C8051F39x/37x

50 MIPS 16 kB Flash, 512B EEPROM Mixed-Signal MCU

Analog Peripherals ('F390/2/4/6/8 and 'F370/4)
- **10-Bit ADC**
  - Programmable throughput up to 500 ksps
  - Up to 16 external inputs, programmable as single-ended or differential
  - Reference from on-chip voltage reference, V_{DD} or external VREF pin
  - Internal or external start of conversion sources
- **Two 10-Bit Current Output DACs**
  - Supports output through resets for continuous operation
- **Comparator**
  - Programmable hysteresis and response time
  - Configurable as interrupt or reset source
- **Precision Temperature Sensor**
  - Accurate to ±2 °C across temperature range with no user calibration

On-Chip Debug
- On-chip debug circuitry facilitates full speed, non-intrusive in-system debug (no emulator required)
- Provides breakpoints, single stepping, inspect/modify memory and registers

Low Power
- 160 µA/MHz Active mode with 49 MHz internal precision oscillator
- 200 nA Stop mode current

Temperature Range
- –40 to +85 °C ('F37x)
- –40 to +105 °C ('F39x)

Package
- 24-Pin QFN ('F390/1/4/5 and 'F37x)
- 20-Pin QFN ('F392/3/6/7/8/9)

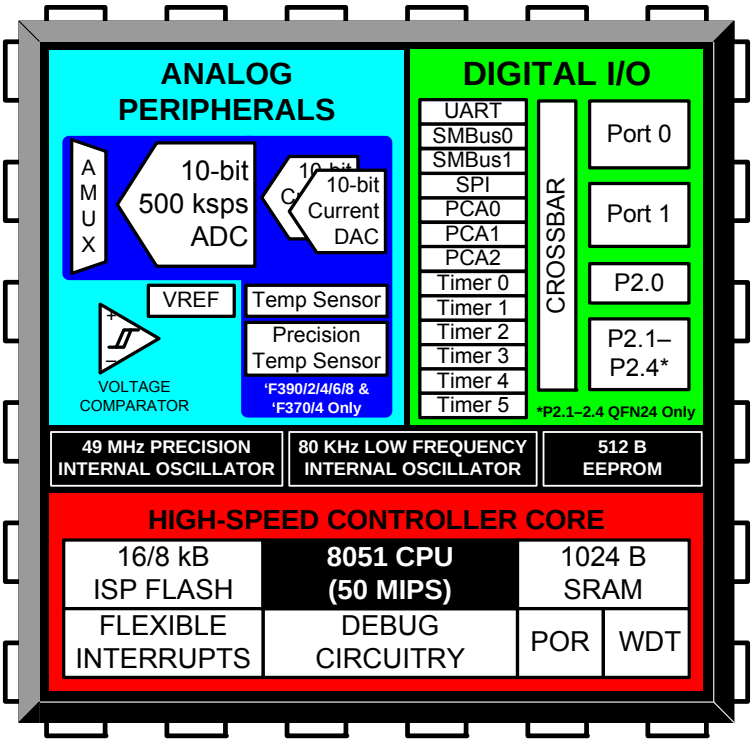
High-Speed 8051 µC Core
- Pipelined instruction architecture; executes 70% of instructions in 1 or 2 system clocks
- Up to 50 MIPS throughput with 50 MHz clock
- Expanded interrupt handler

Memory
- Up to 1 kB internal data RAM (256 + 768)
- Up to 16 kB Flash; In-system programmable in 512-byte Sectors
- 512 bytes of byte-programmable EEPROM; 1 million write/erase cycles ('F37x)

Digital Peripherals
- 21 or 17 Port I/O
- UART, 2 SMBus (I^2C compatible), and SPI serial ports
- Six general purpose 16-bit counter/timers
- 16-Bit programmable counter array (PCA) with three capture/compare modules and PWM functionality

Clock Sources
- 49 MHz ±2% precision internal oscillator
  - Supports crystal-less UART operation
  - Low-power suspend mode with fast wake time
- 80 kHz low-frequency, low-power oscillator
- External oscillator: Crystal, RC, C, or CMOS clock
- Can switch between clock sources on-the-fly; useful in power saving modes

Supply Voltage 1.8 to 3.6 V
- Built-in voltage supply monitor



1. System Overview

C8051F39x/37x devices are fully integrated mixed-signal System-on-a-Chip MCUs. Highlighted features are listed below. Refer to Section “2. Ordering Information” on page 20 for specific product feature selection and part ordering numbers.

- High-speed pipelined 8051-compatible microcontroller core (up to 50 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- True 10-bit 500 ksps 20 or 16-channel single-ended/differential ADC with analog multiplexer
- Two 10-bit Current Output DACs
- Precision temperature sensor with ±2 °C absolute accuracy
- Precision programmable 49 MHz internal oscillator
- Low-power, low-frequency oscillator
- 16 kB of on-chip Flash memory
- 1024 bytes of on-chip RAM
- Co-packaged with 512 bytes of EEPROM memory, accessible via I^2C (C8051F37x)
- Two SMBus/I^2C, UART, and SPI serial interfaces implemented in hardware
- Six general-purpose 16-bit timers
- Programmable Counter/Timer Array (PCA) with three capture/compare modules and Watchdog Timer function
- On-chip Power-On Reset, V_{DD} Monitor, and Temperature Sensor
- On-chip Voltage Comparator
- 21 or 17 Port I/O
- Low-power suspend mode with fast wake-up time

With on-chip Power-On Reset, V_{DD} monitor, Watchdog Timer, and clock oscillator, the C8051F39x/37x devices are truly stand-alone System-on-a-Chip solutions. The Flash memory can be reprogrammed even in-circuit, providing non-volatile data storage, and also allowing field upgrades of the 8051 firmware. User software has complete control of all peripherals, and may individually shut down any or all peripherals for power savings.

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

The C8051F37x devices are specified for 1.8 to 3.6 V operation over the industrial temperature range (–40 to +85 °C), while the C8051F39x devices operate over an extended temperature range (-40 to +105 °C). The C8051F392/3/6/7/8/9 are available in a 20-pin QFN package and the C8051F390/1/4/5 and C8051F37x are available in a 24-pin QFN package. Both package options are lead-free and RoHS compliant. See Section “2. Ordering Information” on page 20 for ordering information. Block diagrams are included in Figure 1.1, Figure 1.2 and Figure 1.3.

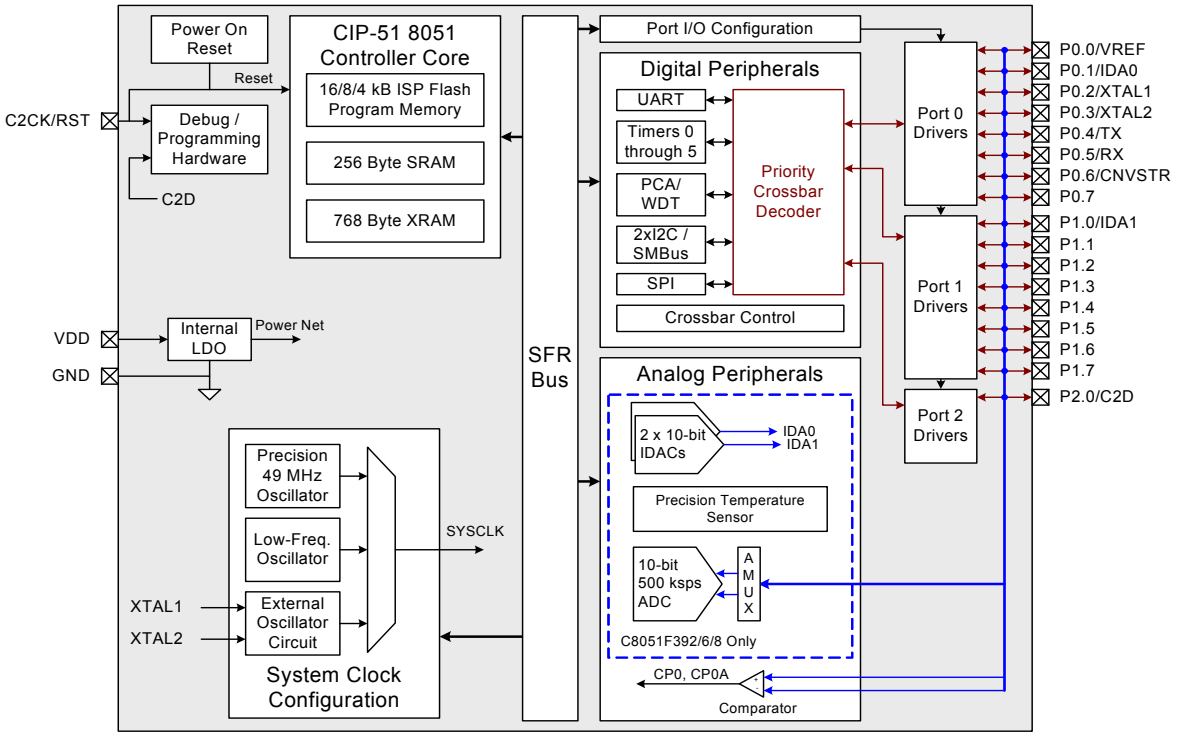


Figure 1.1. C8051F392/3/6/7/8/9 Block Diagram

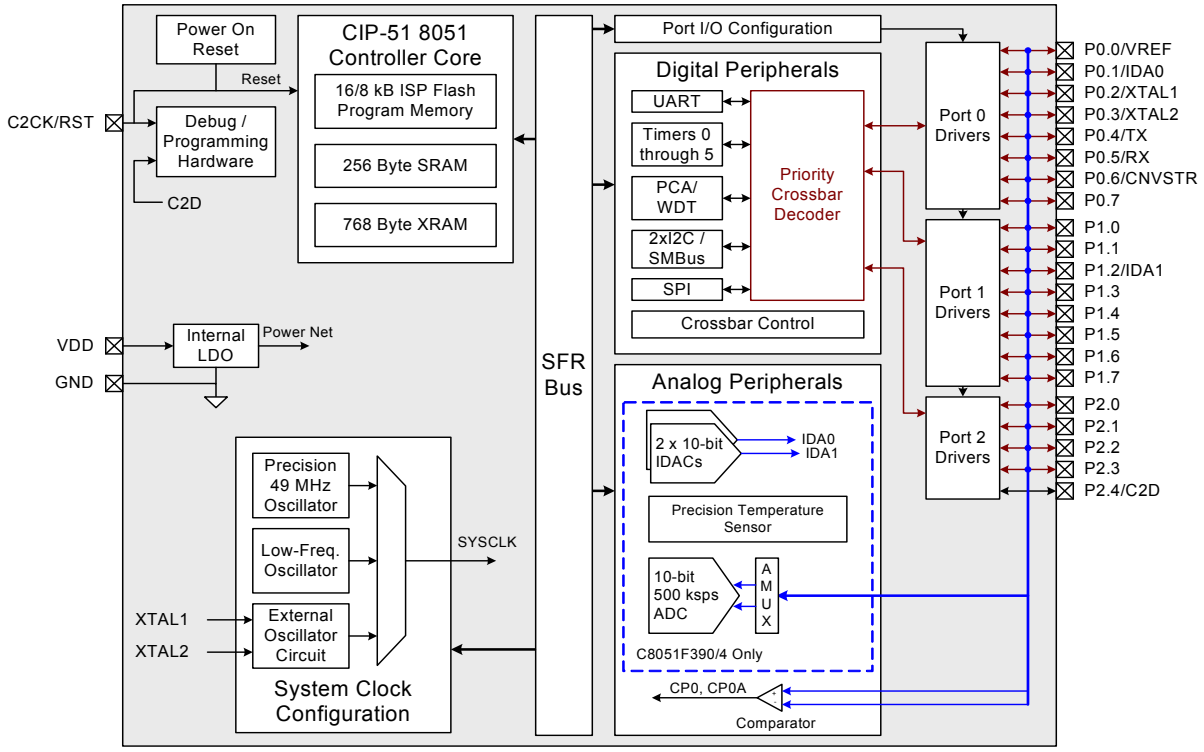


Figure 1.2. C8051F390/1/4/5 Block Diagram

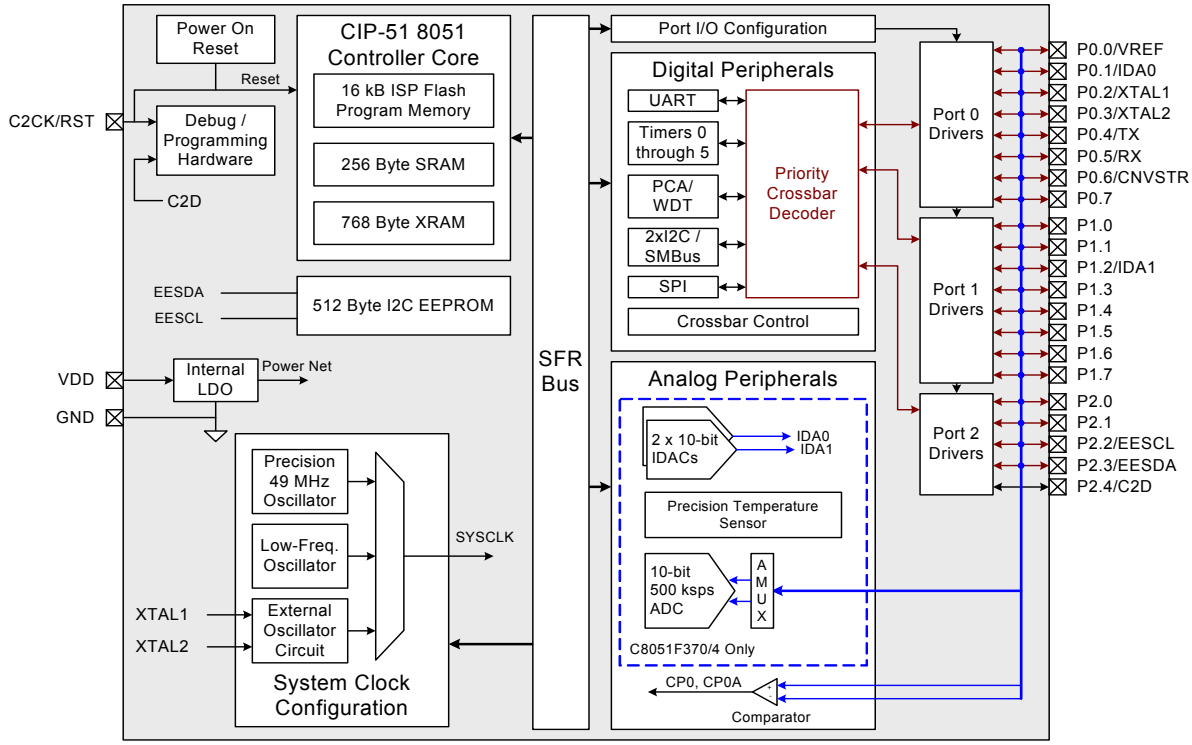


Figure 1.3. C8051F370/1/4/5 Block Diagram

2. Ordering Information

The following features are common to all device in this family:

- 50 MIPS throughput (peak)
- 1 kB of RAM (256 internal bytes and 768 XRAM bytes)
- Calibrated internal 49 MHz oscillator
- Internal 80 kHz oscillator
- Two SMBus/I^2C
- Enhanced SPI, Enhanced UART
- Six Timers
- Three Programmable Counter Array channels
- Analog Comparator
- Lead-free / RoHS Compliant

Table 2.1 shows the features that differentiate the devices in this family.

Table 2.1. Product Selection Guide

Ordering Part Number	Flash Memory (Bytes)	EEPROM (Bytes)	Digital Port I/Os	10-bit ADC Channels	10-bit DAC Channels	On-Chip Voltage Reference	Precision Temperature Sensor	Package 4x4 mm
C8051F370-A-GM	16k	512	21	20	2	Y	Y	QFN-24
C8051F371-A-GM	16k	512	21	—	—	—	—	QFN-24
C8051F374-A-GM	8k	512	21	20	2	Y	Y	QFN-24
C8051F375-A-GM	8k	512	21	—	—	—	—	QFN-24
C8051F390-A-GM	16k	—	21	20	2	Y	Y	QFN-24
C8051F391-A-GM	16k	—	21	—	—	—	—	QFN-24
C8051F392-A-GM	16k	—	17	16	2	Y	Y	QFN-20
C8051F393-A-GM	16k	—	17	—	—	—	—	QFN-20
C8051F394-A-GM	8k	—	21	20	2	Y	Y	QFN-24
C8051F395-A-GM	8k	—	21	—	—	—	—	QFN-24
C8051F396-A-GM	8k	—	17	16	2	Y	Y	QFN-20
C8051F397-A-GM	8k	—	17	—	—	—	—	QFN-20
C8051F398-A-GM	4k	—	17	16	2	Y	Y	QFN-20
C8051F399-A-GM	4k	—	17	—	—	—	—	QFN-20

3. C8051F33x Compatibility

The C8051F39x/37x family is designed to be a pin and code compatible replacement for the C8051F33x device family, with an enhanced feature set. The C8051F39x/37x device should function as a drop-in replacement for the C8051F33x devices in most applications. Table 3.1 lists recommended replacement part numbers for C8051F33x devices. See “3.1. Hardware Incompatibilities” to determine if any changes are necessary when upgrading an existing C8051F33x design to the C8051F39x/37x.

Table 3.1. C8051F33x Replacement Part Numbers

C8051F33x Part Number	C8051F39x/37x Part Number
C8051F330-GM	C8051F396-A-GM
C8051F331-GM	C8051F397-A-GM
C8051F332-GM	C8051F398-A-GM
C8051F333-GM	C8051F399-A-GM
C8051F334-GM	C8051F398-A-GM
C8051F335-GM	C8051F399-A-GM
C8051F336-GM	C8051F392-A-GM
C8051F337-GM	C8051F393-A-GM
C8051F338-GM	C8051F390-A-GM
C8051F339-GM	C8051F391-A-GM

3.1. Hardware Incompatibilities

While the C8051F39x/37x family includes a number of new features not found on the C8051F33x family, there are some differences that should be considered for any design port.

- **Internal High-Frequency Oscillator**: The undivided high-frequency oscillator on the C8051F39x/37x is 49 MHz, whereas the undivided high-frequency oscillator on the C8051F33x is 24.5 MHz. Correspondingly, the internal high frequency divide ratios (IFCN) have doubled. Thus, firmware written for the C8051F33x where the CLKSL[1:0] = 00b will result in the same SYSCLK frequency on the C8051F39x/37x.
- **Fabrication Technology**: The C8051F39x/37x is manufactured using a different technology process than the C8051F33x. As a result, many of the electrical performance parameters will have subtle differences. These differences should not affect most systems but it is nonetheless important to review the electrical parameters for any blocks that are used in the design, and ensure they are compatible with the existing hardware.
- **5 V Tolerance**: The port I/O pins on the C8501F39x/37x are not 5 V tolerant, whereas the port I/O pins on the C8051F33x are 5 V tolerant.
- **Lock Byte Address**: The lock byte for C8051F39x/7x devices with 16 kB of Flash resides at address 0x3FFF, whereas the lock byte for C8051F33x devices with 16 kB of Flash resides at address 0x3DFF. The lock byte for C8051F39x/7x devices with 8 kB of Flash resides at address 0x1FFF, whereas the lock byte for C8051F33x devices with 8 kB of Flash resides at address 0x1DFF.

4. Pin Definitions

Table 4.1. Pin Definitions for the C8051F39x/37x

Name	Pin 'F392/3/6/7/8/9	Pin 'F390/1/4/5	Pin 'F370/1/4/5	Type	Description
V_{DD}	3	4	4		Power Supply Voltage.
GND	2	3	3		Ground. This ground connection is required. The center pad may optionally be connected to ground also.
$\overline{RST}$/ C2CK	4	5	5	D I/O D I/O	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. Clock signal for the C2 Debug Interface.
C2D	5	6	6	D I/O	Bi-directional data signal for the C2 Debug Interface. Shared with P2.0 on 20-pin packaging and P2.4 on 24-pin packaging.
P0.0/ VREF	1	2	2	D I/O or A In A In	Port 0.0. External VREF input.
P0.1 IDA0	20	1	1	D I/O or A In A Out	Port 0.1. IDA0 Output.
P0.2/ XTAL1	19	24	24	D I/O or A In A In	Port 0.2. External Clock Input. This pin is the external oscillator return for a crystal or resonator.
P0.3/ XTAL2	18	23	23	D I/O or A In A I/O or D In	Port 0.3. External Clock Output. For an external crystal or resonator, this pin is the excitation driver. This pin is the external clock input for CMOS, capacitor, or RC oscillator configurations.
P0.4	17	22	22	D I/O or A In	Port 0.4.
P0.5	16	21	21	D I/O or A In	Port 0.5.

Table 4.1. Pin Definitions for the C8051F39x/37x (Continued)

Name	Pin ‘F392/3/6/7/8/9	Pin ’F390/1/4/5	Pin ’F370/1/4/5	Type	Description
P0.6/ CNVSTR	15	20	20	D I/O or A In D In	Port 0.6. ADC0 External Convert Start or IDA0 Update Source Input.
P0.7	14	19	19	D I/O or A In	Port 0.7.
P1.0 IDA1	13	—	—	D I/O or A In A Out	Port 1.0. IDA1 Output.
P1.0		18	18	D I/O or A In	Port 1.0.
P1.1	12	17	17	D I/O or A In	Port 1.1.
P1.2 IDA1	-	16	16	D I/O or A In A Out	Port 1.2. IDA1 Output.
P1.2	11	—	—	D I/O or A In	Port 1.2.
P1.3	10	15	15	D I/O or A In	Port 1.3.
P1.4	9	14	14	D I/O or A In	Port 1.4.
P1.5	8	13	13	D I/O or A In	Port 1.5.
P1.6	7	12	12	D I/O or A In	Port 1.6.
P1.7	6	11	11	D I/O or A In	Port 1.7.
P2.0	5	10	10	D I/O or A In	Port 2.0. (Also C2D on 20-pin Packaging)
P2.1	—	9	9	D I/O or A In	Port 2.1.

Table 4.1. Pin Definitions for the C8051F39x/37x (Continued)

Name	Pin ‘F392/3/6/7/8/9	Pin ’F390/1/4/5	Pin ’F370/1/4/5	Type	Description
P2.2	—	8	—	D I/O or A In	Port 2.2.
P2.2 EESCL	-	—	8	D I/O or A In D I/O	Port 2.2. EEPROM SCL Connection.
P2.3	—	7	—	D I/O or A In	Port 2.3.
P2.3 EESDA	-	—	7	D I/O or A In D I/O	Port 2.3. EEPROM SDA Connection.
P2.4	—	6	6	D I/O	Port 2.4. (Also C2D on 24-pin Packaging)

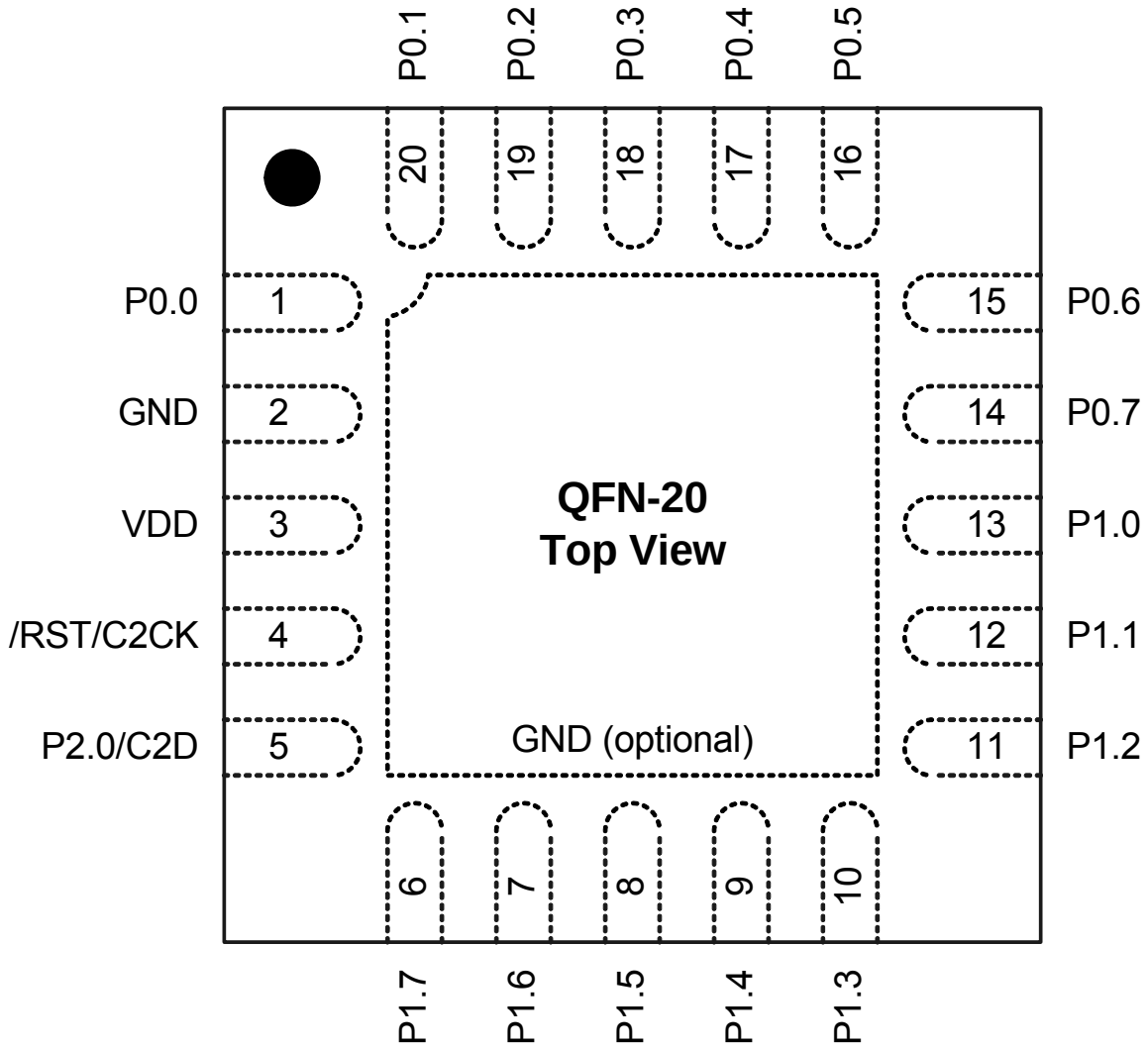


Figure 4.1. C8051F392/3/6/7/8/9 QFN-20 Pinout Diagram (Top View)

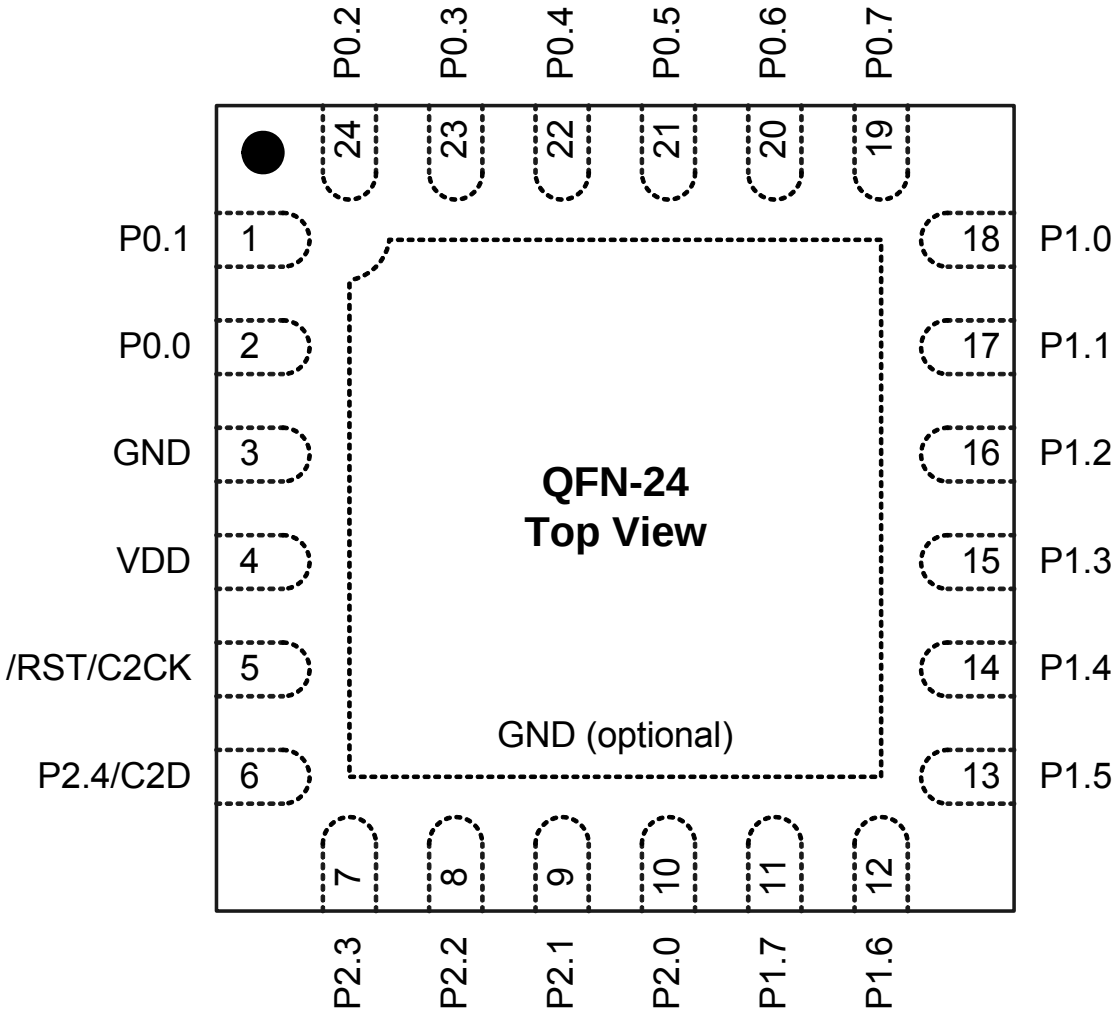


Figure 4.2. C8051F390/1/4/5 Pinout Diagram (Top View)

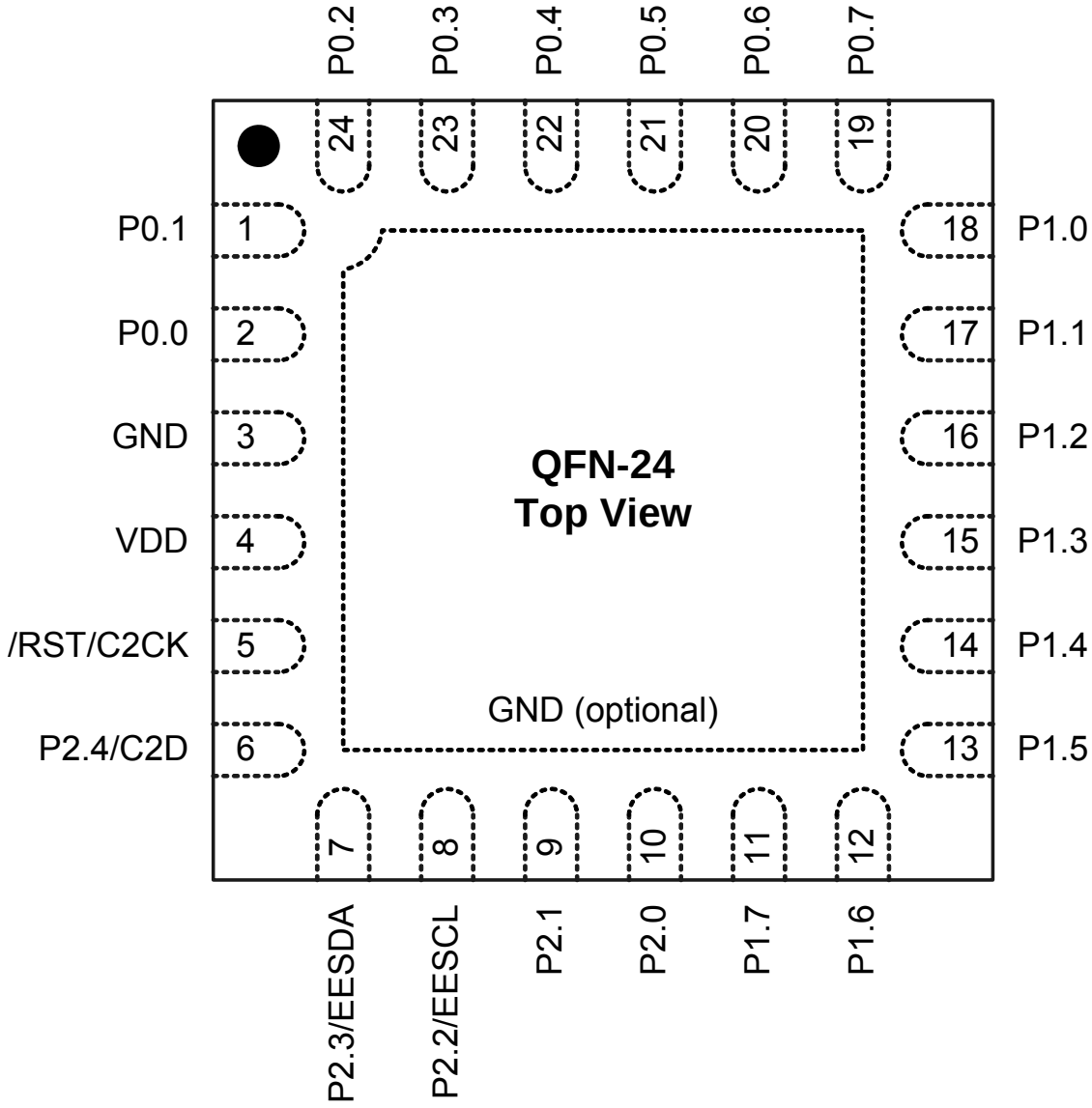


Figure 4.3. C8051F370/1/4/5 Pinout Diagram (Top View)

5. QFN-20 Package Specifications

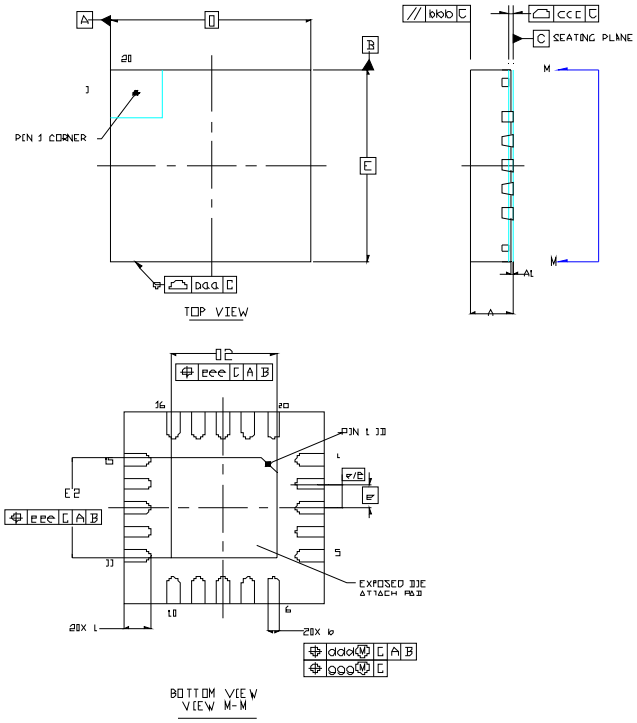


Figure 5.1. QFN-20 Package Drawing

Table 5.1. QFN-20 Package Dimensions

Dimension	Min	Typ	Max	Dimension	Min	Typ	Max
A	0.80	0.85	0.90	L	0.50	0.55	0.60
A1	0.00	0.035	0.05	aaa	—	—	0.10
b	0.20	0.25	0.30	bbb	—	—	0.10
D	4.00 BSC.			ccc	—	—	0.08
D2	2.00	2.10	2.20	ddd	—	—	0.10
e	0.50 BSC.			eee	—	—	0.10
E	4.00 BSC.			ggg			0.05
E2	2.00	2.10	2.20				

Notes:

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This drawing conforms to the JEDEC Solid State Outline MO-220, variation VGGD except for custom features D2, E2, Z, Y, and L which are toleranced per supplier designation.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

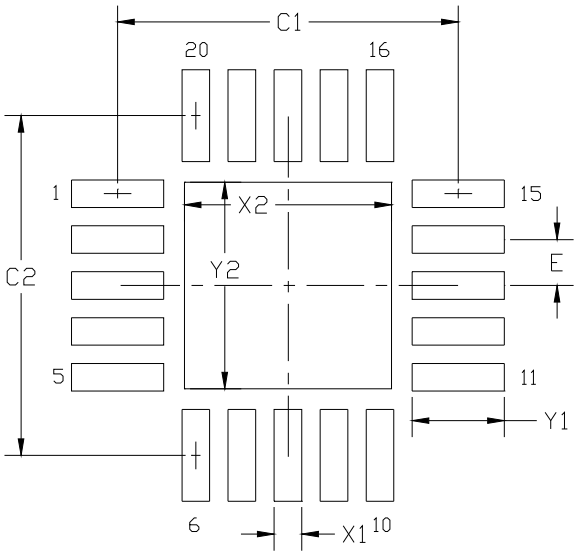


Figure 5.2. QFN-20 Recommended PCB Land Pattern

Table 5.2. QFN-20 PCB Land Pattern Dimensions

Dimension	Max
C1	3.80
C2	3.80
E	0.50
X1	0.30

Dimension	Max
X2	2.20
Y1	1.00
Y2	2.20

Notes:

General

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing is per the ANSI Y14.5M-1994 specification.
3. This Land Pattern Design is based on the IPC-7351 guidelines.

Solder Mask Design

4. All metal pads are to be non-solder mask defined (NSMD). Clearance between the solder mask and the metal pad is to be 60 µm minimum, all the way around the pad.

Stencil Design

5. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
6. The stencil thickness should be 0.125 mm (5 mils).
7. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pins.
8. A 2x2 array of 0.95mm openings on a 1.1 mm pitch should be used for the center pad to assure the proper paste volume (71% Paste Coverage).

Card Assembly

9. A No-Clean, Type-3 solder paste is recommended.
10. The recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

6. QFN-24 Package Specifications

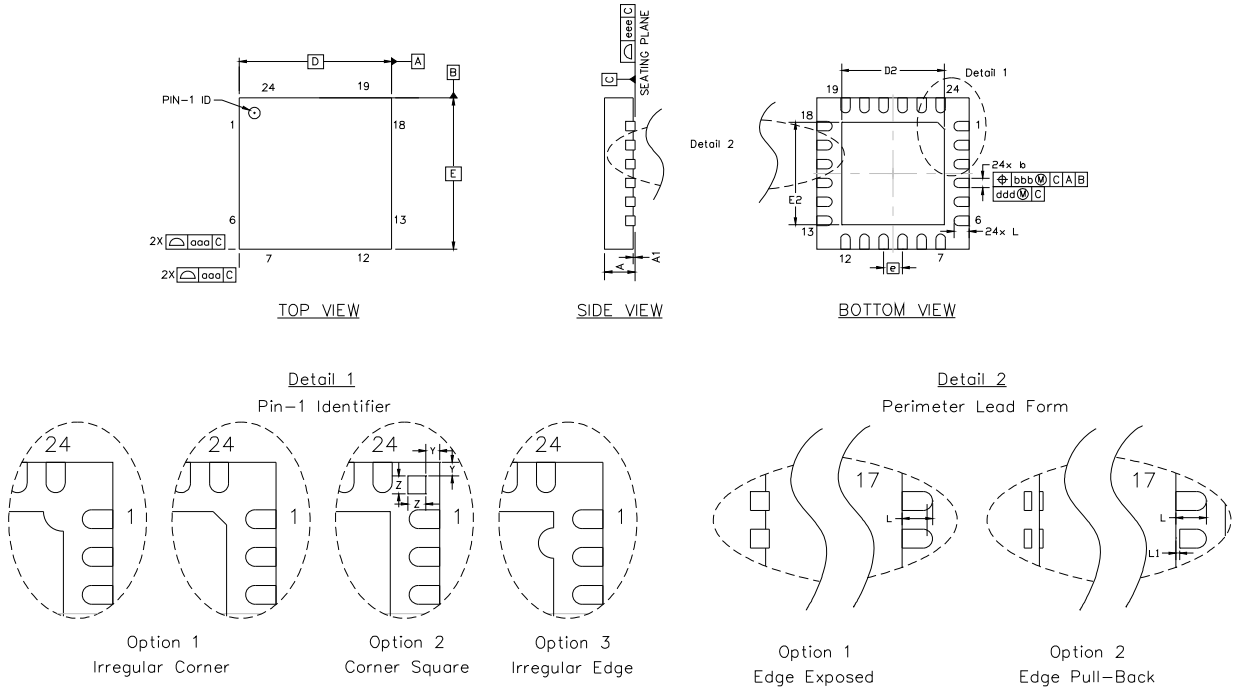


Figure 6.1. QFN-24 Package Drawing

Table 6.1. QFN-24 Package Dimensions

Dimension	Min	Typ	Max		Dimension	Min	Typ	Max
A	0.70	0.75	0.80		L	0.30	0.40	0.50
A1	0.00	0.02	0.05		L1	0.00	—	0.15
b	0.18	0.25	0.30		aaa	—	—	0.15
D	4.00 BSC.				bbb	—	—	0.10
D2	2.55	2.70	2.80		ddd	—	—	0.05
e	0.50 BSC.				eee	—	—	0.08
E	4.00 BSC.				Z	—	0.24	—
E2	2.55	2.70	2.80		Y	—	0.18	—

Notes:

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This drawing conforms to JEDEC Solid State Outline MO-220, variation WGGD except for custom features D2, E2, Z, Y, and L which are toleranced per supplier designation.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

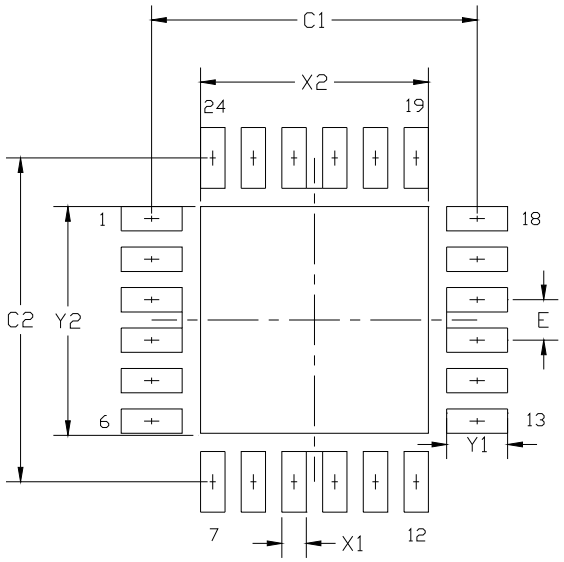


Figure 6.2. QFN-24 Recommended PCB Land Pattern

Table 6.2. QFN-24 PCB Land Pattern Dimensions

Dimension	Min	Max
C1	3.90	4.00
C2	3.90	4.00
E	0.50 BSC	
X1	0.20	0.30

Dimension	Min	Max
X2	2.70	2.80
Y1	0.65	0.75
Y2	2.70	2.80

Notes:

General

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. This Land Pattern Design is based on the IPC-7351 guidelines.

Solder Mask Design

3. All metal pads are to be non-solder mask defined (NSMD). Clearance between the solder mask and the metal pad is to be 60 µm minimum, all the way around the pad.

Stencil Design

4. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
5. The stencil thickness should be 0.125 mm (5 mils).
6. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pads.
7. A 2 x 2 array of 1.10 mm x 1.10 mm openings on a 1.30 mm pitch should be used for the center pad.

Card Assembly

8. A No-Clean, Type-3 solder paste is recommended.
9. The recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

7. Electrical Characteristics

7.1. Absolute Maximum Specifications

Table 7.1. Absolute Maximum Ratings

Parameter	Test Condition	Min	Typ	Max	Unit
Ambient Temperature under Bias		–55	—	125	°C
Storage Temperature		–65	—	150	°C
Voltage on any Port I/O Pin or $\overline{\text{RST}}$ with respect to GND		–0.3	—	V_{DD} + 0.3	V
Voltage on V_{DD} with Respect to GND		–0.3	—	4.2	V
Maximum Total Current through V_{DD} or GND		—	—	100	mA
Maximum Output Current Sunk by $\overline{\text{RST}}$ 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.

7.2. Electrical Characteristics

Table 7.2. Global Electrical Characteristics

–40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), 50 MHz system clock, unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Supply Voltage (V_{DD})	Normal Operation	V_{RST} [1]	3.0	3.6	V
	Writing or Erasing Flash Memory	1.8	3.0	3.6	V
Digital Supply RAM Data Retention Voltage		—	1.5	—	V
SYSCLK (System Clock) [2]		0	—	50	MHz
T_{SYSH} (SYSCLK High Time)		9.5	—	—	ns
T_{SYSL} (SYSCLK Low Time)		9.5	—	—	ns
Specified Operating Temperature Range	C8051F39x	–40	—	+105	°C
	C8051F37x	–40	—	+85	°C
Digital Supply Current—CPU Active (Normal Mode, Fetching Instructions from Flash)					
I_{DD} [3, 4]	V_{DD} = 3.6 V, F = 50 MHz	—	7.1	7.8	mA
	V_{DD} = 3.0 V, F = 50 MHz	—	7.0	7.7	mA
	V_{DD} = 3.6 V, F = 25 MHz	—	4.6	5.2	mA
	V_{DD} = 3.0 V, F = 25 MHz	—	4.5	5.1	mA
	V_{DD} = 3.6 V, F = 1 MHz	—	0.35	—	mA
	V_{DD} = 3.0 V, F = 1 MHz	—	0.35	—	mA
	V_{DD} = 3.0 V, F = 80 kHz	—	0.25	—	mA

Notes:

1. Given in Table 7.4 on page 36.
2. SYSCLK must be at least 32 kHz to enable debugging.
3. Based on device characterization data; Not production tested.
4. Digital Supply Current depends on the particular code being executed. The values in this table are obtained with the CPU executing an “sjmp $” loop, which is the compiled form of a while(1) loop in C. One iteration requires 3 CPU clock cycles, and the Flash memory is read on each cycle. The supply current will vary slightly based on the physical location of the sjmp instruction and the number of Flash address lines that toggle as a result. In the worst case, current can increase by up to 30% if the sjmp loop straddles a 64-byte Flash address boundary (e.g., 0x007F to 0x0080). Real-world code with larger loops and longer linear sequences will have few transitions across the 64-byte boundary.

Table 7.2. Global Electrical Characteristics (Continued)
–40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), 50 MHz system clock, unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Digital Supply Current—CPU Inactive (Idle Mode, Not Fetching Instructions from Flash)					
I_{DD} [3]	V_{DD} = 3.6 V, F = 50 MHz	—	3.9	4.5	mA
	V_{DD} = 3.0 V, F = 50 MHz	—	3.8	4.4	mA
	V_{DD} = 3.6 V, F = 25 MHz	—	2.1	2.5	mA
	V_{DD} = 3.0 V, F = 25 MHz	—	2.0	2.4	mA
	V_{DD} = 3.6 V, F = 1 MHz	—	0.15	—	mA
	V_{DD} = 3.0 V, F = 1 MHz	—	0.15	—	mA
	V_{DD} = 3.0 V, F = 80 kHz	—	0.1	—	mA
Digital Supply Current (Suspend Mode)	Oscillator not running, V_{DD} Monitor Disabled, Regulator running (STOPCF = 0)	—	73	—	µA
Digital Supply Current (Stop Mode)	Oscillator not running, V_{DD} Monitor Disabled, Regulator running (STOPCF = 0)	—	75	—	µA
Digital Supply Current (Stop Mode, Regulator Shutdown)	Oscillator not running, V_{DD} Monitor Disabled, Regulator Shutdown (STOPCF = 1)	—	0.2	—	µA

Notes:
1. Given in Table 7.4 on page 36.
2. SYSCLK must be at least 32 kHz to enable debugging.
3. Based on device characterization data; Not production tested.
4. Digital Supply Current depends on the particular code being executed. The values in this table are obtained with the CPU executing an “sjmp $” loop, which is the compiled form of a while(1) loop in C. One iteration requires 3 CPU clock cycles, and the Flash memory is read on each cycle. The supply current will vary slightly based on the physical location of the sjmp instruction and the number of Flash address lines that toggle as a result. In the worst case, current can increase by up to 30% if the sjmp loop straddles a 64-byte Flash address boundary (e.g., 0x007F to 0x0080). Real-world code with larger loops and longer linear sequences will have few transitions across the 64-byte boundary.

Table 7.3. Port I/O DC Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameters	Test Condition	Min	Typ	Max	Unit
Standard Port I/O					
Output High Voltage	I_{OH} = –3 mA, Port I/O push-pull	V_{DD} – 0.7	—	—	V
	I_{OH} = –10 µA, Port I/O push-pull	V_{DD} – 0.1	—	—	V
	I_{OH} = –10 mA, Port I/O push-pull	—	V_{DD} – 0.8	—	V
Output Low Voltage	I_{OL} = 8.5 mA	—	—	0.6	V
	I_{OL} = 10 µA	—	—	0.1	V
	I_{OL} = 10 mA, 1.8 V ≤ V_{DD} < 2.7 V	—	0.8	—	V
	I_{OL} = 25 mA, 2.7 V ≤ V_{DD} ≤ 3.6 V	—	1.0	—	V
Input High Voltage	1.8 V ≤ V_{DD} < 2.7 V	V_{DD} – 0.4	—	—	V
	2.7 V ≤ V_{DD} ≤ 3.6 V	V_{DD} – 0.5	—	—	V
Input Low Voltage	1.8 V ≤ V_{DD} < 2.7 V	—	—	0.5	V
	2.7 V ≤ V_{DD} ≤ 3.6 V	—	—	0.6	V
Input Leakage Current	Weak Pullup Off	—	—	±1	µA
	Weak Pullup On, V_{IN} = 0 V	—	20	100	µA
EESDA and EESCL (C8051F37x Only)*					
Output Low Voltage (EESDA)	I_{OL} = 0.15 mA, V_{DD} = 1.8 V	—	—	0.2	V
Output Low Voltage (EESDA)	I_{OL} = 2.1 mA, V_{DD} = 3 V	—	—	0.4	V
Output Leakage Current (EESDA)	EEPUE = 0, V_{DD} = 3.6 V, 0 V ≤ V_{OUT} ≤ V_{DD}	—	—	2	µA
Input High Voltage		V_{DD} x 0.7	—	—	V
Input Low Voltage		—	—	V_{DD} x 0.3	V
Input Leakage Current	EEPUE = 0, Standby, V_{DD} = 3.6 V, 0 V ≤ V_{IN} ≤ V_{DD}	—	—	±3	µA
Note: Applicable when interfacing to the C8051F37x EEPROM. Otherwise, standard port I/O characteristics apply.					

