA} is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register, PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register.

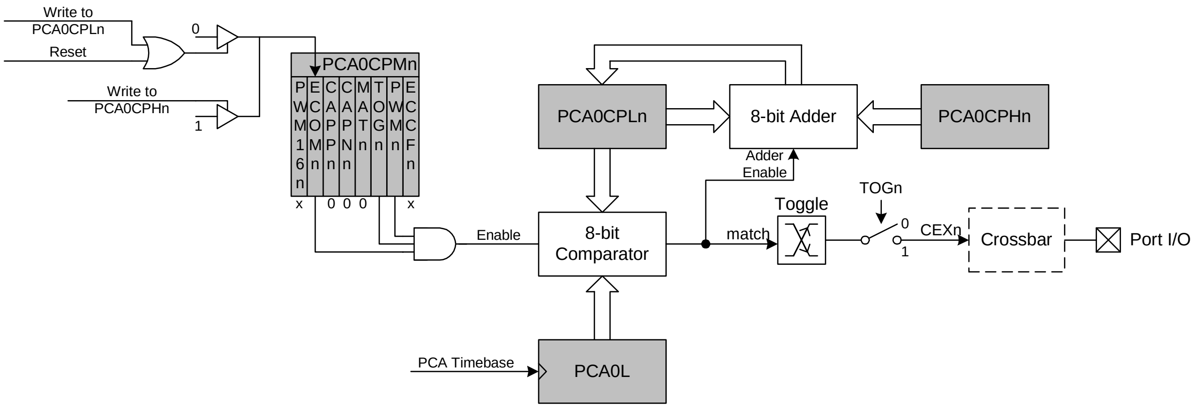


Figure 17.7. PCA Frequency Output Mode

17.2.5. 8-Bit Pulse Width Modulator Mode

Each module can be used independently to generate a pulse width modulated (PWM) output on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA counter/timer. The duty cycle of the PWM output signal is varied using the module's PCA0CPLn capture/compare register. When the value in the low byte of the PCA counter/timer (PCA0L) is equal to the value in PCA0CPLn, the output on the CEXn pin will be set to ‘1’. When the count value in PCA0L overflows, the CEXn output will be set to ‘0’ (see Figure 17.8). Also, when the counter/timer low byte (PCA0L) overflows from 0xFF to 0x00, PCA0CPLn is reloaded automatically with the value stored in the module’s capture/compare high byte (PCA0CPHn) without software intervention. Setting the ECOMn and PWMn bits in the PCA0CPMn register enables 8-Bit Pulse Width Modulator mode. The duty cycle for 8-Bit PWM Mode is given by Equation 17.2.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to ‘0’; writing to PCA0CPHn sets ECOMn to ‘1’.

$$DutyCycle = \frac{(256 - PCA0CPHn)}{256}$$

Equation 17.2. 8-Bit PWM Duty Cycle

Using Equation 17.2, the largest duty cycle is 100% (PCA0CPHn = 0), and the smallest duty cycle is 0.39% (PCA0CPHn = 0xFF). A 0% duty cycle may be generated by clearing the ECOMn bit to ‘0’.