Table 7.4. Reset Electrical Characteristics

–40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
$\overline{RST}$ Output Low Voltage	I_{OL} = 4 mA, V_{DD} = 1.8 to 3.6 V	—	—	0.6	V
$\overline{RST}$ Input Low Voltage		—	—	0.6	V
$\overline{RST}$ Input Pullup Current	$\overline{RST}$ = 0.0 V	—	20	100	µA
V_{DD} POR Threshold (V_{RST})	V_{RST_LOW}	1.7	1.75	1.8	V
	V_{RST_HIGH}	2.4	2.55	2.7	V
Missing Clock Detector Time-out	Time from last system clock rising edge to reset initiation	80	580	800	µs
Reset Time Delay	Delay between release of any reset source and code execution at location 0x0000	—	—	40	µs
Minimum $\overline{RST}$ Low Time to Generate a System Reset		15	—	—	µs
V_{DD} Monitor Turn-on Time		100	—	—	µs
V_{DD} Monitor Supply Current		—	20	50	µA

Table 7.5. Flash Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Flash Size	C8051F390/1/2/3, C8051F370/1		16384		Bytes
	C8051F394/5/6/7, C8051F374/5		8192		Bytes
	C8051F398/9		4096		Bytes
Endurance		20000	100000	—	Erase/Write
Erase Cycle Time		23	25	27	ms
Write Cycle Time		58	61	64	µs

Table 7.6. EEPROM Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
EEPROM Size			512		Bytes
Endurance		1000000	—	—	Write Cycles
Write Cycle Time	16-byte page	—	—	3.5	ms
EESCL Clock Frequency		—	—	400	kHz
Supply Current		—	—	3	µA
		—	—	2	mA
	V_{DD} = 3.6 V, Write	—	—	3	mA

Table 7.7. Internal High-Frequency Oscillator Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), using factory-calibrated settings, unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Oscillator Frequency	C8051F390/1/2/3, C8051F370/1	48	49	50	MHz
Oscillator Supply Current (from V_{DD})	C8051F394/5/6/7, C8051F374/5	—	840	880	µA
Power Supply Sensitivity	C8051F398/9	—	0.12	—	%/V
Temperature Sensitivity		—	90	—	ppm/°C

Table 7.8. Internal Low-Frequency Oscillator Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), using factory-calibrated settings, unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Oscillator Frequency	C8051F390/1/2/3, C8051F370/1	75	80	85	kHz
Oscillator Supply Current (from V_{DD})	C8051F394/5/6/7, C8051F374/5	—	5.5	12	µA
Power Supply Sensitivity	C8051F398/9	—	0.05	—	%/V
Temperature Sensitivity		—	160	—	ppm/°C

Table 7.9. Internal Low-Power Oscillator Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), using factory-calibrated settings, unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Oscillator Frequency	C8051F390/1/2/3, C8051F370/1	18.5	20	21.5	MHz
Power Supply Sensitivity	C8051F394/5/6/7, C8051F374/5	—	0.1	—	%/V
Temperature Sensitivity	C8051F398/9	—	60	—	ppm/°C

Table 7.10. ADC0 Electrical Characteristics

V_{DD} = 3.0 V, VREF = 2.40 V (REFSL = 0), –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
DC Accuracy					
Resolution	C8051F394/5/6/7, C8051F374/5		10		bits
Integral Nonlinearity	C8051F398/9	—	<±0.5	±2.0	LSB
Differential Nonlinearity		—	<±0.5	±1	LSB
Offset Error		–2	0	2	LSB
Full Scale Error		–5	–2	1	LSB
Offset Temperature Coefficient		—	0.005	—	LSB/°C
Dynamic performance (10 kHz sine-wave single-ended input, 1 dB below Full Scale, 500 ksps)					
Signal-to-Noise Plus Distortion		55	58	—	dB
Total Harmonic Distortion	Up to the 5th harmonic	—	–73	—	dB
Spurious-Free Dynamic Range		—	68	—	dB
Conversion Rate					
SAR Conversion Clock		—	—	8.33	MHz
Conversion Time in SAR Clocks		13	—	—	clocks
Track/Hold Acquisition Time		300	—	—	ns
Throughput Rate		—	—	500	ksps
Analog Inputs					
ADC Input Voltage Range	Single Ended (AIN+ – GND) Differential (AIN+ – AIN–)	0 –VREF	— —	VREF VREF	V V
Absolute Pin Voltage with respect to GND	Single Ended or Differential	0	—	V_{DD}	V
Sampling Capacitance (C_{SAMPLE})		—	5	—	pF
Input Multiplexer Impedance (R_{MUX})		—	1.6	—	kΩ
Power Specifications					
Power Supply Current (V_{DD} supplied to ADC0)	Operating Mode, 500 ksps	—	860	1010	µA
Power Supply Rejection	Single Ended (AIN+ – GND) Differential (AIN+ – AIN–)	— —	1.15 2.45	— —	mV/V mV/V

Table 7.11. ADC Temperature Sensor Electrical Characteristics

V_{DD} = 3.0 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Linearity		—	0.75	—	°C
Slope		—	2.92	—	mV/°C
Slope Error*		—	70	—	µV/°C
Offset	Temp = 0 °C	—	785	—	mV
Offset Error*	Temp = 0 °C	—	13	—	mV
Supply Current		—	90	120	µA
Note: Represents one standard deviation from the mean.					

Table 7.12. Precision Temperature Sensor Electrical Characteristics

V_{DD} = 1.8 to 3.6 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Range		–40	—	105	°C
Absolute Error		–2	0.2	+2	°C
Integral Nonlinearity		—	0	±0.4	°C
Resolution			0.0078125		°C
Power Supply Rejection		—	0.05	0.2	°C/V
Supply Current		—	230	280	µA
Clock Frequency (F_{TS0})		320	520	730	kHz

Table 7.13. Voltage Reference Electrical Characteristics

V_{DD} = 3.0 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Internal Reference (REFBE = 1)					
Output Voltage	2.4 V Setting	2.37	2.4	2.43	V
	1.2 V Setting	1.18	1.2	1.22	V
VREF Short-Circuit Current		—	—	9	mA
VREF Temperature Coefficient		—	33	—	ppm/°C
Load Regulation	Load = 0 to 200 µA to AGND	—	6	—	µV/µA
VREF Turn-on Time 1	4.7 µF tantalum, 0.1 µF ceramic bypass	—	1.5	—	ms
VREF Turn-on Time 2	0.1 µF ceramic bypass	—	110	—	µs
Power Supply Rejection	2.4 V Setting	—	3.5	—	mV/V
	1.2 V Setting	—	1.1	—	mV/V
External Reference (REFBE = 0)					
Input Voltage Range		1.0	—	V_{DD}	V
Input Current	Sample Rate = 200 ksps; VREF = 3.0 V	—	3	—	µA
Power Specifications					
Supply Current	REFBE = “1” or TEMPE = “1”	—	70	100	µA

Table 7.14. Voltage Regulator Electrical Characteristics

V_{DD} = 3.0 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Output Voltage		1.73	1.78	1.83	V
Power Supply Sensitivity	Constant Temperature	—	0.5	—	%/V
Temperature Sensitivity	Constant Supply	—	55	—	ppm/°C

Table 7.15. IDAC Electrical Characteristics

V_{DD} = 3.0 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), full-scale output current set to 2 mA, unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Static Performance					
Resolution			10		bits
Integral Nonlinearity		—	<±1	±3	LSB
Differential Nonlinearity	0 to +105 °C, Guaranteed Monotonic	—	<±0.5	±1	LSB
	–40 to 0°C	–1.0	<±0.5	1.3	LSB
Output Compliance Range		0	—	V_{DD} – 1.0	V
Offset Error		—	0	—	µA
Full Scale Error	2 mA Full Scale Output Current	–122	0	40	µA
	1 mA Full Scale Output Current	–61	0	20	µA
	0.5 mA Full Scale Output Current	–31	0	10	µA
Full Scale Error Tempco		—	80	—	ppm/°C
V_{DD} Power Supply Rejection Ratio		—	1.05	—	µA/V
Dynamic Performance					
Output Settling Time to 1/2 LSB	IDA0H:L = 0x3FF to 0x000	—	7	—	µs
Startup Time		—	6.5	—	µs
Gain Variation	1 mA Full Scale Output Current	—	0.1	—	%
	0.5 mA Full Scale Output Current	—	0.1	—	%
Power Consumption					
Power Supply Current (V_{DD} supplied to IDAC)	2 mA Full Scale Output Current	—	2065	—	µA
	1 mA Full Scale Output Current	—	1065	—	µA
	0.5 mA Full Scale Output Current	—	565	—	µA

Table 7.16. Comparator Electrical Characteristics

V_{DD} = 3.0 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Response Time Mode 0, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	370	—	ns
	CP0+ – CP0– = –100 mV	—	135	—	ns
Response Time Mode 3, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	1575	—	ns
	CP0+ – CP0– = –100 mV	—	3705	—	ns
Common-Mode Rejection Ratio		—	0.6	5	mV/V
Positive Hysteresis Mode 0 (CPMD = 00)	CP0HYP1–0 = 00	—	0.5	—	mV
	CP0HYP1–0 = 01	—	8	—	mV
	CP0HYP1–0 = 10	—	16	—	mV
	CP0HYP1–0 = 11	—	32	—	mV
Negative Hysteresis Mode 0 (CPMD = 00)	CP0HYN1–0 = 00	—	0.5	—	mV
	CP0HYN1–0 = 01	—	–8	—	mV
	CP0HYN1–0 = 10	—	–16	—	mV
	CP0HYN1–0 = 11	—	–32	—	mV
Positive Hysteresis Mode 1 (CPMD = 01)	CP0HYP1–0 = 00	—	0.5	—	mV
	CP0HYP1–0 = 01	—	6	—	mV
	CP0HYP1–0 = 10	—	12	—	mV
	CP0HYP1–0 = 11	—	24.5	—	mV
Negative Hysteresis Mode 1 (CPMD = 01)	CP0HYN1–0 = 00	—	0.5	—	mV
	CP0HYN1–0 = 01	—	–6	—	mV
	CP0HYN1–0 = 10	—	–12	—	mV
	CP0HYN1–0 = 11	—	–24.5	—	mV
Positive Hysteresis Mode 2 (CPMD = 10)	CP0HYP1–0 = 00	—	0.7	—	mV
	CP0HYP1–0 = 01	—	4.5	—	mV
	CP0HYP1–0 = 10	—	10	—	mV
	CP0HYP1–0 = 11	—	19	—	mV
Negative Hysteresis Mode 2 (CPMD = 10)	CP0HYN1–0 = 00	—	0.7	—	mV
	CP0HYN1–0 = 01	—	–4.5	—	mV
	CP0HYN1–0 = 10	—	–10	—	mV
	CP0HYN1–0 = 11	—	–19	—	mV
Positive Hysteresis Mode 3 (CPMD = 11)	CP0HYP1–0 = 00	—	1.6	2.3	mV
	CP0HYP1–0 = 01	2	4	6	mV
	CP0HYP1–0 = 10	4.8	8	11	mV
	CP0HYP1–0 = 11	10	15.5	21	mV

Table 7.16. Comparator Electrical Characteristics (Continued)

V_{DD} = 3.0 V, –40 to +105 °C (C8051F39x), –40 to +85 °C (C8051F37x), unless otherwise specified.

Parameter	Test Condition	Min	Typ	Max	Unit
Negative Hysteresis Mode 3 (CPMD = 11)	CP0HYN1–0 = 00	—	1.6	2.3	mV
	CP0HYN1–0 = 01	–6	–4	–2	mV
	CP0HYN1–0 = 10	–11	–8	–4.8	mV
	CP0HYN1–0 = 11	–21	–15.5	–10	mV
Inverting or Non-Inverting Input Voltage Range		–0.25	—	V_{DD} + 0.25	V
Input Capacitance		—	4	—	pF
Input Bias Current		—	0.001	—	nA
Input Offset Voltage		10	—	–10	mV
Power Supply					
Power Supply Rejection		—	0.1	—	mV/V
Power-up Time		—	6.5	—	µs
Supply Current at DC	Mode 0	—	32	50	µA
	Mode 1	—	15	25	µA
	Mode 2	—	5	12	µA
	Mode 3	—	2	8	µA

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

7.3. Typical Performance Curves

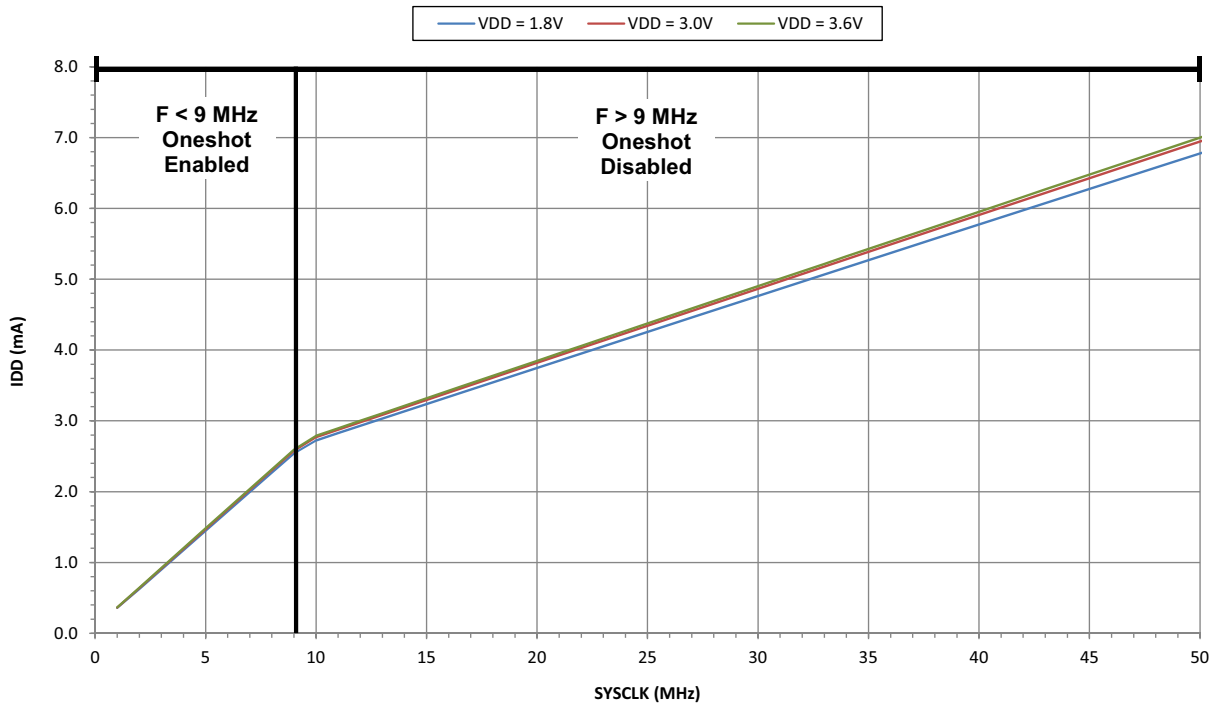


Figure 7.1. Normal Mode Digital Supply Current vs. Frequency

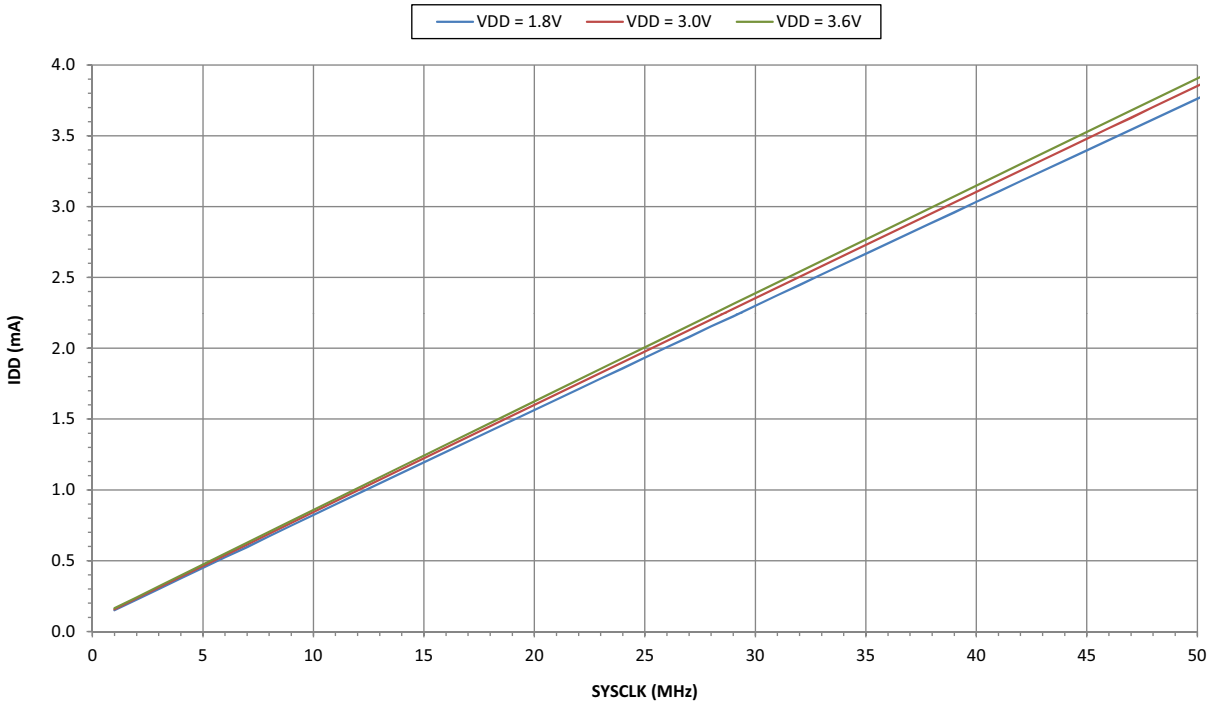


Figure 7.2. Idle Mode Digital Supply Current vs. Frequency

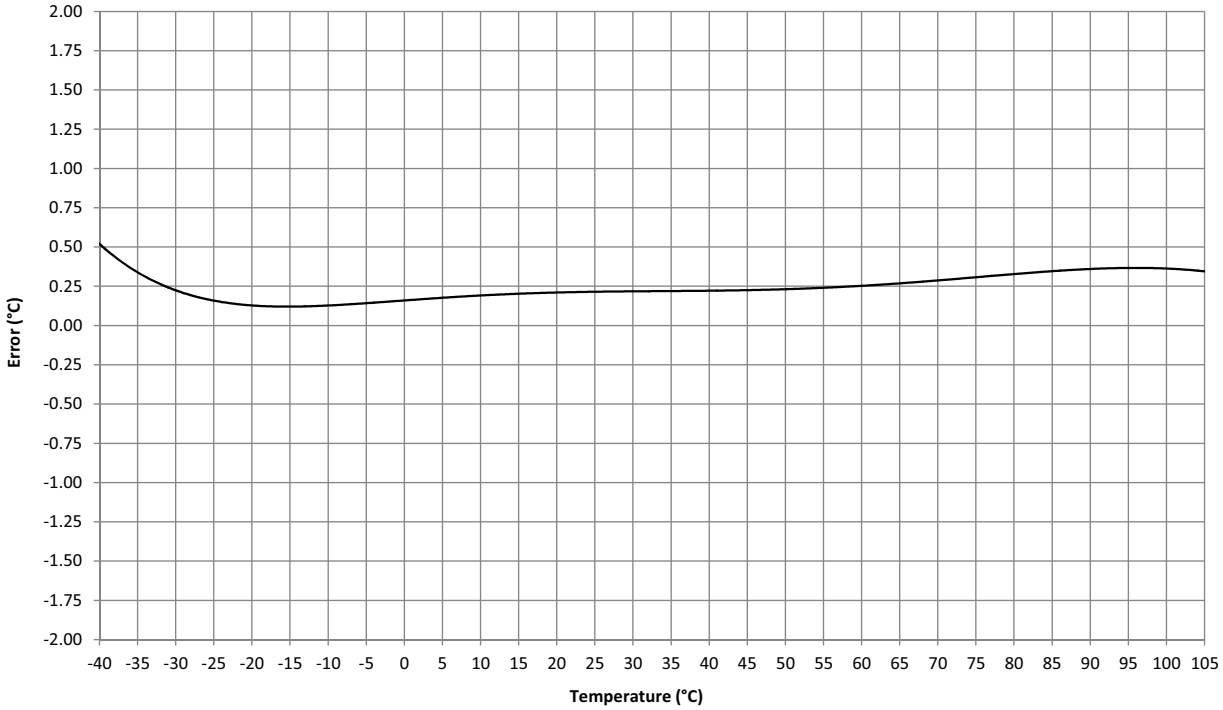


Figure 7.3. Precision Temperature Sensor Error vs. Temperature

8. Precision Temperature Sensor (C8051F390/2/4/6/8 and C8051F370/4 Only)

The precision temperature sensor is a self-contained module that reports the die temperature in degrees Celsius. For the temperature sensor accessed by the ADC, refer to Section 10.

The precision temperature sensor begins a conversion once the TS0STRT bit is set to 1 then cleared to 0 by firmware. The conversion length is specified by TS0CNVL, with a longer conversion time resulting in a more accurate temperature measurement. The TS0DN bit is set by hardware once the temperature sensor block has completed the measurement, and the temperature is available in TS0DATH:TS0DATL.

The precision temperature sensor may also be enabled as an interrupt source by setting the EPTS bit in EIE2. When enabled, the interrupt occurs once TS0DN is set to 1 by hardware.

8.1. Temperature in Two's Complement

The 16-bit word in TS0DATH:TS0DATL is the temperature in degrees Celsius represented in two's complement with 1 weighted sign bit, 8 integer bits, and 7 fractional bits. Equation 8.1 converts the value in TS0DATH:TS0DATL from two's complement binary to decimal.

$$\text{Temperature in } °C = -(TS0DATH[7] \times 2^{8}) + \left(\sum_{n=0}^{6} TS0DATH[n] \times 2^{n+1} + \sum_{n=0}^{7} TS0DATL[n] \times 2^{n-7} \right)$$

Equation 8.1. Temperature Conversion from Two's Complement Binary to Decimal

Where:

- TS0DATH[n] is the n^{th} bit in TS0DATH
- TS0DATL[n] is the n^{th} bit in TS0DATL

Table 8.1 lists several 16-bit values in TS0DATH:TS0DATL and the corresponding temperature.

Table 8.1. Example Temperature Values in TS0DATH:TS0DATL

Hexadecimal	Binary	Temperature (°C)
0x3480	0011 0100 1000 0000	105
0x1400	0001 0100 0000 0000	40
0x0CE0	0000 1100 1110 0000	25.75
0x0080	0000 0000 1000 0000	1
0x0040	0000 0000 0100 0000	0.5
0x0001	0000 0000 0000 0001	1/128
0x0000	0000 0000 0000 0000	0
0xFFFF	1111 1111 1111 1111	–1/128
0xFFC0	1111 1111 1100 0000	–0.5
0xFF80	1111 1111 1000 0000	–1
0xF320	1111 0011 0010 0000	–25.75
0xEC00	1110 1100 0000 0000	–40

SFR Definition 8.1. TS0CN: Temperature Sensor Control

Bit	7	6	5	4	3	2	1	0
Name	TS0STRT	TS0DN				TS0CNVL		
Type	R/W	R	R/W			R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD2; SFR Page = F

Bit	Name	Function
7	TS0STRT	**Temperature Sensor Start.** Firmware must set this bit to 1, then clear this bit to 0 to start a temperature sensor measurement.
6	TS0DN	**Temperature Sensor Finished Flag.** Hardware will set TS0DN to 1 when a temperature sensor measurement is complete. If enabled, a temperature sensor interrupt will be generated. This bit must be cleared to 0 by firmware.
5:3	Reserved	Must Write 000b.
2:0	TS0CNVL	**Temperature Sensor Conversion Length.** This field sets the conversion length of time over which the temperature is calculated. A longer conversion length results in a more accurate measurement. The conversion length in microseconds is derived from the following equation, where TS0CNVL is the 3-bit value held in TS0CNVL[2:0] and F_{TS0} is the precision temperature sensor clock frequency given in Table 7.12. $$\text{Conversion Length in } \mu s = \left(\frac{256}{F_{TS0}} \times 10^6\right) \times (2^{TS0CNVL+1} + 1) + 32$$

SFR Definition 8.2. TS0DATH: Temperature Sensor Output High Byte

Bit	7	6	5	4	3	2	1	0
Name	TS0DATH							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD3; SFR Page = 0

Bit	Name	Function
7:0	TS0DATH	**Temperature Sensor Data Word (MSB).** This byte represents the MSB of the temperature sensor data word. The data word is a 16-bit, 2's complement number.

SFR Definition 8.3. TS0DATL: Temperature Sensor Output Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TS0DATL							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD2; SFR Page = 0

Bit	Name	Function
7:0	TS0DATL	**Temperature Sensor Data Word (LSB).** This byte represents the LSB of the temperature sensor data word. The data word is a 16-bit, 2's complement number.

9. 10-Bit ADC (ADC0, C8051F390/2/4/6/8 and C8051F370/4 Only)

ADC0 on the C8051F390/2/4/6/8 and C8051F370/4 is a 500 ksps, 10-bit successive-approximation-register (SAR) ADC with integrated track-and-hold and a programmable window detector. The ADC is fully configurable under software control via Special Function Registers. The ADC may be configured to measure various different signals using the analog multiplexer described in Section “9.4. ADC0 Analog Multiplexer (C8051F390/2/4/6/8 and C8051F370/4 Only)” on page 61. The voltage reference for the ADC is selected as described in Section “12. Voltage Reference Options” on page 73. The ADC0 subsystem is enabled only when the AD0EN bit in the ADC0 Control register (ADC0CN) is set to logic 1. The ADC0 subsystem is in low power shutdown when this bit is logic 0.

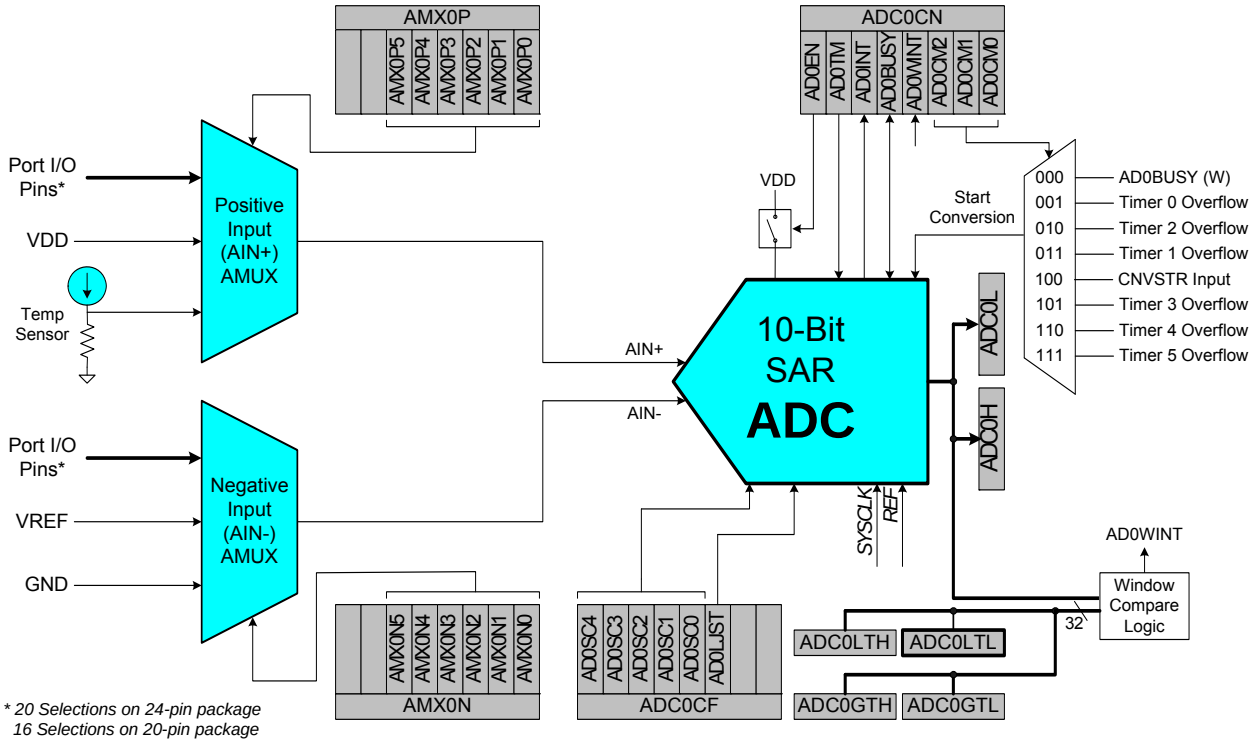


Figure 9.1. ADC0 Functional Block Diagram

9.1. Output Code Formatting

The conversion code format differs between Single-ended and Differential modes. The registers ADC0H and ADC0L contain the high and low bytes of the output conversion code from the ADC at the completion of each conversion. Data can be right-justified or left-justified, depending on the setting of the AD0LJST bit (ADC0CN.0). When in Single-ended Mode, conversion codes are represented as 10-bit unsigned integers. Inputs are measured from 0 to VREF x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADC0H and ADC0L registers are set to 0.

Input Voltage (Single-Ended)	Right-Justified ADC0H:ADC0L (AD0LJST = 0)	Left-Justified ADC0H:ADC0L (AD0LJST = 1)
VREF x 1023/1024	0x03FF	0xFFC0
VREF x 512/1024	0x0200	0x8000
VREF x 256/1024	0x0100	0x4000
0	0x0000	0x0000

When in Differential Mode, conversion codes are represented as 10-bit signed 2s complement numbers. Inputs are measured from –VREF to VREF x 511/512. Example codes are shown below for both right-justified and left-justified data. For right-justified data, the unused MSBs of ADC0H are a sign-extension of the data word. For left-justified data, the unused LSBs in the ADC0L register are set to 0.

Input Voltage (Differential)	Right-Justified ADC0H:ADC0L (AD0LJST = 0)	Left-Justified ADC0H:ADC0L (AD0LJST = 1)
VREF x 511/512	0x01FF	0x7FC0
VREF x 256/512	0x0100	0x4000
0	0x0000	0x0000
–VREF x 256/512	0xFF00	0xC000
–VREF	0xFE00	0x8000

9.2. Modes of Operation

ADC0 has a maximum conversion speed of 500 ksps. The ADC0 conversion clock is a divided version of the system clock, determined by the AD0SC bits in the ADC0CF register.

9.2.1. Starting a Conversion

A conversion can be initiated in one of several 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
6. A Timer 3 overflow
7. A Timer 4 overflow
8. A Timer 5 overflow

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 registers, ADC0H:ADC0L, when bit AD0INT is logic 1. Note that when Timer 2, 3, 4, or 5 overflows are used as the conversion source, Low Byte overflows are used if the timer is in 8-bit mode; High byte overflows are used if the timer is in 16-bit mode. See Section “31. Timers” on page 242 for timer configuration.

Important Note About Using CNVSTR: The CNVSTR input pin also functions as a Port I/O pin. When the CNVSTR input is used as the ADC0 conversion source, the associated pin should be skipped by the Digital Crossbar. See Section “27. Port Input/Output” on page 173 for details on Port I/O configuration.

9.2.2. Tracking Modes

The AD0TM bit in register ADC0CN controls the ADC0 track-and-hold mode. In its default state, the ADC0 input is continuously tracked, except when a conversion is in progress. When the AD0TM bit is logic 1, ADC0 operates in low-power track-and-hold mode. In this mode, each conversion is preceded by a tracking period of three SAR clocks (after the start-of-conversion signal). When the CNVSTR signal is used to initiate conversions in low-power tracking mode, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR. See Figure 9.2 for track and convert timing details. Tracking can also be disabled (shutdown) when the device is in low power standby or sleep modes. Low-power track-and-hold mode is also useful when AMUX settings are frequently changed, due to the settling time requirements described in Section “9.2.3. Settling Time Requirements” on page 54.

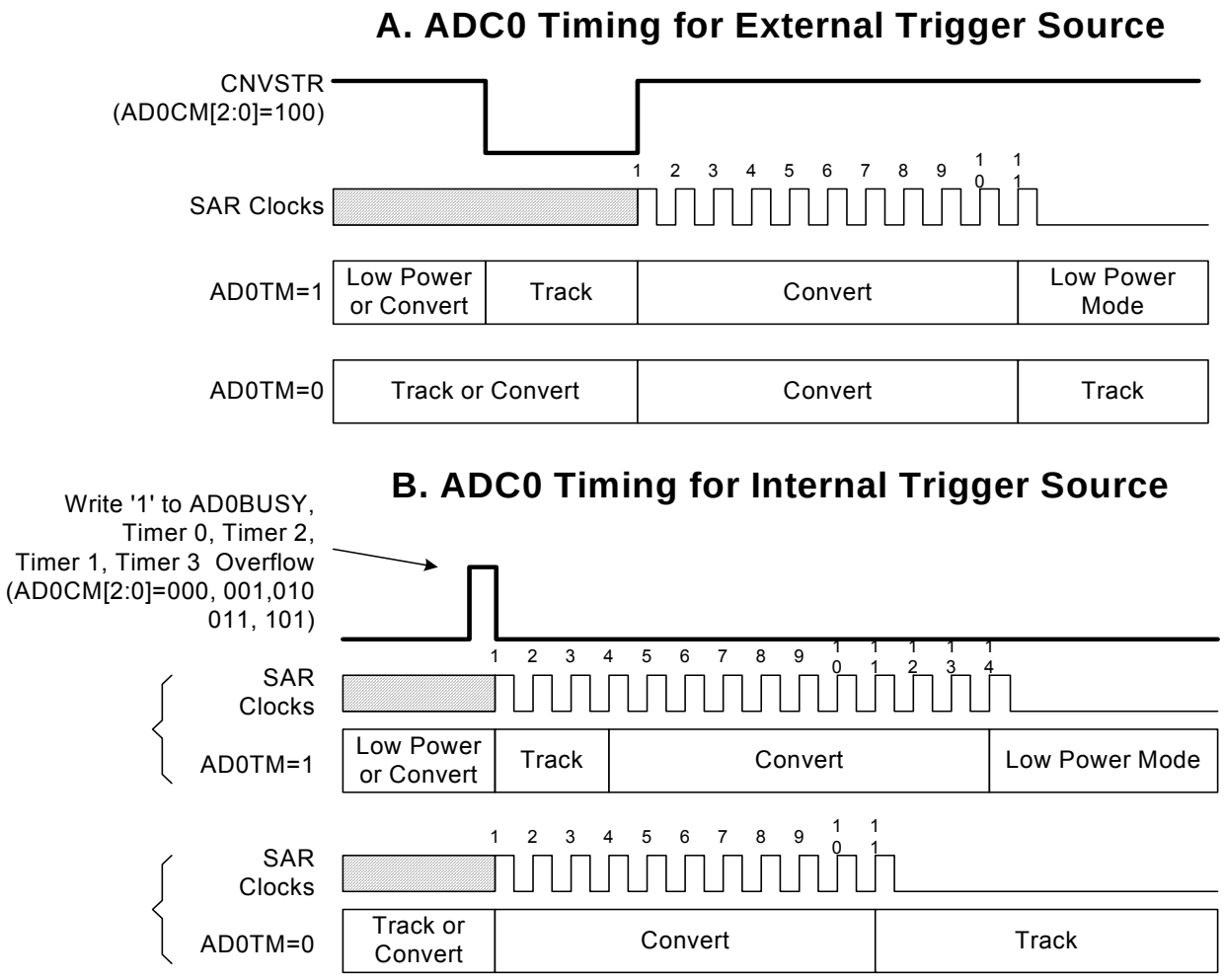


Figure 9.2. 10-Bit ADC Track and Conversion Example Timing

9.2.3. Settling Time Requirements

A minimum tracking time is required before each conversion to ensure that an accurate conversion is performed. 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 low-power tracking mode, three SAR clocks are used for tracking at the start of every conversion. For most applications, these three SAR clocks will meet the minimum tracking time requirements.

Figure 9.3 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 9.1. See Table 7.10 for ADC0 minimum settling time requirements as well as the mux impedance and sampling capacitor values.

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

Equation 9.1. 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).

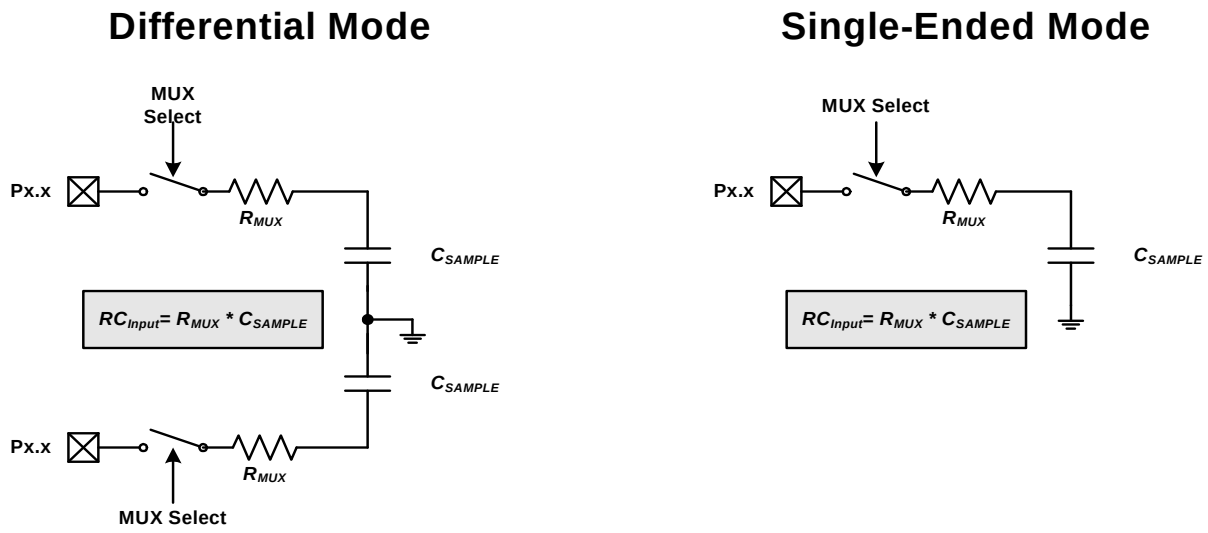


Figure 9.3. ADC0 Equivalent Input Circuits

SFR Definition 9.1. ADC0CF: ADC0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	AD0SC[4:0]					AD0LJST		
Type	R/W					R/W	R/W	
Reset	1	1	1	1	1	0	0	0

SFR Address = 0xBC; SFR Page = All Pages

Bit	Name	Function
7:3	AD0SC[4:0]	**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 the ADC specification Table 7.10. $AD0SC = \frac{SYSCLK}{CLK_{SAR}} - 1$
2	AD0LJST	**ADC0 Left Justify Select.** 0: Data in ADC0H:ADC0L registers are right-justified. 1: Data in ADC0H:ADC0L registers are left-justified.
1:0	Reserved	Must Write 00b.

SFR Definition 9.2. ADC0H: ADC0 Data Word MSB

Bit	7	6	5	4	3	2	1	0
Name	ADC0H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBE; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0H[7:0]	**ADC0 Data Word High-Order Bits.** For AD0LJST = 0: Bits 7–2 will read 000000b. Bits 1–0 are the upper 2 bits of the 10-bit ADC0 Data Word. For AD0LJST = 1: Bits 7–0 are the most-significant bits of the 10-bit ADC0 Data Word.

SFR Definition 9.3. ADC0L: ADC0 Data Word LSB

Bit	7	6	5	4	3	2	1	0
Name	ADC0L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBD; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0L[7:0]	**ADC0 Data Word Low-Order Bits.** For AD0LJST = 0: Bits 7–0 are the lower 8 bits of the 10-bit Data Word. For AD0LJST = 1: Bits 7–6 are the lower 2 bits of the 10-bit Data Word. Bits 5–0 will read 000000b.

SFR Definition 9.4. ADC0CN: ADC0 Control

Bit	7	6	5	4	3	2	1	0
Name	AD0EN	AD0TM	AD0INT	AD0BUSY	AD0WINT	AD0CM[2:0]		
Type	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	AD0EN	**ADC0 Enable Bit.** 0: ADC0 Disabled. ADC0 is in low-power shutdown. 1: ADC0 Enabled. ADC0 is active and ready for data conversions.
6	AD0TM	**ADC0 Track Mode Bit.** 0: Normal Track Mode: When ADC0 is enabled, tracking is continuous unless a conversion is in progress. Conversion begins immediately on start-of-conversion event, as defined by AD0CM[2:0]. 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. Note that there is not a tracking delay when CNVSTR is used (AD0CM[2:0] = 100).
5	AD0INT	**ADC0 Conversion Complete Interrupt Flag.** 0: ADC0 has not completed a data conversion since AD0INT was last cleared. 1: ADC0 has completed a data conversion.
4	AD0BUSY	**ADC0 Busy Bit.**
3	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.
2:0	AD0CM[2:0]	**ADC0 Start of Conversion Mode Select.** 000: ADC0 start-of-conversion source is write of 1 to AD0BUSY. 001: ADC0 start-of-conversion source is overflow of Timer 0. 010: ADC0 start-of-conversion source is overflow of Timer 2. 011: ADC0 start-of-conversion source is overflow of Timer 1. 100: ADC0 start-of-conversion source is rising edge of external CNVSTR. 101: ADC0 start-of-conversion source is overflow of Timer 3. 110: ADC0 start-of-conversion source is overflow of Timer 4. 111: ADC0 start-of-conversion source is overflow of Timer 5.

9.3. Programmable Window Detector

The ADC Programmable Window Detector continuously compares the ADC0 output registers 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. 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 ADC0 Less-Than and ADC0 Greater-Than registers.

SFR Definition 9.5. ADC0GTH: ADC0 Greater Than Data High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTH[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xC4; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0GTH[7:0]	**ADC0 Greater-Than Data Word High-Order Bits.**

SFR Definition 9.6. ADC0GTL: ADC0 Greater-Than Data Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTL[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xC3; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0GTL[7:0]	**ADC0 Greater-Than Data Word Low-Order Bits.**

SFR Definition 9.7. ADC0LTH: ADC0 Less-Than Data High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC6; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0LTH[7:0]	**ADC0 Less-Than Data Word High-Order Bits.**

SFR Definition 9.8. ADC0LTL: ADC0 Less-Than Data Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC5; SFR Page = All Pages

Bit	Name	Function
7:0	ADC0LTL[7:0]	**ADC0 Less-Than Data Word Low-Order Bits.**

9.3.1. Window Detector Example

Figure 9.4 shows two example window comparisons for right-justified, single-ended data, with ADC0LTH:ADC0LTL = 0x0080 (128d) and ADC0GTH:ADC0GTL = 0x0040 (64d). The input voltage can range from 0 to VREF x (1023/1024) with respect to GND, and is represented by a 10-bit unsigned integer value. In the left example, an AD0WINT interrupt will be generated if the ADC0 conversion word (ADC0H:ADC0L) is within the range defined by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL (if 0x0040 < ADC0H:ADC0L < 0x0080). In the right example, and AD0WINT interrupt will be generated if the ADC0 conversion word is outside of the range defined by the ADC0GT and ADC0LT registers (if ADC0H:ADC0L < 0x0040 or ADC0H:ADC0L > 0x0080). Figure 9.5 shows an example using left-justified data with the same comparison values.

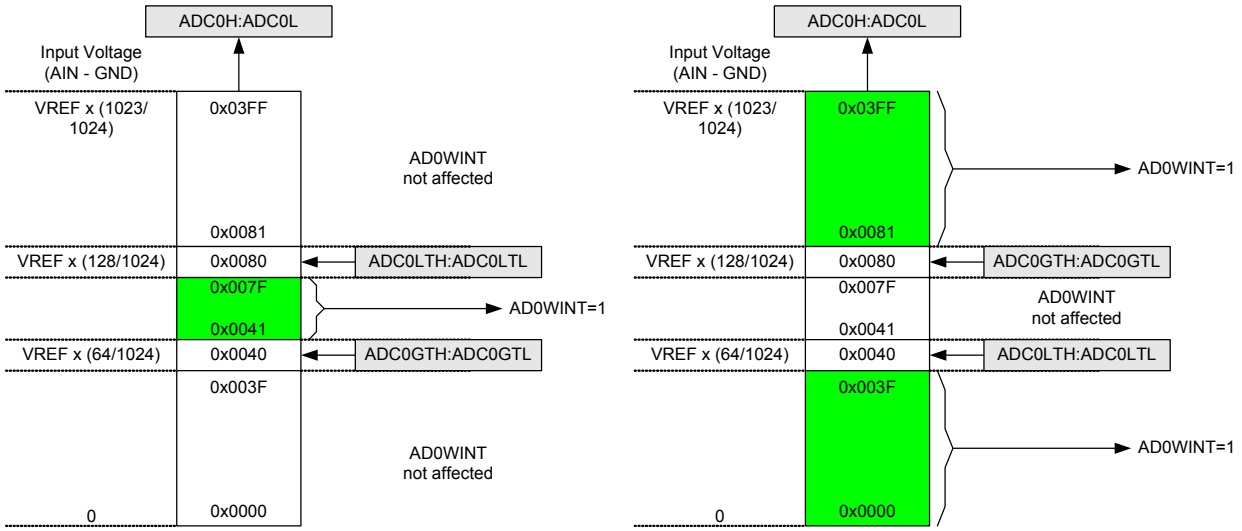


Figure 9.4. ADC Window Compare Example: Right-Justified, Single-Ended Data

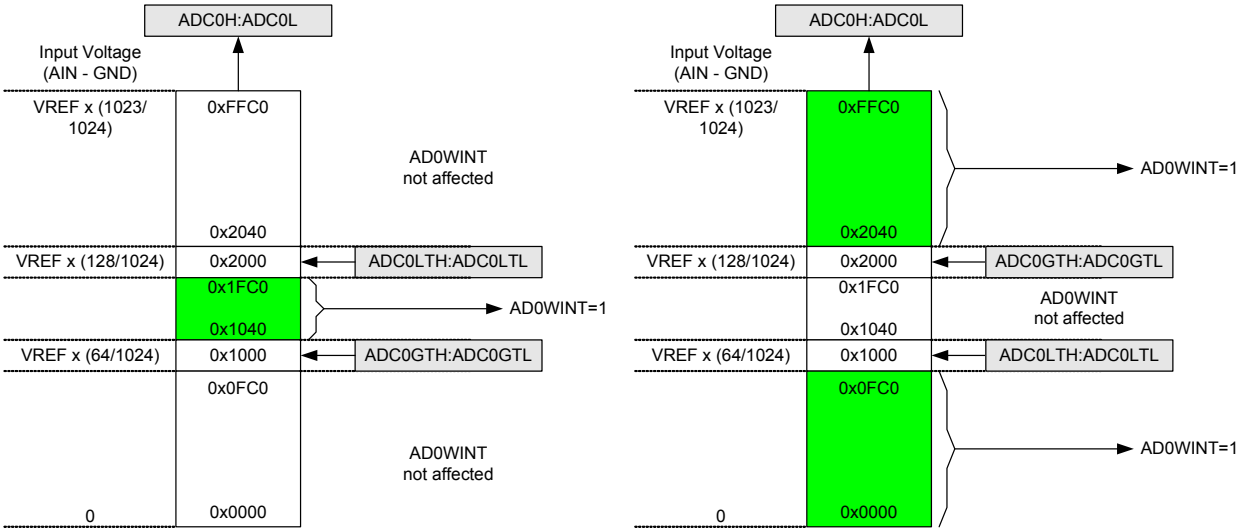


Figure 9.5. ADC Window Compare Example: Left-Justified, Single-Ended Data

9.4. ADC0 Analog Multiplexer (C8051F390/2/4/6/8 and C8051F370/4 Only)

ADC0 on the C8051F390/2/4/6/8 and C8051F370/4 has two analog multiplexers, referred to collectively as AMUX0.

AMUX0 selects the positive and negative inputs to the ADC. Any of the following may be selected as the positive input: Port I/O pins, the on-chip temperature sensor, or the positive power supply (V_{DD}). Any of the following may be selected as the negative input: Port I/O pins, V_{REF}, or GND. **When GND is selected as the negative input, ADC0 operates in Single-ended Mode; all other times, ADC0 operates in Differential Mode.** The ADC0 input channels are selected in the AMX0P and AMX0N registers as described in SFR Definition 9.9 and SFR Definition 9.10.

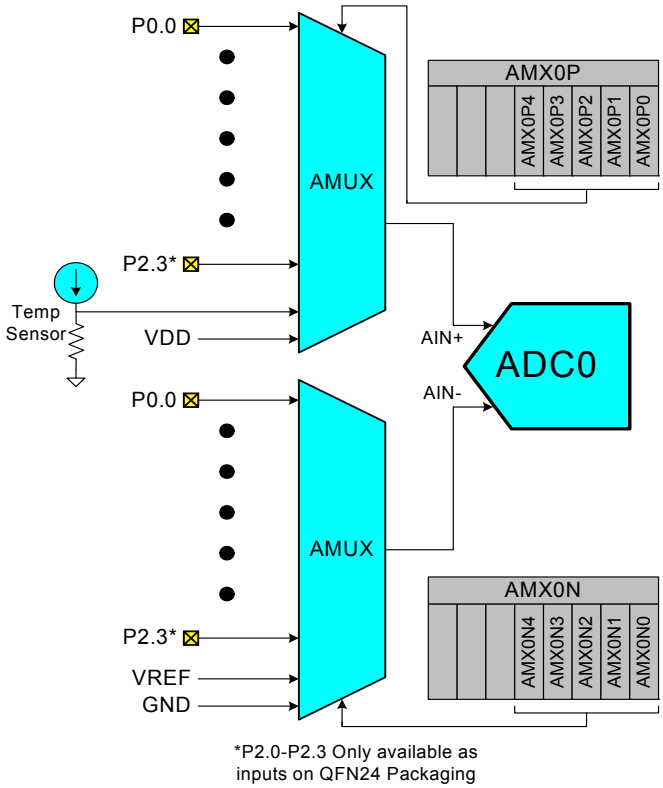


Figure 9.6. ADC0 Multiplexer Block Diagram

Important Note About ADC0 Input Configuration: Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to ‘0’ the corresponding bit in register PnMDIN. To force the Crossbar to skip a Port pin, set to ‘1’ the corresponding bit in register PnSKIP. See Section “27. Port Input/Output” on page 173 for more Port I/O configuration details.

SFR Definition 9.9. AMX0P: AMUX0 Positive Channel Select

Bit	7	6	5	4	3	2	1	0
Name				AMX0P[4:0]				
Type	R	R	R	R/W				
Reset	0	0	0	1	1	1	1	1

SFR Address = 0xBB; SFR Page = All Pages

Bit	Name	Function
7:5	Unused	Read = 000b; Write = Don't Care.
4:0	AMX0P[4:0]	**AMUX0 Positive Input Selection.** 00000: P0.0 00001: P0.1 00010: P0.2 00011: P0.3 00100: P0.4 00101: P0.5 00110: P0.6 00111: P0.7 01000: P1.0 01001: P1.1 01010: P1.2 01011: P1.3 01100: P1.4 01101: P1.5 01110: P1.6 01111: P1.7 10000: Temp Sensor 10001: V_{DD} 10010: P2.0 (C8051F390/1/4/5 and C8051F37x Only) 10011: P2.1 (C8051F390/1/4/5 and C8051F37x Only) 10100: P2.2 (C8051F390/1/4/5 and C8051F37x Only) 10101: P2.3 (C8051F390/1/4/5 and C8051F37x Only) 10110 – 11111: no input selected

SFR Definition 9.10. AMX0N: AMUX0 Negative Channel Select

Bit	7	6	5	4	3	2	1	0
Name				AMX0N[4:0]				
Type	R	R	R	R/W				
Reset	0	0	0	1	1	1	1	1

SFR Address = 0xBA; SFR Page = All Pages

Bit	Name	Function
7:5	Unused	Read = 000b; Write = Don't Care.
4:0	AMX0N[4:0]	**AMUX0 Negative Input Selection.** 00000: P0.0 00001: P0.1 00010: P0.2 00011: P0.3 00100: P0.4 00101: P0.5 00110: P0.6 00111: P0.7 01000: P1.0 01001: P1.1 01010: P1.2 01011: P1.3 01100: P1.4 01101: P1.5 01110: P1.6 01111: P1.7 10000: V_{REF} 10001: GND (ADC in Single-Ended Mode) 10010: P2.0 (C8051F390/1/4/5 and C8051F37x Only) 10011: P2.1 (C8051F390/1/4/5 and C8051F37x Only) 10100: P2.2 (C8051F390/1/4/5 and C8051F37x Only) 10101: P2.3 (C8051F390/1/4/5 and C8051F37x Only) 10110 – 11111: no input selected

10. Temperature Sensor (C8051F390/2/4/6/8 and C8051F370/4 Only)

A fully C8051F33x-compatible temperature sensor is included on the C8051F390/2/4/6/8 and C8051F370/4 and accessed via the ADC multiplexer in single-ended configuration. For the self-contained precision temperature sensor, refer to Section 8.

To use the ADC to measure the temperature sensor, the ADC mux channel should be configured to connect to the temperature sensor. The temperature sensor transfer function is shown in Figure 10.1. The output voltage (V_{TEMP}) is the positive ADC input when the ADC multiplexer is set correctly. The TEMPE bit in register REF0CN enables/disables the temperature sensor, as described in SFR Definition 12.1. While disabled, the temperature sensor defaults to a high impedance state and any ADC measurements performed on the sensor will result in meaningless data. Refer to Table 7.11 for the slope and offset parameters of the temperature sensor.

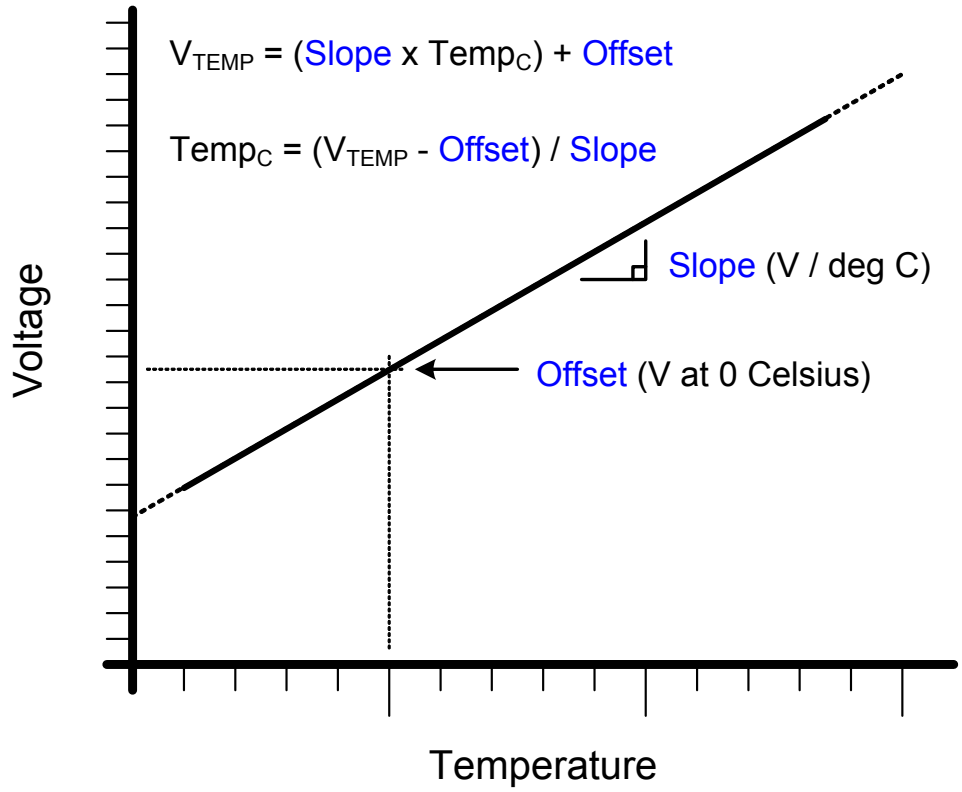


Figure 10.1. Temperature Sensor Transfer Function

10.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements (see Table 7.11 on page 40 for specifications). For absolute temperature measurements, offset and/or gain calibration is recommended.