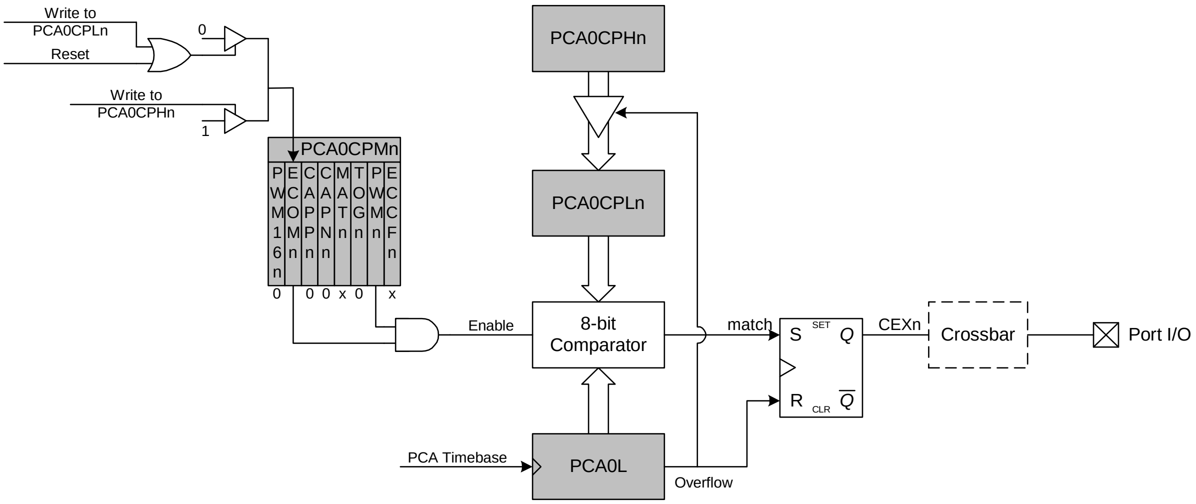


Figure 17.8. PCA 8-Bit PWM Mode Diagram

17.2.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is set to ‘1’; when the counter overflows, CEXn is set to ‘0’. To output a varying duty cycle, new value writes should be synchronized with PCA CCFn match interrupts. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, match interrupts should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. The duty cycle for 16-Bit PWM Mode is given by Equation 17.3.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to ‘0’; writing to PCA0CPHn sets ECOMn to ‘1’.

$$DutyCycle = \frac{(65536 - PCA0CPn)}{65536}$$

Equation 17.3. 16-Bit PWM Duty Cycle

Using Equation 17.3, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to ‘0’.

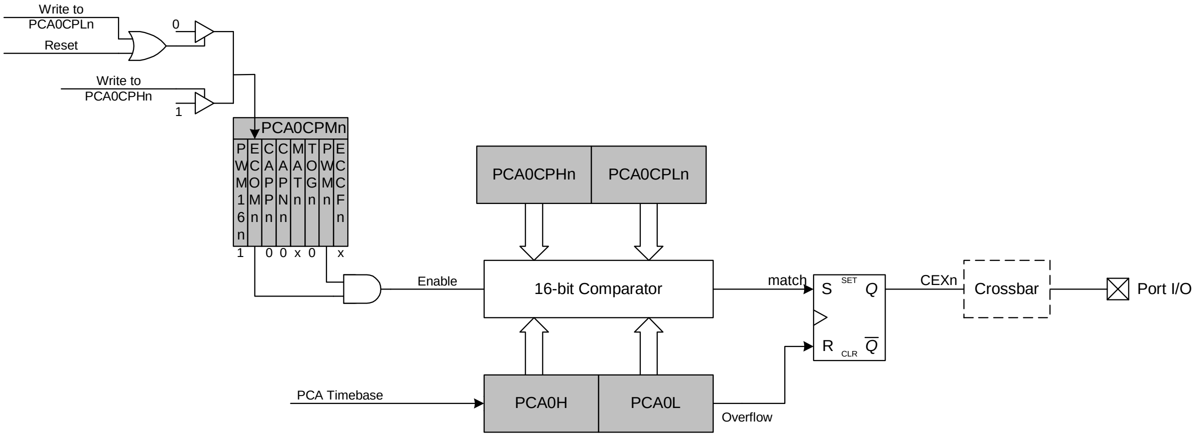


Figure 17.9. PCA 16-Bit PWM Mode

17.3. Watchdog Timer Mode

A programmable watchdog timer (WDT) function is available through the PCA Module 2. The WDT is used to generate a reset if the time between writes to the WDT update register (PCA0CPH2) exceed a specified limit. The WDT can be configured and enabled/disabled as needed by software.