Figure 10.2 shows the typical temperature sensor error assuming a 1-point calibration at 0 °C. **Parameters that affect ADC measurement, in particular the voltage reference value, will also affect temperature measurement.**

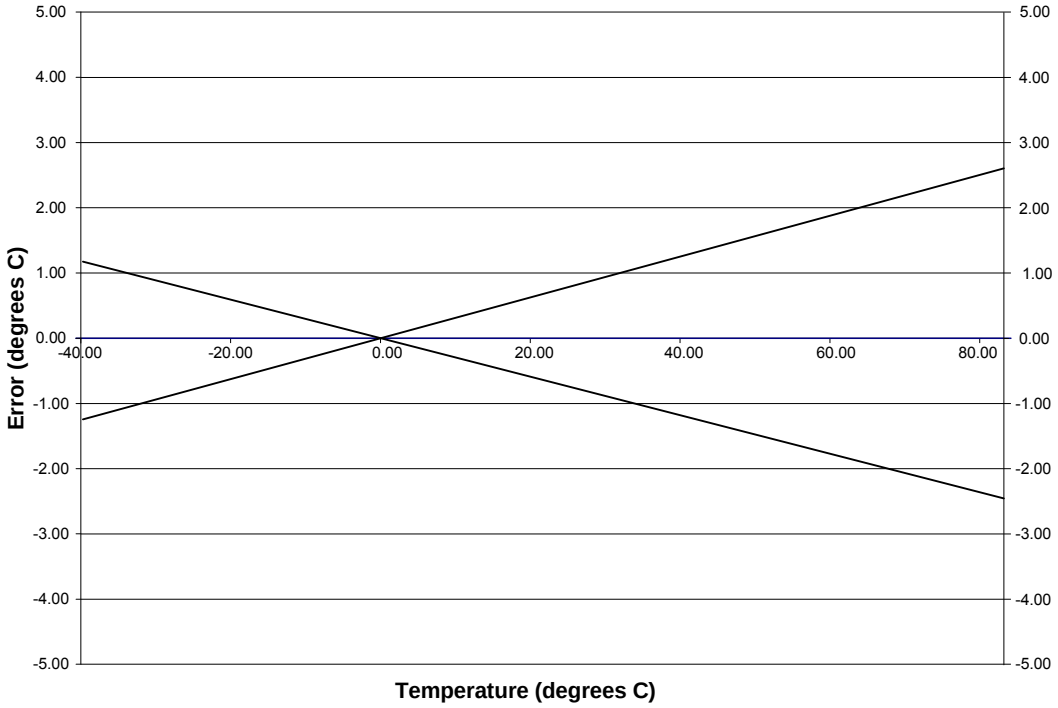


Figure 10.2. Temperature Sensor Error with 1-Point Calibration at 0 °C

11. 10-Bit Current Mode DACs (IDA0, IDA1, C8051F390/2/4/6/8 and C8051F370/4 Only)

The C8051F390/2/4/6/8 and C8051F370/4 devices include two 10-bit current-mode Digital-to-Analog Converters (IDACs). The maximum current output of the IDACs can be adjusted for three different current settings; 0.5 mA, 1 mA, and 2 mA. The IDACs are enabled or disabled with the IDAnEN bit in the Control Register for that IDAC (see SFR Definition 11.1 and SFR Definition 11.4). When IDAnEN is set to 0, the IDAC output behaves as a normal GPIO pin. When IDAnEN is set to 1, the digital output drivers and weak pullup for the IDAC pin are automatically disabled, and the pin is connected to the IDAC output. An internal bandgap bias generator is used to generate a reference current for the IDAC whenever it is enabled. When using an IDAC, the crossbar skip functionality should be enabled on the IDAC output pin, to force the Crossbar to skip the output pin.

11.1. IDAC Output Scheduling

The IDACs feature a flexible output update mechanism which allows for seamless full-scale changes and supports jitter-free updates for waveform generation. Three update modes are provided, allowing IDAC output updates on a write to IDAnH, on a Timer overflow, or on an external pin edge.

11.1.1. Update Output On-Demand

In its default mode (IDAnCN.[6:4] = 111) the IDAC output is updated “on-demand” on a write to the high-byte of the IDAC data register (IDAnH). It is important to note that writes to IDAnL are held in this mode, and have no effect on the IDAC output until a write to IDAnH takes place. If writing a full 10-bit word to the IDAC data registers, the 10-bit data word is written to the low byte (IDAnL) and high byte (IDAnH) data registers. Data is latched into the IDAC after a write to the IDAnH register, **so the write sequence should be IDAnL followed by IDAnH** if the full 10-bit resolution is required. The IDAC can be used in 8-bit mode by initializing IDAnL to the desired value (typically 0x00), and writing data to only IDAnH (see Section 11.3 for information on the format of the 10-bit IDAC data word within the 16-bit SFR space).

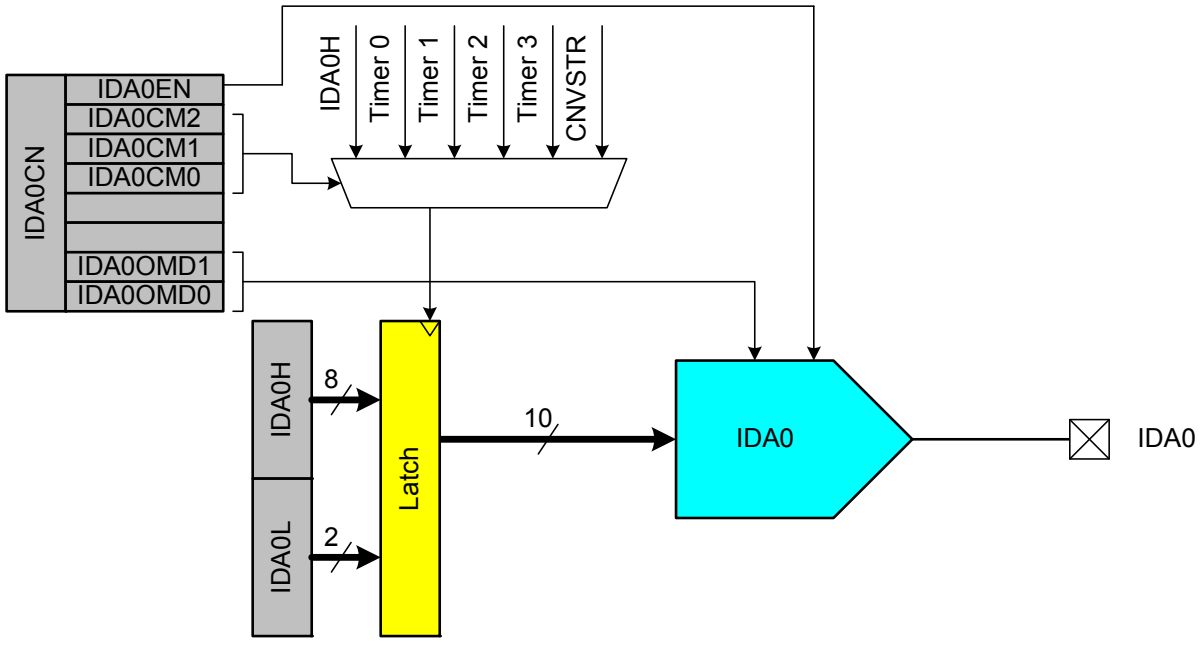


Figure 11.1. IDA0 Functional Block Diagram

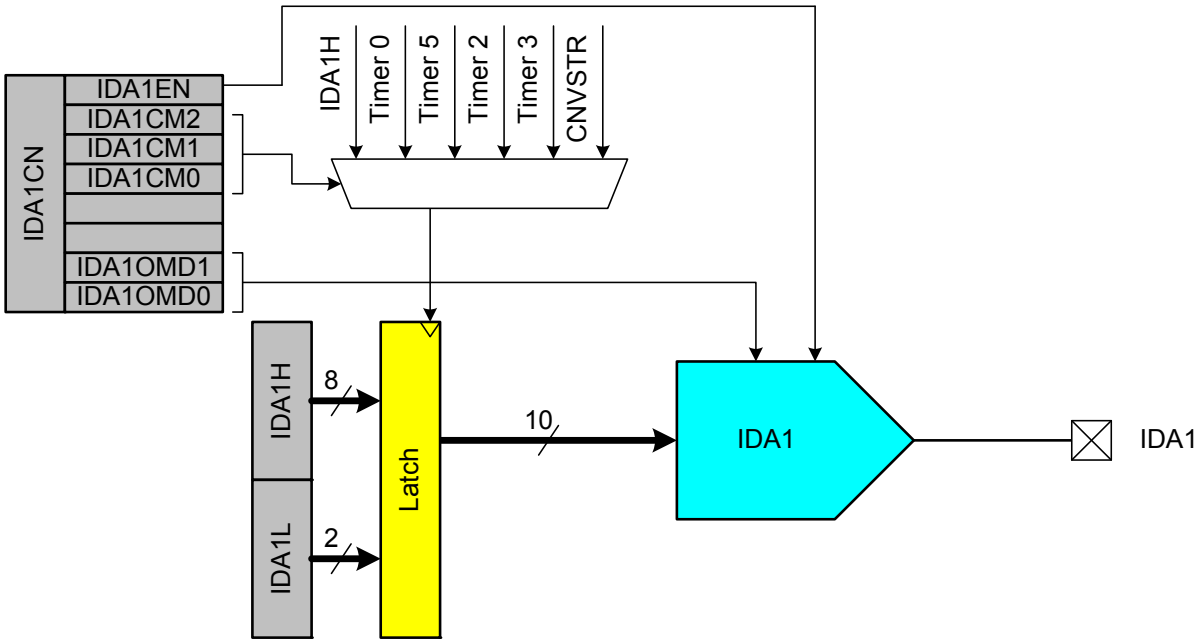


Figure 11.2. IDA1 Functional Block Diagram

11.1.2. Update Output Based on Timer Overflow

The IDAC outputs can use a Timer overflow to schedule an output update event. This feature is useful in systems where the IDAC is used to generate a waveform of a defined sampling rate by eliminating the effects of variable interrupt latency and instruction execution on the timing of the IDAC output. When the IDAnCM bits (IDAnCN.[6:4]) are set to 000, 001, 010 or 011, writes to both IDAC data registers (IDAnL and IDAnH) are held until an associated Timer overflow event occurs, at which time the IDAnH:IDAnL contents are copied to the IDAC input latches, allowing the IDAC output to change to the new value.

11.1.3. Update Output Based on CNVSTR Edge

The IDAC output can also be configured to update on a rising edge, falling edge, or both edges of the external CNVSTR signal. When the IDAnCM bits (IDAnCN.[6:4]) are set to 100, 101, or 110, writes to both IDAC data registers (IDAnL and IDAnH) are held until an edge occurs on the CNVSTR input pin. The particular setting of the IDAnCM bits determines whether IDAC outputs are updated on rising, falling, or both edges of CNVSTR. When a corresponding edge occurs, the IDAnH:IDAnL contents are copied to the IDAC input latches, allowing the IDAC output to change to the new value.

11.2. IDAC Reset Behavior

By default, both IDAC modules revert to a disabled state on any reset source. It is possible to keep the IDAC outputs enabled through all but a POR or VDD monitor reset, however. When the IDAnRP bit in the IDAnCN register is set to 1, any reset other than a POR or VDD monitor reset will not affect the IDAC output. The IDAC output will remain enabled and the value in the IDAC output word is maintained.

11.3. IDAC Output Mapping

The IDAC data registers (IDAnH and IDAnL) are left-justified, meaning that the eight MSBs of the IDAC output word are mapped to bits 7–0 of the IDAnH register, and the two LSBs of the IDAC output word are mapped to bits 7 and 6 of the IDAnL register. The data word mapping for the IDACs is shown in Figure 11.3.

IDAnH								IDAnL							
B9	B8	B7	B6	B5	B4	B3	B2	B1	B0						

Input Data Word (IDAn9–IDAn0)	Output Current IDAnOMD[1:0] = 1x	Output Current IDAnOMD[1:0] = 01	Output Current IDAnOMD[1:0] = 00
0x000	0 mA	0 mA	0 mA
0x001	1/1024 x 2 mA	1/1024 x 1 mA	1/1024 x 0.5 mA
0x200	512/1024 x 2 mA	512/1024 x 1 mA	512/1024 x 0.5 mA
0x3FF	1023/1024 x 2 mA	1023/1024 x 1 mA	1023/1024 x 0.5 mA

Figure 11.3. IDA0 Data Word Mapping

The full-scale output current of the IDAC is selected using the IDAnOMD bits (IDAnCN[1:0]). By default, the IDAC is set to a full-scale output current of 2 mA. The IDAnOMD bits can also be configured to provide full-scale output currents of 1 mA or 0.5 mA, as shown in SFR Definition 11.1 and SFR Definition 11.4.

SFR Definition 11.1. IDA0CN: IDA0 Control

Bit	7	6	5	4	3	2	1	0
Name	IDA0EN	IDA0CM[2:0]				IDA0RP	IDA0OMD[1:0]	
Type	R/W	R/W			R	R/W	R/W	
Reset	0	1	1	1	0	Varies	1	0

SFR Address = 0xB9; SFR Page = 0

Bit	Name	Function
7	IDA0EN	**IDA0 Enable.** 0: IDA0 Disabled. 1: IDA0 Enabled.
6:4	IDA0CM[2:0]	**IDA0 Update Source Select bits.** 000: DAC output updates on Timer 0 overflow. 001: DAC output updates on Timer 1 overflow. 010: DAC output updates on Timer 2 overflow. 011: DAC output updates on Timer 3 overflow. 100: DAC output updates on rising edge of CNVSTR. 101: DAC output updates on falling edge of CNVSTR. 110: DAC output updates on any edge of CNVSTR. 111: DAC output updates on write to IDA0H.
3	Reserved	Write = 0b.
2	IDA0RP	**IDA0 Reset Persistence.** 0: IDA0 is disabled by any reset source. 1: IDA0 will remain enabled through any reset source except a power-on-reset. This bit is reset to 0 by a power on reset, but is sticky through all other reset sources. When setting IDA0RP to 1, IDA0EN must be set to 1 also in the same mov instruction.
1:0	IDA0OMD[1:0]	**IDA0 Output Mode Select bits.** 00: 0.5 mA full-scale output current. 01: 1.0 mA full-scale output current. 1x: 2.0 mA full-scale output current.

SFR Definition 11.2. IDA0H: IDA0 Data Word MSB

Bit	7	6	5	4	3	2	1	0
Name	IDA0[9:2]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x97; SFR Page = 0

Bit	Name	Function
7:0	IDA0[9:2]	**IDA0 Data Word High-Order Bits.** Upper 8 bits of the 10-bit IDA0 Data Word.

SFR Definition 11.3. IDA0L: IDA0 Data Word LSB

Bit	7	6	5	4	3	2	1	0
Name	IDA0[1:0]							
Type	R/W		R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x96; SFR Page = 0

Bit	Name	Function
7:6	IDA0[1:0]	**IDA0 Data Word Low-Order Bits.** Lower 2 bits of the 10-bit IDA0 Data Word.
5:0	Unused	Unused. Read = 000000b. Write = Don't care.

SFR Definition 11.4. IDA1CN: IDA1 Control

Bit	7	6	5	4	3	2	1	0
Name	IDA1EN	IDA1CM[2:0]				IDA1RP	IDA1OMD[1:0]	
Type	R/W	R/W			R	R/W	R/W	
Reset	0	1	1	1	0	Varies	1	0

SFR Address = 0xB9; SFR Page = F

Bit	Name	Function
7	IDA1EN	**IDA1 Enable.** 0: IDA1 Disabled. 1: IDA1 Enabled.
6:4	IDA1CM[2:0]	**IDA0 Update Source Select bits.** 000: DAC output updates on Timer 0 overflow. 001: DAC output updates on Timer 5 overflow. 010: DAC output updates on Timer 2 overflow. 011: DAC output updates on Timer 3 overflow. 100: DAC output updates on rising edge of CNVSTR. 101: DAC output updates on falling edge of CNVSTR. 110: DAC output updates on any edge of CNVSTR. 111: DAC output updates on write to IDA1H.
3	Reserved	Write = 0b.
2	IDA1RP	**IDA1 Reset Persistence.** 0: IDA1 is disabled by any reset source. 1: IDA1 will remain enabled through any reset source except a power-on-reset. This bit is reset to 0 by a power on reset, but is sticky through all other reset sources. When setting IDA1RP to 1, IDA1EN must be set to 1 also in the same move instruction.
1:0	IDA1OMD[1:0]	**IDA1 Output Mode Select bits.** 00: 0.5 mA full-scale output current. 01: 1.0 mA full-scale output current. 1x: 2.0 mA full-scale output current.

SFR Definition 11.5. IDA1H: IDA1 Data Word MSB

Bit	7	6	5	4	3	2	1	0
Name	IDA1[9:2]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x97; SFR Page = F

Bit	Name	Function
7:0	IDA1[9:2]	**IDA1 Data Word High-Order Bits.** Upper 8 bits of the 10-bit IDA1 Data Word.

SFR Definition 11.6. IDA1L: IDA1 Data Word LSB

Bit	7	6	5	4	3	2	1	0
Name	IDA1[1:0]							
Type	R/W		R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x96; SFR Page = F

Bit	Name	Function
7:6	IDA1[1:0]	**IDA1 Data Word Low-Order Bits.** Lower 2 bits of the 10-bit IDA1 Data Word.
5:0	Unused	Unused. Read = 000000b. Write = Don't care.

12. Voltage Reference Options

The Voltage reference multiplexer for the ADC is configurable to use an externally connected voltage reference, the on-chip reference voltage generator routed to the VREF pin, the unregulated power supply voltage (V_{DD}), or the regulated 1.8 V internal supply (see Figure 12.1). The REFSL bit in the Reference Control register (REF0CN, SFR Definition 12.1) selects the reference source for the ADC. For an external source or the on-chip reference, REFSL should be set to 0 to select the VREF pin. To use 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.

The BIASE bit enables the internal voltage bias generator, which is used by many of the analog peripherals on the device. This bias is automatically enabled when any peripheral which requires it is enabled, and it does not need to be enabled manually. The bias generator may be enabled manually by writing a 1 to the BIASE bit in register REF0CN. The electrical specifications for the voltage reference circuit are given in Table 7.13.

The C8051F390/2/4/6/8 and C8051F370/4 devices also include an on-chip voltage reference circuit which consists of a 1.2 V, temperature stable bandgap voltage reference generator and a selectable-gain output buffer amplifier. The buffer is configured for 1x or 2x gain using the REFBGS bit in register REF0CN. On the 1x gain setting the output voltage is nominally 1.2 V, and on the 2x gain setting the output voltage is nominally 2.4 V. The on-chip voltage reference can be driven on the VREF pin by setting the REFBE bit in register REF0CN to a 1. The maximum load seen by the VREF pin must be less than 200 µA to GND. Bypass capacitors of 0.1 µF and 4.7 µF are recommended from the VREF pin to GND, and a minimum of 0.1uF is required. If the on-chip reference is not used, the REFBE bit should be cleared to 0. Electrical specifications for the on-chip voltage reference are given in Table 7.13.

Important Note about the VREF Pin: When using either an external voltage reference or the on-chip reference circuitry, the VREF pin should be configured as an analog pin and skipped by the Digital Crossbar. Refer to Section “27. Port Input/Output” on page 173 for the location of the VREF pin, as well as details of how to configure the pin in analog mode and to be skipped by the crossbar.

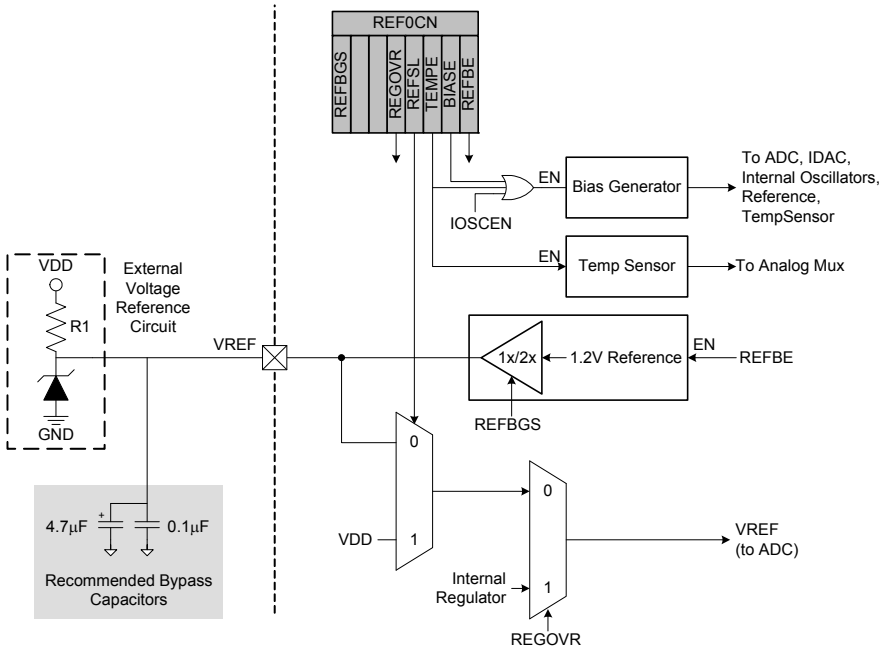


Figure 12.1. Voltage Reference Functional Block Diagram

SFR Definition 12.1. REF0CN: Reference Control

Bit	7	6	5	4	3	2	1	0
Name	REFBGS			REGOVR	REFSL	TEMPE	BIASE	REFBE
Type	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD1; SFR Page = All Pages

Bit	Name	Function
7	REFBGS	**Reference Buffer Gain Select.** This bit selects between 1x and 2x gain for the on-chip voltage reference buffer. 0: 2x Gain 1: 1x Gain
6:5	Unused	Read = 00b; Write = Don't care.
4	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.
3	REFSL	**Voltage Reference Select.** This bit selects the ADCs voltage reference. 0: V_{REF} pin used as voltage reference. 1: V_{DD} used as voltage reference.
2	TEMPE	**Temperature Sensor Enable Bit.** 0: Internal Temperature Sensor off. 1: Internal Temperature Sensor on.
1	BIASE	**Internal Analog Bias Generator Enable Bit.** 0: Internal Bias Generator off. 1: Internal Bias Generator on.
0	REFBE	**On-chip Reference Buffer Enable Bit.** 0: On-chip Reference Buffer off. 1: On-chip Reference Buffer on. Internal voltage reference driven on the V_{REF} pin.

13. Voltage Regulator

C8051F39x/37x devices include an internal regulator that regulates the internal core supply from a V_{DD} supply of 1.8 to 3.6 V. The regulator has two power-saving modes built in to help reduce current consumption in low-power applications. These modes are accessed through the REG0CN register.

13.1. Power Modes

Under default conditions, the internal regulator will remain on when the device enters STOP mode. 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 13.1. REG0CN: Voltage Regulator Control

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

SFR Address = 0xC9; SFR Page = All Pages

Bit	Name	Function
7:4	Reserved	Must Write 0000b.
3	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 or power cycle can reset the device.
2:0	Reserved	Must Write 000b.

14. Comparator0

C8051F39x/37x devices include an on-chip programmable voltage comparator, Comparator0, shown in Figure 14.1.

The Comparator offers programmable response time and hysteresis, an analog input multiplexer, and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0), or an asynchronous “raw” output (CP0A). The asynchronous CP0A signal is available even when the system clock is not active. This allows the Comparator to operate and generate an output with the device in STOP mode. When assigned to a Port pin, the Comparator output may be configured as open drain or push-pull (see Section “27.4. Port I/O Initialization” on page 180). Comparator0 may also be used as a reset source (see Section “24.5. Comparator0 Reset” on page 159), or as a trigger to kill a PCA output channel.

The Comparator0 inputs are selected by the comparator input multiplexer, as detailed in Section “14.1. Comparator Multiplexer” on page 80.

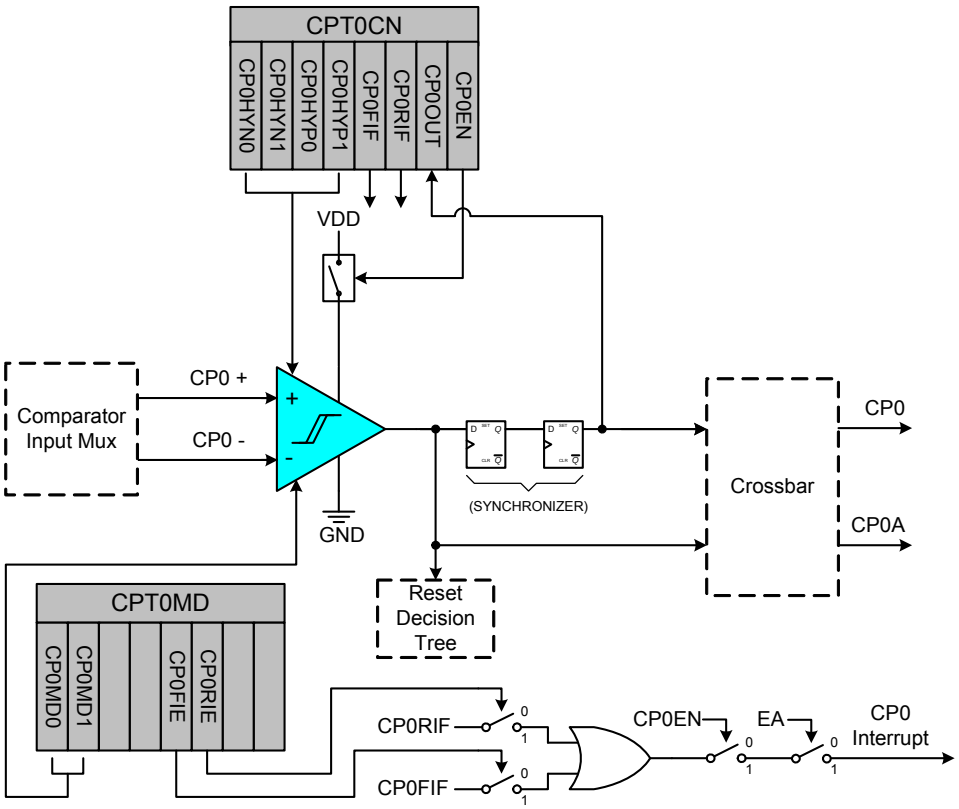


Figure 14.1. Comparator0 Functional Block Diagram

The Comparator output can be polled in software, used as an interrupt source, and/or routed to a Port pin. When routed to a Port pin, the Comparator 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 Comparator output (if assigned to a Port I/O pin via the Crossbar) defaults to the logic low state, and the power supply to the comparator is turned off. See Section “27.3. Priority Crossbar Decoder” on page 178 for details on configuring Comparator outputs via the digital Crossbar. Comparator inputs can be externally driven from –0.25 V to (V_{DD}) + 0.25 V without damage or upset. The complete Comparator electrical specifications are given in Section “7. Electrical Characteristics” on page 32.

The Comparator response time may be configured in software via the CPT0MD register (see SFR Definition 14.2). Selecting a longer response time reduces the Comparator supply current.

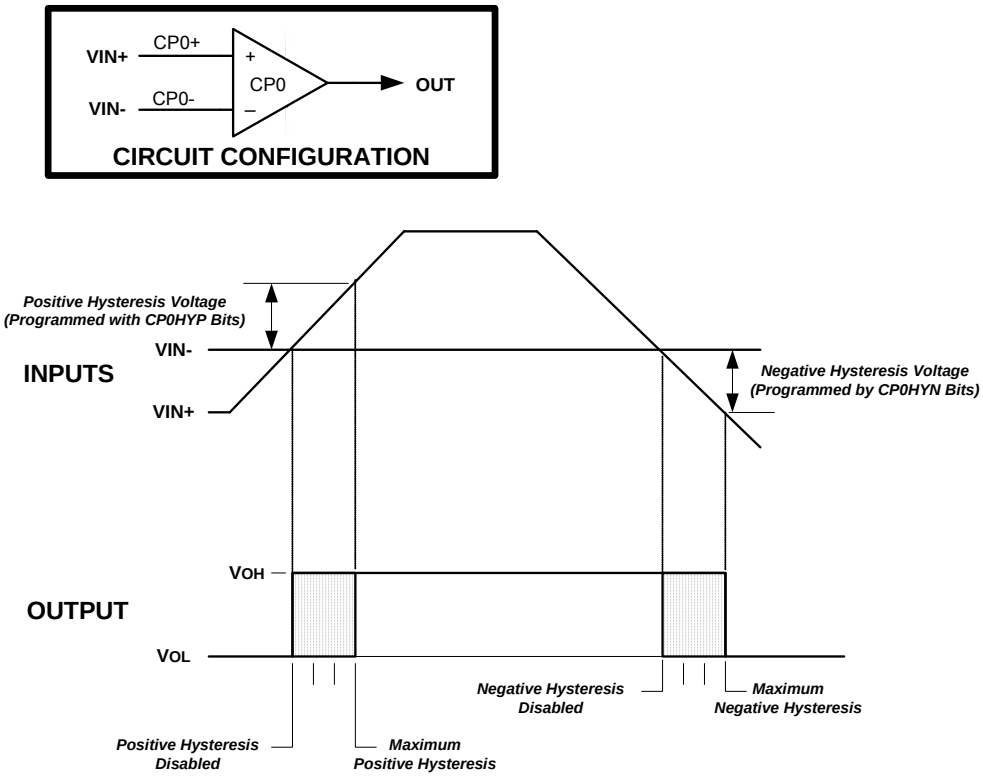


Figure 14.2. Comparator Hysteresis Plot

The Comparator hysteresis is software-programmable via its Comparator 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 Comparator hysteresis is programmed using Bits3–0 in the Comparator Control Register CPT0CN (shown in SFR Definition 14.1). The amount of negative hysteresis voltage is determined by the settings of the CP0HYN bits. As shown in Figure 14.2, settings of 20, 10, or 5 mV of negative hysteresis can be programmed, or negative hysteresis can be disabled. In a similar way, the amount of positive hysteresis is determined by the setting the CP0HYP bits.

Comparator interrupts can be generated on both rising-edge and falling-edge output transitions. (For Interrupt enable and priority control, see Section “20.1. MCU Interrupt Sources and Vectors” on page 118). The CP0FIF flag is set to logic 1 upon a Comparator falling-edge occurrence, and the CP0RIF flag is set to logic 1 upon the Comparator rising-edge occurrence. Once set, these bits remain set until cleared by software. The Comparator rising-edge interrupt mask is enabled by setting CP0RIE to a logic 1. The Comparator0 falling-edge interrupt mask is enabled by setting CP0FIE to a logic 1.

The output state of the Comparator can be obtained at any time by reading the CP0OUT bit. The Comparator is enabled by setting the CP0EN bit to logic 1, and is disabled by clearing this bit to logic 0.

Note that false rising edges and falling edges can be detected when the comparator is first powered on or if changes are made to the hysteresis or response time control bits. Therefore, it is recommended that the rising-edge and falling-edge flags be explicitly cleared to logic 0 a short time after the comparator is enabled or its mode bits have been changed.

SFR Definition 14.1. CPT0CN: Comparator0 Control

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

SFR Address = 0x9B; SFR Page = All Pages

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

SFR Definition 14.2. CPT0MD: Comparator0 Mode Selection

Bit	7	6	5	4	3	2	1	0
Name			CP0RIE	CP0FIE			CP0MD[1:0]	
Type	R	R	R/W	R/W	R	R	R/W	
Reset	0	0	0	0	0	0	1	0

SFR Address = 0x9D; SFR Page = All Pages

Bit	Name	Function
7:6	Unused	Unused. Read = 00b, Write = Don't Care.
5	CP0RIE	**Comparator0 Rising-Edge Interrupt Enable.** 0: Comparator0 Rising-edge interrupt disabled. 1: Comparator0 Rising-edge interrupt enabled.
4	CP0FIE	**Comparator0 Falling-Edge Interrupt Enable.** 0: Comparator0 Falling-edge interrupt disabled. 1: Comparator0 Falling-edge interrupt enabled.
3:2	Unused	Unused. Read = 00b, Write = don't care.
1:0	CP0MD[1:0]	**Comparator0 Mode Select.** These bits affect the response time and power consumption for Comparator0. 00: Mode 0 (Fastest Response Time, Highest Power Consumption) 01: Mode 1 10: Mode 2 11: Mode 3 (Slowest Response Time, Lowest Power Consumption)

14.1. Comparator Multiplexer

C8051F39x/37x devices include an analog input multiplexer to connect Port I/O pins to the comparator inputs. The Comparator0 inputs are selected in the CPT0MX register (SFR Definition 14.3). 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 “27.6. Special Function Registers for Accessing and Configuring Port I/O” on page 185).

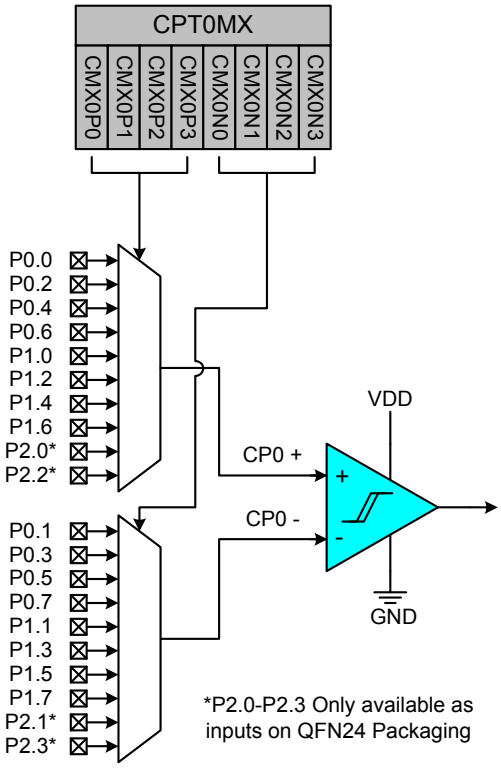


Figure 14.3. Comparator Input Multiplexer Block Diagram

SFR Definition 14.3. CPT0MX: Comparator0 MUX Selection

Bit	7	6	5	4	3	2	1	0
Name	CMX0N[3:0]				CMX0P[3:0]			
Type	R/W				R/W			
Reset	1	1	1	1	1	1	1	1

SFR Address = 0x9F; SFR Page = All Pages

Bit	Name	Function
7:4	CMX0N[3:0]	**Comparator0 Negative Input MUX Selection.** 0000: P0.1 0001: P0.3 0010: P0.5 0011: P0.7 0100: P1.1 0101: P1.3 0110: P1.5 0111: P1.7 1000: P2.1 (C8051F390/1/4/5 and C8051F37x Only) 1001: P2.3 (C8051F390/1/4/5 and C8051F37x Only) 1010-1111: None
3:0	CMX0P[3:0]	**Comparator0 Positive Input MUX Selection.** 0000: P0.0 0001: P0.2 0010: P0.4 0011: P0.6 0100: P1.0 0101: P1.2 0110: P1.4 0111: P1.6 1000: P2.0 (C8051F390/1/4/5 and C8051F37x Only) 1001: P2.2 (C8051F390/1/4/5 and C8051F37x Only) 1010-1111: None

15. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 33), 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 15.1 for a block diagram). The CIP-51 includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 50 MIPS Peak Throughput with 49 MHz Clock
- 0 to 49 MHz Clock Frequency
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

Performance

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

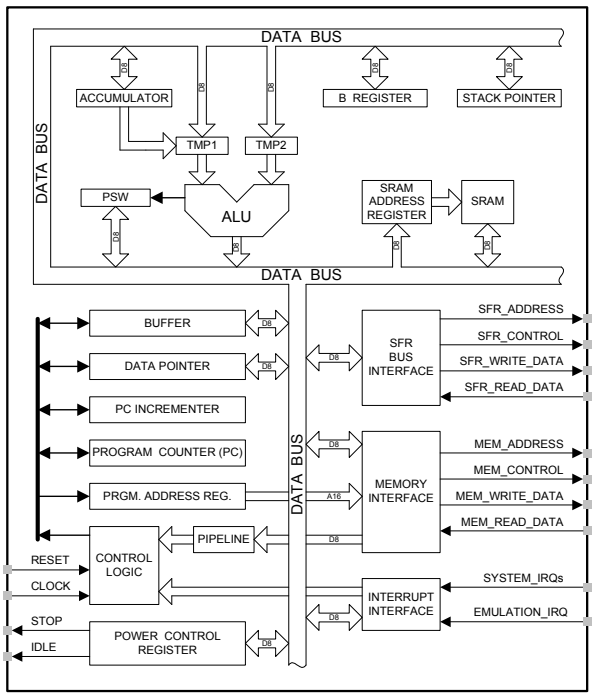


Figure 15.1. CIP-51 Block Diagram

With the CIP-51's maximum system clock at 48 MHz, it has a peak throughput of 48 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	2/4	3	3/5	4	5	4/6	6	8
Number of Instructions	26	50	5	10	7	5	2	1	2	1

Programming and Debugging Support

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

The on-chip debug support logic facilitates full speed in-circuit debugging, allowing the setting of hardware breakpoints, starting, stopping and single stepping through program execution (including interrupt service routines), examination of the program's call stack, and reading/writing the contents of registers and memory. This method of on-chip debugging is completely non-intrusive, requiring no RAM, Stack, timers, or other on-chip resources. C2 details can be found in Section “33. C2 Interface” on page 297.

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.

15.1. 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.

15.1.1. 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 15.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

Table 15.1. 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
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

Table 15.1. CIP-51 Instruction Set Summary (Continued)

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

Table 15.1. CIP-51 Instruction Set Summary (Continued)

Mnemonic	Description	Bytes	Clock Cycles
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	2/4
JNC rel	Jump if Carry is not set	2	2/4
JB bit, rel	Jump if direct bit is set	3	3/5
JNB bit, rel	Jump if direct bit is not set	3	3/5
JBC bit, rel	Jump if direct bit is set and clear bit	3	3/5
Program Branching			
ACALL addr11	Absolute subroutine call	2	4*
LCALL addr16	Long subroutine call	3	5*
RET	Return from subroutine	1	6*
RETI	Return from interrupt	1	6*
AJMP addr11	Absolute jump	2	4*
LJMP addr16	Long jump	3	5*
SJMP rel	Short jump (relative address)	2	4*
JMP @A+DPTR	Jump indirect relative to DPTR	1	4*
JZ rel	Jump if A equals zero	2	2/4*
JNZ rel	Jump if A does not equal zero	2	2/4*
CJNE A, direct, rel	Compare direct byte to A and jump if not equal	3	4/6*
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3	3/5*
CJNE Rn, #data, rel	Compare immediate to Register and jump if not equal	3	3/5*
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3	4/6*
DJNZ Rn, rel	Decrement Register and jump if not zero	2	2/4*
DJNZ direct, rel	Decrement direct byte and jump if not zero	3	3/5*
NOP	No operation	1	1
* Clock cycles for branch instructions with prefetch enabled, Align = 0, FLRT = 0			

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.

15.2. CIP-51 Register Descriptions

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

SFR Definition 15.1. DPL: Data Pointer Low Byte

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

SFR Address = 0x82; SFR Page = All Pages

Bit	Name	Function
7:0	DPL[7:0]	**Data Pointer Low.** The DPL register is the low byte of the 16-bit DPTR.

SFR Definition 15.2. DPH: Data Pointer High Byte

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

SFR Address = 0x83; SFR Page = All Pages

Bit	Name	Function
7:0	DPH[7:0]	**Data Pointer High.** The DPH register is the high byte of the 16-bit DPTR.

SFR Definition 15.3. SP: Stack Pointer

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

SFR Address = 0x81; SFR Page = All Pages

Bit	Name	Function
7:0	SP[7:0]	**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 15.4. ACC: Accumulator

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

SFR Address = 0xE0; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7:0	ACC[7:0]	**Accumulator.** This register is the accumulator for arithmetic operations.

SFR Definition 15.5. B: B Register

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

SFR Address = 0xF0; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7:0	B[7:0]	**B Register.** This register serves as a second accumulator for certain arithmetic operations.

SFR Definition 15.6. PSW: Program Status Word

Bit	7	6	5	4	3	2	1	0
Name	CY	AC	F0	RS[1:0]		OV	F1	PARITY
Type	R/W	R/W	R/W	R/W		R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD0; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	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.
6	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.
5	F0	**User Flag 0.** This is a bit-addressable, general purpose flag for use under software control.
4:3	RS[1:0]	**Register Bank Select.** These bits select which register bank is used during register accesses. 00: Bank 0, Addresses 0x00-0x07 01: Bank 1, Addresses 0x08-0x0F 10: Bank 2, Addresses 0x10-0x17 11: Bank 3, Addresses 0x18-0x1F
2	OV	**Overflow Flag.** This bit 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 bit is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases.
1	F1	**User Flag 1.** This is a bit-addressable, general purpose flag for use under software control.
0	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.

16. Prefetch Engine

The C8051F39x/37x family of devices incorporate a 2-byte prefetch engine. Because the access time of the Flash memory is 40 ns, and the minimum instruction time is roughly 20 ns, the prefetch engine is necessary for full-speed code execution. Instructions are read from Flash memory two bytes at a time by the prefetch engine and given to the CIP-51 processor core to execute. When running linear code (code without any jumps or branches), the prefetch engine allows instructions to be executed at full speed. When a code branch occurs, the processor may be stalled for up to two clock cycles while the next set of code bytes is retrieved from Flash memory.

Note: The prefetch engine should be disabled when the device is in suspend mode to save power.

SFR Definition 16.1. PFE0CN: Prefetch Engine Control

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

SFR Address = 0xB5; SFR Page = All Pages

Bit	Name	Function
7:6	Unused	Unused. Read = 00b, Write = don't care.
5	PFEN	**Prefetch Enable.** This bit enables the prefetch engine. 0: Prefetch engine is disabled. 1: Prefetch engine is enabled.
4:0	Unused	Unused. Read = 00000b. Write = don't care.

17. 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 memory organization of the C8051F39x/37x device family is shown in Figure 17.1. Not shown in Figure 17.1 is 512 bytes of byte-addressable EEPROM available on C8051F37x, accessible by SMBUS/I^2C (see Section 22).

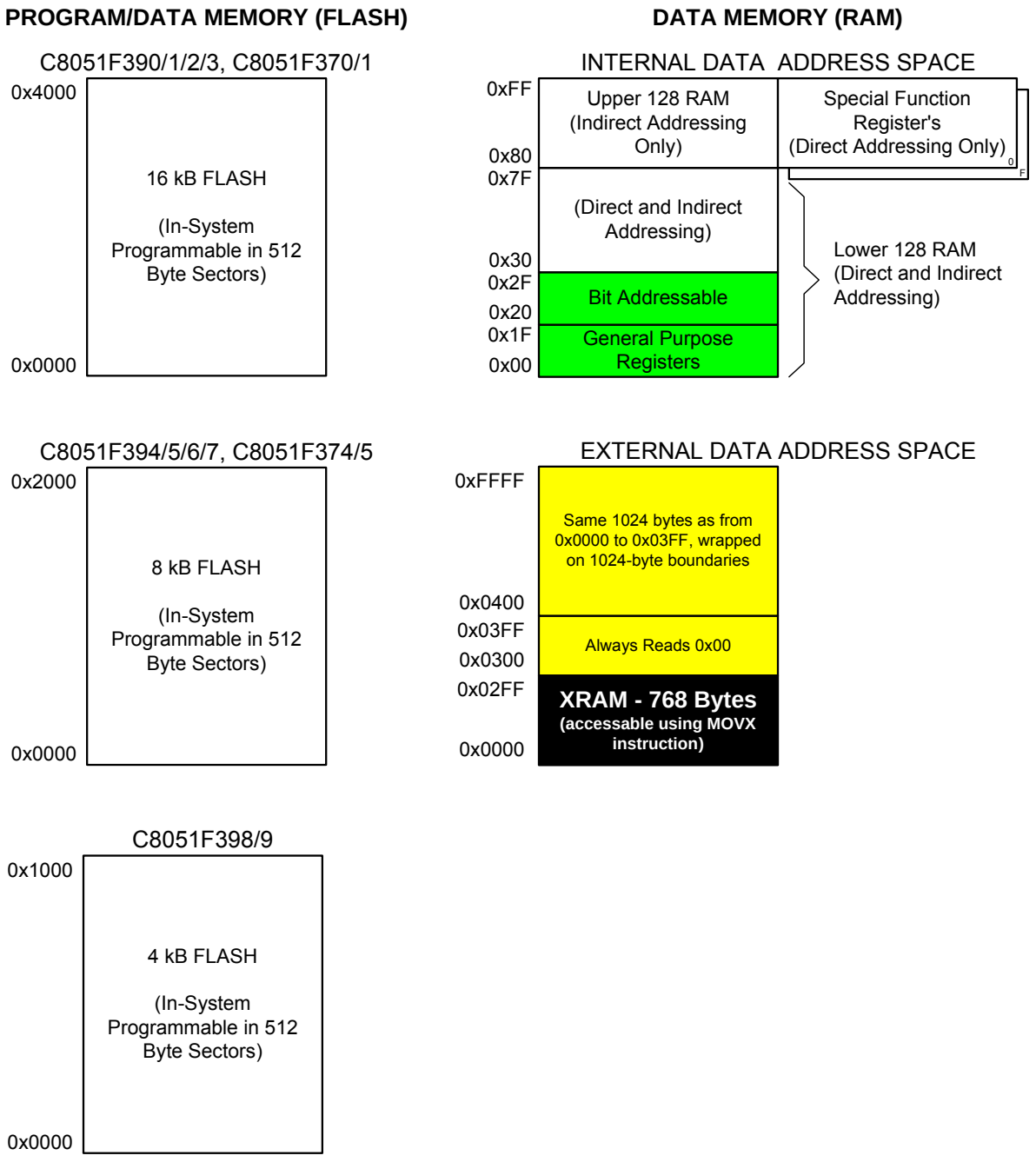


Figure 17.1. C8051F39x/37x Memory Map

17.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051F39x/37x implements 16 kB of this program memory space as in-system, re-programmable Flash memory, organized in a contiguous block from addresses 0x0000 to 0x3FFF. The address 0x3FFF serves as the security lock byte for the device, and addresses above 0x3FFF are reserved.

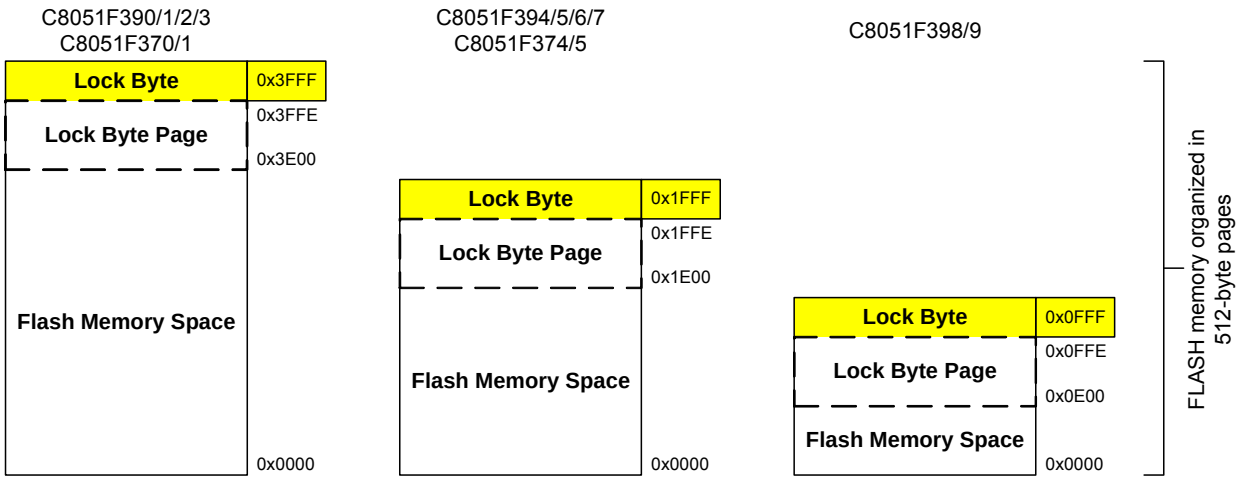


Figure 17.2. Flash Program Memory Map

17.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F39x/37x devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip Flash memory space. MOVC instructions are always used to read Flash memory, while MOVX write instructions are used to erase and write Flash. This Flash access feature provides a mechanism for the C8051F39x/37x to update program code and use the program memory space for non-volatile data storage. Refer to Section “21. Flash Memory” on page 131 for further details.

17.2. Data Memory

The C8051F39x/37x device family includes 1024 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. 768 bytes of this memory is on-chip “external” memory. The data memory map is shown in Figure 17.1 for reference.

17.2.1. Internal RAM

There are 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 17.1 illustrates the data memory organization of the C8051F39x/37x.

17.2.1.1. 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 SFR Definition 15.6). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

17.2.1.2. 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 bit7 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.

17.2.1.3. 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) 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.

17.2.2. External RAM

There are 768 bytes of on-chip RAM mapped into the external data memory space. All of these address locations may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using MOVX indirect addressing mode. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN as shown in SFR Definition 17.1). Note: the MOVX instruction is also used for writes to the Flash memory. See Section “21. Flash Memory” on page 131 for details. The MOVX instruction accesses XRAM by default.

Memory locations between address 0x0300 and 0x03FF will all read back 0x00.

For a 16-bit MOVX operation (@DPTR), the upper 6 bits of the 16-bit external data memory address word are "don't cares". As a result, addresses 0x0000 through 0x03FF are mapped modulo style over the entire 64 k external data memory address range. For example, the XRAM byte at address 0x0000 is shadowed at addresses 0x0400, 0x0800, 0x0C00, 0x1000, etc.

SFR Definition 17.1. EMI0CN: External Memory Interface Control

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

SFR Address = 0xAA; SFR Page = All Pages

Bit	Name	Function
7:2	Unused	Read = 000000b; Write = Don't Care
1:0	PGSEL	**XRAM Page Select.** The PGSEL field provides the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. Since the upper (unused) bits of the register are always zero, the PGSEL determines which page of XRAM is accessed. For Example: If PGSEL = 0x01, addresses 0x0100 through 0x01FF will be accessed.

18. Device ID Registers

The C8051F39x/37x has SFRs that identify the device family and derivative. These SFRs can be read by firmware at runtime to determine the capabilities of the MCU that is executing code. This allows the same firmware image to run on MCUs with different memory sizes and peripherals, and dynamically changing functionality to suit the capabilities of that MCU.

In order for firmware to identify the MCU, it must read two SFRs. DERIVID describes the specific derivative within that device family, and REVID describes the hardware revision of the MCU.

The C8051F39x/37x devices also include four SFRs, SN0 through SN3, that are pre-programmed during production with a unique, 32-bit serial number. The serial number provides a unique identification number for each device and can be read from the application firmware. If the serial number is not used in the application, these four registers can be used as general purpose SFRs.

SFR Definition 18.1. DERIVID: Device Derivative ID

Bit	7	6	5	4	3	2	1	0
Name	DERIVID							
Type	R							
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xAB; SFR Page = 0

Bit	Name	Function
7:0	DERIVID	**Derivative ID.** This read-only register returns the 8-bit derivative ID, which can be used by firmware to identify which device in the product family is being used. 0xD0: C8051F390 0xD1: C8051F391 0xD2: C8051F392 0xD3: C8051F393 0xD4: C8051F394 0xD5: C8051F395 0xD6: C8051F396 0xD7: C8051F397 0xD8: C8051F398 0xD9: C8051F399 0xE0: C8051F370 0xE1: C8051F371 0xE4: C8051F374 0xE5: C8051F375

SFR Definition 18.2. REVISION: Device Revision ID

Bit	7	6	5	4	3	2	1	0
Name	REVISION							
Type	R							
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xAC; SFR Page = 0

Bit	Name	Function
7:0	REVISION	**Device Revision.** This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A.

SFR Definition 18.3. SN3: Serial Number Byte 3

Bit	7	6	5	4	3	2	1	0
Name	SN3							
Type	R							
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xAE; SFR Page = F

Bit	Name	Function
7:0	SN3	**Serial Number Byte 3.** This read-only register returns the MSB (byte 3) of the serial number.

SFR Definition 18.4. SN2: Serial Number Byte 2

Bit	7	6	5	4	3	2	1	0
Name	SN2							
Type	R							
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xAD; SFR Page = F

Bit	Name	Function
7:0	SN2	**Serial Number Byte 2.** This read-only register returns the byte 2 of the serial number.

SFR Definition 18.5. SN1: Serial Number Byte 1

Bit	7	6	5	4	3	2	1	0
Name	SN1							
Type	R							
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xAC; SFR Page = F

Bit	Name	Function
7:0	SN1	**Serial Number Byte 1.** This read-only register returns byte 1 of the serial number.

SFR Definition 18.6. SN0: Serial Number Byte 0

Bit	7	6	5	4	3	2	1	0
Name	SN0							
Type	R							
Reset	Varies	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xAB; SFR Page = F

Bit	Name	Function
7:0	SN0	**Serial Number Byte 0.** This read-only register returns the LSB (byte 0) of the serial number.

19. 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 C8051F39x/37x's resources and peripherals. The CIP-51 controller core 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 C8051F39x/37x. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 19.2 lists the SFRs implemented in the C8051F39x/37x device family.

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 19.3, for a detailed description of each register.

19.1. SFR Paging

The CIP-51 features SFR paging, allowing the device to map many SFRs into the 0x80 to 0xFF memory address space. The SFR memory space has 256 pages. In this way, each memory location from 0x80 to 0xFF can access up to 256 SFRs. The C8051F39x/37x devices utilize two SFR pages: 0x0, and 0xF. Most SFRs are available on both pages. SFR pages are selected using the Special Function Register Page Selection register, SFRPAGE. The procedure for reading and writing an SFR is as follows:

1. Select the appropriate SFR page number using the SFRPAGE register.
2. Use direct accessing mode to read or write the special function register (MOV instruction).

19.2. Interrupts and Automatic SFR Paging

When an interrupt occurs, the current SFRPAGE is pushed onto the SFR page stack. Upon execution of the RETI instruction, the SFR page is automatically restored to the SFR page in use prior to the interrupt. This is accomplished via a five-byte *SFR page stack*, depicted in Figure 19.1. Firmware can read any element of the SFR page stack by setting the SFR Page Stack Index (SFRPGIDX) in the SFR Page Control Register (SFRPGCN) and reading the SFRSTACK register:

Table 19.1. SFR Page Stack

SFRPGIDX Value	SFRSTACK Contains
000b	Value of the first/top* byte of the stack
001b	Value of the second byte of the stack
011b	Value of the third byte of the stack
010b	Value of the forth byte of the stack
100b	Value of the fifth/bottom byte of the stack
***Note:** The first/top byte of the stack can also be directly accessed by reading SFRPAGE.	

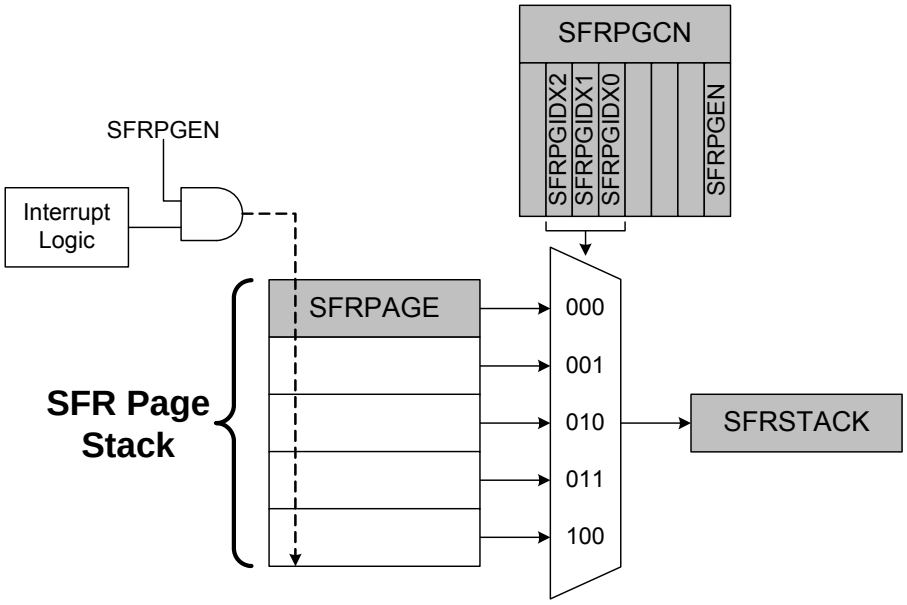


Figure 19.1. SFR Page Stack

Upon an interrupt, hardware performs the five following operations:

1. The value (if any) in the SFRPGIDX = 011b location is pushed to the SFRPAGE = 100b location.
2. The value (if any) in the SFRPGIDX = 010b location is pushed to the SFRPAGE = 011b location.
3. The value (if any) in the SFRPGIDX = 001b location is pushed to the SFRPAGE = 010b location.
4. The current SFRPAGE value is pushed to the SFRPGIDX = 001b location in the stack.
5. SFRPAGE is set to the page corresponding to the flag which generated the interrupt.

On a return from interrupt, hardware performs the four following operations:

1. The SFR page stack is popped resulting in the value in the SFRPGIDX = 001b location returning to the SFRPAGE register, thereby restoring the SFR page context without software intervention.
2. The value in the SFRPGIDX = 010b location of the stack is placed in the SFRPGIDX = 001b location.
3. The value in the SFRPGIDX = 011b location of the stack is placed in the SFRPGIDX = 010b location.
4. The value in the SFRPGIDX = 100b location of the stack is placed in the SFRPGIDX = 011b location.

Automatic switching of the SFR page by hardware upon interrupt entries and exits may be enabled or disabled using the SFR Automatic Page Control Enable Bit (SFRPGEN) located in SFRPGCN. The automatic SFR page switching is enabled after a reset until disabled by firmware.

19.3. SFR Page Stack Example

In this example, the SFR Control register is left in the default enabled state (SFRPGEN set to 1), and the core is executing in-line code that is writing values to Temperature Sensor Control Register (TS0CN). The device is also using the SPI peripheral (SPI0) and the Programmable Counter Array (PCA0) peripheral to generate a PWM output. The PCA is timing a critical control function in its interrupt service routine, therefore, its associated ISR is set to high priority. At this point, the SFR page is set to 0x0F to access the TS0CN SFR. See Figure 19.2.

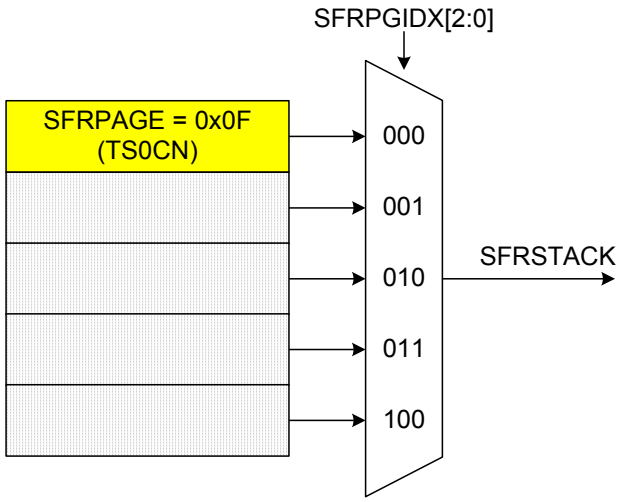


Figure 19.2. SFR Page Stack While Using SFR Page 0x0F To Access TS0CN

The SPI0 interrupt occurs while the core executes in-line code by writing a value to TS0CN. The core vectors to the SPI0 ISR and pushes the current SFR page value (in this case SFR page 0x0F for TS0CN) into the 001b SFRPGIDX location in the SFR page stack. Also, the core automatically places the SFR page (0x00) needed to access the SPI0's special function registers into the SFRPAGE register. See Figure 19.3.

SFRPAGE is considered the top of the SFR page stack. Software may switch to any SFR page by writing a new value to the SFRPAGE register at any time during the SPI0 ISR.

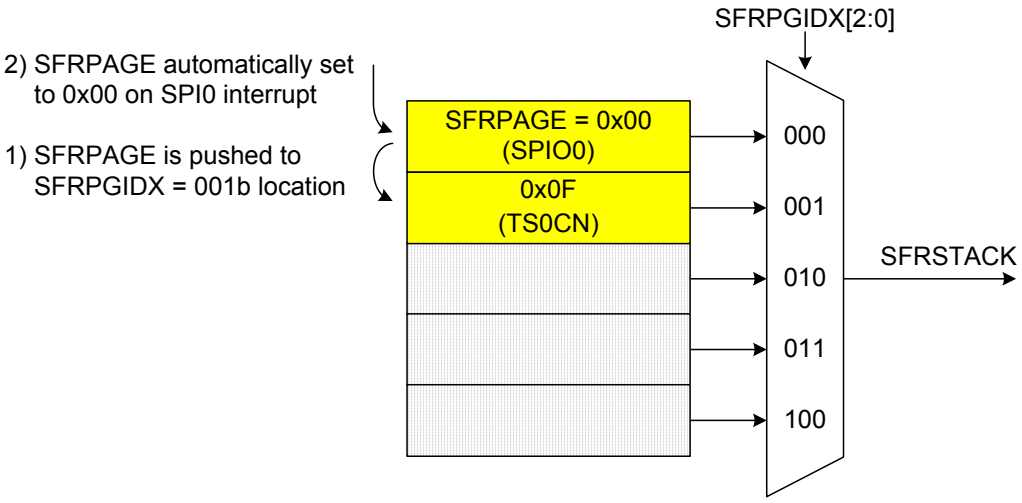


Figure 19.3. SFR Page Stack After SPI0 Interrupt Occurs

While in the SPI0 ISR, a PCA interrupt occurs. Recall the PCA interrupt is configured as a high priority interrupt, while the SPI0 interrupt is configured as a low priority interrupt. Thus, the CIP-51 will now vector to the high priority PCA ISR. Upon doing so, the value that was in the SFRPGIDX = 001b location before the PCA interrupt (in this case SFR page 0x0F for TS0CN) is pushed down to the SFRPGIDX = 010b location. Likewise, the value that was in the SFRPAGE register before the PCA interrupt (SFR page 0x00 for SPI0) is pushed down the stack into the SFRPGIDX = 001b location. Lastly, the CIP-51 will automatically places the SFR page needed to access the PCA0's special function registers into the SFRPAGE register, SFR page 0x00. See Figure 19.4.

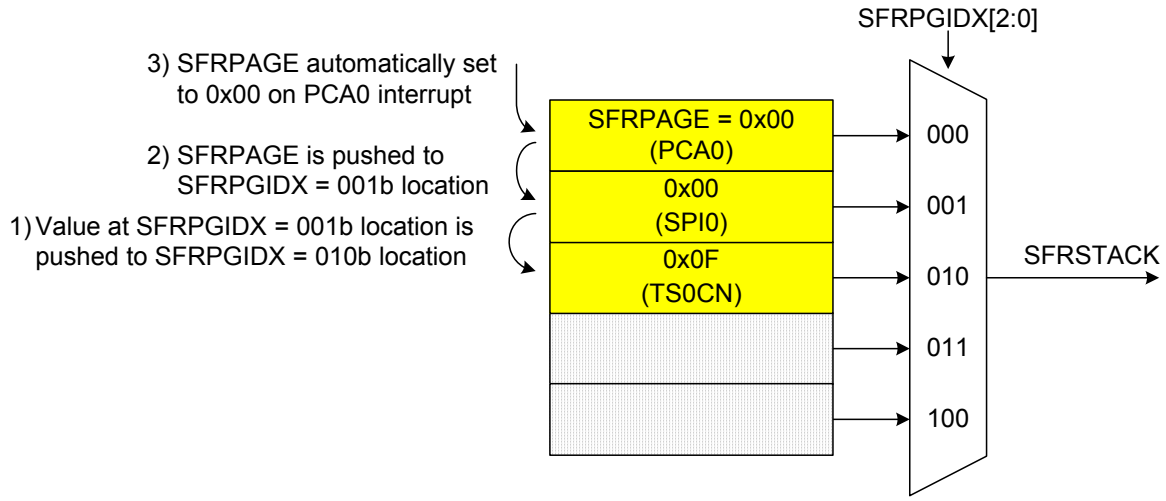


Figure 19.4. SFR Page Stack Upon PCA Interrupt Occurring During a SPI0 ISR

On exit from the PCA0 interrupt service routine, the CIP-51 will return to the SPI0 ISR. On execution of the RETI instruction, SFR page 0x00 used to access the PCA0 registers will be automatically popped off of the SFR page stack, and the contents at the SFRPGIDX = 001b location will be moved to the SFRPAGE register. Software in the SPI0 ISR can continue to access SFRs as it did prior to the PCA interrupt. Likewise, the contents at the SFRPGIDX = 010b location are moved to the SFRPGIDX = 001b location. Recall this was the SFR Page value 0x0F being used to access TS0CN before the SPI0 interrupt occurred. See Figure 19.5.

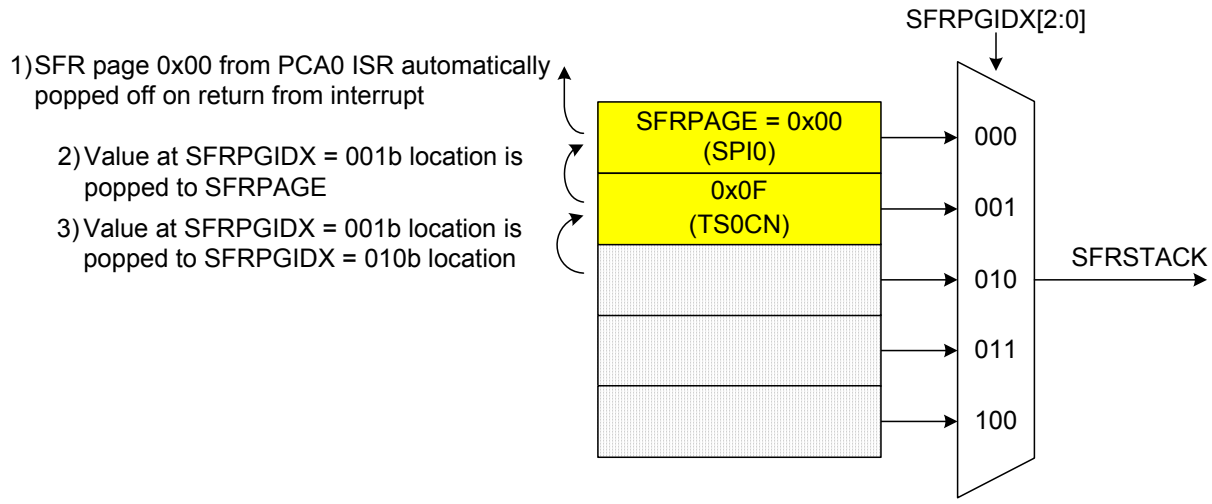


Figure 19.5. SFR Page Stack Upon Return from PCA0 Interrupt

On the execution of the RETI instruction in the SPI0 ISR, the value in SFRPAGE register is overwritten with the contents at the SFRPGIDX = 001b location. The CIP-51 may now access the TS0CN register as it did prior to the interrupts occurring. See Figure 19.6.

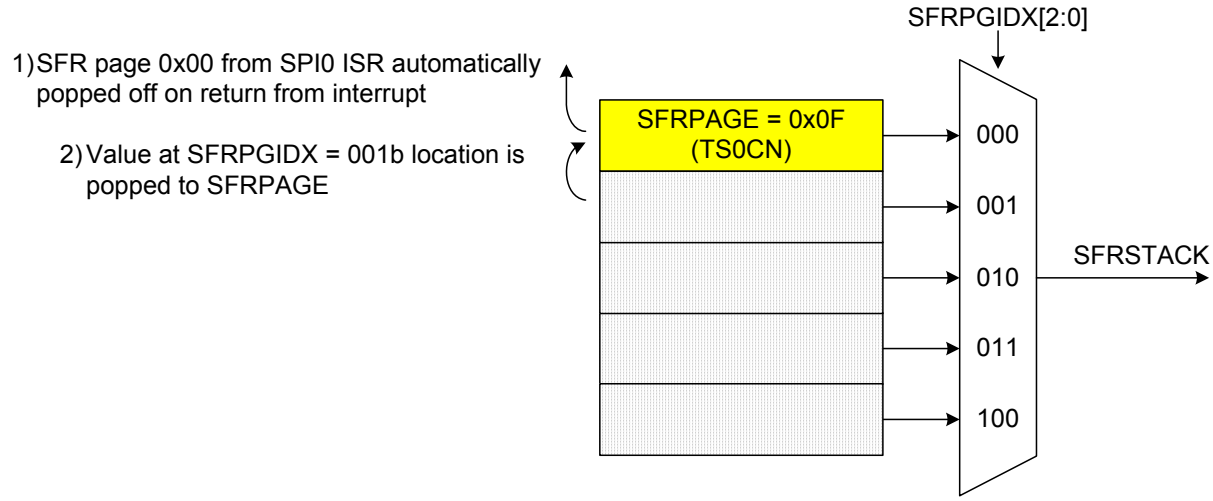


Figure 19.6. SFR Page Stack Upon Return From SPI0 Interrupt

Push operations on the SFR page stack only occur on interrupt service, and pop operations only occur on interrupt exit (execution on the RETI instruction). The automatic switching of the SFRPAGE and operation of the SFR page stack as described above can be disabled in software by clearing the SFR Automatic Page Enable Bit (SFRPGEN) in the SFR Page Control Register (SFRPGCN).

SFR Definition 19.1. SFRPAGE: SFR Page

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

SFR Address = 0xA7; SFR Page = All Pages

Bit	Name	Function
7:0	SFRPAGE[7:0]	**SFR Page Bits.** Represents the SFR Page the C8051 core uses when reading or modifying SFRs. Write: Sets the SFR Page. Read: Byte is the SFR page the C8051 core is using.

SFR Definition 19.2. SFRPGCN: SFR Page Control

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

SFR Address = 0xCF; SFR Page = All Pages

Bit	Name	Function
7	Reserved	Must Write 0b
6:4	SFRPGIDX[2:0]	**SFR Page Stack Index.** This field can be used to access the SFRPAGE values stored in the SFR page stack. It selects which level of the stack is accessible when reading the SFRSTACK register. 000: SFRSTACK contains the value of SFRPAGE, the first/top byte of the SFR page stack 001: SFRSTACK contains the value of the second byte of the SFR page stack 010: SFRSTACK contains the value of the third byte of the SFR page stack 011: SFRSTACK contains the value of the forth byte of the SFR page stack 100: SFRSTACK contains the value of the fifth/bottom byte of the SFR page stack 101: Invalid index 11x: Invalid index
3:1	Reserved	Must Write 000b
0	SFRPGEN	**SFR Automatic Page Control Enable.** This bit is used to enable automatic page switching on ISR entry/exit. When set to 1, the current SFRPAGE value will be pushed onto the SFR page stack, and SFRPAGE will be set to the page corresponding to the flag which generated the interrupt; upon ISR exit, hardware will pop the value from the SFR page stack and restore SFRPAGE. 0: Disable automatic SFR paging. 1: Enable automatic SFR paging.

SFR Definition 19.3. SFRSTACK: SFR Page Stack

Bit	7	6	5	4	3	2	1	0
Name	SFRSTACK							
Type	R							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD3; SFR Page = F

Bit	Name	Function
7:0	SFRSTACK	**SFR Page Stack.** This register is used to access the contents of the SFR page stack. SFRPGIDX in the SFRPGCN register controls which level of the stack this register will access.

Table 19.2. Special Function Register (SFR) Memory Map

Address	Page	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)
F8		SPI0CN	PCA0L	PCA0H	PCA0CPL0	PCA0CPH0	P0MAT	P0MASK	VDM0CN
F0		B	P0MDIN	P1MDIN	P2MDIN	CKCON1		EIP1	PCA0PWM
E8		ADC0CN	PCA0CPL1	PCA0CPH1	PCA0CPL2	PCA0CPH2	P1MAT	P1MASK	RSTSRC
E0	0	ACC	XBR0	XBR1	OSCLCN	IT01CF		EIE1	SMB0ADM
	F								SMB1ADM
D8		PCA0CN	PCA0MD	PCA0CPM0	PCA0CPM1	PCA0CPM2	CRC0AUTO	CRC0CNT	CRC0CN
D0	0	PSW	REF0CN	TS0DATL	TS0DATH	P0SKIP	P1SKIP	P2SKIP	SMB0ADR
	F			TS0CN	SFRSTACK				SMB1ADR
C8		TMR2CN	REG0CN	TMR2RLL	TMR2RLH	TMR2L	TMR2H	PCA0CLR	SFRPGCN
		TMR5CN		TMR5RLL	TMR5RLH	TMR5L	TMR5H		
C0	0	SMB0CN	SMB0CF	SMB0DAT	ADC0GTL	ADC0GTH	ADC0LTL	ADC0LTH	SMBTC
	F	SMB1CN	SMB1CF	SMB1DAT					
B8	0	IP	IDA0CN	AMX0N	AMX0P	ADC0CF	ADC0L	ADC0H	EIP2
	F		IDA1CN						
B0			OSCXCN	OSCICN	OSCICL		PFE0CN	FLSCL	FLKEY
A8	0	IE	CLKSEL	EMI0CN	DERIVID	REVISION			EIE2
	F				SN0	SN1	SN2	SN3	
A0		P2	SPI0CFG	SPI0CKR	SPI0DAT	P0MDOUT	P1MDOUT	P2MDOUT	SFRPAGE
98		SCON0	SBUF0	CRC0FLIP	CPT0CN	CRC0IN	CPT0MD	CRC0DAT	CPT0MX
90	0	P1	TMR3CN	TMR3RLL	TMR3RLH	TMR3L	TMR3H	IDA0L	IDA0H
	F		TMR4CN	TMR4RLL	TMR4RLH	TMR4L	TMR4H	IDA1L	IDA1H
88		TCON	TMOD	TL0	TL1	TH0	TH1	CKCON	PSCTL
80		P0	SP	DPL	DPH	IPH	EIP1H	EIP2H	PCON
		0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)

Notes:

1. SFR Addresses ending in 0x0 or 0x8 are bit-addressable locations and can be used with bitwise instructions.
2. Unless indicated otherwise, SFRs are available on both page 0 and page F.

Table 19.3. Special Function Registers

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

Register	Address	SFR Page	Description	Page
ACC	0xE0	All Pages	Accumulator	89
ADC0CF	0xBC	All Pages	ADC0 Configuration	55
ADC0CN	0xE8	All Pages	ADC0 Control	57
ADC0GTH	0xC4	All Pages	ADC0 Greater-Than Compare High	58
ADC0GTL	0xC3	All Pages	ADC0 Greater-Than Compare Low	58
ADC0H	0xBE	All Pages	ADC0 High	56
ADC0L	0xBD	All Pages	ADC0 Low	56
ADC0LTH	0xC6	All Pages	ADC0 Less-Than Compare Word High	59
ADC0LTL	0xC5	All Pages	ADC0 Less-Than Compare Word Low	59
AMX0N	0xBA	All Pages	AMUX0 Negative Channel Select	63
AMX0P	0xBB	All Pages	AMUX0 Positive Channel Select	62
B	0xF0	All Pages	B Register	90
CKCON	0x8E	All Pages	Clock Control	243
CKCON1	0xF4	All Pages	Clock Control 1	244
CLKSEL	0xA9	All Pages	Clock Select	165
CPT0CN	0x9B	All Pages	Comparator0 Control	78
CPT0MD	0x9D	All Pages	Comparator0 Mode Selection	79
CPT0MX	0x9F	All Pages	Comparator0 MUX Selection	81
CRC0AUTO	0xDD	All Pages	CRC0 Automatic Control	152
CRC0CN	0xDF	All Pages	CRC0 Control	150
CRC0CNT	0xDE	All Pages	CRC0 Automatic Flash Sector Count	153
CRC0DAT	0x9E	All Pages	CRC0 Data Output	151
CRC0FLIP	0x9A	All Pages	CRC0 Bit Flip	154
CRC0IN	0x9C	All Pages	CRC0 Data Input	151
DERIVID	0xAB	0	Device Derivative ID	97
DPH	0x83	All Pages	Data Pointer High	88
DPL	0x82	All Pages	Data Pointer Low	88
EIE1	0xE6	All Pages	Extended Interrupt Enable 1	123
EIE2	0xAF	All Pages	Extended Interrupt Enable 2	126
EIP1	0xF6	All Pages	Extended Interrupt Priority 1	124
EIP1H	0x85	All Pages	Extended Interrupt Priority 1 High	125
EIP2	0xBF	All Pages	Extended Interrupt Priority 2	127
EIP2H	0x86	All Pages	Extended Interrupt Priority 2 High	127

Table 19.3. Special Function Registers (Continued)

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

Register	Address	SFR Page	Description	Page
EMI0CN	0xAA	All Pages	External Memory Interface Control	96
FLKEY	0xB7	All Pages	Flash Lock and Key	138
FLSCL	0xB6	All Pages	Flash Scale	139
IDA0CN	0xB9	0	Current Mode DAC0 Control	69
IDA0H	0x97	0	Current Mode DAC0 High	70
IDA0L	0x96	0	Current Mode DAC0 Low	70
IDA1CN	0xB9	F	Current Mode DAC1 Control	71
IDA1H	0x97	F	Current Mode DAC1 High	72
IDA1L	0x96	F	Current Mode DAC1 Low	72
IE	0xA8	All Pages	Interrupt Enable	120
IP	0xB8	All Pages	Interrupt Priority	121
IPH	0x84	All Pages	Interrupt Priority High	122
IT01CF	0xE4	All Pages	INT0/INT1 Configuration	129
OSCICL	0xB3	All Pages	Internal Oscillator Calibration	166
OSCICN	0xB2	All Pages	Internal Oscillator Control	167
OSCLCN	0xE3	All Pages	Low-Frequency Oscillator Control	168
OSCXCN	0xB1	All Pages	External Oscillator Control	172
P0	0x80	All Pages	Port 0 Latch	186
P0MASK	0xFE	All Pages	Port 0 Mask Configuration	183
P0MAT	0xFD	All Pages	Port 0 Match Configuration	184
P0MDIN	0xF1	All Pages	Port 0 Input Mode Configuration	186
P0MDOUT	0xA4	All Pages	Port 0 Output Mode Configuration	187
P0SKIP	0xD4	All Pages	Port 0 Skip	187
P1	0x90	All Pages	Port 1 Latch	188
P1MASK	0xEE	All Pages	Port 1Mask Configuration	184
P1MAT	0xED	All Pages	Port 1 Match Configuration	185
P1MDIN	0xF2	All Pages	Port 1 Input Mode Configuration	188
P1MDOUT	0xA5	All Pages	Port 1 Output Mode Configuration	189
P1SKIP	0xD5	All Pages	Port 1 Skip	189
P2	0xA0	All Pages	Port 2 Latch	190
P2MDIN	0xF3	All Pages	Port 2 Input Mode Configuration	190
P2MDOUT	0xA6	All Pages	Port 2 Output Mode Configuration	191
P2SKIP	0xD6	All Pages	Port 2 Skip	191
PCA0CLR	0xCE	All Pages	PCA Comparator Clear Control	293

Table 19.3. Special Function Registers (Continued)

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

Register	Address	SFR Page	Description	Page
PCA0CN	0xD8	All Pages	PCA Control	290
PCA0CPH0	0xFC	All Pages	PCA Capture 0 High	296
PCA0CPH1	0xEA	All Pages	PCA Capture 1 High	296
PCA0CPH2	0xEC	All Pages	PCA Capture 2 High	296
PCA0CPL0	0xFB	All Pages	PCA Capture 0 Low	296
PCA0CPL1	0xE9	All Pages	PCA Capture 1 Low	296
PCA0CPL2	0xEB	All Pages	PCA Capture 2 Low	296
PCA0CPM0	0xDA	All Pages	PCA Module 0 Mode Register	294
PCA0CPM1	0xDB	All Pages	PCA Module 1 Mode Register	294
PCA0CPM2	0xDC	All Pages	PCA Module 2 Mode Register	294
PCA0H	0xFA	All Pages	PCA Counter High	295
PCA0L	0xF9	All Pages	PCA Counter Low	295
PCA0MD	0xD9	All Pages	PCA Mode	291
PCA0PWM	0xF7	All Pages	PCA PWM Configuration	292
PCON	0x87	All Pages	Power Control	163
PFE0CN	0xB5	All Pages	Prefetch Engine Control	92
PSCTL	0x8F	All Pages	Program Store R/W Control	137
PSW	0xD0	All Pages	Program Status Word	91
REF0CN	0xD1	All Pages	Voltage Reference Control	74
REG0CN	0xC9	All Pages	Voltage Regulator Control	75
REVISION	0xAC	0	Device Revision	98
RSTSRC	0xEF	All Pages	Reset Source Configuration/Status	160
SBUF0	0x99	All Pages	UART0 Data Buffer	226
SCON0	0x98	All Pages	UART0 Control	225
SFRPAGE	0xBF	All Pages	SFR Page	108
SFRPGCN	0xBF	All Pages	SFR Page Control	109
SFRSTACK	0xBF	F	SFR Page Stack	110
SMB0ADM	0xE7	0	SMBus0 Slave Address Mask	206
SMB0ADR	0xD7	0	SMBus0 Slave Address	205
SMB0CF	0xC1	0	SMBus0 Configuration	198
SMB0CN	0xC0	0	SMBus0 Control	202
SMB0DAT	0xC2	0	SMBus0 Data	209
SMB1ADM	0xE7	F	SMBus1 Slave Address Mask	208
SMB1ADR	0xD7	F	SMBus1 Slave Address	207

Table 19.3. Special Function Registers (Continued)

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

Register	Address	SFR Page	Description	Page
SMB1CF	0xC1	F	SMBus1 Configuration	199
SMB1CN	0xC0	F	SMBus1 Control	203
SMB1DAT	0xC2	F	SMBus1 Data	210
SMBTC	0xC7	All Pages	SMBus Timing Control	200
SN0	0xAB	F	Serial Number Byte 0	100
SN1	0xAC	F	Serial Number Byte 1	99
SN2	0xAD	F	Serial Number Byte 2	99
SN3	0xAE	F	Serial Number Byte 3	98
SP	0x81	All Pages	Stack Pointer	89
SPI0CFG	0xA1	All Pages	SPI Configuration	235
SPI0CKR	0xA2	All Pages	SPI Clock Rate Control	237
SPI0CN	0xF8	All Pages	SPI Control	236
SPI0DAT	0xA3	All Pages	SPI Data	238
TCON	0x88	All Pages	Timer/Counter Control	249
TH0	0x8C	All Pages	Timer/Counter 0 High	252
TH1	0x8D	All Pages	Timer/Counter 1 High	252
TL0	0x8A	All Pages	Timer/Counter 0 Low	251
TL1	0x8B	All Pages	Timer/Counter 1 Low	251
TMOD	0x89	All Pages	Timer/Counter Mode	250
TMR2CN	0xC8	0	Timer/Counter 2 Control	256
TMR2H	0xCD	0	Timer/Counter 2 High	258
TMR2L	0xCC	0	Timer/Counter 2 Low	257
TMR2RLH	0xCB	0	Timer/Counter 2 Reload High	257
TMR2RLL	0xCA	0	Timer/Counter 2 Reload Low	257
TMR3CN	0x91	0	Timer/Counter 3 Control	262
TMR3H	0x95	0	Timer/Counter 3 High	264
TMR3L	0x94	0	Timer/Counter 3 Low	263
TMR3RLH	0x93	0	Timer/Counter 3 Reload High	263
TMR3RLL	0x92	0	Timer/Counter 3 Reload Low	263
TMR4CN	0x91	F	Timer/Counter 4 Control	267
TMR4H	0x95	F	Timer/Counter 4 High	269
TMR4L	0x94	F	Timer/Counter 4 Low	268
TMR4RLH	0x93	F	Timer/Counter 4 Reload High	268
TMR4RLL	0x92	F	Timer/Counter 4 Reload Low	268

Table 19.3. Special Function Registers (Continued)

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

Register	Address	SFR Page	Description	Page
TMR5CN	0xC8	F	Timer/Counter 5 Control	272
TMR5H	0xCD	F	Timer/Counter 5 High	274
TMR5L	0xCC	F	Timer/Counter 5 Low	273
TMR5RLH	0xCB	F	Timer/Counter 5 Reload High	273
TMR5RLL	0xCA	F	Timer/Counter 5 Reload Low	273
TS0CN	0xD2	F	Temperature Sensor Control	48
TS0DATH	0xD3	0	Temperature Sensor Data High	49
TS0DATL	0xD2	0	Temperature Sensor Data Low	49
VDM0CN	0xFF	All Pages	V_{DD} Monitor Control	158
XBR0	0xE1	All Pages	Port I/O Crossbar Control 0	181
XBR1	0xE2	All Pages	Port I/O Crossbar Control 1	182

20. Interrupts

The C8051F39x/37x includes an extended interrupt system supporting multiple interrupt sources with four priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

If interrupts are enabled for the source, an interrupt request is generated when the interrupt-pending flag is set. As soon as execution of the current instruction is complete, the CPU generates an LCALL to a predetermined address to begin execution of an interrupt service routine (ISR). Each ISR must end with 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, and EIE2). 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 a bit to disable an interrupt should be immediately followed by an instruction that has two or more opcode bytes. Using EA (global interrupt enable) as an example:

```
// in 'C':
EA = 0; // clear EA bit.
EA = 0; // this is a dummy instruction with two-byte opcode.

; in assembly:
CLR EA ; clear EA bit.
CLR EA ; this is a dummy instruction with two-byte opcode.
```

For example, if an interrupt is posted during the execution phase of a "CLR EA" opcode (or any instruction which clears a bit to disable an interrupt source), and the instruction is followed by a single-cycle instruction, the interrupt may be taken. However, a read of the enable bit will return a '0' inside the interrupt service routine. When the bit-clearing 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.

20.1. MCU Interrupt Sources and Vectors

The C8051F39x/37x MCUs support 18 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 20.2. Refer to the datasheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

20.1.1. Interrupt Priorities

Each interrupt source can be individually programmed to one of four priority levels. This differs from the traditional two priority levels on the 8051 core. However, the implementation of the extra levels is backwards-compatible with legacy 8051 code.

An interrupt service routine can be preempted by any interrupt of higher priority. Interrupts at the highest priority level cannot be preempted. Each interrupt has two associated priority bits which are used to configure the priority level. For backwards compatibility, the bits are spread across two different registers. The LSBs of the priority setting are stored in the IP, EIP1 and EIP2 registers, while the MSBs are store in the IPH, EIP1H and EIP2H registers. Priority levels according to the MSB and LSB are decoded in Table 20.1. The lowest priority setting is the default for all interrupts. If two or more interrupts are recognized simultaneously, the interrupt with the highest priority is serviced first. If both interrupts have the same priority level, a fixed priority order is used to arbitrate, given in Table 20.2. If legacy 8051 operation is desired, the bits of the “High” priority registers (IPH, EIP1H and EIP2H) should all be configured to 0 (this is the reset value of these registers).

Priority MSB (from IPH, EIP1H or EIP2H)	Priority LSB (from IP, EIP1 or EIP2)	Priority Level
0	0	Priority 0 (lowest priority, default)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3 (highest priority)

Table 20.1. Configurable Interrupt Priority Decoding

20.1.2. 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. If more than one interrupt is pending when the CPU exits an ISR, the CPU will service the next highest priority interrupt that is pending.

Table 20.2. Interrupt Summary

Interrupt Source	Interrupt Vector	Priority Order	Pending Flags	Bit addressable?	Cleared by HW?	Enable Flag
Reset	0x0000	Top	None	N/A	N/A	Always Enabled
External Interrupt 0 ($\overline{INT0}$)	0x0003	0	IE0 (TCON.1)	Y	Y	EX0 (IE.0)
Timer 0 Overflow	0x000B	1	TF0 (TCON.5)	Y	Y	ET0 (IE.1)
External Interrupt 1 ($\overline{INT1}$)	0x0013	2	IE1 (TCON.3)	Y	Y	EX1 (IE.2)
Timer 1 Overflow	0x001B	3	TF1 (TCON.7)	Y	Y	ET1 (IE.3)
UART0	0x0023	4	RI0 (SCON0.0) TI0 (SCON0.1)	Y	N	ES0 (IE.4)
Timer 2 Overflow	0x002B	5	TF2H (TMR2CN.7) TF2L (TMR2CN.6)	Y	N	ET2 (IE.5)
SPI0	0x0033	6	SPIF (SPI0CN.7) WCOL (SPI0CN.6) MODF (SPI0CN.5) RXOVRN (SPI0CN.4)	Y	N	ESPI0 (IE.6)
SMB0	0x003B	7	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)
Port Match	0x0043	8	None	N/A	N/A	EMAT (EIE1.1)
ADC0 Window Compare	0x004B	9	AD0WINT (ADC0CN.3)	Y	N	EWADC0 (EIE1.2)
ADC0 Conversion Complete	0x0053	10	AD0INT (ADC0CN.5)	Y	N	EADC0 (EIE1.3)
Programmable Counter Array	0x005B	11	CF (PCA0CN.7) CCFn (PCA0CN.n) COVF (PCA0PWM.6)	Y	N	EPCA0 (EIE1.4)
Comparator0	0x0063	12	CP0FIF (CPT0CN.4) CP0RIF (CPT0CN.5)	N	N	ECP0 (EIE1.5)
Reserved	0x006B	13	N/A	N/A	N/A	N/A
Timer 3 Overflow	0x0073	14	TF3H (TMR3CN.7) TF3L (TMR3CN.6)	N	N	ET3 (EIE1.7)
SMB1	0x007B	15	SI (SMB0CN.0)	Y	N	ESMB0 (EIE2.0)
Timer 4 Overflow	0x0083	16	TF4H (TMR4CN.7) TF4L (TMR4CN.6)	Y	N	ET4 (EIE2.1)
Timer 5 Overflow	0x008B	17	TF5H (TMR5CN.7) TF5L (TMR5CN.6)	N	N	ET5 (EIE2.2)
Precision Temp Sensor	0x0093	18	TS0DN (TS0CN.6)	N	N	EPTS (EIE2.3)

20.2. Interrupt Register Descriptions

The SFRs used to enable the interrupt sources and set their priority level are described in this section. 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 20.1. IE: Interrupt Enable

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

SFR Address = 0xA8; SFR Page = All Pages; Bit-Addressable

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

SFR Definition 20.2. IP: Interrupt Priority

Bit	7	6	5	4	3	2	1	0
Name		PSPI0	PT2	PS0	PT1	PX1	PT0	PX0
Type	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

SFR Address = 0xB8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	Unused	Read = 1, Write = Don't Care.
6	PSPI0	**Serial Peripheral Interface (SPI0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the SPI0 interrupt.
5	PT2	**Timer 2 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 2 interrupt.
4	PS0	**UART0 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the UART0 interrupt.
3	PT1	**Timer 1 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 1 interrupt.
2	PX1	**External Interrupt 1 Priority Control LSB.** This bit sets the LSB of the priority field for the External Interrupt 1 interrupt.
1	PT0	**Timer 0 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 0 interrupt.
0	PX0	**External Interrupt 0 Priority Control LSB.** This bit sets the LSB of the priority field for the External Interrupt 0 interrupt.

SFR Definition 20.3. IPH: Interrupt Priority High

Bit	7	6	5	4	3	2	1	0
Name		PHSPI0	PHT2	PHS0	PHT1	PHX1	PHT0	PHX0
Type	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

SFR Address = 0x84; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	Unused	Read = 1, Write = Don't Care.
6	PHSPI0	**Serial Peripheral Interface (SPI0) Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the SPI0 interrupt.
5	PHT2	**Timer 2 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 2 interrupt.
4	PHS0	**UART0 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the UART0 interrupt.
3	PHT1	**Timer 1 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 1 interrupt.
2	PHX1	**External Interrupt 1 Priority Control MSB.** This bit sets the MSB of the priority field for the External Interrupt 1 interrupt.
1	PHT0	**Timer 0 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 0 interrupt.
0	PHX0	**External Interrupt 0 Priority Control MSB.** This bit sets the MSB of the priority field for the External Interrupt 0 interrupt.

SFR Definition 20.4. EIE1: Extended Interrupt Enable 1

Bit	7	6	5	4	3	2	1	0
Name	ET3		ECP0	EPCA0	EADC0	EWADC0	EMAT	ESMB0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE6; SFR Page = All Pages

Bit	Name	Function
7	ET3	**Enable Timer 3 Interrupt.** This bit sets the masking of the Timer 3 interrupt. 0: Disable Timer 3 interrupts. 1: Enable interrupt requests generated by the TF3L or TF3H flags.
6	Reserved	Reserved. Must Write 0.
5	ECP0	**Enable Comparator0 (CP0) Interrupt.** This bit sets the masking of the CP0 interrupt. 0: Disable CP0 interrupts. 1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags.
4	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.
3	EADC0	**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.
2	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 (AD0WINT).
1	EMAT	**Enable Port Match Interrupts.** This bit sets the masking of the Port Match Event interrupt. 0: Disable all Port Match interrupts. 1: Enable interrupt requests generated by a Port Match.
0	ESMB0	**Enable SMBus (SMB0) Interrupt.** This bit sets the masking of the SMB0 interrupt. 0: Disable all SMB0 interrupts. 1: Enable interrupt requests generated by SMB0.

SFR Definition 20.5. EIP1: Extended Interrupt Priority 1

Bit	7	6	5	4	3	2	1	0
Name	PT3		PCP0	PPCA0	PADC0	PWADC0	PMAT	PSMB0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF6; SFR Page = All Pages

Bit	Name	Function
7	PT3	**Timer 3 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 3 interrupt.
6	Reserved	Reserved. Must Write 0.
5	PCP0	**Comparator0 (CP0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the CP0 interrupt.
4	PPCA0	**Programmable Counter Array (PCA0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the PCA0 interrupt.
3	PADC0	**ADC0 Conversion Complete Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the ADC0 Conversion Complete interrupt.
2	PWADC0	**ADC0 Window Comparator Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the ADC0 Window interrupt.
1	PMAT	**Port Match Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Port Match Event interrupt.
0	PSMB0	**SMBus (SMB0) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the SMB0 interrupt.

SFR Definition 20.6. EIP1H: Extended Interrupt Priority 1 High

Bit	7	6	5	4	3	2	1	0
Name	PHT3		PHCP0	PHPCA0	PHADC0	PHWADC0	PHMAT	PHSMB0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x85; SFR Page = All Pages

Bit	Name	Function
7	PHT3	**Timer 3 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 3 interrupt.
6	Reserved	Reserved. Must Write 0.
5	PHCP0	**Comparator0 (CP0) Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the CP0 interrupt.
4	PHPCA0	**Programmable Counter Array (PCA0) Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the PCA0 interrupt.
3	PHADC0	**ADC0 Conversion Complete Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the ADC0 Conversion Complete interrupt.
2	PHWADC0	**ADC0 Window Comparator Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the ADC0 Window interrupt.
1	PHMAT	**Port Match Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Port Match Event interrupt.
0	PHSMB0	**SMBus (SMB0) Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the SMB0 interrupt.

SFR Definition 20.7. EIE2: Extended Interrupt Enable 2

Bit	7	6	5	4	3	2	1	0
Name					EPTS	ET5	ET4	ESMB1
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xAF; SFR Page = All Pages

Bit	Name	Function
7:4	Reserved	Must Write 0000b.
3	EPTS	**Enable Precision Temperature Sensor Interrupt.** This bit sets the masking of the Precision Temperature Sensor interrupt. 0: Disable Precision Temperature Sensor interrupts. 1: Enable interrupt requests generated by the Precision Temperature Sensor.
2	ET5	**Enable Timer 5 Interrupt.** This bit sets the masking of the Timer 5 interrupt. 0: Disable Timer 5 interrupts. 1: Enable interrupt requests generated by the TF5L or TF5H flags.
1	ET4	**Enable Timer 4 Interrupt.** This bit sets the masking of the Timer 4 interrupt. 0: Disable Timer 4 interrupts. 1: Enable interrupt requests generated by the TF4L or TF4H flags.
0	ESMB1	**Enable SMBus (SMB1) Interrupt.** This bit sets the masking of the SMB1 interrupt. 0: Disable all SMB1 interrupts. 1: Enable interrupt requests generated by SMB1.

SFR Definition 20.8. EIP2: Extended Interrupt Priority 2

Bit	7	6	5	4	3	2	1	0
Name					PPTS	PT5	PT4	PSMB1
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBF; SFR Page = All Pages

Bit	Name	Function
7:4	Reserved	Must Write 0000b.
3	PPTS	**Precision Temperature Sensor Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Precision Temperature Sensor interrupt.
2	PT5	**Timer 5 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 5 interrupt.
1	PT4	**Timer 4 Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the Timer 4 interrupt.
0	PSMB1	**SMBus (SMB1) Interrupt Priority Control LSB.** This bit sets the LSB of the priority field for the SMB1 interrupt.

SFR Definition 20.9. EIP2H: Extended Interrupt Priority 2 High

Bit	7	6	5	4	3	2	1	0
Name					PHPTS	PHT5	PHT4	PHSMB1
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x86; SFR Page = All Pages

Bit	Name	Function
7:4	Reserved	Must Write 0000b.
3	PHPTS	**Precision Temperature Sensor Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Precision Temperature Sensor interrupt.
2	PHT5	**Timer 5 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 5 interrupt.
1	PHT4	**Timer 4 Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the Timer 4 interrupt.
0	PHSMB1	**SMBus (SMB1) Interrupt Priority Control MSB.** This bit sets the MSB of the priority field for the SMB1 interrupt.

20.3. External Interrupts $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$

The $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL ($\overline{\text{INT0}}$ Polarity) and IN1PL ($\overline{\text{INT1}}$ Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (Section "31.1. Timer 0 and Timer 1" on page 245) select level or edge sensitive. The table below lists the possible configurations.

IT0	IN0PL	$\overline{\text{INT0}}$ 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	$\overline{\text{INT1}}$ 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

$\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ are assigned to Port pins as defined in the IT01CF register (see SFR Definition 20.10). Note that $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ Port pin assignments are independent of any Crossbar assignments. $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ 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 $\overline{\text{INT0}}$ and/or $\overline{\text{INT1}}$, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register XBR0 (see Section "27.3. Priority Crossbar Decoder" on page 178 for complete details on configuring the Crossbar).

IE0 (TCON.1) and IE1 (TCON.3) serve as the interrupt-pending flags for the $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupts, respectively. If an $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ 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.

SFR Definition 20.10. IT01CF: INT0/INT1 Configuration

Bit	7	6	5	4	3	2	1	0
Name	IN1PL	IN1SL[2:0]			IN0PL	IN0SL[2:0]		
Type	R/W	R/W			R/W	R/W		
Reset	0	0	0	0	0	0	0	1

SFR Address = 0xE4; SFR Page = All Pages

Bit	Name	Function
7	IN1PL	**$\overline{\text{INT1}}$ Polarity.** 0: /INT1 input is active low. 1: /INT1 input is active high.
6:4	IN1SL[2:0]	**$\overline{\text{INT1}}$ 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. 000: Select P0.0 001: Select P0.1 010: Select P0.2 011: Select P0.3 100: Select P0.4 101: Select P0.5 110: Select P0.6 111: Select P0.7
3	IN0PL	**$\overline{\text{INT0}}$ Polarity.** 0: $\overline{\text{INT0}}$ input is active low. 1: $\overline{\text{INT0}}$ input is active high.
2:0	IN0SL[2:0]	**$\overline{\text{INT0}}$ Port Pin Selection Bits.** These bits select which Port pin is assigned to $\overline{\text{INT0}}$. Note that this pin assignment is independent of the Crossbar; $\overline{\text{INT0}}$ 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. 000: Select P0.0 001: Select P0.1 010: Select P0.2 011: Select P0.3 100: Select P0.4 101: Select P0.5 110: Select P0.6 111: Select P0.7

21. Flash Memory

On-chip, re-programmable Flash memory is included for program code and non-volatile data storage. The Flash memory can be programmed in-system, a single byte at a time, through the C2 interface or by software using the MOVX instruction. Once cleared to logic 0, a Flash bit must be erased to set it back to logic 1. Flash bytes would typically be erased (set to 0xFF) before being reprogrammed. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operation is not required. Code execution is stalled during a Flash write/erase operation. Refer to Section “7. Electrical Characteristics” on page 32 for complete Flash memory electrical characteristics.

21.1. Programming The Flash Memory

The simplest means of programming the Flash memory is through the C2 interface using programming tools provided by Silicon Labs or a third party vendor. This is the only means for programming a non-initialized device. For details on the C2 commands to program Flash memory, see Section “33. C2 Interface” on page 297.

To ensure the integrity of Flash contents, it is strongly recommended that the on-chip V_{DD} Monitor be enabled in any system that includes code that writes and/or erases Flash memory from software. See Section 21.4 for more details.

21.1.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The Flash Lock and Key Register (FLKEY) must be written with the correct key codes, in sequence, before Flash operations may be performed. The key codes are: 0xA5, 0xF1. The timing does not matter, but the codes must be written in order. If the key codes are written out of order, or the wrong codes are written, Flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a Flash write or erase is attempted before the key codes have been written properly. The Flash lock resets after each write or erase; the key codes must be written again before a following Flash operation can be performed. The FLKEY register is detailed in SFR Definition 21.2.

21.1.2. Flash Erase Procedure

The Flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before writing to Flash memory using MOVX, Flash write operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1 (this directs the MOVX writes to target Flash memory); and (2) Writing the Flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software.

A write to Flash memory can clear bits to logic 0 but cannot set them; only an erase operation can set bits to logic 1 in Flash. **A byte location to be programmed should be erased before a new value is written.** The Flash memory is organized in 512-byte pages. The erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire 512-byte page, perform the following steps:

1. Disable interrupts (recommended).
2. Set thePSEE bit (register PSCTL).
3. Set the PSWE bit (register PSCTL).
4. Write the first key code to FLKEY: 0xA5.
5. Write the second key code to FLKEY: 0xF1.
6. Using the MOVX instruction, write a data byte to any location within the 512-byte page to be erased.
7. Clear the PSWE and PSEE bits.

21.1.3. Flash Write Procedure

Flash bytes are programmed by software with the following sequence:

1. Disable interrupts (recommended).
2. Erase the 512-byte Flash page containing the target location, as described in Section 21.1.2.
3. Set the PSWE bit (register PSCTL).
4. Clear the PSEE bit (register PSCTL).
5. Write the first key code to FLKEY: 0xA5.
6. Write the second key code to FLKEY: 0xF1.
7. Using the MOVX instruction, write a single data byte to the desired location within the 512-byte sector.
8. Clear the PSWE bit.

Steps 5–7 must be repeated for each byte to be written. After Flash writes are complete, PSWE should be cleared so that MOVX instructions do not target program memory.

21.2. Non-volatile Data Storage

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

21.3. Security Options

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

A Security Lock Byte located in Flash user space offers protection of the Flash program memory from access (reads, writes, or erases) by unprotected code or the C2 interface. See Section "17. Memory Organization" on page 93 for the location of the security byte. The Flash security mechanism allows the user to lock *n* 512-byte Flash pages, starting at page 0 (addresses 0x0000 to 0x01FF), where *n* is the 1's complement number represented by the Security Lock Byte. **Note that the page containing the Flash Security Lock Byte is unlocked when no other Flash pages are locked (all bits of the Lock Byte are '1') and locked when any other Flash pages are locked (any bit of the Lock Byte is '0').** An example is shown in Figure 21.1.

Security Lock Byte:	11111101b
1s Complement:	00000010b
Flash pages locked:	3 (First two Flash pages + Lock Byte Page)

Figure 21.1. Security Byte Decoding

The level of Flash security depends on the Flash access method. The three Flash access methods that can be restricted are reads, writes, and erases from the C2 debug interface, user firmware executing on unlocked pages, and user firmware executing on locked pages. Table 21.1 summarizes the Flash security features of the C8051F39x/37x devices.

Table 21.1. Flash Security Summary

Action	C2 Debug Interface	User Firmware executing from:	
		an unlocked page	a locked page
Read, Write or Erase unlocked pages (except page with Lock Byte)	Permitted	Permitted	Permitted
Read, Write or Erase locked pages (except page with Lock Byte)	Not Permitted	Flash Error Reset	Permitted
Read or Write page containing Lock Byte (if no pages are locked)	Permitted	Permitted	Permitted
Read or Write page containing Lock Byte (if any page is locked)	Not Permitted	Flash Error Reset	Permitted
Read contents of Lock Byte (if no pages are locked)	Permitted	Permitted	Permitted
Read contents of Lock Byte (if any page is locked)	Not Permitted	Flash Error Reset	Permitted
Erase page containing Lock Byte (if no pages are locked)	Permitted	Permitted	N/A

Table 21.1. Flash Security Summary

Erase page containing Lock Byte—Unlock all pages (if any page is locked)	C2 Device Erase Only	Flash Error Reset	Flash Error Reset
Lock additional pages (change 1s to 0s in the Lock Byte)	Not Permitted	Flash Error Reset	Flash Error Reset
Unlock individual pages (change 0s to 1s in the Lock Byte)	Not Permitted	Flash Error Reset	Flash Error Reset
Read, Write or Erase Reserved Area	Not Permitted	Flash Error Reset	Flash Error Reset

C2 Device Erase—Erases all Flash pages including the page containing the Lock Byte.