With the WDTE bit set in the PCA0MD register, Module 2 operates as a watchdog timer (WDT). The Module 2 high byte is compared to the PCA counter high byte; the Module 2 low byte holds the offset to be used when WDT updates are performed. **The Watchdog Timer is enabled on reset. Writes to some PCA registers are restricted while the Watchdog Timer is enabled.**

17.3.1. Watchdog Timer Operation

While the WDT is enabled:

- PCA counter is forced on.
- Writes to PCA0L and PCA0H are not allowed.
- PCA clock source bits (CPS2–CPS0) are frozen.
- PCA Idle control bit (CIDL) is frozen.
- Module 2 is forced into software timer mode.
- Writes to the module 2 mode register (PCA0CPM2) are disabled.

While the WDT is enabled, writes to the CR bit will not change the PCA counter state; the counter will run until the WDT is disabled. The PCA counter run control (CR) will read zero if the WDT is enabled but user software has not enabled the PCA counter. If a match occurs between PCA0CPH2 and PCA0H while the WDT is enabled, a reset will be generated. To prevent a WDT reset, the WDT may be updated with a write of any value to PCA0CPH2. Upon a PCA0CPH2 write, PCA0H plus the offset held in PCA0CPL2 is loaded into PCA0CPH2 (See Figure 17.10).

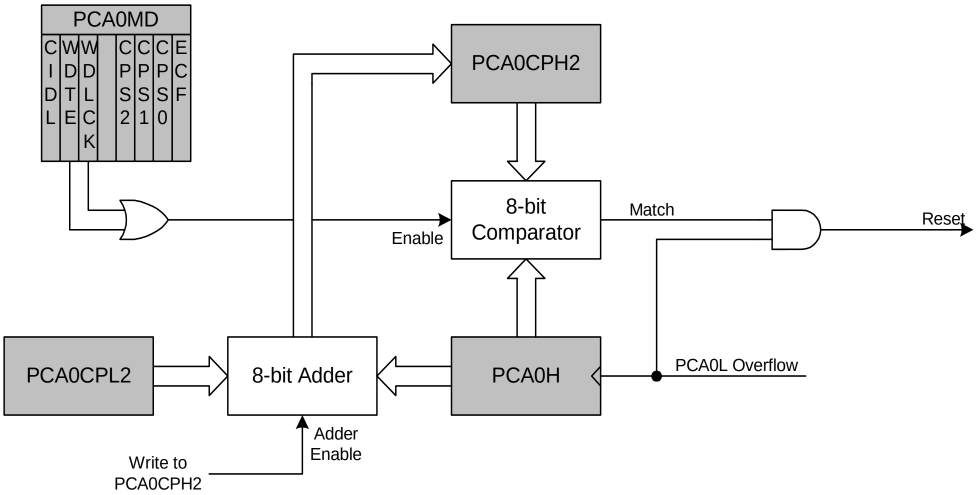


Figure 17.10. PCA Module 2 with Watchdog Timer Enabled

Note that the 8-bit offset held in PCA0CPH2 is compared to the upper byte of the 16-bit PCA counter. This offset value is the number of PCA0L overflows before a reset. Up to 256 PCA clocks may pass before the first PCA0L overflow occurs, depending on the value of the PCA0L when the update is performed. The total offset is then given (in PCA clocks) by Equation 17.4, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL2) + (256 - PCA0L)$$

Equation 17.4. Watchdog Timer Offset in PCA Clocks

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH2 and PCA0H. Software may force a WDT reset by writing a ‘1’ to the CCF2 flag (PCA0CN.2) while the WDT is enabled.

17.3.2. Watchdog Timer Usage

To configure the WDT, perform the following tasks:

- Disable the WDT by writing a ‘0’ to the WDTE bit.
- Select the desired PCA clock source (with the CPS2–CPS0 bits).
- Load PCA0CPL2 with the desired WDT update offset value.
- Configure the PCA Idle mode (set CIDL if the WDT should be suspended while the CPU is in Idle mode).
- Enable the WDT by setting the WDTE bit to ‘1’.
- Reload the WDT by writing any value to PCA0CPH2.

The PCA clock source and Idle mode select cannot be changed while the WDT is enabled. The Watchdog Timer is enabled by setting the WDTE or WDLCK bits in the PCA0MD register. When WDLCK is set, the WDT cannot be disabled until the next system reset. If WDLCK is not set, the WDT is disabled by clearing the WDTE bit.