Flash Error Reset —Not permitted; Causes Flash Error Device Reset (FERROR bit in RSTSRC is '1' after reset).

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

21.4. Flash Write and Erase Guidelines

Any system which contains routines which write or erase Flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of V_{DD}, system clock frequency, or temperature. This accidental execution of Flash modifying code can result in alteration of Flash memory contents causing a system failure that is only recoverable by re-Flashing the code in the device. The following guidelines are recommended for any system which contains routines which write or erase Flash from code.

21.4.1. V_{DD} Maintenance and the V_{DD} Monitor

1. If the system power supply is subject to voltage or current "spikes," add sufficient transient protection devices to the power supply to ensure that the supply voltages listed in the Absolute Maximum Ratings table are not exceeded.
2. Make certain that the minimum V_{DD} rise time specification of 1 ms is met. If the system cannot meet this rise time specification, then add an external V_{DD} brownout circuit to the $\overline{RST}$ pin of the device that holds the device in reset until V_{DD} reaches 2.7 V and re-asserts $\overline{RST}$ if V_{DD} drops below 2.7 V.
3. Enable the on-chip V_{DD} monitor and enable the V_{DD} monitor as a reset source as early in code as possible. This should be the first set of instructions executed after the Reset Vector. For 'C'-based systems, this will involve modifying the startup code added by the C compiler. See your compiler documentation for more details. Make certain that there are no delays in software between enabling the V_{DD} monitor and enabling the V_{DD} monitor as a reset source. Code examples showing this can be found in “AN201: Writing to Flash from Firmware", available from the Silicon Laboratories web site.
4. As an added precaution, explicitly enable the V_{DD} monitor and enable the V_{DD} monitor as a reset source inside the functions that write and erase Flash memory. The V_{DD} monitor enable instructions should be placed just after the instruction to set PSWE to a 1, but before the Flash write or erase operation instruction.
5. Make certain that all writes to the RSTSRC (Reset Sources) register use direct assignment operators and explicitly DO NOT use the bit-wise operators (such as AND or OR). For example, "RSTSRC = 0x02" is correct. "RSTSRC |= 0x02" is incorrect.
6. Make certain that all writes to the RSTSRC register explicitly set the PORSF bit to a '1'. Areas to check are initialization code which enables other reset sources, such as the Missing Clock Detector or Comparator, for example, and instructions which force a Software Reset. A global search on "RSTSRC" can quickly verify this.

21.4.2. PSWE Maintenance

7. Reduce the number of places in code where the PSWE bit (b0 in PSCTL) is set to a '1'. There should be exactly one routine in code that sets PSWE to a '1' to write Flash bytes and one routine in code that sets PSWE and PSEE both to a '1' to erase Flash pages.
8. Minimize the number of variable accesses while PSWE is set to a '1'. Handle pointer address updates and loop variable maintenance outside the "PSWE = 1;... PSWE = 0;" area. Code examples showing this can be found in “AN201: Writing to Flash from Firmware", available from the Silicon Laboratories web site.
9. Disable interrupts prior to setting PSWE to a '1' and leave them disabled until after PSWE has been reset to 0. Any interrupts posted during the Flash write or erase operation will be serviced in priority order after the Flash operation has been completed and interrupts have been re-enabled by software.
10. Make certain that the Flash write and erase pointer variables are not located in XRAM. See your compiler documentation for instructions regarding how to explicitly locate variables in different memory areas.
11. Add address bounds checking to the routines that write or erase Flash memory to ensure that a routine called with an illegal address does not result in modification of the Flash.

21.4.3. System Clock

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

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

SFR Definition 21.1. PSCTL: Program Store R/W Control

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

SFR Address = 0x8F; SFR Page = All Pages

Bit	Name	Function
7:2	Unused	Read = 000000b, Write = don't care.
1	PSEE	**Program Store Erase Enable** Setting this bit (in combination with PSWE) allows an entire page of Flash program memory to be erased. If this bit is logic 1 and Flash writes are enabled (PSWE is logic 1), a write to Flash memory using the MOVX instruction will erase the entire page that contains the location addressed by the MOVX instruction. The value of the data byte written does not matter. 0: Flash program memory erasure disabled. 1: Flash program memory erasure enabled.
0	PSWE	**Program Store Write Enable** Setting this bit allows writing a byte of data to the Flash program memory using the MOVX write instruction. The Flash location should be erased before writing data. 0: Writes to Flash program memory disabled. 1: Writes to Flash program memory enabled; the MOVX write instruction targets Flash memory.

SFR Definition 21.2. FLKEY: Flash Lock and Key

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

SFR Address = 0xB7; SFR Page = All Pages

Bit	Name	Function
7:0	FLKEY[7:0]	**Flash Lock and Key Register.** Write: This register provides a lock and key function for Flash erasures and writes. Flash writes and erases are enabled by writing 0xA5 followed by 0xF1 to the FLKEY register. Flash writes and erases are automatically disabled after the next write or erase is complete. If any writes to FLKEY are performed incorrectly, or if a Flash write or erase operation is attempted while these operations are disabled, the Flash will be permanently locked from writes or erasures until the next device reset. If an application never writes to Flash, it can intentionally lock the Flash by writing a non-0xA5 value to FLKEY from software. Read: When read, bits 1–0 indicate the current Flash lock state. 00: Flash is write/erase locked. 01: The first key code has been written (0xA5). 10: Flash is unlocked (writes/erases allowed). 11: Flash writes/erases disabled until the next reset.

SFR Definition 21.3. FLSCL: Flash Scale

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

SFR Address = 0xB6; SFR Page = All Pages

Bit	Name	Function
7	FOSE	**Flash One-shot Enable** This bit enables the Flash read one-shot (recommended). If the Flash one-shot is disabled, the Flash sense amps are enabled for a full clock cycle during Flash reads, increasing the device power consumption. 0: Flash one-shot disabled. 1: Flash one-shot enabled.
6:5	Reserved	Must Write 00b.
4	FLRT	**Flash Read Timing** This bit should be programmed to the smallest allowed value, according to the system clock speed. 0: SYSCLK ≤ 25 MHz. 1: SYSCLK ≤ 50 MHz.
3:0	Reserved	Must Write 0000b.

22. EEPROM (C8051F37x)

The C8051F37x devices contain 512 bytes of byte-programmable EEPROM. The EEPROM is accessible by a 2-wire bus, available on EESDA and EESCL pins, which correspond to P2.2 and P2.3 respectively.

The EEPROM operates as a slave. The master can be either the SMBUS1 peripheral of the C8051F37x, internally connected to EESDA and EESCL, or an external master connected externally to the EESDA and EESCL pins.

22.1. EEPROM Communication Protocol

Communication between the master and the EEPROM consists of two types of operations: writes and reads. An overview of both operations is as follows:

- The master generates the clock on EESCL.
- Communication begins when the master generates a START condition by causing a falling edge in EESDA when EESCL is logic high.
- The master sends the slave address byte. See Section 22.1.1.
- The EEPROM acknowledges the receipt of the slave address byte generating an ACK. See Section 22.1.2.
- The master performs a read or write operation based on the setting of the R/W bit in the slave address byte. See Section 22.2 and Section 22.3.
- Throughout communication, the state of EESDA represents one bit of valid data when EESCL is logic high:
  - The master is permitted to change the state of EESDA when EESCL is logic high only to generate a START or STOP condition. Any changes in the EESDA line while the EESCL line is logic high will be interpreted as a START or STOP condition by the EEPROM.
  - The master or EEPROM is permitted to change the state of EESDA when EESCL is logic low.
- Communication terminates when the master generates a STOP condition by causing a rising edge in EESDA when EESCL is logic high.
- If necessary, the master can reset the communication with the EEPROM. See Section 22.1.4.

22.1.1. Slave Address Byte

The master begins a transmission by sending a START condition followed by the slave address byte (SAB).

Slave Address Byte (SAB) Definition

Bit	7	6	5	4	3	2	1	0
Name	SLA						ADDR MSB	R/W
Value	1	0	1	0	0	0	Varies	Varies

Bit	Name	Function
7:2	SLA	**Slave Address of EEPROM.** Always 101000b.
1	ADDR MSB	**Most Significant Addressing Bit.** This bit is concatenated to the 8-bit address counter to create a 9-bit address used by EEPROM read and write operations. 0: Address locations 0x000 to 0x0FF are targeted by the EEPROM operations. 1: Address locations 0x100 to 0x1FF are targeted by the EEPROM operations.
0	R/W	**EEPROM Read/Write Direction Bit.** Instructs the EEPROM to perform a read or write operation 0: Perform an EEPROM write operation 1: Perform an EEPROM read operation

Figure 22.1. Slave Address Byte Definition

22.1.2. Acknowledgement (ACK)

During an acknowledgement (ACK), the master or EEPROM forces the EESDA line to a logic low when EESCL is logic high.

22.1.3. Not-Acknowledgement (NACK)

During a not-acknowledgement (NACK), the master or EEPROM allows the EESDA line to be pulled up to a logic high when EESCL is logic high.

22.1.4. Reset

The EEPROM can be reset in case the SMBus communication is accidentally interrupted (e.g. power loss) or needs to be terminated mid-stream. The reset is initialized when the master device creates a START condition. To do this, it may be necessary for the master device to monitor EESDA up to nine times while cycling the EESCL signal. During this process, the master checks for a logic high on EESDA for each rising edge of EESCL.

22.2. Write Operation

Up to sixteen successive bytes may be written to the EEPROM within a single write operation. These writes are in 16-byte pages and must be page aligned. To write to the EEPROM:

1. The master sends the START condition and the slave address byte with the R/W bit cleared to 0.
2. The EEPROM generates an ACK.
3. The master sends the write address location (A[7:0]) to the EEPROM.
4. The EEPROM stores the address location in its address counter and generates an ACK.
5. The master transmits the data byte (D[7:0]) to the EEPROM.
6. The EEPROM increments four least significant bits of the address counter by 1 and generates an ACK. The four most significant bits of the address counter are unchanged.
7. The master can repeat Steps 5 and 6 up to fifteen more times.
8. The master generates a STOP condition.
9. The EEPROM begins its internal programming cycle.
10. The master transmits a START condition and slave address with the R/W bit cleared to 0:
    a. If the EEPROM does not generate an ACK, repeat Steps 8 and 9.
    b. If the EEPROM does generate and ACK, the EEPROM internal programming cycle is complete.

Note: The write address sets the upper 4 bits of the EEPROM write address pointer, and this value does not change. The lower four bits increment by 1. Because of this, the write cannot occur across 16-byte page boundaries in the same I2C write operation.

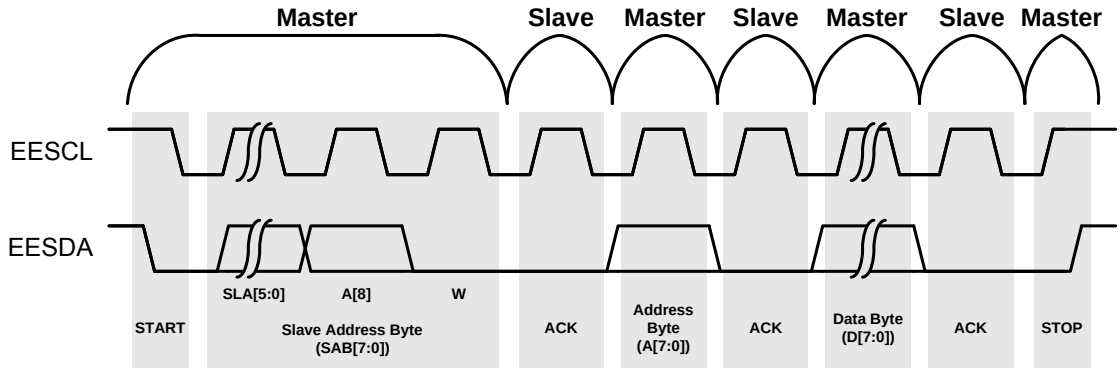


Figure 22.2. Write Operation (Single Byte)

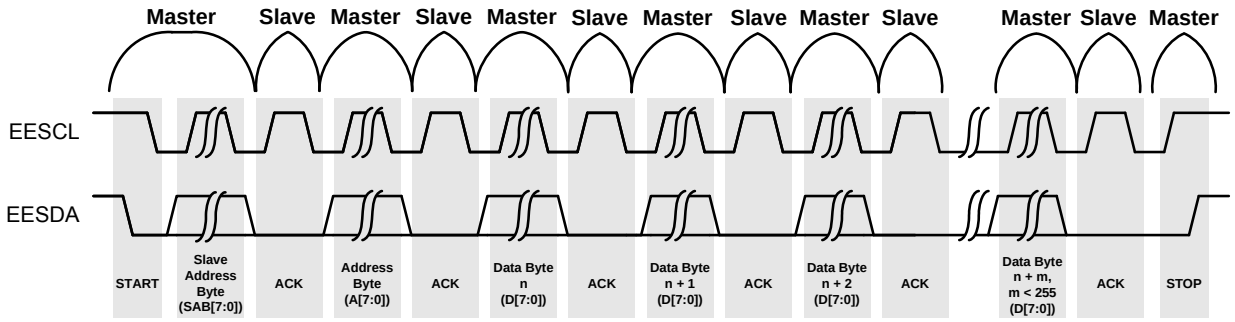


Figure 22.3. Write Operation (Multiple Bytes)

22.3. Read Operation

There are two operations to read the EEPROM: current address read and selective address read. Both read operations can read up to 256 bytes within a single read operation.

22.3.1. Current Address Read

A current address read accesses the data at the EEPROM internal address counter's current location.

The address counter in the EEPROM maintains the address of the last byte accessed, incremented by one. For example, if the previous operation was a read or write operation addressed to address location n, the internal address counter automatically increments to address n+1.

To perform a current address read operation:

1. The master sends the START condition and the slave address byte with the R/W bit set to 1.
2. The EEPROM generates an ACK and transmits the byte of data (D[7:0]) stored at the address specified by the address counter. This address will be the address from the last read or write operation incremented by one.
3. The EEPROM increments the internal address counter by one.
4. (Optional) To read additional bytes:
    a. The master generates an ACK.
    b. The EEPROM transmits the byte of data stored at the address specified by the address counter.
    c. The EEPROM increments the internal address counter by one.
    d. Repeat Step 4a through 4c until the master is done reading bytes.
5. The master generates a NACK.
6. The master generates a STOP condition.
7. The EEPROM terminates the transmission.

Note: If the previous operation targeted the last byte of the EEPROM, the EEPROM will transmit the data from address location 0x00 for a current address read operation.

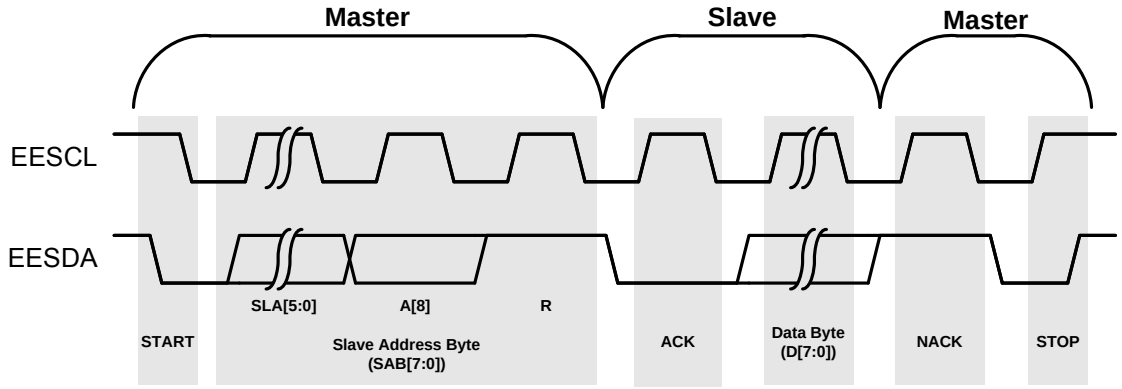


Figure 22.4. Current Address Read Operation (Single Byte)

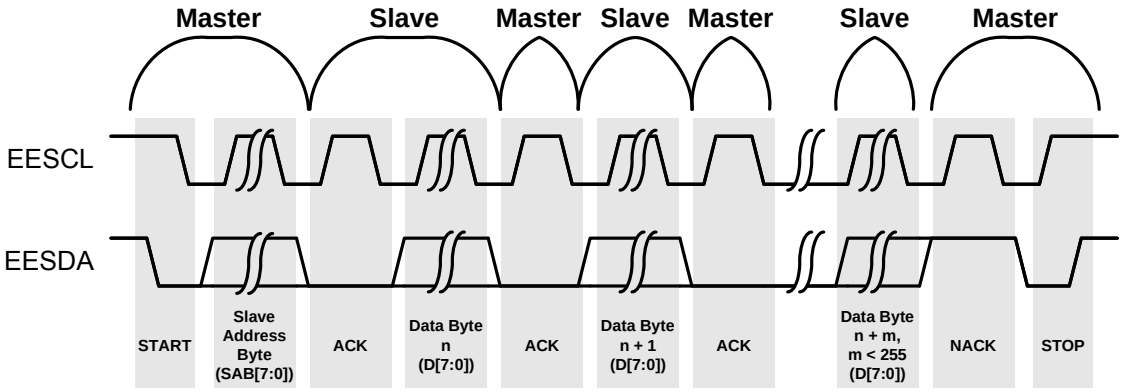


Figure 22.5. Current Address Read Operation (Multiple Bytes)

22.3.2. Selective Address Read

In a selective address read operation, the master selects the target memory location for the read operation.

To perform a selective address read:

1. The master sends the START condition and the slave address byte with the R/W bit set to 1.
2. The EEPROM generates an ACK.
3. The master sends the read memory address (A[7:0]) to the EEPROM.
4. The EEPROM stores the address in the address counter and generates an ACK.
5. The master again sends the slave address byte with the R/W bit set to 1.
6. The EEPROM generates an ACK.
7. The EEPROM sends the byte of data (D[7:0]) specified by the address counter.
8. The EEPROM increments the internal address counter by one.
9. (Optional) To read additional bytes:
    a. The master generates an ACK.
    b. The EEPROM sends the byte of data (D[7:0]) specified by the address counter.
    c. The EEPROM increments the internal address counter by one.
    d. Repeat Steps9a through 9c until the master reads all of the desired bytes.
10. The master generates a NACK.
11. The master generates a STOP condition.
12. The EEPROM terminates the transmission.

Note: If the selective read operation overflows the top of memory, the EEPROM address counter will wrap, and the EEPROM transmit the data from address location 0x00.

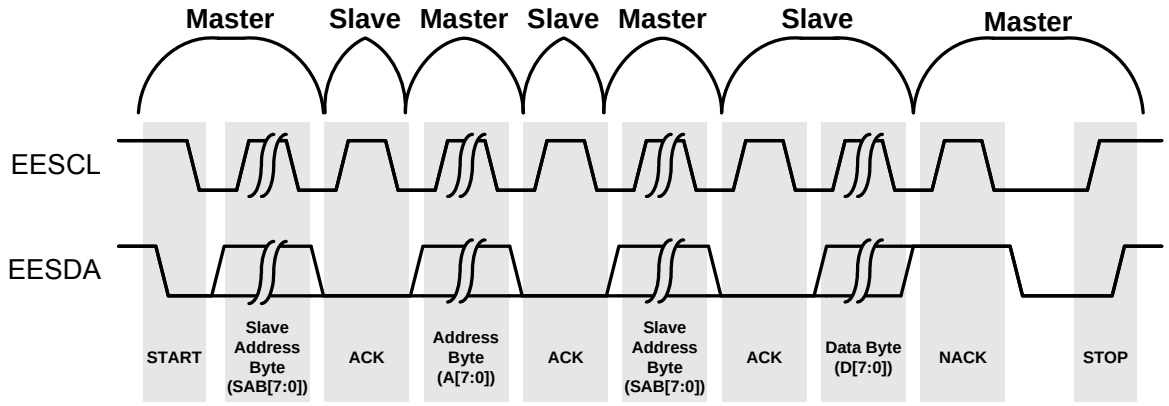


Figure 22.6. Selective Address Read (Single Byte)

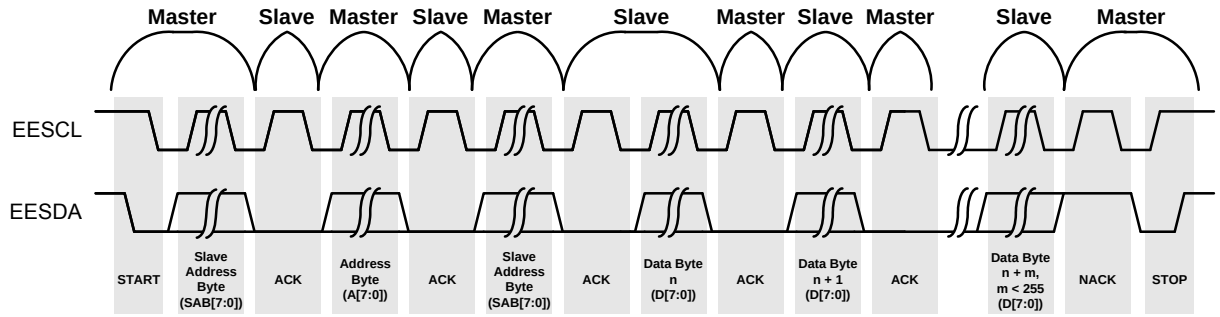


Figure 22.7. Selective Address Read (Multiple Bytes)

23. Cyclic Redundancy Check Unit (CRC0)

C8051F39x/37x devices include a cyclic redundancy check unit (CRC0) that can perform a CRC using a 16-bit polynomial. CRC0 accepts a stream of 8-bit data written to the CRC0IN register. CRC0 posts the 16-bit result to an internal register. The internal result register may be accessed indirectly using the CRC0PNT bits and CRC0DAT register, as shown in Figure 23.1. CRC0 also has a bit reverse register for quick data manipulation.

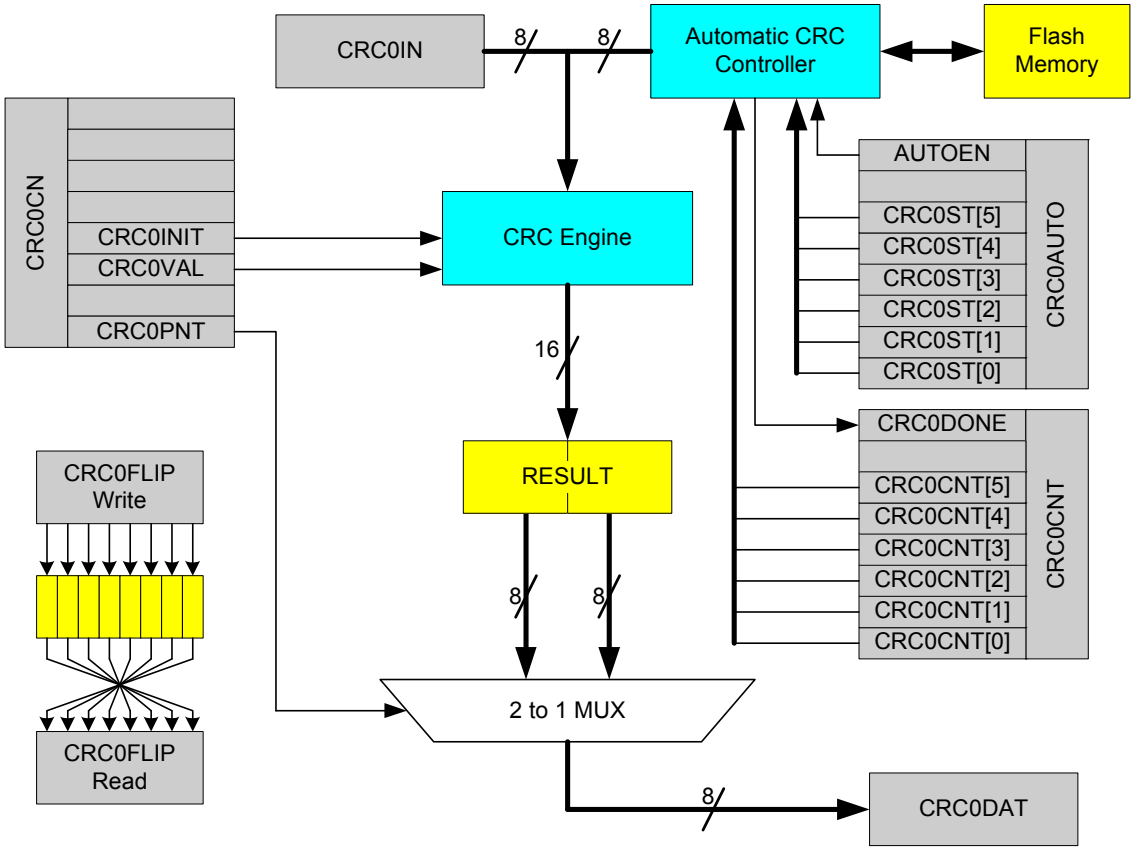


Figure 23.1. CRC0 Block Diagram

23.1. CRC Algorithm

The C8051F39x/37x CRC unit generates a CRC result equivalent to the following algorithm:

1. XOR the input with the most-significant bits of the current CRC result. If this is the first iteration of the CRC unit, the current CRC result will be the set initial value (0x00000000 or 0xFFFFFFFF).

2a. If the MSB of the CRC result is set, shift the CRC result and XOR the result with the selected polynomial.

2b. If the MSB of the CRC result is not set, shift the CRC result.

Repeat Steps 2a/2b for the number of input bits (8). The algorithm is also described in the following example.

The 16-bit C8051F39x/37x CRC algorithm can be described by the following code:

```
unsigned short UpdateCRC (unsigned short CRC_acc, unsigned char CRC_input)
{
   unsigned char i;                    // loop counter

   #define POLY 0x1021

   // Create the CRC "dividend" for polynomial arithmetic (binary arithmetic
   // with no carries)
   CRC_acc = CRC_acc ^ (CRC_input << 8);

   // "Divide" the poly into the dividend using CRC XOR subtraction
   // CRC_acc holds the "remainder" of each divide
   //
   // Only complete this division for 8 bits since input is 1 byte
   for (i = 0; i < 8; i++)
   {
      // Check if the MSB is set (if MSB is 1, then the POLY can "divide"
      // into the "dividend")
      if ((CRC_acc & 0x8000) == 0x8000)
      {
         // if so, shift the CRC value, and XOR "subtract" the poly
         CRC_acc = CRC_acc << 1;
         CRC_acc ^= POLY;
      }
      else
      {
         // if not, just shift the CRC value
         CRC_acc = CRC_acc << 1;
      }
   }

   // Return the final remainder (CRC value)
   return CRC_acc;
}
```

Table 23.1 lists several input values and the associated outputs using the 16-bit C8051F39x/37x CRC algorithm:

Table 23.1. Example 16-bit CRC Outputs

Input	Output
0x63	0xBD35
0x8C	0xB1F4
0x7D	0x4ECA
0xAA, 0xBB, 0xCC	0x6CF6
0x00, 0x00, 0xAA, 0xBB, 0xCC	0xB166

23.2. Preparing for a CRC Calculation

To prepare CRC0 for a CRC calculation, software should set the initial value of the result. The polynomial used for the CRC computation is 0x1021. The CRC0 result may be initialized to one of two values: 0x0000 or 0xFFFF. The following steps can be used to initialize CRC0.

1. Select the initial result value (Set CRC0VAL to 0 for 0x0000 or 1 for 0xFFFF).
2. Set the result to its initial value (Write 1 to CRC0INIT).

23.3. Performing a CRC Calculation

Once CRC0 is initialized, the input data stream is sequentially written to CRC0IN, one byte at a time. The CRC0 result is automatically updated after each byte is written. The CRC engine may also be configured to automatically perform a CRC on one or more 256 byte blocks read from Flash. The following steps can be used to automatically perform a CRC on Flash memory.

1. Prepare CRC0 for a CRC calculation as shown above.
2. Write the index of the starting page to CRC0AUTO.
3. Set the AUTOEN bit to 1 in CRC0AUTO.
4. Write the number of 256 byte blocks to perform in the CRC calculation to CRC0CNT.
5. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes. **See the note in SFR Definition 23.1. CRC0CN: CRC0 Control for more information on how to properly initiate a CRC calculation.**
6. Clear the AUTOEN bit in CRC0AUTO.
7. Read the CRC result using the procedure below.

23.4. Accessing the CRC0 Result

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

23.5. CRC0 Bit Reverse Feature

CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 23.2. Each byte of data written to CRC0FLIP is read back bit reversed. For example, if 0xC0 is written to CRC0FLIP, the data read back is 0x03. Bit reversal is a useful mathematical function used in algorithms such as the FFT.

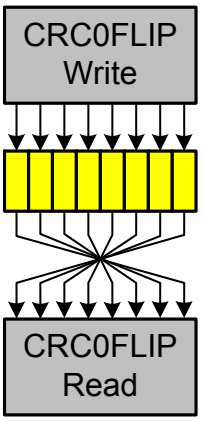


Figure 23.2. Bit Reverse Register

SFR Definition 23.1. CRC0CN: CRC0 Control

Bit	7	6	5	4	3	2	1	0
Name					CRC0INIT	CRC0VAL		CRC0PNT
Type	R	R	R	R	R/W	R/W	R	R/W
Reset	0	0	0	1	0	0	0	0

SFR Address = 0xDF; SFR Page = All Pages

Bit	Name	Function
7:4	Unused	Read = 0001b; Write = Don't Care.
3	CRC0INIT	**CRC0 Result Initialization Bit.** Writing a 1 to this bit initializes the entire CRC result based on CRC0VAL.
2	CRC0VAL	**CRC0 Set Value Initialization Bit.** This bit selects the set value of the CRC result. 0: CRC result is set to 0x00000000 on write of 1 to CRC0INIT. 1: CRC result is set to 0xFFFFFFFF on write of 1 to CRC0INIT.
1	Unused	Read = 0b; Write = Don't Care.
0	CRC0PNT	**CRC0 Result Pointer.** Specifies the byte of the CRC result to be read/written on the next access to CRC0DAT. The value of these bits will auto-increment upon each read or write. 0: CRC0DAT accesses bits 7–0 of the 16-bit CRC result. 1: CRC0DAT accesses bits 15–8 of the 16-bit CRC result.

Note: Upon initiation of an automatic CRC calculation, the three cycles following a write to CRC0CN that initiate a CRC operation must only contain instructions which execute in the same number of cycles as the number of bytes in the instruction. An example of such an instruction is a 3-byte MOV that targets the CRC0FLIP register. When programming in C, the dummy value written to CRC0FLIP should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.

SFR Definition 23.2. CRC0IN: CRC0 Data Input

Bit	7	6	5	4	3	2	1	0
Name	CRC0IN[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9C; SFR Page = All Pages

Bit	Name	Function
7:0	CRC0IN[7:0]	**CRC0 Data Input.** Each write to CRC0IN results in the written data being computed into the existing CRC result according to the CRC algorithm described in Section 23.1

SFR Definition 23.3. CRC0DAT: CRC0 Data Output

Bit	7	6	5	4	3	2	1	0
Name	CRC0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9E; SFR Page = All Pages

Bit	Name	Function
7:0	CRC0DAT[7:0]	**CRC0 Data Output.** Each read or write performed on CRC0DAT targets the CRC result bits pointed to by the CRC0 Result Pointer (CRC0PNT bits in CRC0CN).

SFR Definition 23.4. CRC0AUTO: CRC0 Automatic Control

Bit	7	6	5	4	3	2	1	0
Name	AUTOEN		CRC0ST[5:0]					
Type	R/W	R/W	R/W					
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xDD; SFR Page = All Pages

Bit	Name	Function
7	AUTOEN	**Automatic CRC Calculation Enable.** When AUTOEN is set to 1, any write to CRC0CN will initiate an automatic CRC starting at Flash sector CRC0ST and continuing for CRC0CNT sectors.
6	Reserved	Must write 0b.
5:0	CRC0ST[5:0]	**Automatic CRC Calculation Starting Block.** These bits specify the Flash block to start the automatic CRC calculation. The starting address of the first Flash block included in the automatic CRC calculation is CRC0ST x Block Size. **Note:** The block size is 256 bytes.

SFR Definition 23.5. CRC0CNT: CRC0 Automatic Flash Sector Count

Bit	7	6	5	4	3	2	1	0
Name	CRCDONE		CRC0CNT[5:0]					
Type	R	R/W	R/W					
Reset	1	0	0	0	0	0	0	0

SFR Address = 0xDE; SFR Page = All Pages

Bit	Name	Function
7	CRCDONE	**CRCDONE Automatic CRC Calculation Complete.** Set to 0 when a CRC calculation is in progress. Code execution is stopped during a CRC calculation; therefore, reads from firmware will always return 1.
6	Reserved	Must write 0b.
5:0	CRC0CNT[5:0]	**Automatic CRC Calculation Block Count.** These bits specify the number of Flash blocks to include in an automatic CRC calculation. The last address of the last Flash block included in the automatic CRC calculation is (CRC0ST+CRC0CNT) x Block Size - 1. **Notes:** **1.** The block size is 256 bytes.

SFR Definition 23.6. CRC0FLIP: CRC0 Bit Flip

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

SFR Address = 0x9A; SFR Page = All Pages

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

24. Reset Sources

Reset circuitry allows the controller to be easily placed in a predefined default condition. Upon entering this reset state, the following events 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 $\overline{RST}$ 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. The Watchdog Timer is enabled with the system clock divided by 12 as its clock source. Program execution begins at location 0x0000.

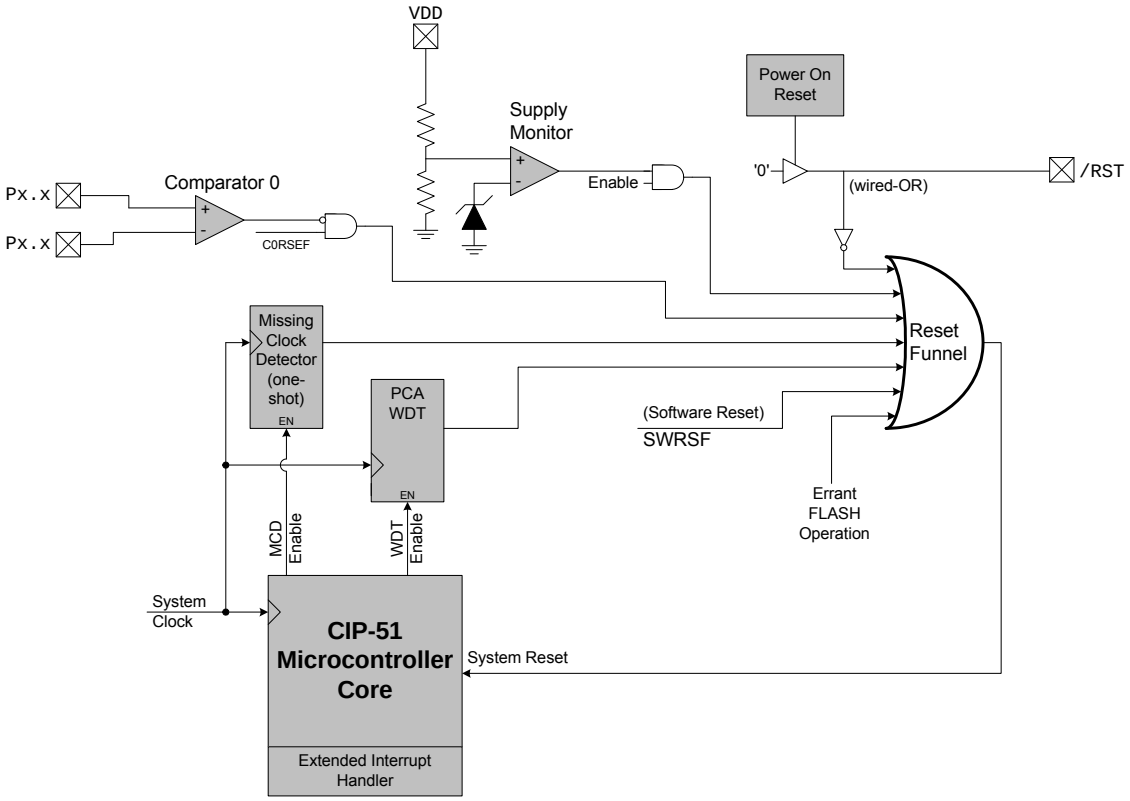


Figure 24.1. Reset Sources

24.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin is driven low until V_{DD} settles above V_{RST}. A 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 V_{RST}). Figure 24.2. plots the power-on and V_{DD} monitor reset timing. For ramp times less than 1 ms, the power-on reset delay ($T_{PORDelay}$) is typically less than 0.3 ms.

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 should be assumed to be undefined after a power-on reset. The V_{DD} monitor is enabled following a power-on reset.

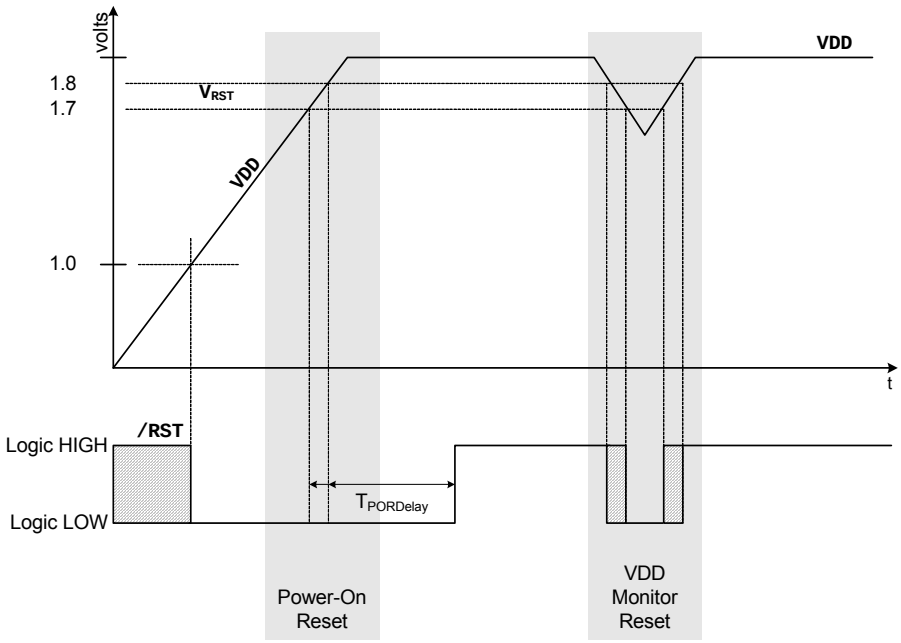


Figure 24.2. Power-On and V_{DD} Monitor Reset Timing

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

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 24.2). 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 enabled after power-on resets. Its defined state (enabled/disabled) is not altered by any other reset source. For example, if the V_{DD} monitor is disabled by code and a software reset is performed, the V_{DD} monitor will still be disabled after the reset.

Important Note: If the V_{DD} monitor is being turned on from a disabled state, it should be enabled before it is selected as a reset source. Selecting the V_{DD} monitor as a reset source before it is enabled and stabilized may cause a system reset. In some applications, this reset may be undesirable. If this is not desirable in the application, a delay should be introduced between enabling the monitor and selecting it as a reset source. The procedure for enabling the V_{DD} monitor and configuring it as a reset source from a disabled state is shown below:

1. Enable the V_{DD} monitor (VDMEN bit in VDM0CN = '1').
2. If necessary, wait for the V_{DD} monitor to stabilize.
3. Select the V_{DD} monitor as a reset source (PORSF bit in RSTSRC = '1').

See Figure 24.2 for V_{DD} monitor timing; note that the power-on-reset delay is not incurred after a V_{DD} monitor reset. See Section "7. Electrical Characteristics" on page 32 for complete electrical characteristics of the V_{DD} monitor.

SFR Definition 24.1. VDM0CN: V_{DD} Monitor Control

Bit	7	6	5	4	3	2	1	0
Name	VDMEN	VDDSTAT	VDMLVL					
Type	R/W	R	R/W	R	R	R	R	R
Reset	Varies	Varies	0	0	0	0	0	0

SFR Address = 0xFF; SFR Page = All Pages

Bit	Name	Function
7	VDMEN	**V_{DD} Monitor Enable.** This bit turns the V_{DD} monitor circuit on/off. The V_{DD} Monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC (SFR Definition 24.2). Selecting the V_{DD} monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the V_{DD} Monitor and selecting it as a reset source. 0: V_{DD} Monitor Disabled. 1: V_{DD} Monitor Enabled.
6	VDDSTAT	**V_{DD} Status.** This bit indicates the current power supply status (V_{DD} Monitor output). 0: V_{DD} is at or below the V_{DD} monitor threshold. 1: V_{DD} is above the V_{DD} monitor threshold.
5	VDMLVL	**VDD Monitor Level Select.** 0: VDD Monitor Threshold is set to VRST-LOW. 1: VDD Monitor Threshold is set to VRST-HIGH. This setting is required for any system with firmware that writes and/or erases Flash.
4:0	Unused	Read = 000000b; Write = Don't care.

24.3. External Reset

The external $\overline{\text{RST}}$ pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the $\overline{\text{RST}}$ pin generates a reset; an external pullup and/or decoupling of the $\overline{\text{RST}}$ pin may be necessary to avoid erroneous noise-induced resets. See Section “7. Electrical Characteristics” on page 32 for complete $\overline{\text{RST}}$ pin specifications. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

24.4. 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 $\overline{\text{RST}}$ pin is unaffected by this reset.

24.5. 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 $\overline{\text{RST}}$ pin is unaffected by this reset.

24.6. 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 “32.4. Watchdog Timer Mode” on page 286; 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 $\overline{\text{RST}}$ pin is unaffected by this reset.

24.7. Flash Error Reset

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

- A Flash write or erase is attempted above user code space. This occurs when PSWE is set to ‘1’ and a MOVX write operation targets an address above address 0x3DFF.
- A Flash read is attempted above user code space. This occurs when a MOVC operation targets an address above address 0x3DFF.
- A Program read is attempted above user code space. This occurs when user code attempts to branch to an address above 0x3DFF.
- A Flash read, write or erase attempt is restricted due to a Flash security setting (see Section “21.3. Security Options” on page 133).

The FERROR bit (RSTSRC.6) is set following a Flash error reset. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

24.8. 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 this reset.

SFR Definition 24.2. RSTSRC: Reset Source

Bit	7	6	5	4	3	2	1	0
Name		FERROR	C0RSEF	SWRSF	WDTRSF	MCDRSF	PORSF	PINRSF
Type	R	R	R/W	R/W	R	R/W	R/W	R
Reset	0	Varies	Varies	Varies	Varies	Varies	Varies	Varies

SFR Address = 0xEF; SFR Page = All Pages

Bit	Name	Description	Write	Read
7	Unused	**Unused.**	Don't care.	0
6	FERROR	**Flash Error Reset Flag.**	N/A	Set to '1' if Flash read/ write/erase error caused the last reset.
5	C0RSEF	**Comparator0 Reset Enable and Flag.**	Writing a '1' enables Comparator0 as a reset source (active-low).	Set to '1' if Comparator0 caused the last reset.
4	SWRSF	**Software Reset Force and Flag.**	Writing a '1' forces a system reset.	Set to '1' if last reset was caused by a write to SWRSF.
3	WDTRSF	**Watchdog Timer Reset Flag.**	N/A	Set to '1' if Watchdog Timer overflow caused the last reset.
2	MCDRSF	**Missing Clock Detector Enable and Flag.**	Writing a '1' enables the Missing Clock Detector. The MCD triggers a reset if a missing clock condition is detected.	Set to '1' if Missing Clock Detector timeout caused the last reset.
1	PORSF	**Power-On / V_{DD} Monitor Reset Flag, and V_{DD} monitor Reset Enable.**	Writing a '1' enables the V_{DD} monitor as a reset source. **Writing '1' to this bit before the V_{DD} monitor is enabled and stabilized may cause a system reset.**	Set to '1' anytime a power-on or V_{DD} monitor reset occurs. **When set to '1' all other RSTSRC flags are indeterminate.**
0	PINRSF	**HW Pin Reset Flag.**	N/A	Set to '1' if $\overline{\text{RST}}$ pin caused the last reset.

Note: Do not use read-modify-write operations on this register

25. Power Management Modes

The C8051F39x/37x devices have three software programmable power management modes: idle, stop, and Suspend. Idle mode and stop mode are part of the standard 8051 architecture, while suspend mode is an enhanced power-saving mode implemented by the high-speed oscillator.

Idle mode halts the CPU while leaving the peripherals and clocks active. In stop mode, the CPU is halted, all interrupts and timers (except the Missing Clock Detector) are inactive, and the internal oscillator is stopped (analog peripherals remain in their selected states; the external oscillator is not affected). Suspend mode is similar to Stop mode in that the internal oscillator and CPU are halted, but the device can wake on events such as a Port Mismatch, Comparator low output, or a Timer 3 overflow. 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 and suspend mode consume the least power because the majority of the device is shut down with no clocks active. SFR Definition 25.1 describes the Power Control Register (PCON) used to control the C8051F39x/37x's Stop and Idle power management modes. Suspend mode is controlled by the SUSPEND bit in the OSCICN register (SFR Definition 26.3).

Although the C8051F39x/37x has idle, stop, and suspend modes available, more control over the device power can be achieved 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 oscillators lowers power consumption considerably, at the expense of reduced functionality.

25.1. Idle Mode

Setting the Idle Mode Select bit (PCON.0) causes the hardware 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.

Note: 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. Therefore, instructions that set the IDLE bit should be followed by an instruction that has two 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 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 "24.6. PCA Watchdog Timer Reset" on page 159 for more information on the use and configuration of the WDT.

25.2. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the controller core to enter Stop mode as soon as the instruction that sets the bit completes execution. Before entering stop mode, the system clock must be sourced by the internal high-frequency oscillator. 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 device 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 REG01CN should be set to 1 prior to setting the STOP bit (see SFR Definition 25.1). 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.

25.3. Suspend Mode

Setting the SUSPEND bit (OSCICN.5) causes the hardware to halt the CPU and the high-frequency internal oscillator, and go into suspend mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. Most digital peripherals are not active in suspend mode. The exception to this is the Port Match feature and Timer 3, when it is run from an external oscillator source or the internal low-frequency oscillator.

Suspend mode can be terminated by four types of events, a port match (described in Section “27.5. Port Match” on page 183), a Timer 3 overflow (described in Section “31.3. Timer 3” on page 259), a Comparator low output (if enabled), or a device reset event. Note that in order to run Timer 3 in suspend mode, the timer must be configured to clock from either the external clock source or the internal low-frequency oscillator source. When suspend mode is terminated, the device will continue execution on the instruction following the one that set the SUSPEND bit. If the wake event (port match or Timer 3 overflow) was configured to generate an interrupt, the interrupt will be serviced upon waking the device. If suspend mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

SFR Definition 25.1. PCON: Power Control

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

SFR Address = 0x87; SFR Page = All Pages

Bit	Name	Function
7:2	GF[5:0]	**General Purpose Flags 5–0.** These are general purpose flags for use under software control.
1	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 (internal oscillator stopped).
0	IDLE	**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.)

26. Oscillators and Clock Selection

C8051F39x/37x devices include a programmable internal high-frequency oscillator, a programmable internal low-frequency oscillator, an internal low-power oscillator, and an external oscillator drive circuit. The internal high-frequency oscillator can be enabled/disabled and calibrated using the OSCICN and OSCICL registers, as shown in Figure 26.1. The internal low-frequency oscillator can be enabled/disabled and calibrated using the OSCLCN register. The internal low-power oscillator is automatically enabled and disabled when selected and deselected as the system clock. The system clock can be sourced by the external oscillator circuit or any internal oscillator. The internal high-frequency and low-frequency oscillators offer a selectable post-scaling feature.

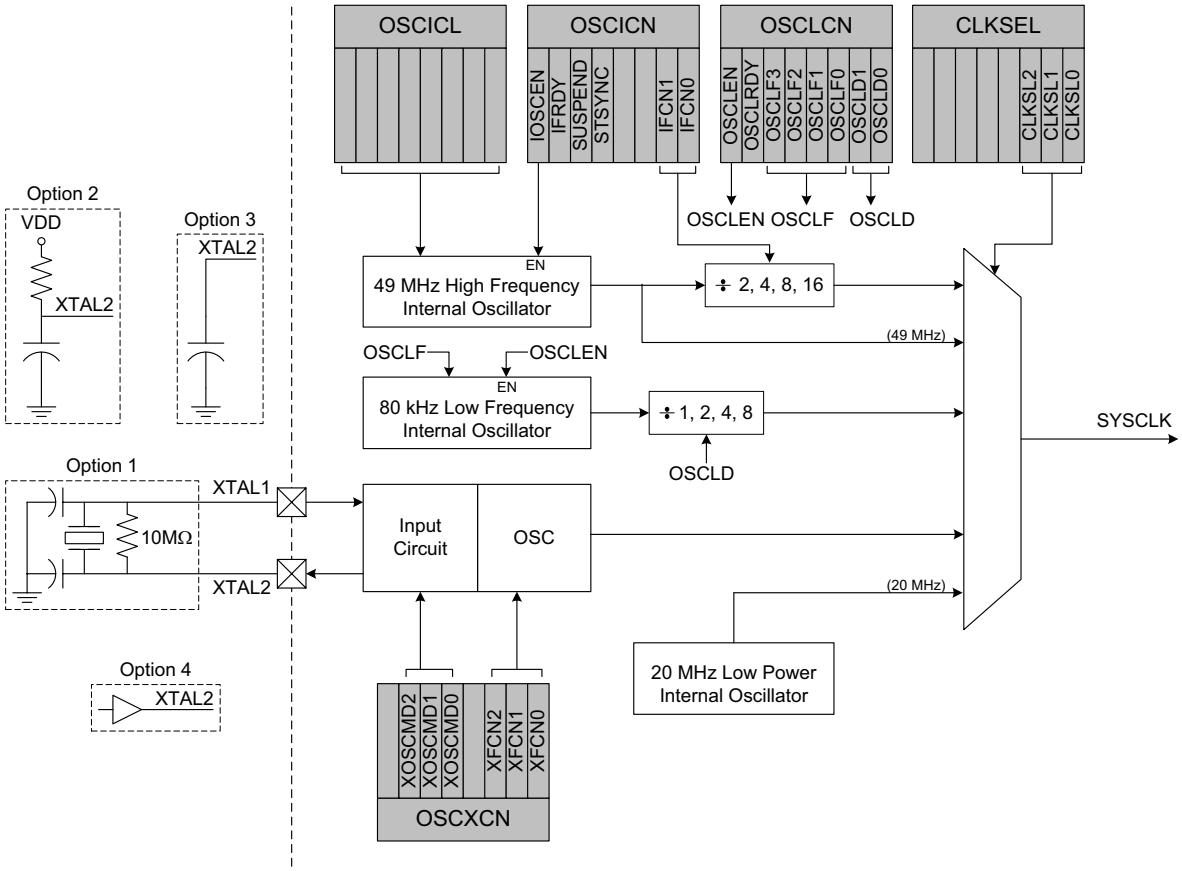


Figure 26.1. Oscillator Options

26.1. System Clock Selection

The CLKSL[2:0] bits in register CLKSEL select which oscillator source is used as the system clock. CLKSL[2:0] must be set to 001b for the system clock to run from the external oscillator; however the external oscillator may still clock certain peripherals (timers, PCA) when the internal oscillator is selected as the system clock. The system clock may be switched on-the-fly between any of the oscillator sources so long as the selected clock source is enabled and has settled.

The internal high-frequency and low-frequency oscillators require little start-up time and may be selected as the system clock immediately following the register write which enables the oscillator. The external RC and C modes also typically require no startup time.

External crystals and ceramic resonators however, typically require a start-up time before they are settled and ready for use. The Crystal Valid Flag (XTLVLD in register OSCXCN) is set to '1' by hardware when the external crystal or ceramic resonator is settled. **In crystal mode, to avoid reading a false XTLVLD, software should delay at least 1 ms between enabling the external oscillator and checking XTLVLD.**

SFR Definition 26.1. CLKSEL: Clock Select

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

SFR Address = 0xA9; SFR Page = All Pages

Bit	Name	Function
7:3	Unused	Read = 00000b; Write = Don't Care
2:0	CLKSL[2:0]	**System Clock Source Select Bits.** 000: SYSCLK derived from the Internal High-Frequency Oscillator and scaled per the IFCN bits in register OSCICN. 001: SYSCLK derived from the External Oscillator circuit.* 010: SYSCLK derived from the Internal Low-Frequency Oscillator and scaled per the OSCLD bits in register OSCLCN. 011: SYSCLK derived directly from the Internal High-Frequency Oscillator.* 100: Reserved. 101: SYSCLK derived from the Internal Low-Power Oscillator. 110: Reserved. 111: Reserved.
Note: Prior to switching to a system clock frequency > 25 MHz, ensure that the FLRT bit in SFR Definition 21.3. FLSCL: Flash Scale has been set appropriately to ensure proper flash read timing.		

26.2. Programmable Internal High-Frequency (H-F) Oscillator

All C8051F39x/37x devices include a programmable internal high-frequency oscillator that defaults as the system clock after a system reset. The internal oscillator period can be adjusted via the OSCICL register as defined by SFR Definition 26.2.

On C8051F39x/37x devices, OSCICL is factory calibrated to obtain a 49 MHz base frequency.

The system clock may be derived directly from the programmed internal oscillator, or from a divided version, with factors of 2, 4, 8, or 16, as defined by the IFCN bits in register OSCICN. The divide value defaults to 16 following a reset.

26.2.1. Internal Oscillator Suspend Mode

When software writes a logic 1 to SUSPEND (OSCICN.5), the internal oscillator is suspended. If the system clock is derived from the internal oscillator, the input clock to the peripheral or CIP-51 will be stopped until one of the following events occur:

- Port 0 Match Event.
- Port 1 Match Event.
- Comparator 0 enabled and output is logic 0.
- Timer3 Overflow Event.

When one of the oscillator awakening events occur, the internal oscillator, CIP-51, and affected peripherals resume normal operation, regardless of whether the event also causes an interrupt. The CPU resumes execution at the instruction following the write to SUSPEND.

SFR Definition 26.2. OSCICL: Internal H-F Oscillator Calibration

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

SFR Address = 0xB3; SFR Page = All Pages

Bit	Name	Function
7	Unused	Unused. Read = 0; Write = Don't Care
6:0	OSCICL[6:0]	**Internal Oscillator Calibration Bits.** These bits determine the internal oscillator period. When set to 0000000b, the H-F oscillator operates at its fastest setting. When set to 1111111b, the H-F oscillator operates at its slowest setting. The reset value is factory calibrated to generate an internal oscillator frequency of 49 MHz.

SFR Definition 26.3. OSCICN: Internal H-F Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	IOSCEN	IFRDY	SUSPEND	STSYNC			IFCN[1:0]	
Type	R/W	R	R/W	R	R	R	R/W	
Reset	1	1	0	0	0	0	0	0

SFR Address = 0xB2; SFR Page = All Pages

Bit	Name	Function
7	IOSCEN	**Internal H-F Oscillator Enable Bit.** 0: Internal H-F Oscillator Disabled. 1: Internal H-F Oscillator Enabled.
6	IFRDY	**Internal H-F Oscillator Frequency Ready Flag.** 0: Internal H-F Oscillator is not running at programmed frequency. 1: Internal H-F Oscillator is running at programmed frequency.
5	SUSPEND	**Internal Oscillator Suspend Enable Bit.** Setting this bit to logic 1 places the internal oscillator in SUSPEND mode. The internal oscillator resumes operation when one of the SUSPEND mode awakening events occurs.
4	STSYNC	**Suspend Timer Synchronization Bit.** This bit is used to indicate when it is safe to read and write the registers associated with the suspend wake-up timer. If a suspend wake-up source other than the timer has brought the oscillator out of suspend mode, it may take up to three timer clocks before the timer can be read or written. When STSYNC reads '1', reads and writes of the timer register should not be performed. When STSYNC reads '0', it is safe to read and write the timer registers.
3:2	Unused	Unused. Read = 00b; Write = Don't Care
1:0	IFCN[1:0]	**Internal H-F Oscillator Frequency Divider Control Bits.** These bits control the oscillator clock divider when the clock divider is selected as the system clock source. 00: Internal H-F divide ratio = 16. 01: Internal H-F divide ratio = 8. 10: Internal H-F divide ratio = 4. 11: Internal H-F divide ratio = 2.

26.3. Programmable Internal Low-Frequency (L-F) Oscillator

All C8051F39x/37x devices include a programmable low-frequency internal oscillator, which is calibrated to a nominal frequency of 80 kHz. The low-frequency oscillator circuit includes a divider that can be changed to divide the clock by 1, 2, 4, or 8, using the OSCLD bits in the OSCLCN register (see SFR Definition 26.4). Additionally, the OSCLF[3:0] bits can be used to adjust the oscillator's output frequency.

26.3.1. Calibrating the Internal L-F Oscillator

Timers 2 and 3 include capture functions that can be used to capture the oscillator frequency, when running from a known time base. When either Timer 2 or Timer 3 is configured for L-F Oscillator Capture Mode, a falling edge (Timer 2) or rising edge (Timer 3) of the low-frequency oscillator's output will cause a capture event on the corresponding timer. As a capture event occurs, the current timer value (TMRnH:TMRnL) is copied into the timer reload registers (TMRnRLH:TMRnRLL). By recording the difference between two successive timer capture values, the low-frequency oscillator's period can be calculated. The OSCLF bits can then be adjusted to produce the desired oscillator frequency.

SFR Definition 26.4. OSCLCN: Internal L-F Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	OSCLEN	OSCLRDY	OSCLF[3:0]				OSCLD[1:0]	
Type	R/W	R	R.W				R/W	
Reset	0	0	Varies	Varies	Varies	Varies	0	0

SFR Address = 0xE3; SFR Page = All Pages

Bit	Name	Function
7	OSCLEN	**Internal L-F Oscillator Enable.** 0: Internal L-F Oscillator Disabled. 1: Internal L-F Oscillator Enabled.
6	OSCLRDY	**Internal L-F Oscillator Ready.** 0: Internal L-F Oscillator frequency not stabilized. 1: Internal L-F Oscillator frequency stabilized. **Note:** OSCLRDY is only set back to 0 in the event of a device reset or a change to the OSCLD[1:0] bits.
5:2	OSCLF[3:0]	**Internal L-F Oscillator Frequency Control Bits.** Fine-tune control bits for the Internal L-F oscillator frequency. When set to 0000b, the L-F oscillator operates at its fastest setting. When set to 1111b, the L-F oscillator operates at its slowest setting. The OSCLF bits should only be changed by firmware when the L-F oscillator is disabled (OSCLEN = 0).
1:0	OSCLD[1:0]	**Internal L-F Oscillator Divider Select.** 00: Divide by 8 selected. 01: Divide by 4 selected. 10: Divide by 2 selected. 11: Divide by 1 selected.

26.4. Internal Low-Power Oscillator

All C8051F39x/37x devices include a low-power internal oscillator with a nominal frequency of 20 MHz. The low-power oscillator is automatically enabled when selected as the system clock and disabled when not in use. See Table 7.9, “Internal Low-Power Oscillator Electrical Characteristics,” on page 38 for complete oscillator specifications.

26.5. External Oscillator Drive Circuit

The external oscillator circuit may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. Figure 26.1 shows a block diagram of the four external oscillator options. The external oscillator is enabled and configured using the OSCXCN register (see SFR Definition 26.5).

Important Note on External Oscillator Usage: Port pins must be configured when using the external oscillator circuit. When the external oscillator drive circuit is enabled in crystal/resonator mode, Port pins P0.2 and P0.3 are used as XTAL1 and XTAL2, respectively. When the external oscillator drive circuit is enabled in capacitor, RC, or CMOS clock mode, Port pin P0.3 is used as XTAL2. The Port I/O Crossbar should be configured to skip the Port pin used by the oscillator circuit; see Section “27.3. Priority Crossbar Decoder” on page 178 for Crossbar configuration. Additionally, when using the external oscillator circuit in crystal/resonator, capacitor, or RC mode, the associated Port pins should be configured as **analog inputs**. In CMOS clock mode, the associated pin should be configured as a **digital input**. See Section “27.4. Port I/O Initialization” on page 180 for details on Port input mode selection.

The external oscillator output may be selected as the system clock or used to clock some of the digital peripherals (e.g. Timers, PCA, etc.). See the data sheet chapters for each digital peripheral for details. See Section “7. Electrical Characteristics” on page 32 for complete oscillator specifications.

26.5.1. External Crystal Mode

If a crystal or ceramic resonator is used as the external oscillator, the crystal/resonator and a 10 MΩ resistor must be wired across the XTAL1 and XTAL2 pins as shown in Figure 26.1, “Crystal Mode”. Appropriate loading capacitors should be added to XTAL1 and XTAL2, and both pins should be configured for analog I/O with the digital output drivers disabled.

The capacitors shown in the external crystal configuration provide the load capacitance required by the crystal for correct oscillation. These capacitors are “in series” as seen by the crystal and “in parallel” with the stray capacitance of the XTAL1 and XTAL2 pins.

Note: The recommended load capacitance depends upon the crystal and the manufacturer. Refer to the crystal data sheet when completing these calculations.

The equation for determining the load capacitance for two capacitors is

$$C_L = \frac{C_A \times C_B}{C_A + C_B} + C_S$$

Where:
C_A and C_B are the capacitors connected to the crystal leads.
C_S is the total stray capacitance of the PCB.
The stray capacitance for a typical layout where the crystal is as close as possible to the pins is 2-5 pF per pin.

If C_A and C_B are the same (C), then the equation becomes

$$C_L = \frac{C}{2} + C_S$$

For example, a tuning-fork crystal of 32 kHz with a recommended load capacitance of 12.5 pF should use the configuration shown in Figure 26.1, Option 1. With a stray capacitance of 3 pF per pin (6 pF total), the 13 pF capacitors yield an equivalent capacitance of 12.5 pF across the crystal, as shown in Figure 26.2.

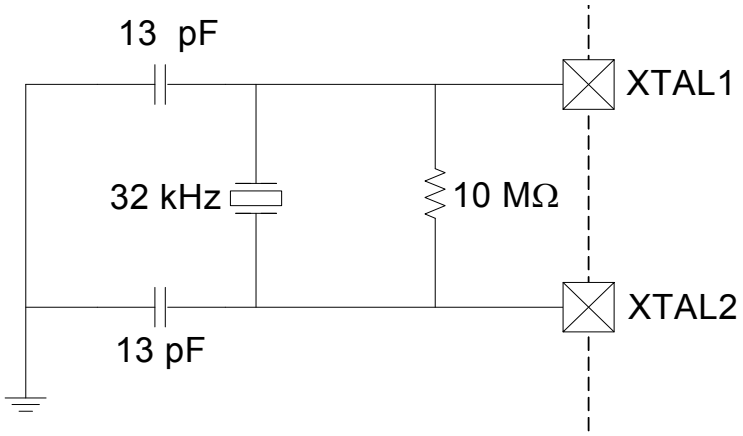


Figure 26.2. External Crystal Example

Important Note on External Crystals: Crystal oscillator circuits are quite sensitive to PCB layout. The crystal should be placed as close as possible to the XTAL pins on the device. The traces should be as short as possible and shielded with ground plane from any other traces which could introduce noise or interference.

When using an external crystal, the external oscillator drive circuit must be configured by software for *Crystal Oscillator Mode* or *Crystal Oscillator Mode with divide by 2 stage*. The divide by 2 stage ensures that the clock derived from the external oscillator has a duty cycle of 50%. The External Oscillator Frequency Control value (XFCN) must also be specified based on the crystal frequency (see SFR Definition 26.5).

When the crystal oscillator is first enabled, the external oscillator valid detector allows software to determine when the external system clock is valid and running. Switching to the external oscillator before the crystal oscillator has stabilized can result in unpredictable behavior. The recommended procedure for starting the crystal is:

1. Configure XTAL1 and XTAL2 for analog I/O.
2. Disable the XTAL1 and XTAL2 digital output drivers by writing 1s to the appropriate bits in the Port Latch register.
3. Configure and enable the external oscillator.
4. Wait at least 1 ms.
5. Poll for XTLVLD $\geq$ 1.
6. Switch the system clock to the external oscillator.

26.5.2. 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 26.1, “RC Mode”. 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, according to Equation , where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and R = the pull-up resistor value in kΩ.

$$f = 1.23 \times 10^3 / (R \times C)$$

Equation 26.1. RC Mode Oscillator Frequency

For example: If the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

f = 1.23(10^3) / RC = 1.23 (10^3) / [246 x 50] = 0.1 MHz = 100 kHz

Referring to the table in SFR Definition 26.5, the required XFCN setting is 010b.

26.5.3. External Capacitor Example

If a capacitor is used as an external oscillator for the MCU, the circuit should be configured as shown in Figure 26.1, “C Mode”. 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 according to Equation , where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and V_{DD} = the MCU power supply in Volts.

$$f = (KF) / (C \times V_{DD})$$

Equation 26.2. C Mode Oscillator Frequency

For example: Assume V_{DD} = 3.0 V and f = 150 kHz:

f = KF / (C x VDD)
0.150 MHz = KF / (C x 3.0)

Since the frequency of roughly 150 kHz is desired, select the K Factor from the table in SFR Definition 26.5 (OSCXCN) as KF = 22:

0.150 MHz = 22 / (C x 3.0)
C x 3.0 = 22 / 0.150 MHz
C = 146.6 / 3.0 pF = 48.8 pF

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

SFR Definition 26.5. OSCXCN: External Oscillator Control

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

SFR Address = 0xB1; SFR Page = All Pages

Bit	Name	Function
7	XCLKVLD	**External Oscillator Valid Flag.** Provides External Oscillator status and is valid at all times for all modes of operation except External CMOS Clock Mode and External CMOS Clock Mode with divide by 2. In these modes, XCLKVLD always returns 0. 0: External Oscillator is unused or not yet stable. 1: External Oscillator is running and stable.
6:4	XOSCMD[2:0]	**External Oscillator Mode Select.** 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. 110: Crystal Oscillator Mode. 111: Crystal Oscillator Mode with divide-by-2 stage.
3	Unused	Read = 0; Write = don't care
2:0	XFCN[2:0]	**External Oscillator Frequency Control Bits.** Set according to the desired frequency for RC mode. Set according to the desired K Factor for C mode.

XFCN	Crystal Mode	RC Mode	C Mode
000	f ≤ 20 kHz	f ≤ 25 kHz	K Factor = 0.87
001	20 kHz < f ≤ 58 kHz	25 kHz < f ≤ 50 kHz	K Factor = 2.6
010	58 kHz < f ≤ 155 kHz	50 kHz < f ≤ 100 kHz	K Factor = 7.8
011	155 kHz < f ≤ 415 kHz	100 kHz < f ≤ 200 kHz	K Factor = 22
100	415 kHz < f ≤ 1.1 MHz	200 kHz < f ≤ 400 kHz	K Factor = 66
101	1.1 MHz < f ≤ 3.1 MHz	400 kHz < f ≤ 800 kHz	K Factor = 189
110	3.1 MHz < f ≤ 8.2 MHz	800 kHz < f ≤ 1.6 MHz	K Factor = 741
111	8.2 MHz < f ≤ 25 MHz	1.6 MHz < f ≤ 3.2 MHz	K Factor = 2222

27. Port Input/Output

Digital and analog resources are available through 17 (C8051F392/3/6/7/8/9) or 21 (C8051F390/1/4/5 and C8051F37x) I/O pins. Port pins P0.0-P2.3 can be defined as general-purpose I/O (GPIO), assigned to one of the internal digital resources, or assigned to an analog function as shown in Figure 27.3. Port pin P2.4 on the C8051F390/1/4/5 and C8051F37x and P2.0 on the C8051F392/3/6/7/8/9 can be used as GPIO and are shared with the C2 Interface Data signal (C2D). 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. Note that 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 27.3 and Figure 27.4). The registers XBR0 and XBR1, defined in SFR Definition 27.1 and SFR Definition 27.2, are used to select internal digital functions.

The Port I/O cells are configured as either push-pull or open-drain in the Port Output Mode registers (PnMDOUT, where n = 0,1). Complete Electrical Specifications for Port I/O are given in Section "7. Electrical Characteristics" on page 32.

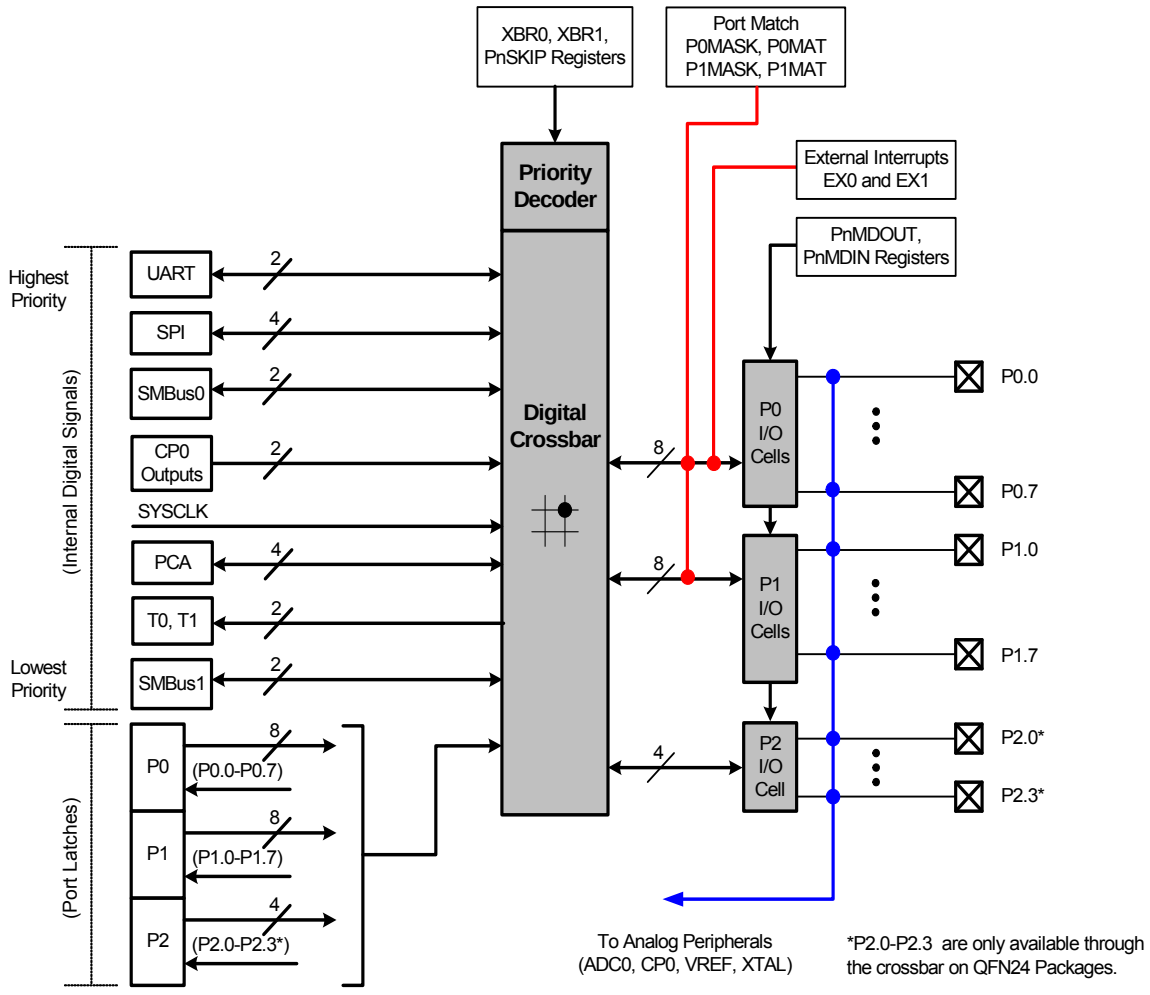


Figure 27.1. Port I/O Functional Block Diagram

27.1. Port I/O Modes of Operation

Port pins P0.0 - P2.3 use the Port I/O cell shown in Figure 27.2. Each Port I/O cell can be configured by software for analog I/O or digital I/O using the PnMDIN registers. On reset, all Port I/O cells default to a high impedance state with weak pull-ups enabled. Until the crossbar is enabled (XBARE = ‘1’), both the high and low port I/O drive circuits are explicitly disabled on all crossbar pins.

27.1.1. Port Pins Configured for Analog I/O

Any pins to be used as Comparator or ADC input, external oscillator input/output, VREF, or IDAC output should be configured for analog I/O (PnMDIN.n = ‘1’). When a pin is configured for analog I/O, its weak pullup, digital driver, and digital receiver are disabled. Port pins configured for analog I/O will always read back a value of ‘0’.

Configuring pins as analog I/O saves power and isolates the Port pin from digital interference. Port pins configured as digital I/O may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

27.1.2. Port Pins Configured For Digital I/O

Any pins to be used by digital peripherals (UART, SPI, SMBus, etc.), external event trigger functions, or as GPIO should be configured as digital I/O (PnMDIN.n = ‘1’). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PnMDOUT registers.

Push-pull outputs (PnMDOUT.n = ‘1’) drive the Port pad to the VDD or GND supply rails based on the output logic value of the Port pin. Open-drain outputs have the high side driver disabled; therefore, they only drive the Port pad to GND when the output logic value is ‘0’ and become high impedance inputs (both high low drivers turned off) when the output logic value is ‘1’.

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the Port pad to the VDD supply voltage to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven to GND to minimize power consumption, and they may be globally disabled by setting WEAKPUD to ‘1’. The user should ensure that digital I/O are always internally or externally pulled or driven to a valid logic state to minimize power consumption. Port pins configured for digital I/O always read back the logic state of the Port pad, regardless of the output logic value of the Port pin.

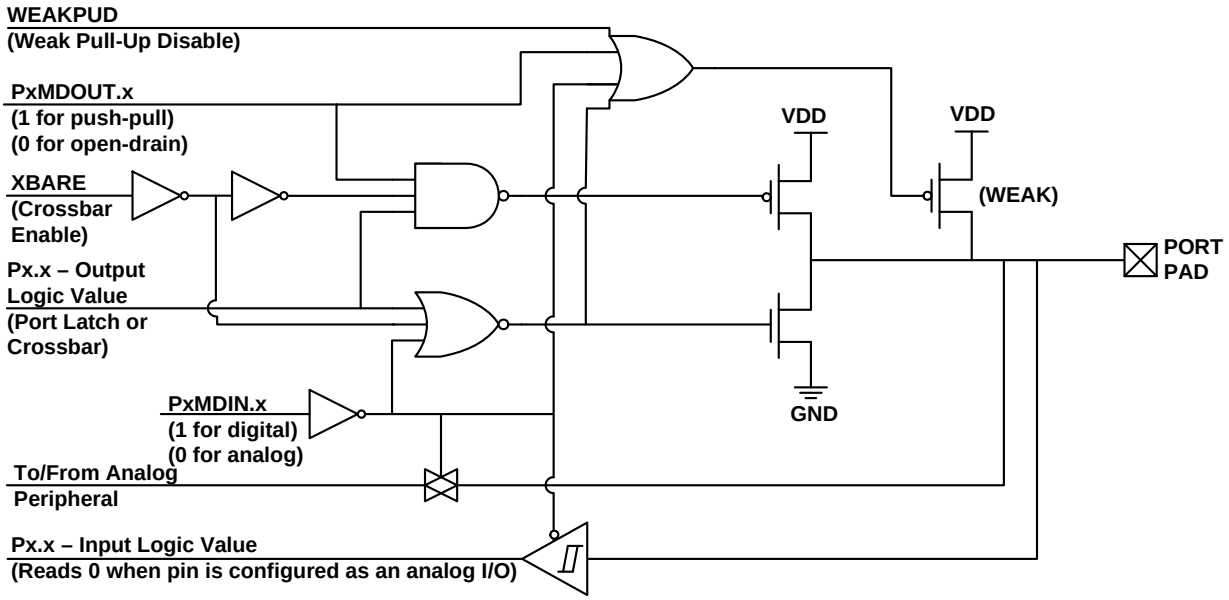


Figure 27.2. Port I/O Cell Block Diagram

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

Port I/O pins P0.0 - P2.3 can be assigned to various analog, digital, and external interrupt functions. The Port pins assigned to analog functions should be configured for analog I/O, and Port pins assigned to digital or external interrupt functions should be configured for digital I/O.

27.2.1. Assigning Port I/O Pins to Analog Functions

Table 27.1 shows all available analog functions that require Port I/O assignments. **Port pins selected for these analog functions should have their corresponding bit in PnSKIP set to '1'.** This reserves the pin for use by the analog function and does not allow it to be claimed by the Crossbar. Table 27.1 shows the potential mapping of Port I/O to each analog function.

Table 27.1. Port I/O Assignment for Analog Functions

Analog Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
ADC Input	P0.0 - P2.3	AMX0P, AMX0N, PnSKIP, PnMDIN
Comparator0 Input	P0.0 - P2.3	CPT0MX, PnSKIP, PnMDIN
Voltage Reference (VREF0)	P0.0	REF0CN, PnSKIP, PnMDIN
Current DAC Output (IDA0)	P0.1	IDA0CN, PnSKIP, PnMDIN
Current DAC Output (IDA1)	P1.0 (20-pin devices) P1.2 (24-pin devices)	IDA1CN, PnSKIP, PnMDIN
External Oscillator in Crystal Mode (XTAL1)	P0.2	OSCXCN, PnSKIP, PnMDIN
External Oscillator in RC, C, or Crystal Mode (XTAL2)	P0.3	OSCXCN, PnSKIP, PnMDIN

27.2.2. Assigning Port I/O Pins to Digital Functions

Any Port pins not assigned to analog functions may be assigned to digital functions or used as GPIO. Most digital functions rely on the Crossbar for pin assignment; however, some digital functions bypass the Crossbar in a manner similar to the analog functions listed above. **Port pins used by these digital functions and any Port pins selected for use as GPIO should have their corresponding bit in PnSKIP set to '1'.** Table 27.2 shows all available digital functions and the potential mapping of Port I/O to each digital function.

Table 27.2. Port I/O Assignment for Digital Functions

Digital Function	Potentially Assignable Port Pins	SFR(s) Used for Assignment
UART0, SPI0, SMBus0, SMBus1, CP0, CP0A, SYSCLK, PCA0 (CEX0-2 and ECI), T0 or T1.	Any Port pin available for assignment by the Crossbar. This includes P0.0 - P2.3 pins which have their PnSKIP bit set to '0'. **Note:** The Crossbar will always assign UART0 pins to P0.4 and P0.5.	XBR0, XBR1
Any pin used for GPIO	P0.0 - P2.4	P0SKIP, P1SKIP, P2SKIP

27.2.3. Assigning Port I/O Pins to External Event Trigger Functions

External event trigger functions can be used to trigger an interrupt or wake the device from a low power mode when a transition occurs on a digital I/O pin. The event trigger functions do not require dedicated pins and will function on both GPIO pins (PnSKIP = ’1’) and pins in use by the Crossbar (PnSKIP = ‘0’). External event trigger functions cannot be used on pins configured for analog I/O. Table 27.3 shows all available external event trigger functions.

Table 27.3. Port I/O Assignment for External Event Trigger Functions

Event Trigger Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
External Interrupt 0	P0.0 - P0.7	IT01CF
External Interrupt 1	P0.0 - P0.7	IT01CF
Port Match	P0.0 - P1.7	P0MASK, P0MAT P1MASK, P1MAT

27.3. Priority Crossbar Decoder

The Priority Crossbar Decoder (Figure 27.3) 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 PnSKIP registers are set. The PnSKIP registers allow software to skip Port pins that are to be used for analog input, dedicated functions, or GPIO.

Important Note on Crossbar Configuration: If a Port pin is claimed by a peripheral without use of the Crossbar, its corresponding PnSKIP bit should be set. This applies to P0.0 if VREF is used, P0.3 and/or P0.2 if the external oscillator circuit is enabled, P0.6 if the ADC or IDAC 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 27.3 shows all of the potential peripheral-to-pin assignments available to the crossbar. Note that this does not mean any peripheral can always be assigned to the highlighted pins. The actual pin assignments are determined by the priority of the enabled peripherals.

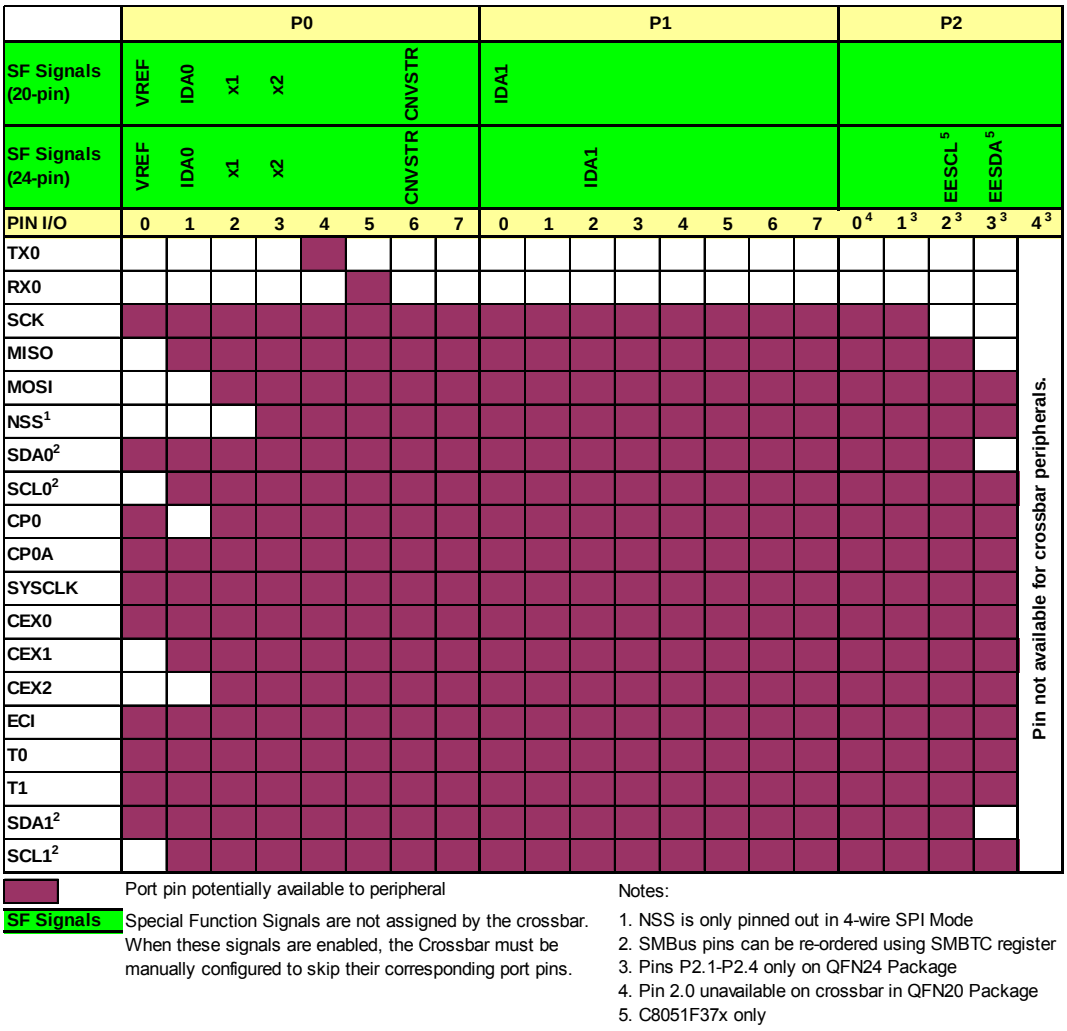


Figure 27.3. Crossbar Priority Decoder - Possible Pin Assignments

Registers XBR0 and XBR1 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); when the UART is selected, the Crossbar assigns both pins associated with the UART (TX and RX). UART0 pin assignments are fixed for bootloading purposes: UART TX0 is always assigned to P0.4; UART RX0 is always assigned to P0.5. Standard Port I/Os appear contiguously after the prioritized functions have been assigned.

Figure 27.4 shows an example of the resulting pin assignments of the device with UART0, SMBus, and CEX0 enabled, the XTAL1 (P0.2) and XTAL2 (P0.3) pins skipped (P0SKIP = 0x0C). UART0 is the highest priority and it will be assigned first. The UART can only appear on P0.4 and P0.5, so that is where it is assigned. The next-highest enabled peripheral is SMBus0. P0.0 and P0.1 are free, so SMBus0 takes these two pins. The last peripheral enabled is the PCA's CEX0 pin. P0.0, P0.1, P0.4 and P0.5 are already occupied by higher-priority peripherals. Additionally, P0.2 and P0.3 are set to be skipped by the crossbar. The CEX0 signal ends up getting routed to P0.6, as it is the next available pin. The other pins on the device are available for use as general-purpose digital I/O or analog functions.

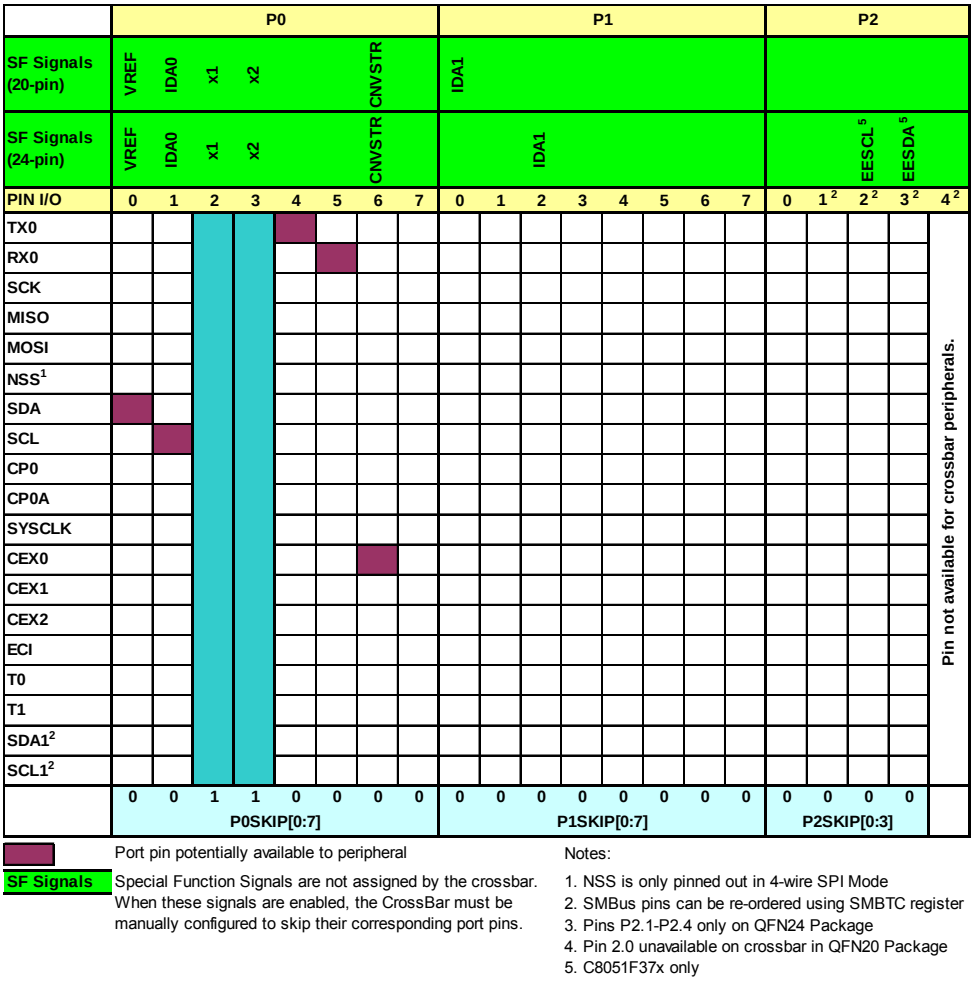


Figure 27.4. Crossbar Priority Decoder Example

Important Notes: The SPI can be operated in either 3-wire or 4-wire modes, pending the state of the NSSMD1–NSSMD0 bits in register SPI0CN. According to the SPI mode, the NSS signal may or may not be routed to a Port pin. The order in which SMBus pins are assigned is defined by the SMBnSWAP bits in the SMBTC register.

27.4. Port I/O Initialization

Port I/O initialization consists of the following steps:

1. Select the input mode (analog or digital) for all Port pins, using the Port Input Mode register (PnMDIN).
2. Select the output mode (open-drain or push-pull) for all Port pins, using the Port Output Mode register (PnMDOUT).
3. Select any pins to be skipped by the I/O Crossbar using the Port Skip registers (PnSKIP).
4. Assign Port pins to desired peripherals.
5. Enable the Crossbar (XBARE = ‘1’).

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 pullup, digital driver, and digital receiver are 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 PnSKIP). Port input mode is set in the PnMDIN 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 27.8 for the PnMDIN register details.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). 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 PnMDOUT settings. When the WEAKPUD bit in XBR1 is ‘0’, a weak pullup is enabled for all Port I/O configured as open-drain. WEAKPUD does not affect the push-pull Port I/O. Furthermore, the weak pullup is turned off on an output that is driving a ‘0’ to avoid unnecessary power dissipation.

Registers XBR0 and XBR1 must be loaded with the appropriate values to select the digital I/O functions required by the design. Setting the XBARE bit in XBR1 to ‘1’ enables the Crossbar. Until the Crossbar is enabled, the external pins remain as standard Port I/O (in input mode), 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.

The Crossbar must be enabled to use Port pins as standard Port I/O in output mode. Port output drivers are disabled while the Crossbar is disabled.

SFR Definition 27.1. XBR0: Port I/O Crossbar Register 0

Bit	7	6	5	4	3	2	1	0
Name	EEPUE	SMB1E	CP0AE	CP0E	SYSCKE	SMB0E	SPI0E	URT0E
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE1; SFR Page = All Pages

Bit	Name	Function
7	EEPUE	**EEPROM Pullup Enable.** 0: On-chip strong pullups not active. 1: On-chip strong pullups active on pins P2.2 and P2.3.
6	SMB1E	**SMBus1 I/O Enable.** 0: SMBus1 I/O unavailable at Port pins. 1: SMBus1 I/O routed to Port pins.
5	CP0AE	**Comparator0 Asynchronous Output Enable.** 0: Asynchronous CP0 unavailable at Port pin. 1: Asynchronous CP0 routed to Port pin.
4	CP0E	**Comparator0 Output Enable.** 0: CP0 unavailable at Port pin. 1: CP0 routed to Port pin.
3	SYSCKE	**/SYSCLK Output Enable.** 0: /SYSCLK unavailable at Port pin. 1: /SYSCLK output routed to Port pin.
2	SMB0E	**SMBus0 I/O Enable.** 0: SMBus0 I/O unavailable at Port pins. 1: SMBus0 I/O routed to Port pins.
1	SPI0E	**SPI I/O Enable.** 0: SPI I/O unavailable at Port pins. 1: SPI I/O routed to Port pins. Note that the SPI can be assigned either 3 or 4 GPIO pins.
0	URT0E	**UART I/O Output Enable.** 0: UART I/O unavailable at Port pin. 1: UART TX0, RX0 routed to Port pins P0.4 and P0.5.

SFR Definition 27.2. XBR1: Port I/O Crossbar Register 1

Bit	7	6	5	4	3	2	1	0
Name	WEAKPUD	XBARE	T1E	T0E	ECIE		PCA0ME[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE2; SFR Page = All Pages

Bit	Name	Function
7	WEAKPUD	**Port I/O Weak Pullup Disable.** 0: Weak Pullups enabled (except for Ports whose I/O are configured for analog mode). 1: Weak Pullups disabled.
6	XBARE	**Crossbar Enable.** 0: Crossbar disabled. 1: Crossbar enabled.
5	T1E	**T1 Enable.** 0: T1 unavailable at Port pin. 1: T1 routed to Port pin.
4	T0E	**T0 Enable.** 0: T0 unavailable at Port pin. 1: T0 routed to Port pin.
3	ECIE	**PCA0 External Counter Input Enable.** 0: ECI unavailable at Port pin. 1: ECI routed to Port pin.
2	Unused	Read = 0b; Write = Don't Care.
1:0	PCA0ME[1:0]	**PCA Module I/O 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.

27.5. Port Match

Port match functionality allows system events to be triggered by a logic value change on P0 or P1. A software controlled value stored in the PnMATCH registers specifies the expected or normal logic values of P0 and P1. A Port mismatch event occurs if the logic levels of the Port's input pins no longer match the software controlled value. This allows Software to be notified if a certain change or pattern occurs on P0 or P1 input pins regardless of the XBRn settings.

The PnMASK registers can be used to individually select which P0 and P1 pins should be compared against the PnMATCH registers. A Port mismatch event is generated if (P0 & P0MASK) does not equal (P0MATCH & P0MASK) or if (P1 & P1MASK) does not equal (P1MATCH & P1MASK).

A Port mismatch event may be used to generate an interrupt or wake the device from a low power mode, such as IDLE or SUSPEND. See the Interrupts and Power Options chapters for more details on interrupt and wake-up sources.

SFR Definition 27.3. P0MASK: Port 0 Mask Register

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

SFR Address = 0xFE; SFR Page = All Pages

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

SFR Definition 27.4. P0MAT: Port 0 Match Register

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

SFR Address = 0xFD; SFR Page = All Pages

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

SFR Definition 27.5. P1MASK: Port 1 Mask Register

Bit	7	6	5	4	3	2	1	0
Name	P1MASK[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xEE; SFR Page = All Pages

Bit	Name	Function
7:0	P1MASK[7:0]	**Port 1 Mask Value.** Selects P1 pins to be compared to the corresponding bits in P1MAT. 0: P1.n pin logic value is ignored and cannot cause a Port Mismatch event. 1: P1.n pin logic value is compared to P1MAT.n.

SFR Definition 27.6. P1MAT: Port 1 Match Register

Bit	7	6	5	4	3	2	1	0
Name	P1MAT[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xED; SFR Page = All Pages

Bit	Name	Function
7:0	P1MAT[7:0]	**Port 1 Match Value.** Match comparison value used on Port 1 for bits in P1MASK which are set to ‘1’. 0: P1.n pin logic value is compared with logic LOW. 1: P1.n pin logic value is compared with logic HIGH.

27.6. Special Function Registers for Accessing and Configuring Port I/O

All Port I/O are accessed through corresponding special function registers (SFRs) that are both byte addressable and bit addressable. When writing to a Port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the Port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the Crossbar, the Port register can always read its corresponding Port I/O pin). The exception to this is the execution of the read-modify-write instructions that target a Port Latch register as the destination. 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 SETB, when the destination is an individual bit in a Port SFR. For these instructions, the value of the latch register (not the pin) is read, modified, and written back to the SFR.

Each Port has a corresponding PnSKIP register which allows its individual Port pins to be assigned to digital functions or skipped by the Crossbar. All Port pins used for analog functions, GPIO, or dedicated digital functions such as the EMIF should have their PnSKIP bit set to ‘1’.

The Port input mode of the I/O pins is defined using the Port Input Mode registers (PnMDIN). Each Port cell can be configured for analog or digital I/O. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is P2.4, which can only be used for digital I/O.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). 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 PnMDOUT settings.

SFR Definition 27.7. P0: Port 0

Bit	7	6	5	4	3	2	1	0
Name	P0[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0x80; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:0	P0[7:0]	**Port 0 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P0.n Port pin is logic LOW. 1: P0.n Port pin is logic HIGH.

SFR Definition 27.8. P0MDIN: Port 0 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P0MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF1; SFR Page = All Pages

Bit	Name	Function
7:0	P0MDIN[7:0]	**Analog Configuration Bits for P0.7–P0.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P0.n pin is configured for analog mode. 1: Corresponding P0.n pin is not configured for analog mode.

SFR Definition 27.9. P0MDOUT: Port 0 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P0MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA4; SFR Page = All Pages

Bit	Name	Function
7:0	P0MDOUT[7:0]	**Output Configuration Bits for P0.7–P0.0 (respectively).** These bits are ignored if the corresponding bit in register P0MDIN is logic 0. 0: Corresponding P0.n Output is open-drain. 1: Corresponding P0.n Output is push-pull.

SFR Definition 27.10. P0SKIP: Port 0 Skip

Bit	7	6	5	4	3	2	1	0
Name	P0SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD4; SFR Page = All Pages

Bit	Name	Function
7:0	P0SKIP[7:0]	**Port 0 Crossbar Skip Enable Bits.** These bits select Port 0 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO 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 27.11. P1: Port 1

Bit	7	6	5	4	3	2	1	0
Name	P1[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0x90; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:0	P1[7:0]	**Port 1 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P1.n Port pin is logic LOW. 1: P1.n Port pin is logic HIGH.

SFR Definition 27.12. P1MDIN: Port 1 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P1MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF2; SFR Page = All Pages

Bit	Name	Function
7:0	P1MDIN[7:0]	**Analog Configuration Bits for P1.7–P1.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P1.n pin is configured for analog mode. 1: Corresponding P1.n pin is not configured for analog mode.

SFR Definition 27.13. P1MDOUT: Port 1 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P1MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA5; SFR Page = All Pages

Bit	Name	Function
7:0	P1MDOUT[7:0]	**Output Configuration Bits for P1.7–P1.0 (respectively).** These bits are ignored if the corresponding bit in register P1MDIN is logic 0. 0: Corresponding P1.n Output is open-drain. 1: Corresponding P1.n Output is push-pull.

SFR Definition 27.14. P1SKIP: Port 1 Skip

Bit	7	6	5	4	3	2	1	0
Name	P1SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD5; SFR Page = All Pages

Bit	Name	Function
7:0	P1SKIP[7:0]	**Port 1 Crossbar Skip Enable Bits.** These bits select Port 1 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P1.n pin is not skipped by the Crossbar. 1: Corresponding P1.n pin is skipped by the Crossbar.

SFR Definition 27.15. P2: Port 2

Bit	7	6	5	4	3	2	1	0
Name				P2[4:0]				
Type	R	R	R	R/W				
Reset	0	0	0	1	1	1	1	1

SFR Address = 0xA0; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:5	Unused	**Unused.**	Don't Care	000b
4:0	P2[4:0]	**Port 2 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P2.n Port pin is logic LOW. 1: P2.n Port pin is logic HIGH.
Note: Pins P2.1-P2.4 are only available in QFN24-packaged devices.				

SFR Definition 27.16. P2MDIN: Port 2 Input Mode

Bit	7	6	5	4	3	2	1	0
Name					P2MDIN[7:0]			
Type	R	R	R	R	R/W			
Reset	0	0	0	0	1	1	1	1

SFR Address = 0xF3; SFR Page = All Pages

Bit	Name	Function
7:4	Unused	Read = 0000b; Write = Don't Care
3:0	P2MDIN[3:0]	**Analog Configuration Bits for P2.3–P2.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P2.n pin is configured for analog mode. 1: Corresponding P2.n pin is not configured for analog mode.
Note: Pins P2.1-P2.4 are only available in QFN24-packaged devices.		

SFR Definition 27.17. P2MDOUT: Port 2 Output Mode

Bit	7	6	5	4	3	2	1	0
Name				P2MDOUT[4:0]				
Type	R	R	R	R/W				
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA6; SFR Page = All Pages

Bit	Name	Function
7:5	Unused	Read = 000b; Write = Don't Care
4:0	P2MDOUT[4:0]	**Output Configuration Bits for P2.4–P2.0 (respectively).** These bits are ignored if the corresponding bit in register P2MDIN is logic 0. 0: Corresponding P2.n Output is open-drain. 1: Corresponding P2.n Output is push-pull.
Note: P2.0 is not available for analog input in the QFN20-packaged devices, and P2.1-P2.4 are only available in the QFN24-packaged devices.		

SFR Definition 27.18. P2SKIP: Port 2 Skip

Bit	7	6	5	4	3	2	1	0
Name					P2SKIP[7:0]			
Type	R	R	R	R	R/W			
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD6; SFR Page = All Pages

Bit	Name	Function
7:4	Unused	Read = 0000b; Write = Don't Care
3:0	P2SKIP[3:0]	**Port 2 Crossbar Skip Enable Bits.** These bits select Port 2 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P2.n pin is not skipped by the Crossbar. 1: Corresponding P2.n pin is skipped by the Crossbar.
Note: P2.0 is not available for crossbar peripherals in the QFN20-packaged devices, and P2.1-P2.4 are only available in the QFN24-packaged devices.		

28. SMBus0 and SMBus1 (I^2C Compatible)

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. The C8051F39x/37x devices contain two SMBus interfaces, SMBus0 and SMBus1.

Reads and writes to the SMBus 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 1/20th of the system clock as a 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 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. The SMBus peripherals can be fully driven by software (i.e., software accepts/rejects slave addresses, and generates ACKs), or hardware slave address recognition and automatic ACK generation can be enabled to minimize software overhead. A block diagram of the SMBus0 peripheral and the associated SFRs is shown in Figure 28.1. SMBus1 is identical, with the exception of the available timer options for the clock source, and the timer used to implement the SCL low timeout feature. Refer to the specific SFR definitions for more details.

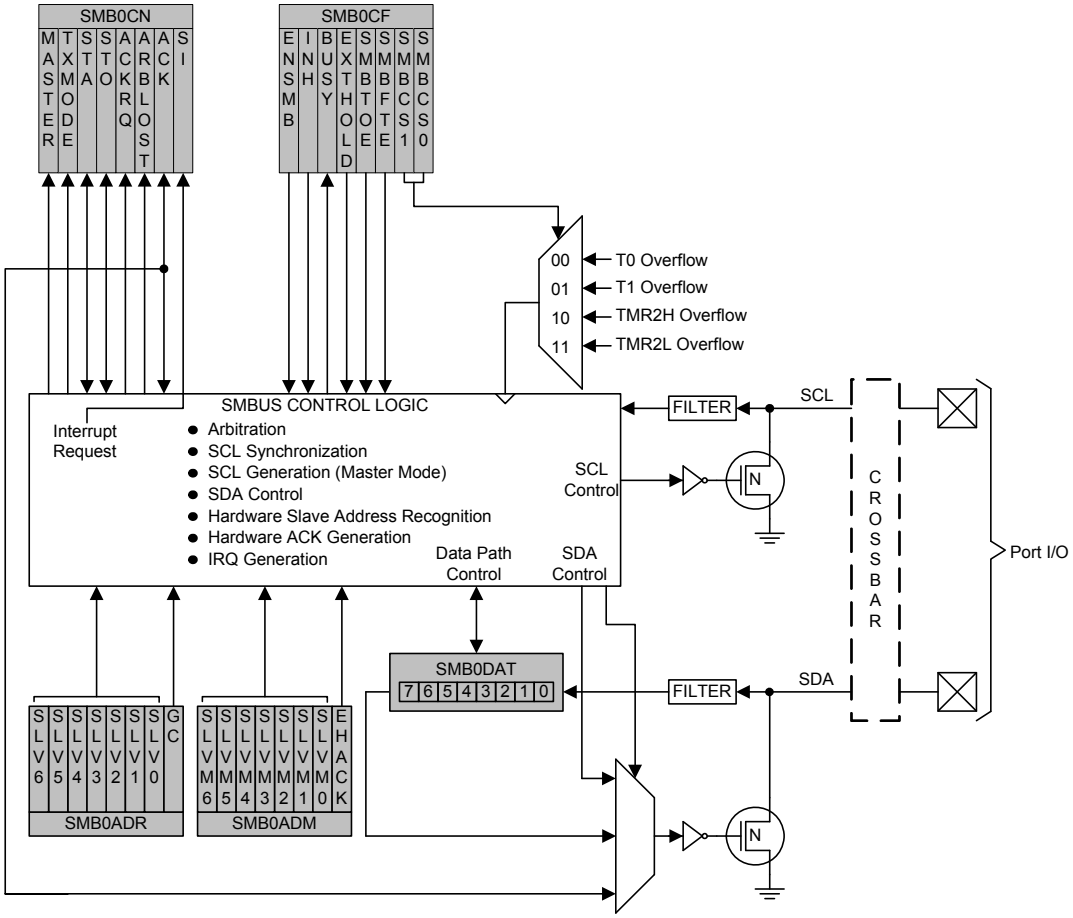


Figure 28.1. SMBus0 Block Diagram

28.1. Supporting Documents

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

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

28.2. SMBus Configuration

Figure 28.2 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. However, the maximum voltage on any port pin must conform to Table 7.1. 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.

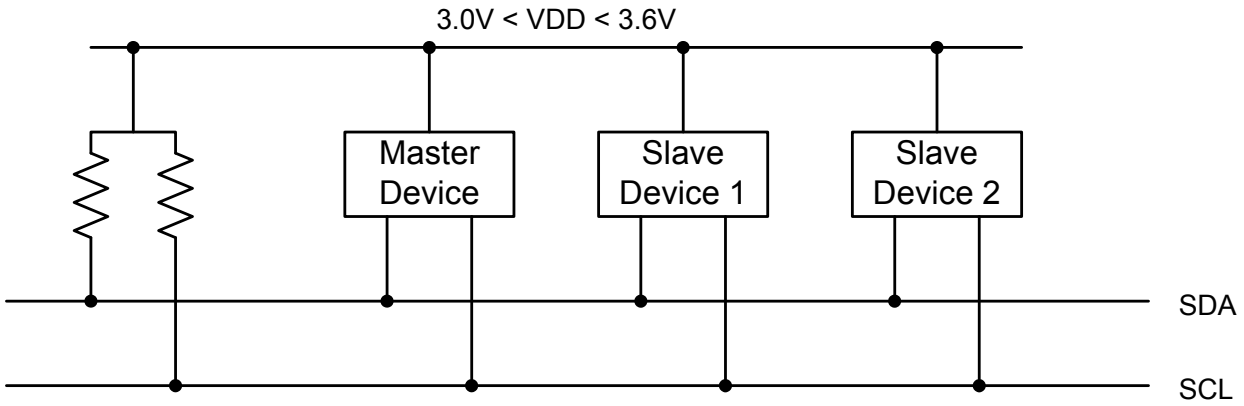


Figure 28.2. Typical SMBus Configuration

28.3. 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. It is not necessary to specify one device as the Master in a system; any device who 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. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 28.3). 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 28.3 illustrates a typical SMBus transaction.