The WDT is enabled following any reset. The PCA0 counter clock defaults to the system clock divided by 12, PCA0L defaults to 0x00, and PCA0CPL2 defaults to 0x00. Using Equation 17.4, this results in a WDT timeout interval of 3072 system clock cycles. Table 17.3 lists some example timeout intervals for typical system clocks, assuming SYSCLK / 12 as the PCA clock source.

Table 17.3. Watchdog Timer Timeout Intervals[1]

System Clock (Hz)	PCA0CPL2	Timeout Interval (ms)
24,500,000	255	32.1
24,500,000	128	16.2
24,500,000	32	4.1
12,250,000	255	64.2
12,250,000	128	32.4
12,250,000	32	8.3
6,125,000	255	128.4
6,125,000	128	64.7
6,125,000	32	16.6
3,062,500[2]	255	257
3,062,500[2]	128	129.5
3,062,500[2]	32	33.1

Notes:

1. Assumes SYSCLK/12 as the PCA clock source, and a PCA0L value of 0x00 at the update time.
2. Internal reset frequency for SYSCLK (Internal Oscillator/8).

17.4. Register Descriptions for PCA

Following are detailed descriptions of the special function registers related to the operation of the PCA.

SFR Definition 17.1. PCA0CN: PCA Control

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
CF	CR	—	—	—	CCF2	CCF1	CCF0	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address:
							(bit addressable)	0xD8

Bit7:	CF: PCA Counter/Timer Overflow Flag. Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
Bit6:	CR: PCA Counter/Timer Run Control. This bit enables/disables the PCA Counter/Timer. 0: PCA Counter/Timer disabled. 1: PCA Counter/Timer enabled.
Bits5–3:	UNUSED. Read = 000b, Write = don't care.
Bit2:	CCF2: PCA Module 2 Capture/Compare Flag. This bit is set by hardware when a match or capture occurs. When the CCF2 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
Bit1:	CCF1: PCA Module 1 Capture/Compare Flag. This bit is set by hardware when a match or capture occurs. When the CCF1 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
Bit0:	CCF0: PCA Module 0 Capture/Compare Flag. This bit is set by hardware when a match or capture occurs. When the CCF0 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

SFR Definition 17.2. 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 2 is used as the Watchdog Timer.
0: Watchdog Timer disabled.
1: PCA Module 2 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 clock 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 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.

SFR Definition 17.3. PCA0CPMn: PCA Capture/Compare Mode

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Reset Value
PWM16n	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	SFR Address: 0xDA, 0xDB, 0xDC

PCA0CPMn Address:
PCA0CPM0 = 0xDA (n = 0)
PCA0CPM1 = 0xDB (n = 1)
PCA0CPM2 = 0xDC (n = 2)

Bit7: PWM16n: 16-bit Pulse Width Modulation Enable.
This bit selects 16-bit mode when Pulse Width Modulation mode is enabled (PWMn = 1).
0: 8-bit PWM selected.
1: 16-bit PWM selected.

Bit6: ECOMn: Comparator Function Enable.
This bit enables/disables the comparator function for PCA Module n.
0: Disabled.
1: Enabled.

Bit5: CAPPn: Capture Positive Function Enable.
This bit enables/disables the positive edge capture for PCA Module n.
0: Disabled.
1: Enabled.

Bit4: CAPNn: Capture Negative Function Enable.
This bit enables/disables the negative edge capture for PCA Module n.
0: Disabled.
1: Enabled.

Bit3: MATn: Match Function Enable.
This bit enables/disables the match function for PCA Module n. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCFn bit in PCA0MD register to be set to logic 1.
0: Disabled.
1: Enabled.

Bit2: TOGn: Toggle Function Enable.
This bit enables/disables the toggle function for PCA Module n. When enabled, matches of the PCA counter with a module's capture/compare register cause the logic level on the CEXn pin to toggle. If the PWMn bit is also set to logic 1, the module operates in Frequency Output Mode.
0: Disabled.
1: Enabled.

Bit1: PWMn: Pulse Width Modulation Mode Enable.
This bit enables/disables the PWM function for PCA Module n. When enabled, a pulse width modulated signal is output on the CEXn pin. 8-bit PWM is used if PWM16n is cleared; 16-bit mode is used if PWM16n is set to logic 1. If the TOGn bit is also set, the module operates in Frequency Output Mode.
0: Disabled.
1: Enabled.

Bit0: ECCFn: Capture/Compare Flag Interrupt Enable.
This bit sets the masking of the Capture/Compare Flag (CCFn) interrupt.
0: Disable CCFn interrupts.
1: Enable a Capture/Compare Flag interrupt request when CCFn is set.

SFR Definition 17.4. PCA0L: PCA Counter/Timer 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:
								0xF9

Bits 7–0: PCA0L: PCA Counter/Timer Low Byte.
The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.

SFR Definition 17.5. PCA0H: PCA Counter/Timer 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:
								0xFA

Bits 7–0: PCA0H: PCA Counter/Timer High Byte.
The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer.

SFR Definition 17.6. PCA0CPLn: PCA Capture Module 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: 0xFB, 0xE9, 0xEB

PCA0CPLn Address: PCA0CPL0 = 0xFB (n = 0)
PCA0CPL1 = 0xE9 (n = 1)
PCA0CPL2 = 0xEB (n = 2)

Bits7–0: PCA0CPLn: PCA Capture Module Low Byte.
The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture Module n.

SFR Definition 17.7. PCA0CPHn: PCA Capture Module 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: 0xFC, 0xEA, 0xEC

PCA0CPHn Address: PCA0CPH0 = 0xFC (n = 0)
PCA0CPH1 = 0xEA (n = 1)
PCA0CPH2 = 0xEC(n = 2)

Bits7–0: PCA0CPHn: PCA Capture Module High Byte.
The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture Module n.

18. Revision Specific Behavior

This chapter contains behavioral differences between C8051T60x "REV C" and behavior as stated in the data sheet.

These deviations will be resolved in the next revision of the device.

18.1. Revision Identification

The Lot ID Code on the top side of the device package can be used for decoding device revision information. On C8051T60x devices the revision letter is the first letter of the Lot ID Code.

Figure 18.1 and Figure 18.2 show how to find the Lot ID Code on the top side of the device package.

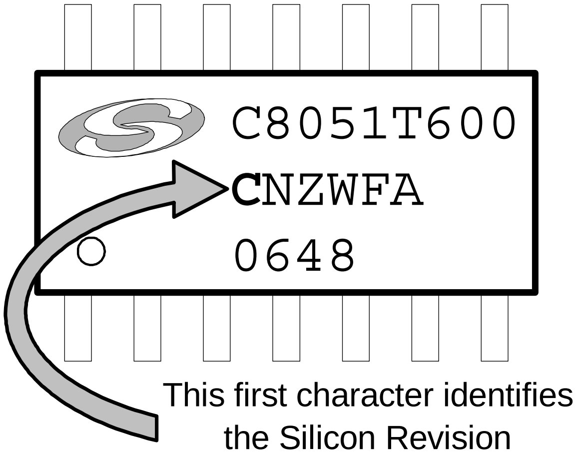


Figure 18.1. Device Package - SOIC 14

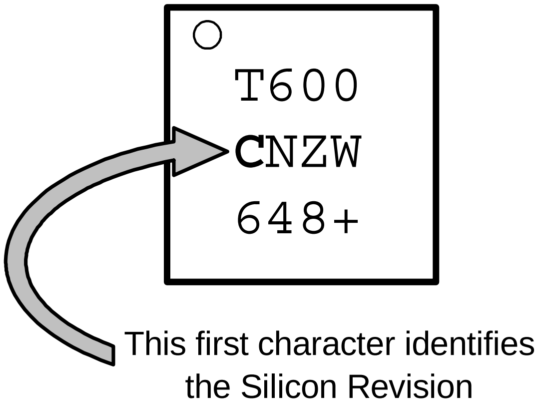


Figure 18.2. Device Package - QFN 10

18.2. SAR Clock Maximum

The maximum SAR clock for "REV C" devices is slower than specified in the data sheet.