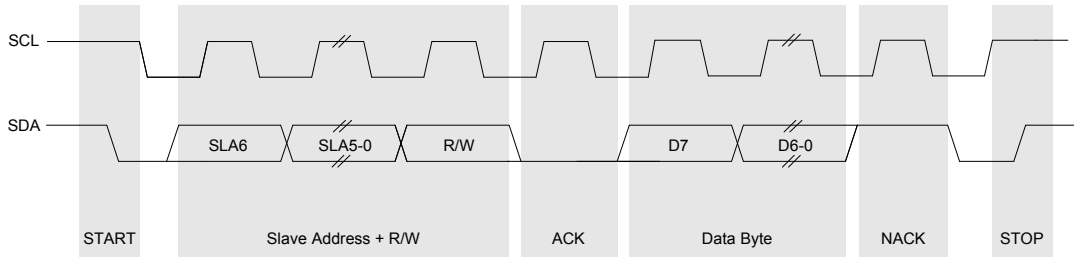


Figure 28.3. SMBus Transaction

28.3.1. Transmitter vs. Receiver

On the SMBus communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

28.3.2. 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 “28.3.5. SCL High (SMBus Free) Timeout” on page 195). 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.

28.3.3. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I2C, 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.

28.3.4. 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.

For the SMBus0 interface, Timer 3 is used to implement SCL low timeouts. Timer 4 is used on the SMBus1 interface for SCL low timeouts. The SCL low timeout feature is enabled by setting the SMBnTOE bit in SMBnCF. The associated timer is forced to reload when SCL is high, and allowed to count when SCL is

low. With the associated timer enabled and configured to overflow after 25 ms (and SMBnTOE set), the timer interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

28.3.5. 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 SMBnFTE bit in SMBnCF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods (as defined by the timer configured for the SMBus clock source). If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. A clock source is required for free timeout detection, even in a slave-only implementation.

28.4. 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
- Optional hardware recognition of slave address and automatic acknowledgement of address/data

SMBus interrupts are generated for each data byte or slave address that is transferred. When hardware acknowledgment is disabled, the point at which the interrupt is generated depends on whether the hardware is acting as a data transmitter or receiver. When a transmitter (i.e., sending address/data, receiving an ACK), this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data (i.e., receiving address/data, sending an ACK), this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. If hardware acknowledgment is enabled, these interrupts are always generated after the ACK cycle. See Section 28.5 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 SMBnCN (SMBus Control register) to find the cause of the SMBus interrupt. The SMBnCN register is described in Section 28.4.4; Table 28.5 provides a quick SMBnCN decoding reference.

28.4.1. SMBus Configuration Register

The SMBus Configuration register (SMBnCF) 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 28.1. SMBus Clock Source Selection

SMBnCS1	SMBnCS0	SMBus0 Clock Source	SMBus1 Clock Source
0	0	Timer 0 Overflow	Timer 0 Overflow
0	1	Timer 1 Overflow	Timer 5 Overflow
1	0	Timer 2 High Byte Overflow	Timer 2 High Byte Overflow
1	1	Timer 2 Low Byte Overflow	Timer 2 Low Byte Overflow

The SMBnCS1–0 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 28.1.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 SMBus0 and SMBus1 clock rates simultaneously. Timer configuration is covered in Section “31. Timers” on page 242.

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

Equation 28.1. 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 28.1. 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 28.2.

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

Equation 28.2. Typical SMBus Bit Rate

Figure 28.4 shows the typical SCL generation described by Equation 28.2. 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 28.1.

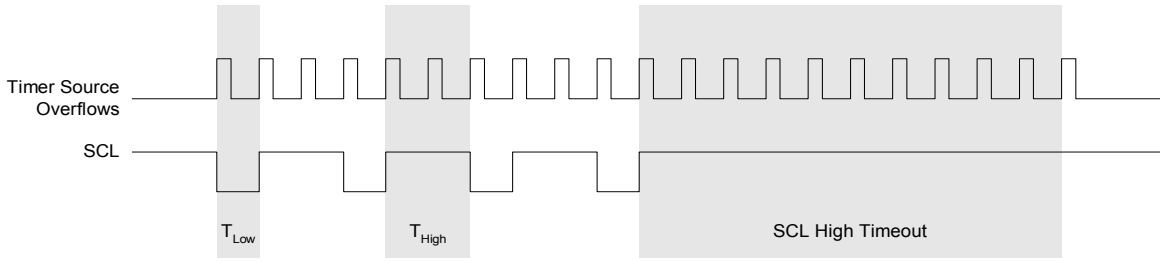


Figure 28.4. Typical SMBus SCL Generation

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 28.2 shows the min-

imum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

Table 28.2. 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. When using software acknowledgement, 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 SMBnTOE bit set, Timer 3 (SMBus0) and Timer 5 (SMBus1) should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “28.3.4. SCL Low Timeout” on page 194). The SMBus interface will force the associated timer to reload while SCL is high, and allow the timer to count when SCL is low. The timer interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBnFTE 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 28.4).

28.4.2. SMBus Pin Swap

The SMBus peripherals are assigned to pins using the priority crossbar decoder. By default, the SMBus signals are assigned to port pins starting with SDA on the lower-numbered pin, and SCL on the next available pin. The SMBnSWAP bits in the SMBTC register can be set to 1 to reverse the order in which the SMBus signals are assigned.

28.4.3. SMBus Timing Control

The SMBnSDD field in the SMBTC register are used to restrict the detection of a START condition under certain circumstances. In some systems where there is significant mis-match between the impedance or the capacitance on the SDA and SCL lines, it may be possible for SCL to fall after SDA during an address or data transfer. Such an event can cause a false START detection on the bus. These kind of events are not expected in a standard SMBus or I2C-compliant system. **In most systems this parameter should not be adjusted, and it is recommended that it be left at its default value.**

By default, if the SCL falling edge is detected after the falling edge of SDA (i.e. one SYSCLK cycle or more), the device will detect this as a START condition. The SMBnSDD field is used to increase the amount of hold time that is required between SDA and SCL falling before a START is recognized. An additional 2, 4, or 8 SYSCLKs can be added to prevent false START detection in systems where the bus conditions warrant this.

SFR Definition 28.1. SMB0CF: SMBus Clock/Configuration

Bit	7	6	5	4	3	2	1	0
Name	ENSMB0	INH0	BUSY0	EXTHOLD0	SMB0TOE	SMB0FTE	SMB0CS[1:0]	
Type	R/W	R/W	R	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC1; SFR Page = 0

Bit	Name	Function
7	ENSMB0	**SMBus0 Enable.** This bit enables the SMBus0 interface when set to 1. When enabled, the interface constantly monitors the SDA0 and SCL0 pins.
6	INH0	**SMBus0 Slave Inhibit.** When this bit is set to logic 1, the SMBus0 does not generate an interrupt when slave events occur. This effectively removes the SMBus0 slave from the bus. Master Mode interrupts are not affected.
5	BUSY0	**SMBus0 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.
4	EXTHOLD0	**SMBus0 Setup and Hold Time Extension Enable.** This bit controls the SDA0 setup and hold times according to Table 28.2. 0: SDA0 Extended Setup and Hold Times disabled. 1: SDA0 Extended Setup and Hold Times enabled.
3	SMB0TOE	**SMBus0 SCL Timeout Detection Enable.** This bit enables SCL low timeout detection. If set to logic 1, the SMBus0 forces Timer 3 to reload while SCL0 is high and allows Timer 3 to count when SCL0 goes low. If Timer 3 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL0 is high. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus0 communication.
2	SMB0FTE	**SMBus0 Free Timeout Detection Enable.** When this bit is set to logic 1, the bus will be considered free if SCL0 and SDA0 remain high for more than 10 SMBus clock source periods.
1:0	SMB0CS[1:0]	**SMBus0 Clock Source Selection.** These two bits select the SMBus0 clock source, which is used to generate the SMBus0 bit rate. The selected device should be configured according to Equation 28.1. 00: Timer 0 Overflow 01: Timer 1 Overflow 10: Timer 2 High Byte Overflow 11: Timer 2 Low Byte Overflow

SFR Definition 28.2. SMB1CF: SMBus Clock/Configuration

Bit	7	6	5	4	3	2	1	0
Name	ENSMB1	INH1	BUSY1	EXTHOLD1	SMB1TOE	SMB1FTE	SMB1CS[1:0]	
Type	R/W	R/W	R	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC1; SFR Page = F

Bit	Name	Function
7	ENSMB1	**SMBus1 Enable.** This bit enables the SMBus1 interface when set to 1. When enabled, the interface constantly monitors the SDA1 and SCL1 pins.
6	INH1	**SMBus1 Slave Inhibit.** When this bit is set to logic 1, the SMBus1 does not generate an interrupt when slave events occur. This effectively removes the SMBus1 slave from the bus. Master Mode interrupts are not affected.
5	BUSY1	**SMBus1 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.
4	EXTHOLD1	**SMBus1 Setup and Hold Time Extension Enable.** This bit controls the SDA1 setup and hold times according to Table 28.2. 0: SDA1 Extended Setup and Hold Times disabled. 1: SDA1 Extended Setup and Hold Times enabled.
3	SMB1TOE	**SMBus1 SCL Timeout Detection Enable.** This bit enables SCL low timeout detection. If set to logic 1, the SMBus1 forces Timer 4 to reload while SCL1 is high and allows Timer 4 to count when SCL1 goes low. If Timer 4 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL1 is high. Timer 4 should be programmed to generate interrupts at 25 ms, and the Timer 4 interrupt service routine should reset SMBus1 communication.
2	SMB1FTE	**SMBus1 Free Timeout Detection Enable.** When this bit is set to logic 1, the bus will be considered free if SCL1 and SDA1 remain high for more than 10 SMBus clock source periods.
1:0	SMB1CS[1:0]	**SMBus1 Clock Source Selection.** These two bits select the SMBus1 clock source, which is used to generate the SMBus1 bit rate. The selected device should be configured according to Equation 28.1. 00: Timer 0 Overflow 01: Timer 5 Overflow 10: Timer 2 High Byte Overflow 11: Timer 2 Low Byte Overflow

SFR Definition 28.3. SMBTC: SMBus Timing and Pin Control

Bit	7	6	5	4	3	2	1	0
Name	SMB1SWAP	SMB0SWAP			SMB1SDD[1:0]		SMB0SDD[1:0]	
Type	R/W	R/W	R/W	R/W	R/W		R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC7; SFR Page = All Pages

Bit	Name	Function
7	SMB1SWAP	**SMBus1 Swap Pins** This bit swaps the order of the SMBus1 pins on the cross-bar. This should be set to 1 when accessing the EEPROM. 0: SDA1 is mapped to the lower-numbered port pin, and SCL1 is mapped to the higher-numbered port pin. 1: SCL1 is mapped to the lower-numbered port pin, and SDA1 is mapped to the higher-numbered port pin.
6	SMB0SWAP	**SMBus0 Swap Pins** This bit swaps the order of the SMBus1 pins on the cross-bar. This should be set to 1 when accessing the EEPROM. 0: SDA0 is mapped to the lower-numbered port pin, and SCL0 is mapped to the higher-numbered port pin. 1: SCL0 is mapped to the lower-numbered port pin, and SDA0 is mapped to the higher-numbered port pin.
5:4	Reserved	Must Write 00b.
3:2	SMB1SDD[1:0]	**SMBus1 Start Detection Window** These bits increase the hold time requirement between SDA falling and SCL falling for START detection. 00: No additional hold time requirement (0-1 SYSCLK). 01: Increase hold time window to 2-3 SYSCLKs. 10: Increase hold time window to 4-5 SYSCLKs. 11: Increase hold time window to 8-9 SYSCLKs.
1:0	SMB0SDD[1:0]	**SMBus0 Start Detection Window** These bits increase the hold time requirement between SDA falling and SCL falling for START detection. 00: No additional hold time window (0-1 SYSCLK). 01: Increase hold time window to 2-3 SYSCLKs. 10: Increase hold time window to 4-5 SYSCLKs. 11: Increase hold time window to 8-9 SYSCLKs.

28.4.4. SMBnCN Control Register

SMBnCN is used to control the interface and to provide status information (see SFR Definition 28.4). The higher four bits of SMBnCN (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

STA and STO indicate that a START and/or STOP has been detected or generated since the last SMBus interrupt. STA and STO are also used to generate START and STOP conditions when operating as a master. Writing a 1 to STA will cause the SMBus interface to enter Master Mode and generate a START when the bus becomes free (STA is not cleared by hardware after the START is generated). Writing a 1 to STO while in Master Mode will cause the interface to generate a STOP and end the current transfer after the next ACK cycle. If STO and STA are both set (while in Master Mode), a STOP followed by a START will be generated.

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 28.3 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.

28.4.4.1. Software ACK Generation

When the EHACK bit in register SMBnADM is cleared to 0, the firmware on the device must detect incoming slave addresses and ACK or NACK the slave address and incoming data bytes. As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received during 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.

28.4.4.2. Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. More detail about automatic slave address recognition can be found in Section 28.4.5. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 28.3 lists all sources for hardware changes to the SMBnCN bits. Refer to Table 28.5 for SMBus status decoding using the SMBnCN register.

SFR Definition 28.4. SMB0CN: SMBus Control

Bit	7	6	5	4	3	2	1	0
Name	MASTER0	TXMODE0	STA0	STO0	ACKRQ0	ARBLOST0	ACK0	SI0
Type	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC0; SFR Page = 0; Bit-Addressable

Bit	Name	Description	Read	Write
7	MASTER0	**SMBus0 Master/Slave Indicator.** This read-only bit indicates when the SMBus0 is operating as a master.	0: SMBus0 operating in slave mode. 1: SMBus0 operating in master mode.	N/A
6	TXMODE0	**SMBus0 Transmit Mode Indicator.** This read-only bit indicates when the SMBus0 is operating as a transmitter.	0: SMBus0 in Receiver Mode. 1: SMBus0 in Transmitter Mode.	N/A
5	STA0	**SMBus0 Start Flag.**	0: No Start or repeated Start detected. 1: Start or repeated Start detected.	0: No Start generated. 1: When Configured as a Master, initiates a START or repeated START.
4	STO0	**SMBus0 Stop Flag.**	0: No Stop condition detected. 1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).	0: No STOP condition is transmitted. 1: When configured as a Master, causes a STOP condition to be transmitted after the next ACK cycle. Cleared by Hardware.
3	ACKRQ0	**SMBus0 Acknowledge Request.**	0: No ACK requested 1: ACK requested	N/A
2	ARBLOST0	**SMBus0 Arbitration Lost Indicator.**	0: No arbitration error. 1: Arbitration Lost	N/A
1	ACK0	**SMBus0 Acknowledge.**	0: NACK received. 1: ACK received.	0: Send NACK 1: Send ACK
0	SI0	**SMBus0 Interrupt Flag.** This bit is set by hardware under the conditions listed in Table 28.3. SI0 must be cleared by software. While SI0 is set, SCL0 is held low and the SMBus0 is stalled.	0: No interrupt pending 1: Interrupt Pending	0: Clear interrupt, and initiate next state machine event. 1: Force interrupt.

SFR Definition 28.5. SMB1CN: SMBus Control

Bit	7	6	5	4	3	2	1	0
Name	MASTER1	TXMODE1	STA1	STO1	ACKRQ1	ARBLOST1	ACK1	SI1
Type	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC0; SFR Page = F; Bit-Addressable

Bit	Name	Description	Read	Write
7	MASTER1	**SMBus1 Master/Slave Indicator.** This read-only bit indicates when the SMBus1 is operating as a master.	0: SMBus1 operating in slave mode. 1: SMBus1 operating in master mode.	N/A
6	TXMODE1	**SMBus1 Transmit Mode Indicator.** This read-only bit indicates when the SMBus1 is operating as a transmitter.	0: SMBus1 in Receiver Mode. 1: SMBus1 in Transmitter Mode.	N/A
5	STA1	**SMBus1 Start Flag.**	0: No Start or repeated Start detected. 1: Start or repeated Start detected.	0: No Start generated. 1: When Configured as a Master, initiates a START or repeated START.
4	STO1	**SMBus1 Stop Flag.**	0: No Stop condition detected. 1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).	0: No STOP condition is transmitted. 1: When configured as a Master, causes a STOP condition to be transmitted after the next ACK cycle. Cleared by Hardware.
3	ACKRQ1	**SMBus1 Acknowledge Request.**	0: No ACK requested 1: ACK requested	N/A
2	ARBLOST1	**SMBus1 Arbitration Lost Indicator.**	0: No arbitration error. 1: Arbitration Lost	N/A
1	ACK1	**SMBus1 Acknowledge.**	0: NACK received. 1: ACK received.	0: Send NACK 1: Send ACK
0	SI1	**SMBus1 Interrupt Flag.** This bit is set by hardware under the conditions listed in Table 28.3. SI1 must be cleared by software. While SI1 is set, SCL1 is held low and the SMBus1 is stalled.	0: No interrupt pending 1: Interrupt Pending	0: Clear interrupt, and initiate next state machine event. 1: Force interrupt.

Table 28.3. Sources for Hardware Changes to SMBnCN

Bit	Set by Hardware When:	Cleared by Hardware When:
MASTERn	■ A START is generated.	■ A STOP is generated. ■ Arbitration is lost.
TXMODEn	■ START is generated. ■ SMBnDAT is written before the start of an SMBus frame.	■ A START is detected. ■ Arbitration is lost. ■ SMBnDAT is not written before the start of an SMBus frame.
STAn	■ A START followed by an address byte is received.	■ Must be cleared by software.
STOn	■ A STOP is detected while addressed as a slave. ■ Arbitration is lost due to a detected STOP.	■ A pending STOP is generated.
ACKRQn	■ A byte has been received and an ACK response value is needed (only when hardware ACK is not enabled).	■ After each ACK cycle.
ARBLOSTn	■ A repeated START is detected as a MASTER when STAn is low (unwanted repeated START). ■ SCLn is sensed low while attempting to generate a STOP or repeated START condition. ■ SDAn is sensed low while transmitting a 1 (excluding ACK bits).	■ Each time SIn is cleared.
ACKn	■ The incoming ACK value is low (ACKNOWLEDGE).	■ The incoming ACK value is high (NOT ACKNOWLEDGE).
SIn	■ 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.

28.4.5. Hardware Slave Address Recognition

The SMBus hardware has the capability to automatically recognize incoming slave addresses and send an ACK without software intervention. Automatic slave address recognition is enabled by setting the EHACK bit in register SMB0ADM to 1. This will enable both automatic slave address recognition and automatic hardware ACK generation for received bytes (as a master or slave). More detail on automatic hardware ACK generation can be found in Section 28.4.4.2.

The registers used to define which address(es) are recognized by the hardware are the SMBus Slave Address register and the SMBus Slave Address Mask register. A single address or range of addresses (including the General Call Address 0x00) can be specified using these two registers. The most-significant seven bits of the two registers are used to define which addresses will be ACKed. A 1 in bit positions of the slave address mask SLVM[6:0] enable a comparison between the received slave address and the hardware's slave address SLV[6:0] for those bits. A 0 in a bit of the slave address mask means that bit will be treated as a "don't care" for comparison purposes. In this case, either a 1 or a 0 value are acceptable on

the incoming slave address. Additionally, if the GCn bit in register SMBnADR is set to 1, hardware will recognize the General Call Address (0x00). Table 28.4 shows some example parameter settings and the slave addresses that will be recognized by hardware under those conditions.

Table 28.4. Hardware Address Recognition Examples (EHACK = 1)

Hardware Slave Address SLVn[6:0]	Slave Address Mask SLVMn[6:0]	GCn bit	Slave Addresses Recognized by Hardware
0x34	0x7F	0	0x34
0x34	0x7F	1	0x34, 0x00 (General Call)
0x34	0x7E	0	0x34, 0x35
0x34	0x7E	1	0x34, 0x35, 0x00 (General Call)
0x70	0x73	0	0x70, 0x74, 0x78, 0x7C

SFR Definition 28.6. SMB0ADR: SMBus0 Slave Address

Bit	7	6	5	4	3	2	1	0
Name	SLV0[6:0]							GC0
Type	R/W							R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD7; SFR Page = 0

Bit	Name	Function
7:1	SLV0[6:0]	**SMBus Hardware Slave Address.** Defines the SMBus0 Slave Address(es) for automatic hardware acknowledgment. Only address bits which have a 1 in the corresponding bit position in SLVM0[6:0] are checked against the incoming address. This allows multiple addresses to be recognized.
0	GC0	**General Call Address Enable.** When hardware address recognition is enabled (EHACK0 = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware. 0: General Call Address is ignored. 1: General Call Address is recognized.

SFR Definition 28.7. SMB0ADM: SMBus0 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM0[6:0]							EHACK0
Type	R/W							R/W
Reset	1	1	1	1	1	1	1	0

SFR Address = 0xE7; SFR Page = 0

Bit	Name	Function
7:1	SLVM0[6:0]	**SMBus0 Slave Address Mask.** Defines which bits of register SMB0ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM0[6:0] enables comparisons with the corresponding bit in SLV0[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
0	EHACK0	**Hardware Acknowledge Enable.** Enables hardware acknowledgement of slave address and received data bytes. 0: Firmware must manually acknowledge all incoming address and data bytes. 1: Automatic Slave Address Recognition and Hardware Acknowledge is Enabled.

SFR Definition 28.8. SMB1ADR: SMBus1 Slave Address

Bit	7	6	5	4	3	2	1	0
Name	SLV1[6:0]							GC1
Type	R/W							R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD7; SFR Page = F

Bit	Name	Function
7:1	SLV1[6:0]	**SMBus1 Hardware Slave Address.** Defines the SMBus1 Slave Address(es) for automatic hardware acknowledgment. Only address bits which have a 1 in the corresponding bit position in SLVM1[6:0] are checked against the incoming address. This allows multiple addresses to be recognized.
0	GC1	**General Call Address Enable.** When hardware address recognition is enabled (EHACK1 = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware. 0: General Call Address is ignored. 1: General Call Address is recognized.

SFR Definition 28.9. SMB1ADM: SMBus1 Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM1[6:0]							EHACK1
Type	R/W							R/W
Reset	1	1	1	1	1	1	1	0

SFR Address = 0xE7; SFR Page = F

Bit	Name	Function
7:1	SLVM1[6:0]	**SMBus1 Slave Address Mask.** Defines which bits of register SMB1ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM1[6:0] enables comparisons with the corresponding bit in SLV1[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
0	EHACK1	**Hardware Acknowledge Enable.** Enables hardware acknowledgement of slave address and received data bytes. 0: Firmware must manually acknowledge all incoming address and data bytes. 1: Automatic Slave Address Recognition and Hardware Acknowledge is Enabled.

28.4.6. Data Register

The SMBus Data register SMBnDAT 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 SIn flag is set. Software should not attempt to access the SMBnDAT register when the SMBus is enabled and the SIn 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 SMBnDAT is always shifted out MSB first. After a byte has been received, the first bit of received data is located at the MSB of SMBnDAT. While data is being shifted out, data on the bus is simultaneously being shifted in. SMBnDAT 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 SMBnDAT.

SFR Definition 28.10. SMB0DAT: SMBus Data

Bit	7	6	5	4	3	2	1	0
Name	SMB0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC2; SFR Page = 0

Bit	Name	Function
7:0	SMB0DAT[7:0]	**SMBus0 Data.** The SMB0DAT register contains a byte of data to be transmitted on the SMBus0 serial interface or a byte that has just been received on the SMBus0 serial interface. The CPU can read from or write to this register whenever the SI0 serial interrupt flag (SMB0CN.0) is set to logic 1. The serial data in the register remains stable as long as the SI0 flag is set. When the SI0 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.

SFR Definition 28.11. SMB1DAT: SMBus Data

Bit	7	6	5	4	3	2	1	0
Name	SMB1DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC2; SFR Page = F

Bit	Name	Function
7:0	SMB1DAT[7:0]	**SMBus1 Data.** The SMB1DAT register contains a byte of data to be transmitted on the SMBus1 serial interface or a byte that has just been received on the SMBus1 serial interface. The CPU can read from or write to this register whenever the SI1 serial interrupt flag (SMB1CN.0) is set to logic 1. The serial data in the register remains stable as long as the SI1 flag is set. When the SI1 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.

28.5. 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 an arbitration or generates a STOP. An SMBus interrupt is generated at the end of all SMBus byte frames. The position of the ACK interrupt when operating as a receiver depends on whether hardware ACK generation is enabled. As a receiver, the interrupt for an ACK occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled. As a transmitter, interrupts occur **after** the ACK, regardless of whether hardware ACK generation is enabled or not.

28.5.1. Write Sequence (Master)

During a write sequence, an SMBus master writes data to a slave device. The master in this transfer will be a transmitter during the address byte, and a transmitter during all data bytes. 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. The interface will switch to Master Receiver Mode if SMB0DAT is not written following a Master Transmitter interrupt. Figure 28.5 shows a typical master write sequence. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

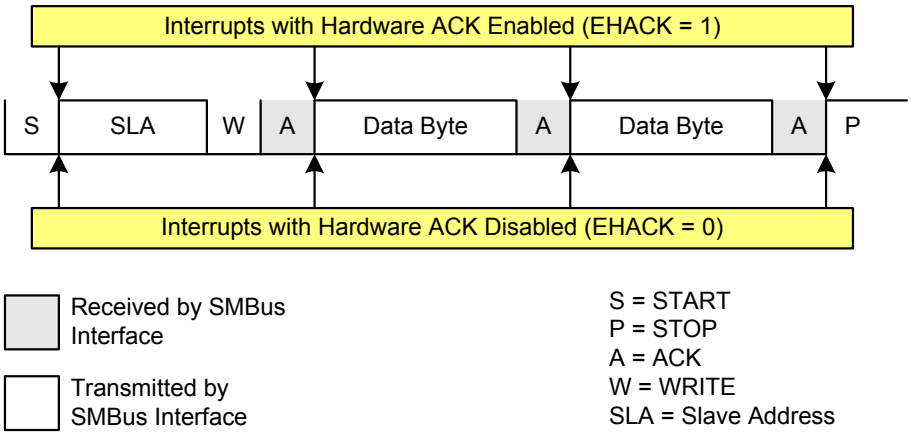


Figure 28.5. Typical Master Write Sequence

28.5.2. Read Sequence (Master)

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. 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.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

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 for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 28.6 shows a typical master read sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

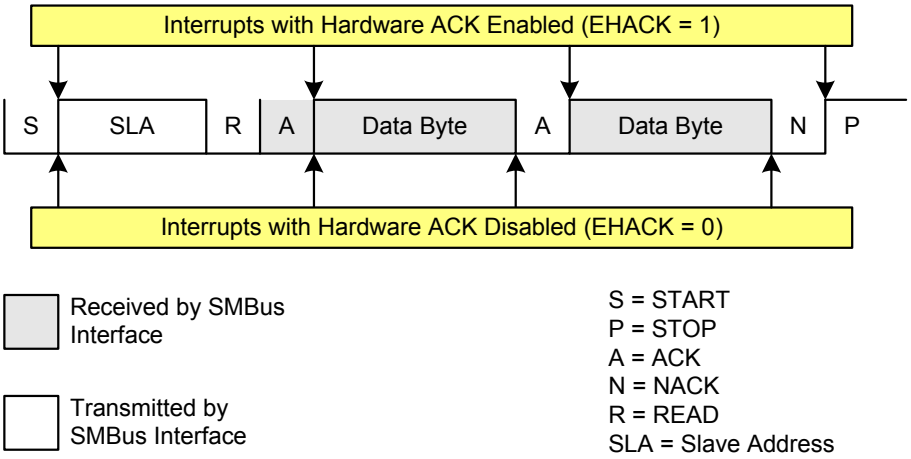


Figure 28.6. Typical Master Read Sequence

28.5.3. Write Sequence (Slave)

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. 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. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), 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.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

The interface exits Slave Receiver Mode after receiving a STOP. The interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 28.7 shows a typical slave write sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

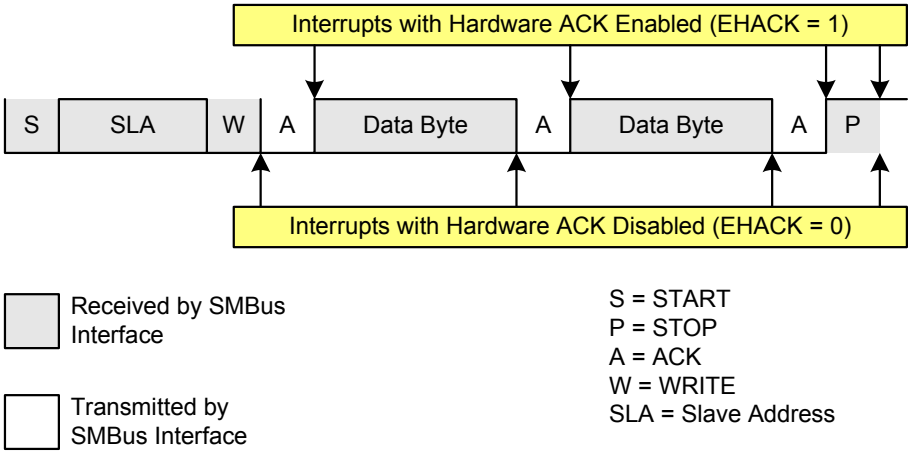


Figure 28.7. Typical Slave Write Sequence

28.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. 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. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters slave transmitter mode, and transmits one or more bytes of data. 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 (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. The interface will switch to slave receiver mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 28.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

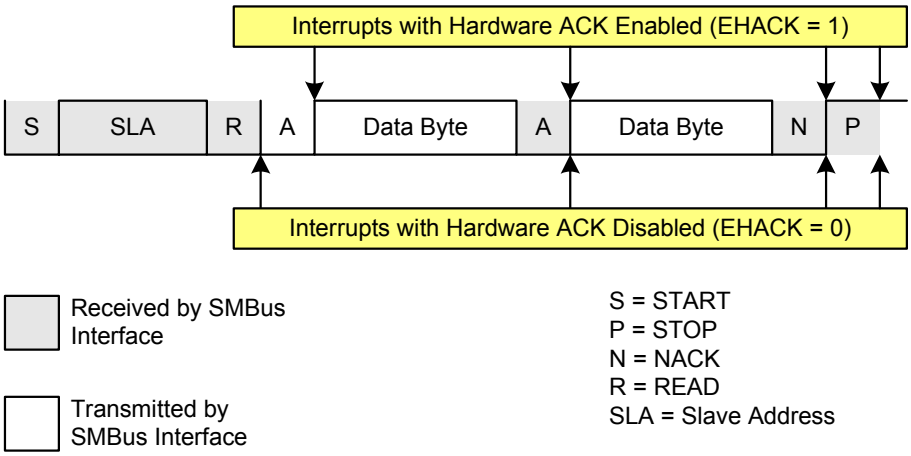


Figure 28.8. Typical Slave Read Sequence

28.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. The appropriate actions to take in response to an SMBus event depend on whether hardware slave address recognition and ACK generation is enabled or disabled. Table 28.5 describes the typical actions when hardware slave address recognition and ACK generation is disabled. Table 28.6 describes the typical actions when hardware slave address recognition and ACK generation is enabled. In the tables, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the SMBus specification. Highlighted responses are allowed by hardware but do not conform to the SMBus specification.

Table 28.5. SMBus Status Decoding: Hardware ACK Disabled (EHACK = 0)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	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
	1100	0	0	0	A master data or address byte was transmitted; NACK received.	Set STA to restart transfer.	1	0	X	1110
						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	1100
						End transfer with STOP.	0	1	X	—
						End transfer with STOP and start another transfer.	1	1	X	—
						Send repeated START.	1	0	X	1110
						Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT).	0	0	X	1000
Master Receiver	1000	1	0	X	A master data byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1	1000
						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	1110
						Send ACK followed by repeated START.	1	0	1	1110
						Send NACK to indicate last byte, and send repeated START.	1	0	0	1110
						Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	1	1100
						Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	0	1100

Table 28.5. SMBus Status Decoding: Hardware ACK Disabled (EHACK = 0) (Continued)

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

Table 28.5. SMBus Status Decoding: Hardware ACK Disabled (EHACK = 0) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Bus Error Condition	0010	0	1	X	Lost arbitration while attempting a repeated START.	Abort failed transfer.	0	0	X	—
						Reschedule failed transfer.	1	0	X	1110
	0001	0	1	X	Lost arbitration due to a detected STOP.	Abort failed transfer.	0	0	X	—
						Reschedule failed transfer.	1	0	X	1110
	0000	1	1	X	Lost arbitration while transmitting a data byte as master.	Abort failed transfer.	0	0	0	—
						Reschedule failed transfer.	1	0	0	1110

Table 28.6. SMBus Status Decoding: Hardware ACK Enabled (EHACK = 1)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Transmitter	1110	0	0	X	A master START was generated.	Load slave address + R/W into SMB0DAT.	0	0	X	1100
	1100	0	0	0	A master data or address byte was transmitted; NACK received.	Set STA to restart transfer.	1	0	X	1110
						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	1100
						End transfer with STOP.	0	1	X	—
						End transfer with STOP and start another transfer.	1	1	X	—
						Send repeated START.	1	0	X	1110
						Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). Set ACK for initial data byte.	0	0	1	1000

Table 28.6. SMBus Status Decoding: Hardware ACK Enabled (EHACK = 1) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Receiver	1000	0	0	1	A master data byte was received; ACK sent.	Set ACK for next data byte; Read SMB0DAT.	0	0	1	1000
						Set NACK to indicate next data byte as the last data byte; Read SMB0DAT.	0	0	0	1000
						Initiate repeated START.	1	0	0	1110
						Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	X	1100
		0	0	0	A master data byte was received; NACK sent (last byte).	Read SMB0DAT; send STOP.	0	1	0	—
						Read SMB0DAT; Send STOP followed by START.	1	1	0	1110
						Initiate repeated START.	1	0	0	1110
						Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	X	1100
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X	0001
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X	0100
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X	0001
	0101	0	X	X	An illegal STOP or bus error was detected while a Slave Transmission was in progress.	Clear STO.	0	0	X	—

Table 28.6. SMBus Status Decoding: Hardware ACK Enabled (EHACK = 1) (Continued)

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

29. 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 “29.1. Enhanced Baud Rate Generation” on page 221). Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

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

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

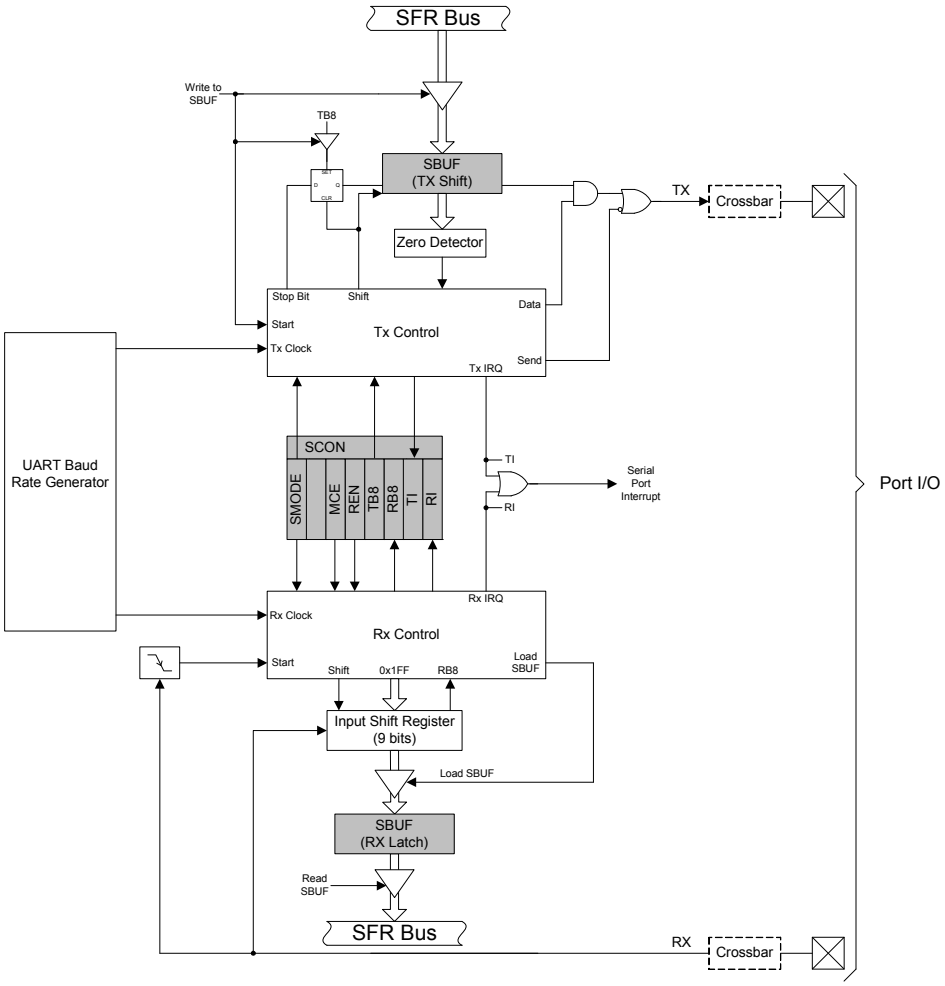


Figure 29.1. UART0 Block Diagram

29.1. 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 29.2), 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.

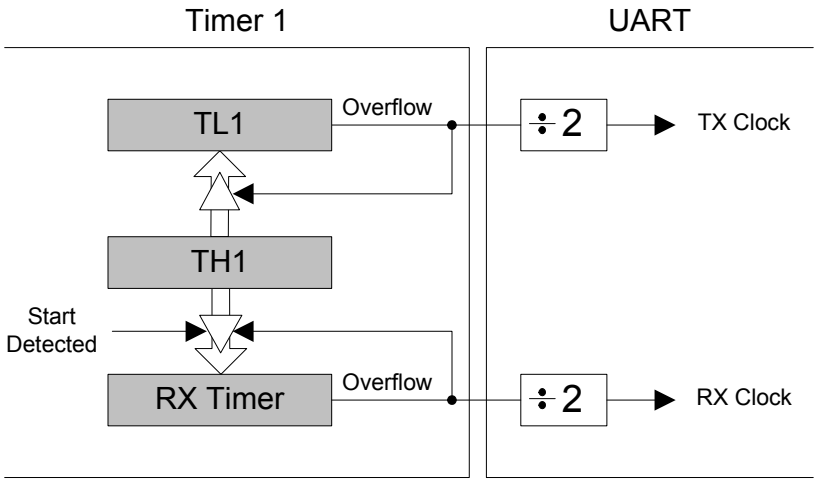


Figure 29.2. UART0 Baud Rate Logic

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see Section “31.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload” on page 247). 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 six sources: SYSCLK, SYSCLK/4, SYSCLK/12, SYSCLK/48, the external oscillator clock/8, or an external input T1. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 29.1-A and Equation 29.1-B.

A) $$\text{UartBaudRate} = \frac{1}{2} \times \text{T1\_Overflow\_Rate}$$

B) $$\text{T1\_Overflow\_Rate} = \frac{\text{T1}_{CLK}}{256 - \text{TH1}}$$

Equation 29.1. 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 “31. Timers” on page 242. A quick reference for typical baud rates and system clock frequencies is given in Table 29.1 through Table 29.2. The internal oscillator may still generate the system clock when the external oscillator is driving Timer 1.

29.2. 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 29.3.

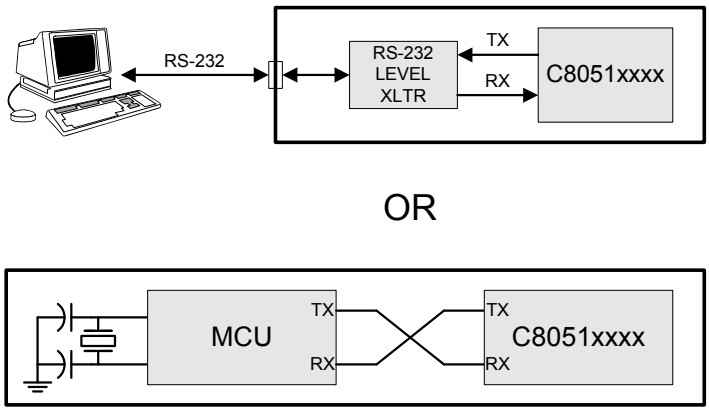


Figure 29.3. UART Interconnect Diagram

29.2.1. 8-Bit UART

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

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

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

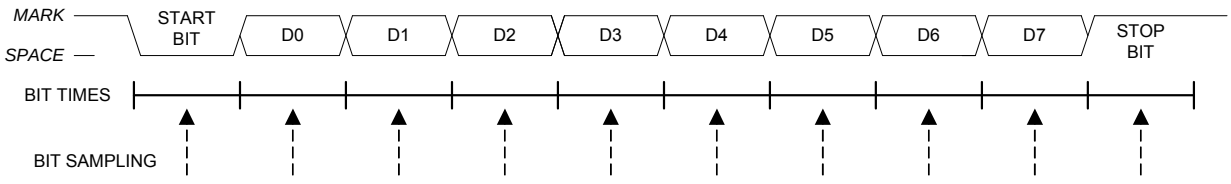


Figure 29.4. 8-Bit UART Timing Diagram

29.2.2. 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.

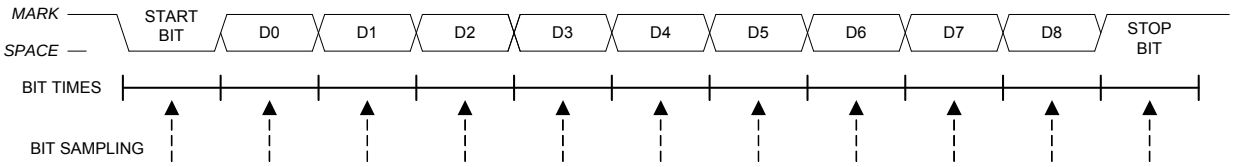


Figure 29.5. 9-Bit UART Timing Diagram

29.3. 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 (SCON0.5) 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 1 (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).

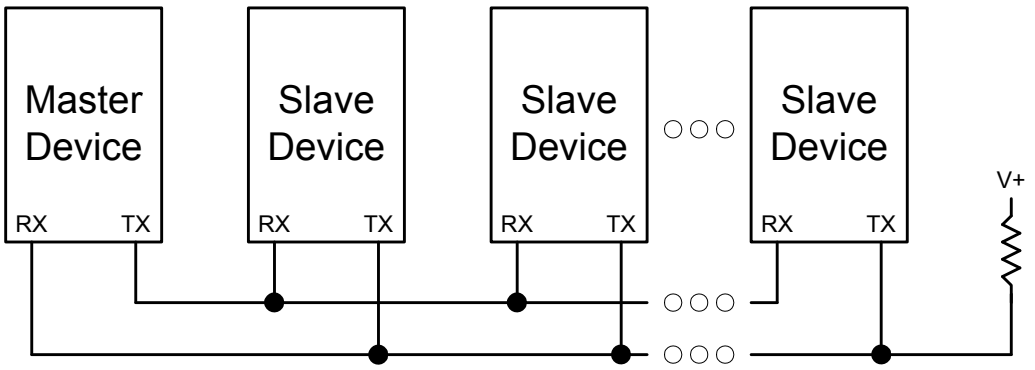


Figure 29.6. UART Multi-Processor Mode Interconnect Diagram

SFR Definition 29.1. SCON0: Serial Port 0 Control

Bit	7	6	5	4	3	2	1	0
Name	S0MODE		MCE0	REN0	TB80	RB80	TI0	RI0
Type	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	0	0	0	0	0	0

SFR Address = 0x98; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	S0MODE	**Serial Port 0 Operation Mode.** Selects the UART0 Operation Mode. 0: 8-bit UART with Variable Baud Rate. 1: 9-bit UART with Variable Baud Rate.
6	Unused	Unused. Read = 1b, Write = Don't Care.
5	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.
4	REN0	**Receive Enable.** 0: UART0 reception disabled. 1: UART0 reception enabled.
3	TB80	**Ninth Transmission Bit.** The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0).
2	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.
1	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.
0	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 29.2. SBUF0: Serial (UART0) Port Data Buffer

Bit	7	6	5	4	3	2	1	0
Name	SBUF0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x99; SFR Page = All Pages

Bit	Name	Function
7:0	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 initiates the transmission. A read of SBUF0 returns the contents of the receive latch.

Table 29.1. Timer Settings for Standard Baud Rates Using The Internal 49 MHz Oscillator

	Frequency: 49 MHz						
	Target Baud Rate (bps)	Baud Rate % Error	Oscillator Divide Factor	Timer Clock Source	SCA1–SCA0 (pre-scale select)[1]	T1M[1]	Timer 1 Reload Value (hex)
SYSCLK from Internal Osc.	230400	–0.32%	212	SYSCLK	XX[2]	1	0x96
	115200	0.15%	426	SYSCLK	XX	1	0x2B
	57600	–0.32%	848	SYSCLK/4	01	0	0x96
	28800	0.15%	1704	SYSCLK/12	00	0	0xB9
	9600	–0.32%	5088	SYSCLK/48	00	0	0xCB
	2400	0.15%	20448	SYSCLK/48	10	0	0x2B

Notes:
1. SCA1–SCA0 and T1M bit definitions can be found in Section 31.1.
2. X = Don't care.

Table 29.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator

	Frequency: 22.1184 MHz						
	Target Baud Rate (bps)	Baud Rate % Error	Oscillator Divide Factor	Timer Clock Source	SCA1–SCA0 (pre-scale select)[1]	T1M[1]	Timer 1 Reload Value (hex)
SYSCLK from External Osc.	230400	0.00%	96	SYSCLK	XX[2]	1	0xD0
	115200	0.00%	192	SYSCLK	XX	1	0xA0
	57600	0.00%	384	SYSCLK	XX	1	0x40
	28800	0.00%	768	SYSCLK / 12	00	0	0xE0
	14400	0.00%	1536	SYSCLK / 12	00	0	0xC0
	9600	0.00%	2304	SYSCLK / 12	00	0	0xA0
	2400	0.00%	9216	SYSCLK / 48	10	0	0xA0
	1200	0.00%	18432	SYSCLK / 48	10	0	0x40
SYSCLK from Internal Osc.	230400	0.00%	96	EXTCLK / 8	11	0	0xFA
	115200	0.00%	192	EXTCLK / 8	11	0	0xF4
	57600	0.00%	384	EXTCLK / 8	11	0	0xE8
	28800	0.00%	768	EXTCLK / 8	11	0	0xD0
	14400	0.00%	1536	EXTCLK / 8	11	0	0xA0
	9600	0.00%	2304	EXTCLK / 8	11	0	0x70

Notes:
1. SCA1–SCA0 and T1M bit definitions can be found in Section 31.1.
2. X = Don't care.

30. Enhanced Serial Peripheral Interface (SPI0)

The Enhanced Serial Peripheral Interface (SPI0) provides access to a flexible, full-duplex synchronous serial bus. SPI0 can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select SPI0 in slave mode, or to disable Master Mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a chip-select output in master mode, or disabled for 3-wire operation. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

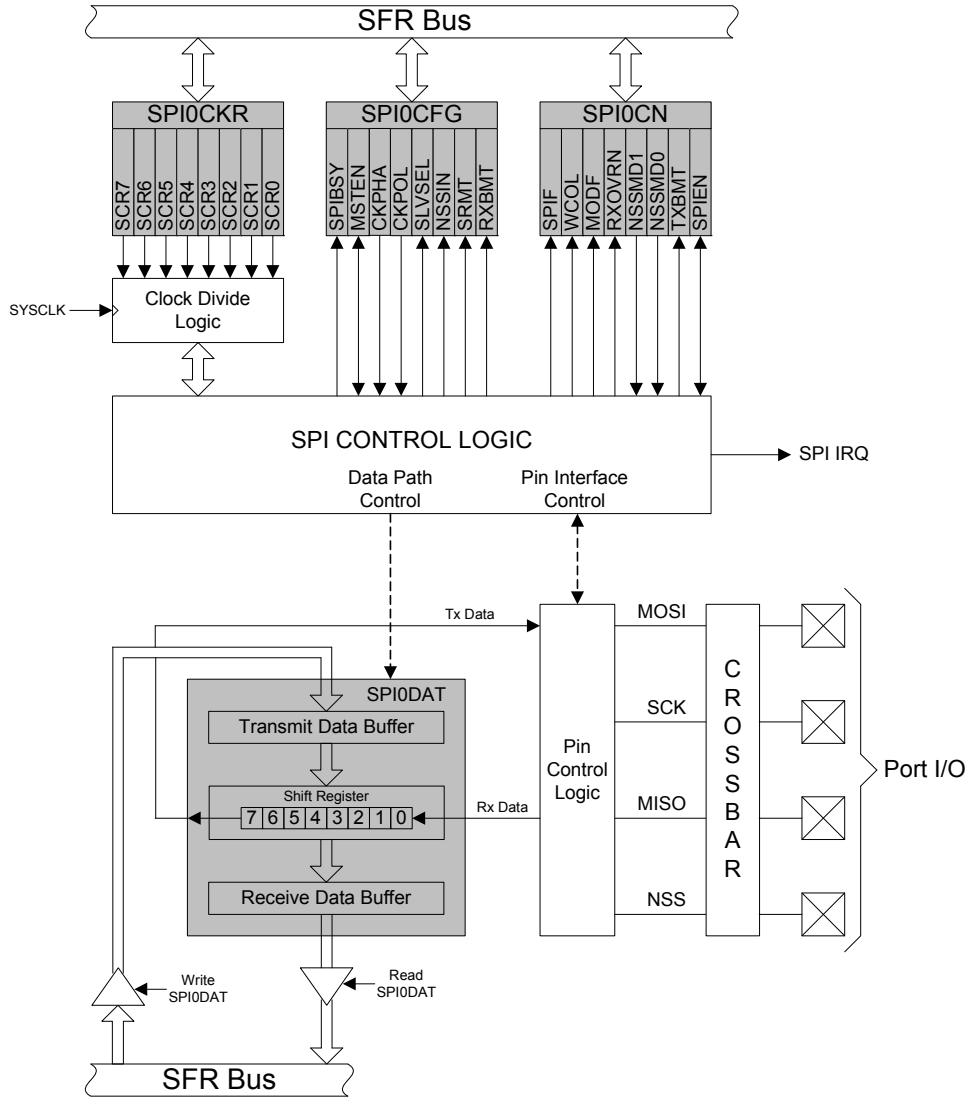


Figure 30.1. SPI Block Diagram

30.1. Signal Descriptions

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

30.1.1. Master Out, Slave In (MOSI)

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

30.1.2. Master In, Slave Out (MISO)

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

30.1.3. Serial Clock (SCK)

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

30.1.4. Slave Select (NSS)

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

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

See Figure 30.2, Figure 30.3, and Figure 30.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device. See Section “27. Port Input/Output” on page 173 for general purpose port I/O and crossbar information.

30.2. SPI0 Master Mode Operation

A SPI master device initiates all data transfers on a SPI bus. SPI0 is placed in master mode by setting the Master Enable flag (MSTEN, SPI0CN.6). Writing a byte of data to the SPI0 data register (SPI0DAT) when in master mode writes to the transmit buffer. If the SPI shift register is empty, the byte in the transmit buffer is moved to the shift register, and a data transfer begins. The SPI0 master immediately shifts out the data serially on the MOSI line while providing the serial clock on SCK. The SPIF (SPI0CN.7) flag is set to logic 1 at the end of the transfer. If interrupts are enabled, an interrupt request is generated when the SPIF flag is set. While the SPI0 master transfers data to a slave on the MOSI line, the addressed SPI slave device simultaneously transfers the contents of its shift register to the SPI master on the MISO line in a full-duplex operation. Therefore, the SPIF flag serves as both a transmit-complete and receive-data-ready flag. The data byte received from the slave is transferred MSB-first into the master's shift register. When a byte is fully shifted into the register, it is moved to the receive buffer where it can be read by the processor by reading SPI0DAT.

When configured as a master, SPI0 can operate in one of three different modes: multi-master mode, 3-wire single-master mode, and 4-wire single-master mode. The default, multi-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In this mode, NSS is an input to the device, and is used to disable the master SPI0 when another master is accessing the bus. When NSS is pulled low in this mode, MSTEN (SPI0CN.6) and SPIEN (SPI0CN.0) are set to 0 to disable the SPI master device, and a Mode Fault is generated (MODF, SPI0CN.5 = 1). Mode Fault will generate an interrupt if enabled. SPI0 must be manually re-enabled in software under these circumstances. In multi-master systems, devices will typically default to being slave devices while they are not acting as the system master device. In multi-master mode, slave devices can be addressed individually (if needed) using general-purpose I/O pins. Figure 30.2 shows a connection diagram between two master devices in multiple-master mode.