On "REV C" devices, the maximum SAR clock for full-performance operation is 4 MHz. If the SAR clock is operated faster than 4 MHz on "REV C" devices, there will be degradation to SNR and linearity performance of the ADC.

This issue will be corrected for "REV D" and later devices.

18.3. V_{DD} Monitor Oscillation

On "REV C" devices, when the V_{DD} Supply Monitor is enabled, the device may go into and come back out of reset very quickly when the supply is very close to the supply monitor's reset threshold. This will cause the /RST line on the device to pulse low and high accordingly, and may affect systems which are connecting /RST to other devices as well.

This issue will be corrected for "REV D" and later devices.

19. C2 Interface

C8051T600/1/2/3/4/5 devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow EPROM programming and in-system debugging with the production part installed in the end application. The C2 interface operates using only two pins: a bi-directional data signal (C2D), and a clock input (C2CK). See the C2 Interface Specification for details on the C2 protocol.

19.1. C2 Interface Registers

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

C2 Register Definition 19.1. C2ADD: C2 Address

								Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands, and to report status information. Address register writes select a register, while address register reads return status information.

Bits7–0: **Write:** Select target Data Read or Data Write register location:

Address	Description
0x00	Selects the Device ID Register (read only)
0x01	Selects the Revision ID Register (read only)
0x02	Selects the C2 Device Control Register
0xDF	Selects the C2 EPROM Programming Control Register
0xBF	Selects the C2 EPROM Data Register
0xAF	Selects the C2 EPROM Address High Byte Register
0xAE	Selects the C2 EPROM Address Low Byte Register
0xA9	Selects the CRC0 Register
0xAA	Selects the CRC1 Register
0xAB	Selects the CRC2 Register
0xAC	Selects the CRC3 Register
0x80	Selects the Port0 SFR
0xF1	Selects the Port0 Input Mode SFR
0xA4	Selects the Port0 Output Mode SFR

Bit7: **Read**: OTPBusy. A ‘1’ indicates that the hardware is performing a write, read, or CRC operation.
Bit6: **Read**: OTPError. A ‘1’ indicates that the most recent OTP operation failed.
Bits5-0: **Read**: Don’t Care.

C2 Register Definition 19.2. DEVICEID: C2 Device ID

								Reset Value
								00010000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

C2ADD: 0x00

This read-only register returns the 8-bit device ID: 0x10.

C2 Register Definition 19.3. REVID: C2 Revision ID

								Reset Value
								Variable
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

C2ADD: 0x01

This read-only register returns the 8-bit revision ID: For example, Revision A = 0x00.

C2 Register Definition 19.4. DEVCTL: C2 Device State

								Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

C2ADD: 0x02

Bits7–0	DEVSTATE: Device Control Register This register is used to halt the device for EPROM operations via the C2 interface. Refer to the OTP chapter for more information.

C2 Register Definition 19.5. EPCTL: C2 EPROM Programming Control

								Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

C2ADD: 0xDF

Bits7–0	EPCTL: EPROM Programming Control Register This register is used to enable EPROM programming via the C2 interface. Refer to the OTP chapter for more information.

C2 Register Definition 19.6. EPDAT: C2 EPROM Data

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset Value
								00000000

C2ADD: 0xBF

Bits7–0: EPDAT: C2 EPROM Data Register
This register is used to pass EPROM data during C2 EPROM operations.

C2 Register Definition 19.7. EPADDRH: C2 EPROM Address High Byte

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset Value
								00000000

C2ADD: 0xAF

Bits7–0: EPADDRH: C2 EPROM Address High Byte Register
This register is used to set the EPROM address location during C2 EPROM operations.

C2 Register Definition 19.8. EPADDRL: C2 EPROM Address Low Byte

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset Value
								00000000

C2ADD: 0xAE

Bits7–0: EPADDRL: C2 EPROM Address Low Byte Register
This register is used to set the EPROM address location during C2 EPROM operations.

C2 Register Definition 19.9. CRC0: CRC Byte 0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Reset Value: 00000000

C2ADD: 0xA9

Bits7–0: CRC Byte 0.
A write to this register initiates a 16-bit CRC of one 256-byte block of EPROM memory. The byte written to CRC0 is the upper byte of the 16-bit address where the CRC will begin. The lower byte of the beginning address is always 0x00. When complete, the 16-bit result will be available in CRC1 (MSB) and CRC0 (LSB). See **Section “11.3. Program Memory CRC” on page 89**.

C2 Register Definition 19.10. CRC1: CRC Byte 1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Reset Value: 00000000

C2ADD: 0xAB

Bits7–0: CRC Byte 1.
A write to this register initiates a 32-bit CRC on the entire program memory space. The CRC begins at address 0x0000. When complete, the 32-bit result is stored in CRC3 (MSB), CRC2, CRC1, and CRC0 (LSB). See **Section “11.3. Program Memory CRC” on page 89**.

C2 Register Definition 19.11. CRC2: CRC Byte 2

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Reset Value: 00000000

C2ADD: 0xAC

Bits7–0: CRC Byte 2. See **Section “11.3. Program Memory CRC” on page 89**.

C2 Register Definition 19.12. CRC3: CRC Byte 3

								Reset Value
								00000000
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
							C2ADD:	0xAD

Bits7–0: CRC Byte 3. See **Section “11.3. Program Memory CRC” on page 89**.

19.2. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and EPROM programming functions may be performed. This is possible because C2 communication is typically performed when the device is in the halt state, where all on-chip peripherals and user software are stalled. In this halted state, the C2 interface can safely 'borrow' the C2CK (normally $\overline{\text{RST}}$) and C2D (normally P0.7) pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application when performing debug functions. These external resistors are not necessary for production boards. A typical isolation configuration is shown in Figure 19.1.

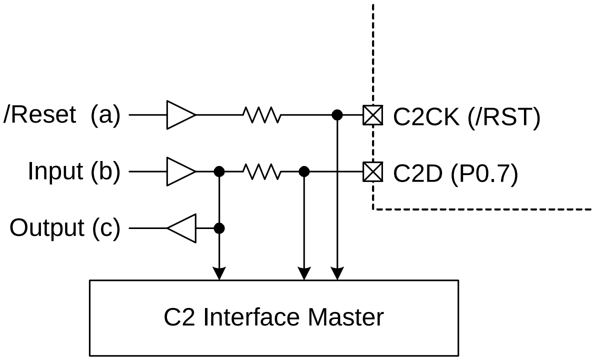


Figure 19.1. Typical C2 Pin Sharing

The configuration in Figure 19.1 assumes the following:

1. The user input (b) cannot change state while the target device is halted.
2. The $\overline{\text{RST}}$ pin on the target device is used as an input only.

Additional resistors may be necessary depending on the specific application.

NOTES:

CONTACT INFORMATION

Silicon Laboratories Inc.

400 West Cesar Chavez
Austin, TX 78701
Email: MCUinfo@silabs.com
Internet: www.silabs.com

The information in this document is believed to be accurate in all respects at the time of publication but is subject to change without notice. Silicon Laboratories assumes no responsibility for errors and omissions, and disclaims responsibility for any consequences resulting from the use of information included herein. Additionally, Silicon Laboratories assumes no responsibility for the functioning of undescribed features or parameters. Silicon Laboratories reserves the right to make changes without further notice. Silicon Laboratories makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Silicon Laboratories assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. Silicon Laboratories products are not designed, intended, or authorized for use in applications intended to support or sustain life, or for any other application in which the failure of the Silicon Laboratories product could create a situation where personal injury or death may occur. Should Buyer purchase or use Silicon Laboratories products for any such unintended or unauthorized application, Buyer shall indemnify and hold Silicon Laboratories harmless against all claims and damages.

Silicon Laboratories and Silicon Labs are trademarks of Silicon Laboratories Inc.
Other products or brandnames mentioned herein are trademarks or registered trademarks of their respective holders.

Downloaded from Datasheet.su