3-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. In this mode, NSS is not used, and is not mapped to an external port pin through the crossbar. Any slave devices that must be addressed in this mode should be selected using general-purpose I/O pins. Figure 30.3 shows a connection diagram between a master device in 3-wire master mode and a slave device.

4-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 1. In this mode, NSS is configured as an output pin, and can be used as a slave-select signal for a single SPI device. In this mode, the output value of NSS is controlled (in software) with the bit NSSMD0 (SPI0CN.2). Additional slave devices can be addressed using general-purpose I/O pins. Figure 30.4 shows a connection diagram for a master device in 4-wire master mode and two slave devices.

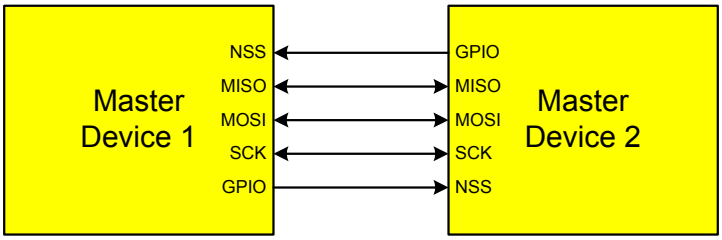


Figure 30.2. Multiple-Master Mode Connection Diagram

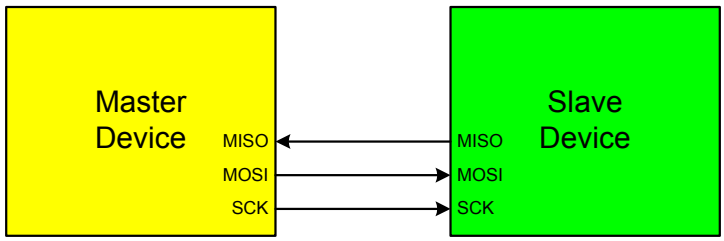


Figure 30.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram

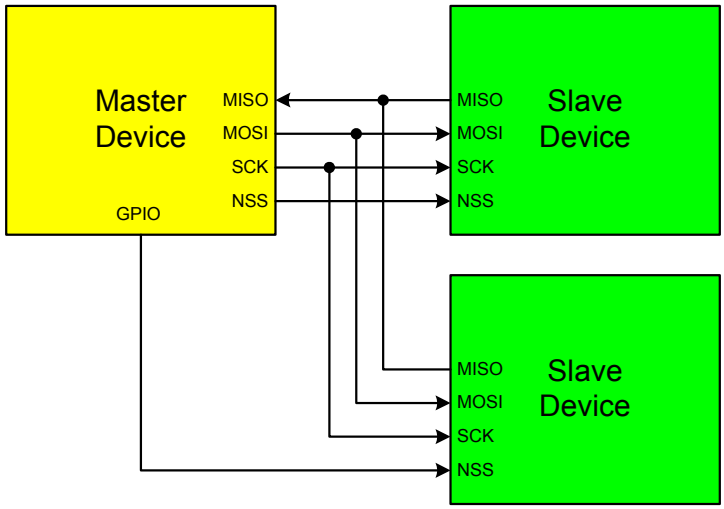


Figure 30.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram

30.3. SPI0 Slave Mode Operation

When SPI0 is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted through the shift register, the SPIF flag is set to logic 1, and the byte is copied into the receive buffer. Data is read from the receive buffer by reading SPI0DAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the shift register by writing to SPI0DAT. Writes to SPI0DAT are double-buffered, and are placed in the transmit buffer first. If the shift register is empty, the contents of the transmit buffer will immediately be transferred into the shift register. When the shift register already contains data, the SPI will load the shift register with the transmit buffer's contents after the last SCK edge of the next (or current) SPI transfer.

When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. The default, 4-wire slave mode, is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In 4-wire mode, the NSS signal is routed to a port pin and configured as a digital input. SPI0 is enabled when NSS is logic 0, and disabled when NSS is logic 1. The bit counter is reset on a falling edge of NSS. Note that the NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 30.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

The 3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and is not mapped to an external port pin through the crossbar. Since there is no way of

uniquely addressing the device in 3-wire slave mode, SPI0 must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling SPI0 with the SPIEN bit. Figure 30.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

30.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

All of the following bits must be cleared by software.

- The SPI Interrupt Flag, SPIF (SPI0CN.7) is set to logic 1 at the end of each byte transfer. This flag can occur in all SPI0 modes.
- The Write Collision Flag, WCOL (SPI0CN.6) is set to logic 1 if a write to SPI0DAT is attempted when the transmit buffer has not been emptied to the SPI shift register. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written.This flag can occur in all SPI0 modes.
- The Mode Fault Flag MODF (SPI0CN.5) is set to logic 1 when SPI0 is configured as a master, and for multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTEN and SPIEN bits in SPI0CN are set to logic 0 to disable SPI0 and allow another master device to access the bus.
- The Receive Overrun Flag RXOVRN (SPI0CN.4) is set to logic 1 when configured as a slave, and a transfer is completed and the receive buffer still holds an unread byte from a previous transfer. The new byte is not transferred to the receive buffer, allowing the previously received data byte to be read. The data byte which caused the overrun is lost.

30.5. Serial Clock Phase and Polarity

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPI0 Configuration Register (SPI0CFG). The CKPHA bit (SPI0CFG.5) selects one of two clock phases (edge used to latch the data). The CKPOL bit (SPI0CFG.4) selects between an active-high or active-low clock. Both master and slave devices must be configured to use the same clock phase and polarity. SPI0 should be disabled (by clearing the SPIEN bit, SPI0CN.0) when changing the clock phase or polarity. The clock and data line relationships for master mode are shown in Figure 30.5. For slave mode, the clock and data relationships are shown in Figure 30.6 and Figure 30.7. Note that CKPHA should be set to 0 on both the master and slave SPI when communicating between two Silicon Labs C8051 devices.

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 30.3 controls the master mode serial clock frequency. This register is ignored when operating in slave mode. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower. When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS (in 4-wire slave mode), and the serial input data synchronously with the slave's system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less than 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the slave's system clock.

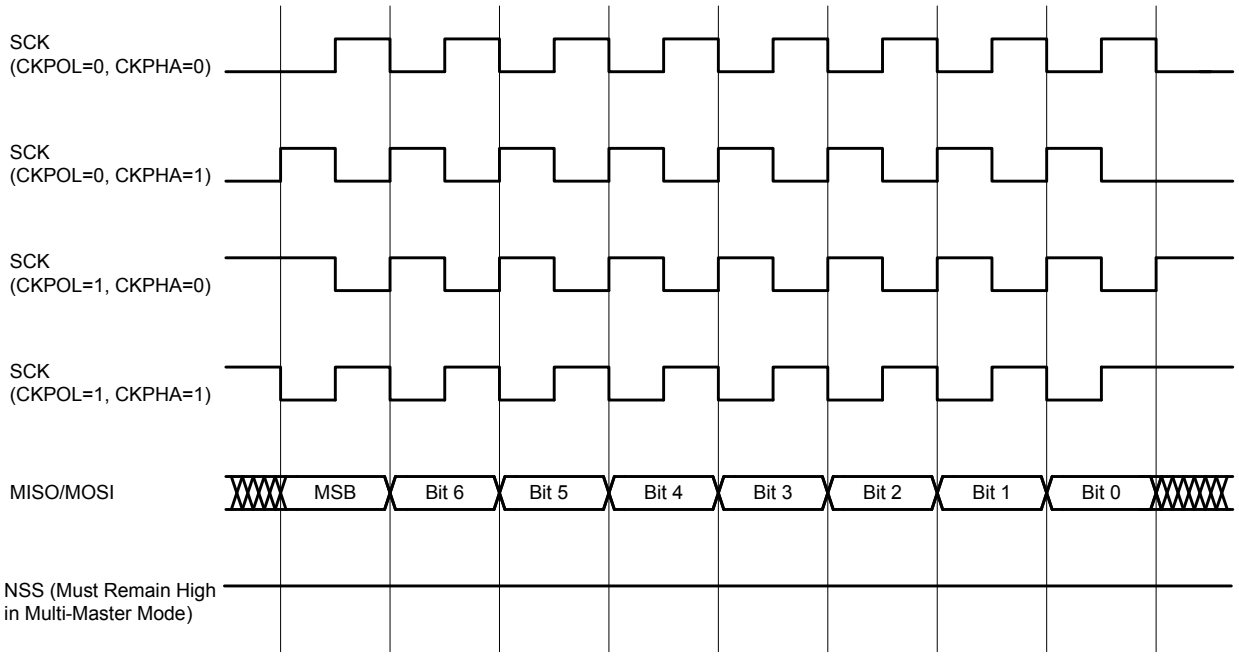


Figure 30.5. Master Mode Data/Clock Timing

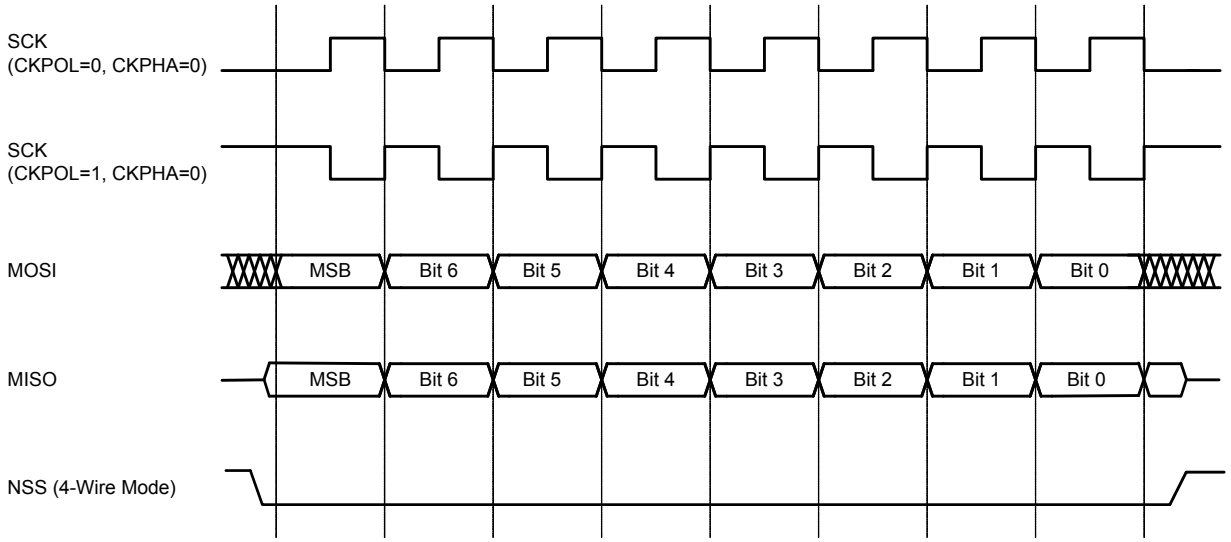


Figure 30.6. Slave Mode Data/Clock Timing (CKPHA = 0)

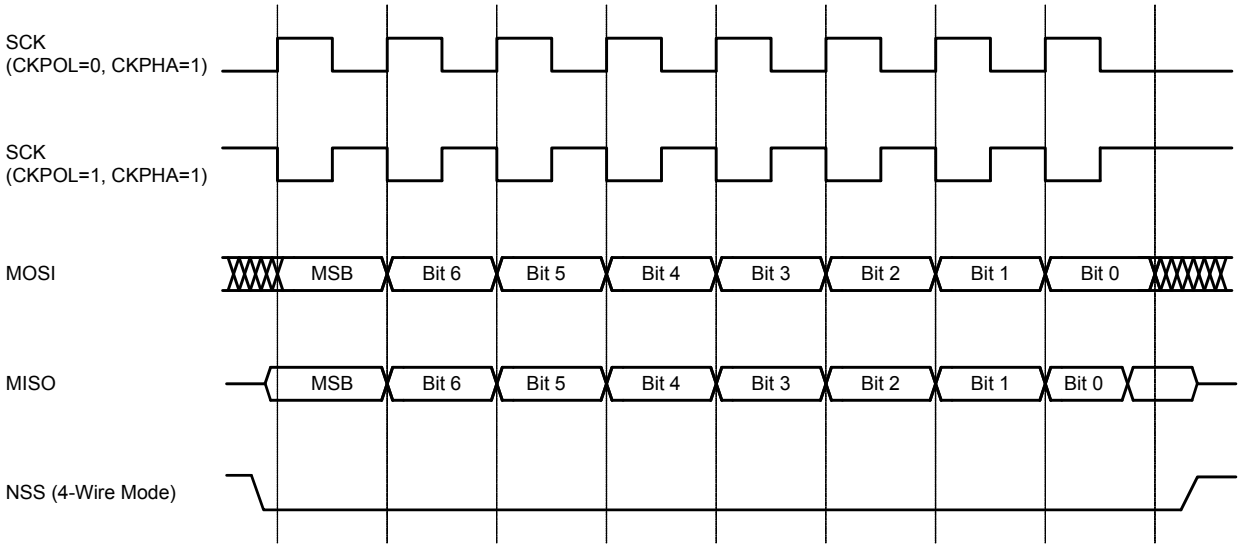


Figure 30.7. Slave Mode Data/Clock Timing (CKPHA = 1)

30.6. SPI Special Function Registers

SPI0 is accessed and controlled through four special function registers in the system controller: SPI0CN Control Register, SPI0DAT Data Register, SPI0CFG Configuration Register, and SPI0CKR Clock Rate Register. The four special function registers related to the operation of the SPI0 Bus are described in the following figures.

SFR Definition 30.1. SPI0CFG: SPI0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	SPIBSY	MSTEN	CKPHA	CKPOL	SLVSEL	NSSIN	SRMT	RXBMT
Type	R	R/W	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	1	1	1

SFR Address = 0xA1; SFR Page = All Pages

Bit	Name	Function
7	SPIBSY	**SPI Busy.** This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode).
6	MSTEN	**Master Mode Enable.** 0: Disable master mode. Operate in slave mode. 1: Enable master mode. Operate as a master.
5	CKPHA	**SPI0 Clock Phase.** 0: Data centered on first edge of SCK period.* 1: Data centered on second edge of SCK period.*
4	CKPOL	**SPI0 Clock Polarity.** 0: SCK line low in idle state. 1: SCK line high in idle state.
3	SLVSEL	**Slave Selected Flag.** This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input.
2	NSSIN	**NSS Instantaneous Pin Input.** This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched.
1	SRMT	**Shift Register Empty (valid in slave mode only).** This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode.
0	RXBMT	**Receive Buffer Empty (valid in slave mode only).** This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode.

Note: In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 30.1 for timing parameters.

SFR Definition 30.2. SPI0CN: SPI0 Control

Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSMD[1:0]		TXBMT	SPIEN
Type	R/W	R/W	R/W	R/W	R/W		R	R/W
Reset	0	0	0	0	0	1	1	0

SFR Address = 0xF8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	SPIF	**SPI0 Interrupt Flag.** This bit is set to logic 1 by hardware at the end of a data transfer. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
6	WCOL	**Write Collision Flag.** This bit is set to logic 1 if a write to SPI0DAT is attempted when TXBMT is 0. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
5	MODF	**Mode Fault Flag.** This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
4	RXOVRN	**Receive Overrun Flag (valid in slave mode only).** This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
3:2	NSSMD[1:0]	**Slave Select Mode.** Selects between the following NSS operation modes: (See Section 30.2 and Section 30.3). 00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin. 01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device. 1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0.
1	TXBMT	**Transmit Buffer Empty.** This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer.
0	SPIEN	**SPI0 Enable.** 0: SPI disabled. 1: SPI enabled.

SFR Definition 30.3. SPI0CKR: SPI0 Clock Rate

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

SFR Address = 0xA2; SFR Page = All Pages

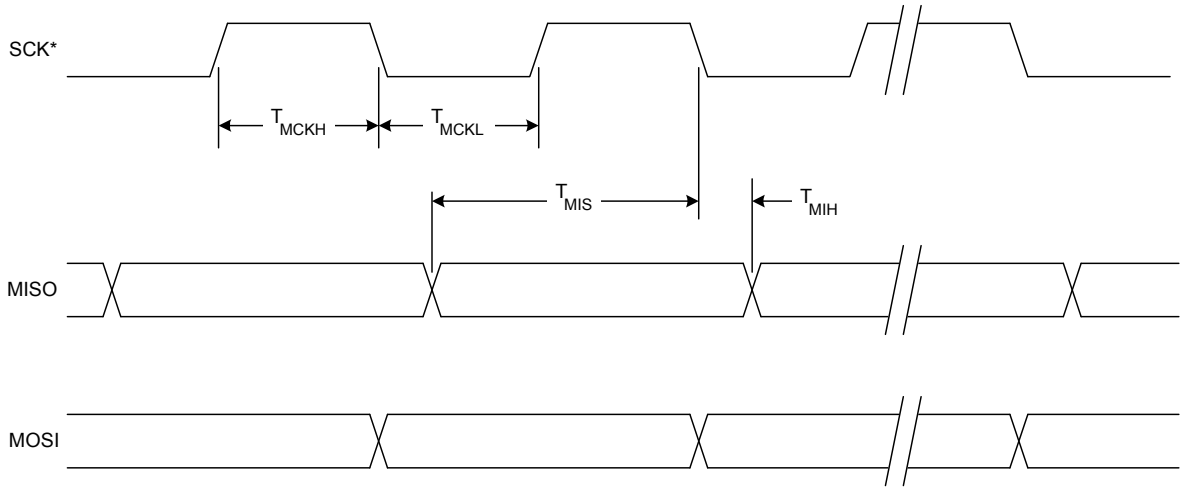
Bit	Name	Function
7:0	SCR[7:0]	**SPI0 Clock Rate.** These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where *SYSCLK* is the system clock frequency and *SPI0CKR* is the 8-bit value held in the SPI0CKR register. $f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR[7:0] + 1)}$ for 0 <= SPI0CKR <= 255 Example: If SYSCLK = 2 MHz and SPI0CKR = 0x04, $f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$ $f_{SCK} = 200kHz$

SFR Definition 30.4. SPI0DAT: SPI0 Data

Bit	7	6	5	4	3	2	1	0
Name	SPI0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

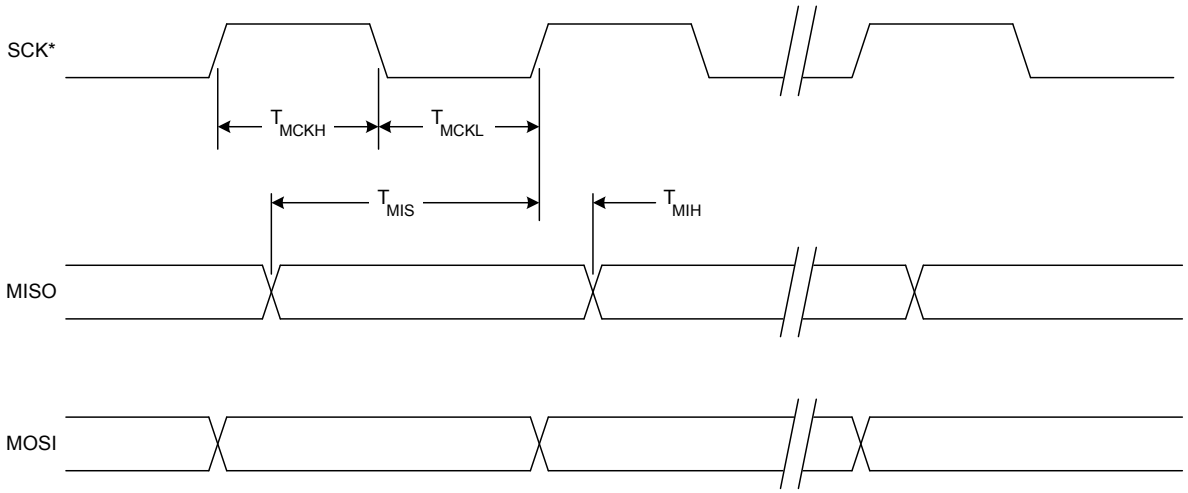
SFR Address = 0xA3; SFR Page = All Pages

Bit	Name	Function
7:0	SPI0DAT[7:0]	**SPI0 Transmit and Receive Data.** The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0DAT places the data into the transmit buffer and initiates a transfer when in Master Mode. A read of SPI0DAT returns the contents of the receive buffer.



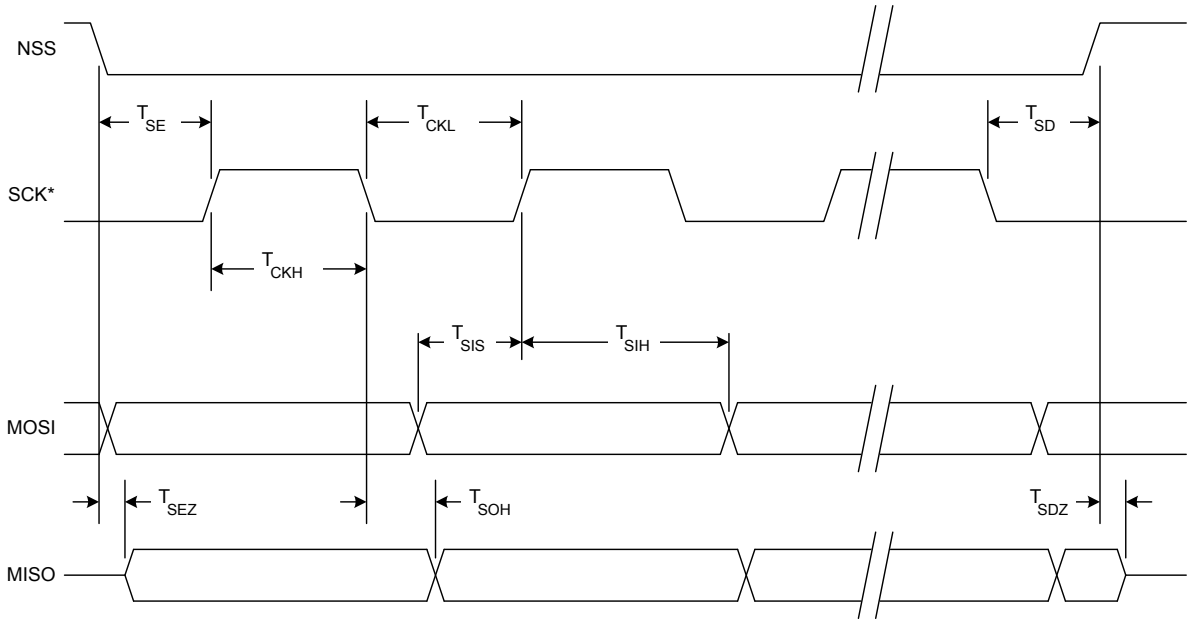
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 30.8. SPI Master Timing (CKPHA = 0)



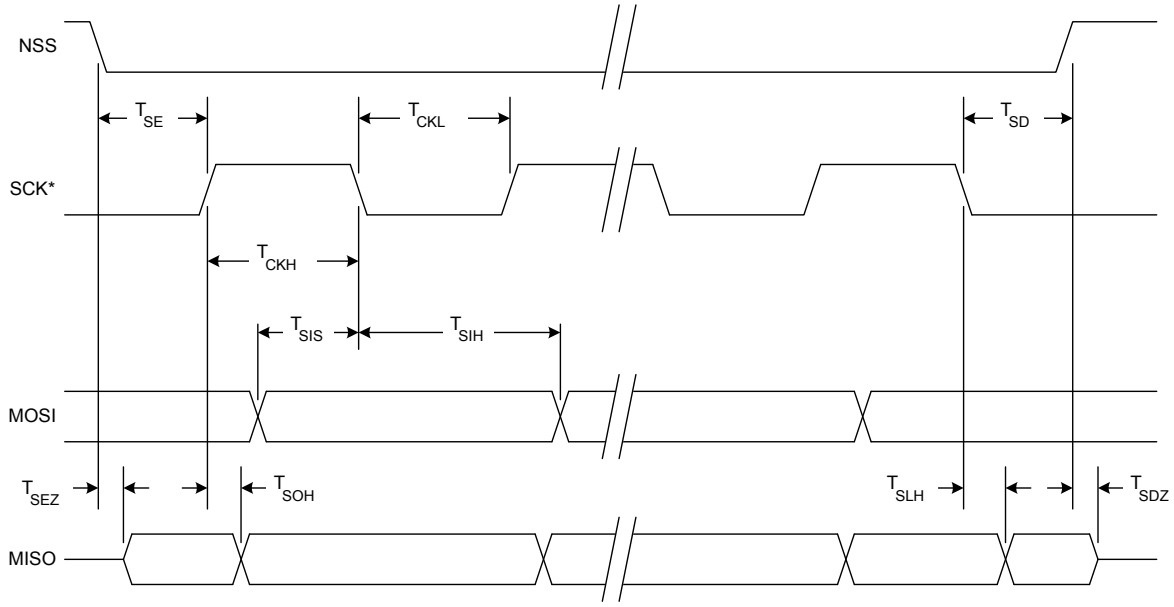
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 30.9. SPI Master Timing (CKPHA = 1)



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

Figure 30.10. SPI Slave Timing (CKPHA = 0)



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

Figure 30.11. SPI Slave Timing (CKPHA = 1)

Table 30.1. SPI Slave Timing Parameters

Parameter	Description	Min	Max	Units
Master Mode Timing (See Figure 30.8 and Figure 30.9)				
T_{MCKH}	SCK High Time	1 x T_{SYSCLK}	—	ns
T_{MCKL}	SCK Low Time	1 x T_{SYSCLK}	—	ns
T_{MIS}	MISO Valid to SCK Shift Edge	1 x T_{SYSCLK} + 20	—	ns
T_{MIH}	SCK Shift Edge to MISO Change	0	—	ns
Slave Mode Timing (See Figure 30.10 and Figure 30.11)				
T_{SE}	NSS Falling to First SCK Edge	2 x T_{SYSCLK}	—	ns
T_{SD}	Last SCK Edge to NSS Rising	2 x T_{SYSCLK}	—	ns
T_{SEZ}	NSS Falling to MISO Valid	—	4 x T_{SYSCLK}	ns
T_{SDZ}	NSS Rising to MISO High-Z	—	4 x T_{SYSCLK}	ns
T_{CKH}	SCK High Time	5 x T_{SYSCLK}	—	ns
T_{CKL}	SCK Low Time	5 x T_{SYSCLK}	—	ns
T_{SIS}	MOSI Valid to SCK Sample Edge	2 x T_{SYSCLK}	—	ns
T_{SIH}	SCK Sample Edge to MOSI Change	2 x T_{SYSCLK}	—	ns
T_{SOH}	SCK Shift Edge to MISO Change	—	4 x T_{SYSCLK}	ns
T_{SLH}	Last SCK Edge to MISO Change (CKPHA = 1 ONLY)	6 x T_{SYSCLK}	8 x T_{SYSCLK}	ns
Note: T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).				

31. Timers

Each MCU includes six counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and four are 16-bit auto-reload timer for use with the 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, 3, 4, and 5 offer 16-bit and split 8-bit timer functionality with auto-reload.

Timer 0 and Timer 1 Modes:	Timer 2, 3, 4, and 5 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 31.1 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, 3, 4, and 5 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 (T0 or T1). Events with a frequency of up to one-fourth the system clock 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.

SFR Definition 31.1. CKCON: Clock Control

Bit	7	6	5	4	3	2	1	0
Name	T3MH	T3ML	T2MH	T2ML	T1M	T0M	SCA[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8E; SFR Page = All Pages

Bit	Name	Function
7	T3MH	**Timer 3 High Byte Clock Select.** Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only). 0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN. 1: Timer 3 high byte uses the system clock.
6	T3ML	**Timer 3 Low Byte Clock Select.** Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode. 0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN. 1: Timer 3 low byte uses the system clock.
5	T2MH	**Timer 2 High Byte Clock Select.** Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only). 0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN. 1: Timer 2 high byte uses the system clock.
4	T2ML	**Timer 2 Low Byte Clock Select.** 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.
3	T1	**Timer 1 Clock Select.** Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1. 0: Timer 1 uses the clock defined by the prescale bits SCA[1:0]. 1: Timer 1 uses the system clock.
2	T0	**Timer 0 Clock Select.** Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1. 0: Counter/Timer 0 uses the clock defined by the prescale bits SCA[1:0]. 1: Counter/Timer 0 uses the system clock.
1:0	SCA[1:0]	**Timer 0/1 Prescale Bits.** These bits control the Timer 0/1 Clock Prescaler: 00: System clock divided by 12 01: System clock divided by 4 10: System clock divided by 48 11: External clock divided by 8 (synchronized with the system clock)

SFR Definition 31.2. CKCON1: Clock Control 1

Bit	7	6	5	4	3	2	1	0
Name					T5MH	T5ML	T4MH	T4ML
Type	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF4; SFR Page = All Pages

Bit	Name	Function
7:4	Unused	Read = 0000b; Write = don't care
3	T5MH	**Timer 5 High Byte Clock Select.** Selects the clock supplied to the Timer 5 high byte (split 8-bit timer mode only). 0: Timer 5 high byte uses the clock defined by the T5XCLK bit in TMR5CN. 1: Timer 5 high byte uses the system clock.
2	T5ML	**Timer 5 Low Byte Clock Select.** Selects the clock supplied to Timer 5. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode. 0: Timer 5 low byte uses the clock defined by the T5XCLK bit in TMR5CN. 1: Timer 5 low byte uses the system clock.
1	T4MH	**Timer 4 High Byte Clock Select.** Selects the clock supplied to the Timer 4 high byte (split 8-bit timer mode only). 0: Timer 4 high byte uses the clock defined by the T4XCLK bit in TMR4CN. 1: Timer 4 high byte uses the system clock.
0	T4ML	**Timer 4 Low Byte Clock Select.** Selects the clock supplied to Timer 4. If Timer 4 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer. 0: Timer 4 low byte uses the clock defined by the T4XCLK bit in TMR4CN. 1: Timer 4 low byte uses the system clock.

31.1. Timer 0 and Timer 1

Each timer is implemented as a 16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register (Section “20.2. Interrupt Register Descriptions” on page 120); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (Section “20.2. Interrupt Register Descriptions” on page 120). 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.

31.1.1. 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 in TCON is set and an interrupt will occur if Timer 0 interrupts are enabled.

The C/T0 bit in the TMOD register selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to Section “27.3. Priority Crossbar Decoder” on page 178 for information on selecting and configuring external I/O pins). Clearing C/T selects the clock defined by the T0M bit in register CKCON. 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 31.1).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 in the TMOD register is logic 0 or the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 20.10). Setting GATE0 to 1 allows the timer to be controlled by the external input signal $\overline{INT0}$ (see Section “20.2. Interrupt Register Descriptions” on page 120), facilitating pulse width measurements

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

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

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal $\overline{INT0}$ is used with Timer 1; the /INT1 polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 20.10).

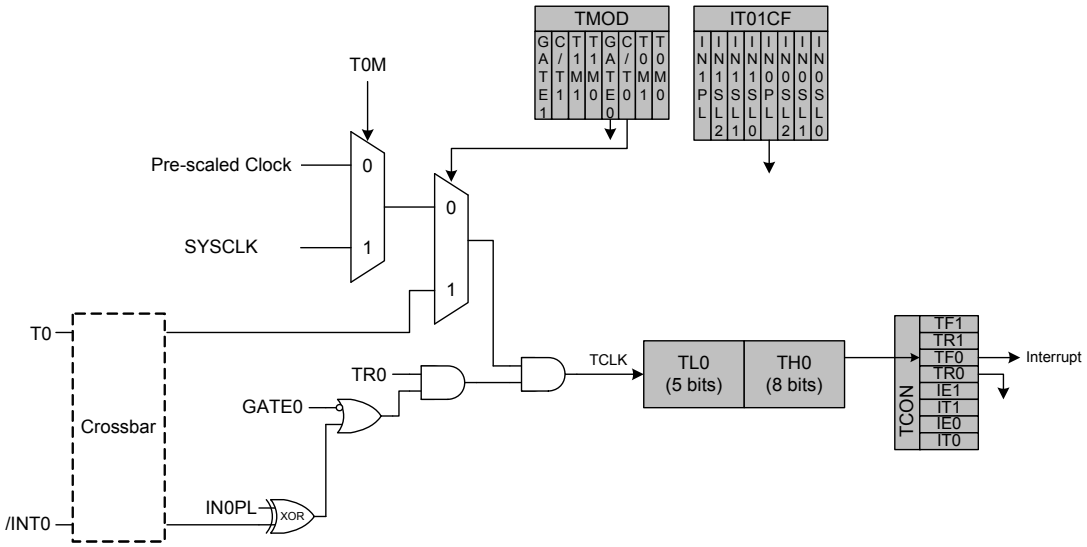


Figure 31.1. T0 Mode 0 Block Diagram

31.1.2. 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.

31.1.3. 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 in the TCON register 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 in the TMOD register is logic 0 or when the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF (see Section “20.3. External Interrupts $\overline{INT0}$ and $\overline{INT1}$” on page 128 for details on the external input signals $\overline{INT0}$ and $\overline{INT1}$).

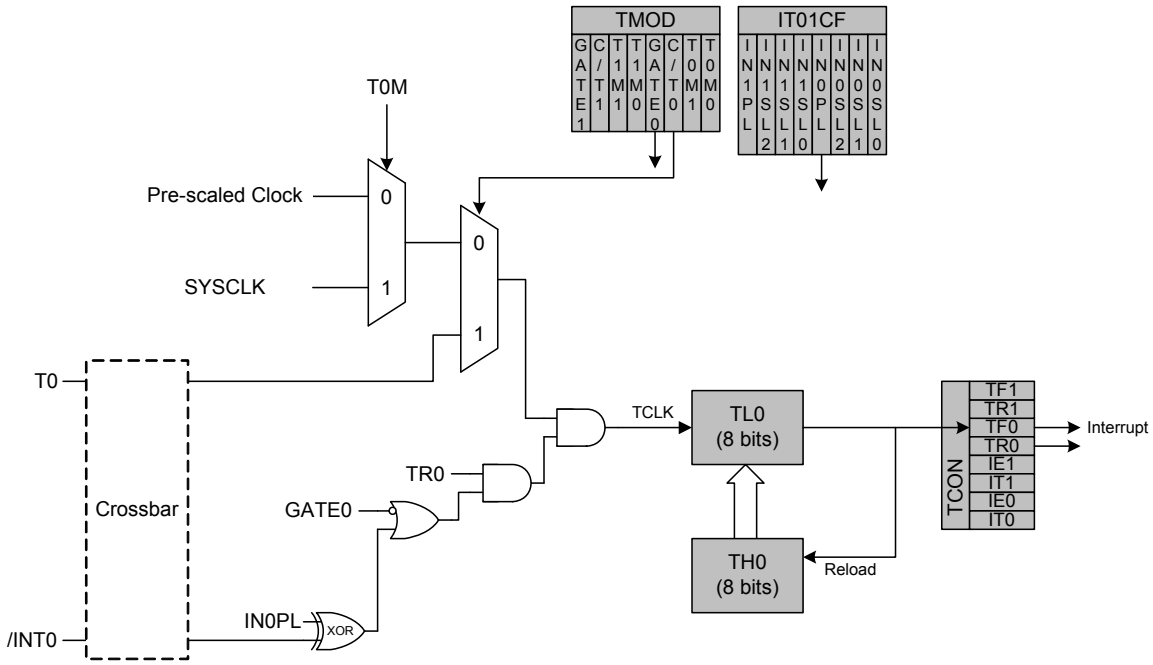


Figure 31.2. T0 Mode 2 Block Diagram

31.1.4. 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.

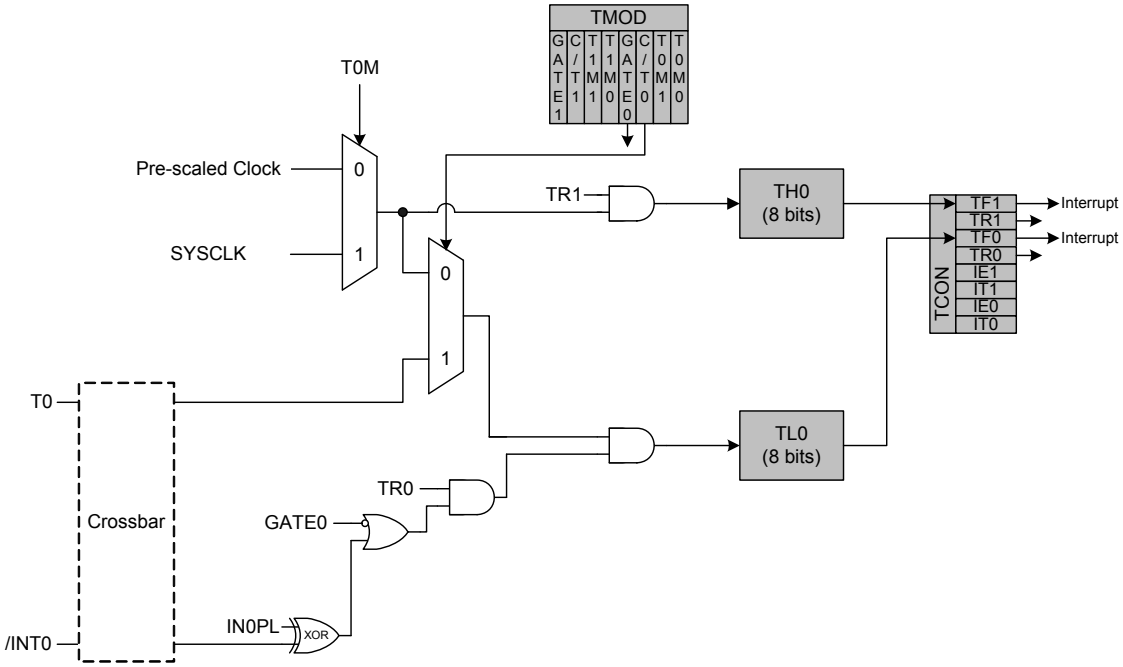


Figure 31.3. T0 Mode 3 Block Diagram

SFR Definition 31.3. TCON: Timer Control

Bit	7	6	5	4	3	2	1	0
Name	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x88; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	TF1	**Timer 1 Overflow Flag.** Set to 1 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.
6	TR1	**Timer 1 Run Control.** Timer 1 is enabled by setting this bit to 1.
5	TF0	**Timer 0 Overflow Flag.** Set to 1 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.
4	TR0	**Timer 0 Run Control.** Timer 0 is enabled by setting this bit to 1.
3	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 in edge-triggered mode.
2	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 20.10). 0: /INT1 is level triggered. 1: /INT1 is edge triggered.
1	IE0	**External Interrupt 0.** 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 0 service routine in edge-triggered mode.
0	IT0	**Interrupt 0 Type Select.** This bit selects whether the configured $\overline{\text{INT0}}$ interrupt will be edge or level sensitive. $\overline{\text{INT0}}$ is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 20.10). 0: $\overline{\text{INT0}}$ is level triggered. 1: $\overline{\text{INT0}}$ is edge triggered.

SFR Definition 31.4. TMOD: Timer Mode

Bit	7	6	5	4	3	2	1	0
Name	GATE1	C/T1	T1M[1:0]		GATE0	C/T0	T0M[1:0]	
Type	R/W	R/W	R/W		R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x89; SFR Page = All Pages

Bit	Name	Function
7	GATE1	**Timer 1 Gate Control.** 0: Timer 1 enabled when TR1 = 1 irrespective of $\overline{INT1}$ logic level. 1: Timer 1 enabled only when TR1 = 1 AND $\overline{INT1}$ is active as defined by bit IN1PL in register IT01CF (see SFR Definition 20.10).
6	C/T1	**Counter/Timer 1 Select.** 0: Timer: Timer 1 incremented by clock defined by T1M bit in register CKCON. 1: Counter: Timer 1 incremented by high-to-low transitions on external pin (T1).
5:4	T1M[1:0]	**Timer 1 Mode Select.** These bits select the Timer 1 operation mode. 00: Mode 0, 13-bit Counter/Timer 01: Mode 1, 16-bit Counter/Timer 10: Mode 2, 8-bit Counter/Timer with Auto-Reload 11: Mode 3, Timer 1 Inactive
3	GATE0	**Timer 0 Gate Control.** 0: Timer 0 enabled when TR0 = 1 irrespective of $\overline{INT0}$ logic level. 1: Timer 0 enabled only when TR0 = 1 AND $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 20.10).
2	C/T0	**Counter/Timer 0 Select.** 0: Timer: Timer 0 incremented by clock defined by T0M bit in register CKCON. 1: Counter: Timer 0 incremented by high-to-low transitions on external pin (T0).
1:0	T0M[1:0]	**Timer 0 Mode Select.** These bits select the Timer 0 operation mode. 00: Mode 0, 13-bit Counter/Timer 01: Mode 1, 16-bit Counter/Timer 10: Mode 2, 8-bit Counter/Timer with Auto-Reload 11: Mode 3, Two 8-bit Counter/Timers

SFR Definition 31.5. TL0: Timer 0 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8A; SFR Page = All Pages

Bit	Name	Function
7:0	TL0[7:0]	**Timer 0 Low Byte.** The TL0 register is the low byte of the 16-bit Timer 0.

SFR Definition 31.6. TL1: Timer 1 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8B; SFR Page = All Pages

Bit	Name	Function
7:0	TL1[7:0]	**Timer 1 Low Byte.** The TL1 register is the low byte of the 16-bit Timer 1.

SFR Definition 31.7. TH0: Timer 0 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8C; SFR Page = All Pages

Bit	Name	Function
7:0	TH0[7:0]	**Timer 0 High Byte.** The TH0 register is the high byte of the 16-bit Timer 0.

SFR Definition 31.8. TH1: Timer 1 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8D; SFR Page = All Pages

Bit	Name	Function
7:0	TH1[7:0]	**Timer 1 High Byte.** The TH1 register is the high byte of the 16-bit Timer 1.

31.2. 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.

31.2.1. 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 31.4, 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.

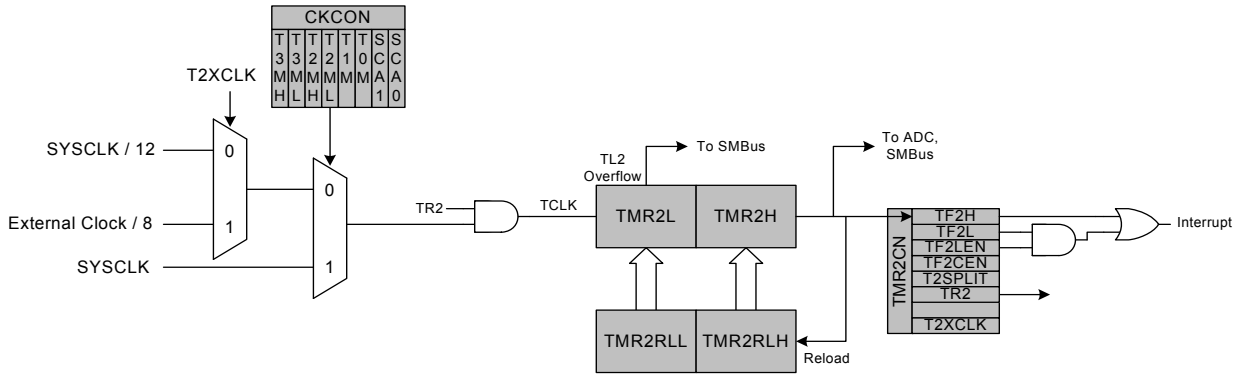


Figure 31.4. Timer 2 16-Bit Mode Block Diagram

31.2.2. 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 31.5. 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

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.

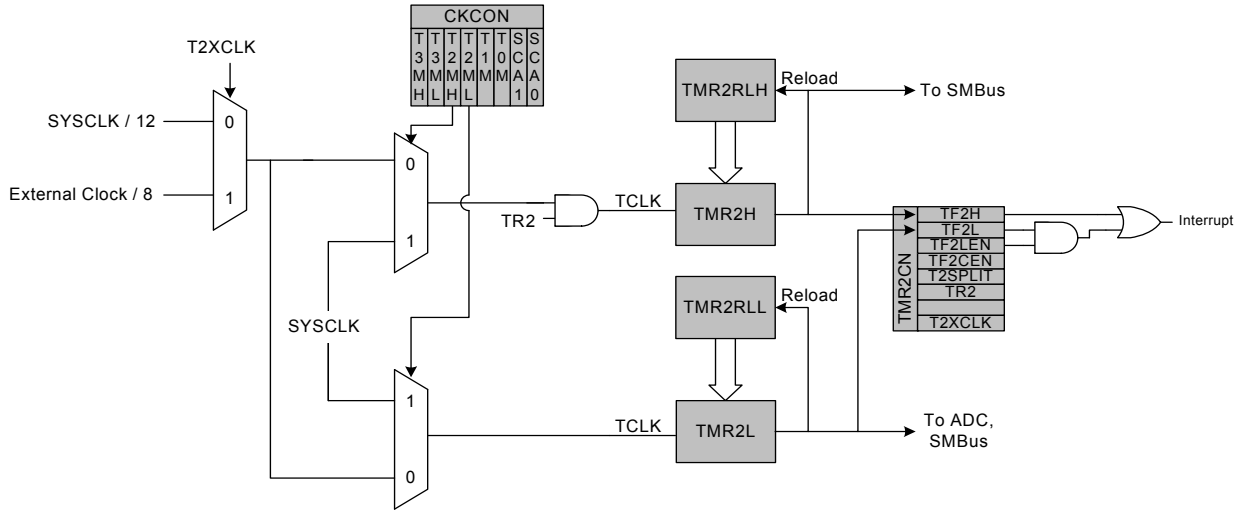


Figure 31.5. Timer 2 8-Bit Mode Block Diagram

31.2.3. Low-Frequency Oscillator (LFO) Capture Mode

The Low-Frequency Oscillator Capture Mode allows the LFO clock to be measured against the system clock or an external oscillator source. Timer 2 can be clocked from the system clock, the system clock divided by 12, or the external oscillator divided by 8, depending on the T2ML (CKCON.4), and T2XCLK settings.

Setting TF2CEN to 1 enables the LFO Capture Mode for Timer 2. In this mode, T2SPLIT should be set to 0, as the full 16-bit timer is used. Upon a falling edge of the low-frequency oscillator, the contents of Timer 2 (TMR2H:TMR2L) are loaded into the Timer 2 reload registers (TMR2RLH:TMR2RLL) and the TF2H flag is set. By recording the difference between two successive timer capture values, the LFO clock frequency can be determined with respect to the Timer 2 clock. The Timer 2 clock should be much faster than the LFO to achieve an accurate reading.

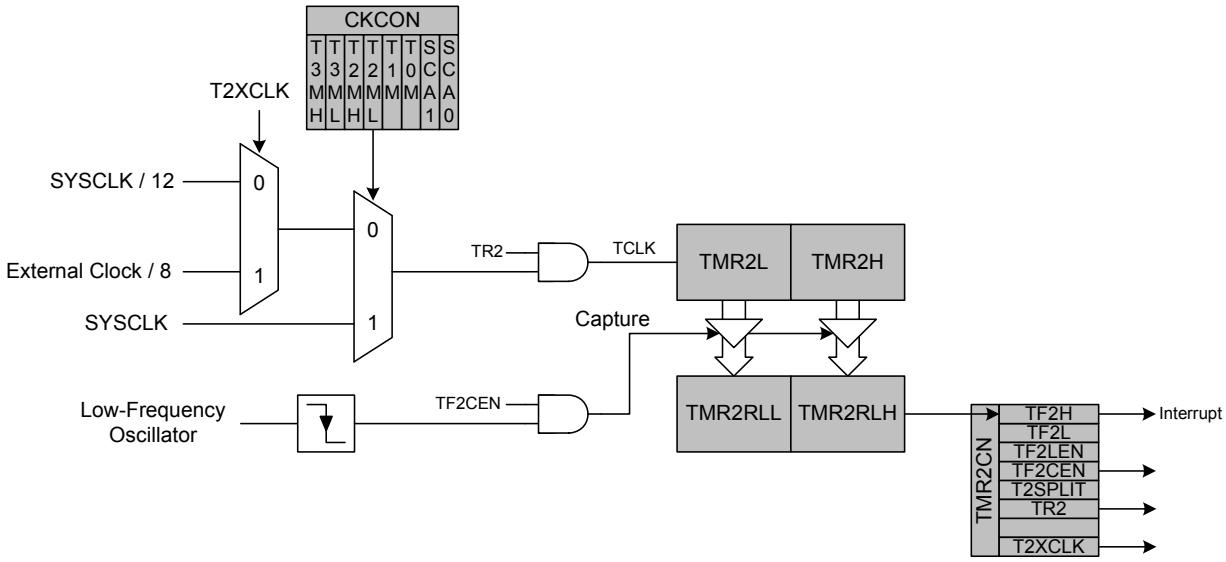


Figure 31.6. Timer 2 Low-Frequency Oscillation Capture Mode Block Diagram

SFR Definition 31.9. TMR2CN: Timer 2 Control

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

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

Bit	Name	Function
7	TF2H	**Timer 2 High Byte Overflow Flag.** Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF2L	**Timer 2 Low Byte Overflow Flag.** Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. TF2L will be set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware.
5	TF2LEN	**Timer 2 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 2 Low Byte interrupts. If Timer 2 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 2 overflows.
4	TF2CEN	**Timer 2 Low-Frequency Oscillator Capture Enable.** When set to 1, this bit enables Timer 2 Low-Frequency Oscillator Capture Mode. If TF2CEN is set and Timer 2 interrupts are enabled, an interrupt will be generated on a falling edge of the low-frequency oscillator output, and the current 16-bit timer value in TMR2H:TMR2L will be copied to TMR2RLH:TMR2RLL.
3	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.
2	TR2	**Timer 2 Run Control.** Timer 2 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in split mode.
1	Unused	Unused. Read = 0b; Write = Don't Care
0	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 clock is the system clock divided by 12. 1: Timer 2 clock is the external clock divided by 8 (synchronized with SYSCLK).

SFR Definition 31.10. TMR2RLL: Timer 2 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCA; SFR Page = 0

Bit	Name	Function
7:0	TMR2RLL[7:0]	**Timer 2 Reload Register Low Byte.** TMR2RLL holds the low byte of the reload value for Timer 2.

SFR Definition 31.11. TMR2RLH: Timer 2 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCB; SFR Page = 0

Bit	Name	Function
7:0	TMR2RLH[7:0]	**Timer 2 Reload Register High Byte.** TMR2RLH holds the high byte of the reload value for Timer 2.

SFR Definition 31.12. TMR2L: Timer 2 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCC; SFR Page = 0

Bit	Name	Function
7:0	TMR2L[7:0]	**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 31.13. TMR2H Timer 2 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCD; SFR Page = 0

Bit	Name	Function
7:0	TMR2H[7:0]	**Timer 2 Low 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.

31.3. Timer 3

Timer 3 is a 16-bit timer formed by two 8-bit SFRs: TMR3L (low byte) and TMR3H (high byte). Timer 3 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T3SPLIT bit (TMR3CN.3) defines the Timer 3 operation mode.

Timer 3 may be clocked by the system clock, the system clock divided by 12, the external oscillator source divided by 8, or the internal low-frequency oscillator divided by 8. The external clock mode is ideal for real-time clock (RTC) functionality, where the internal high-frequency oscillator drives the system clock while Timer 3 is clocked by an external oscillator source. Note that the external oscillator source divided by 8 and the LFO source divided by 8 are synchronized with the system clock when in all operating modes except suspend. When the internal oscillator is placed in suspend mode, The external clock/8 signal or the LFO/8 output can directly drive the timer. This allows the use of an external clock or the LFO to wake up the device from suspend mode. The timer will continue to run in suspend mode and count up. When the timer overflow occurs, the device will wake from suspend mode, and begin executing code again. The timer value may be set prior to entering suspend, to overflow in the desired amount of time (number of clocks) to wake the device. If a wake-up source other than the timer wakes the device from suspend mode, it may take up to three timer clocks before the timer registers can be read or written. During this time, the STSYNC bit in register OSCICN will be set to 1, to indicate that it is not safe to read or write the timer registers.

Important Note: In internal LFO/8 mode, the divider for the internal LFO must be set to 1 for proper functionality. The timer will not operate if the LFO divider is not set to 1.

31.3.1. 16-bit Timer with Auto-Reload

When T3SPLIT (TMR3CN.3) is zero, Timer 3 operates as a 16-bit timer with auto-reload. Timer 3 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 3 reload registers (TMR3RLH and TMR3RLL) is loaded into the Timer 3 register as shown in Figure 31.7, and the Timer 3 High Byte Overflow Flag (TMR3CN.7) is set. If Timer 3 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 3 overflow. Additionally, if Timer 3 interrupts are enabled and the TF3LEN bit is set (TMR3CN.5), an interrupt will be generated each time the lower 8 bits (TMR3L) overflow from 0xFF to 0x00.

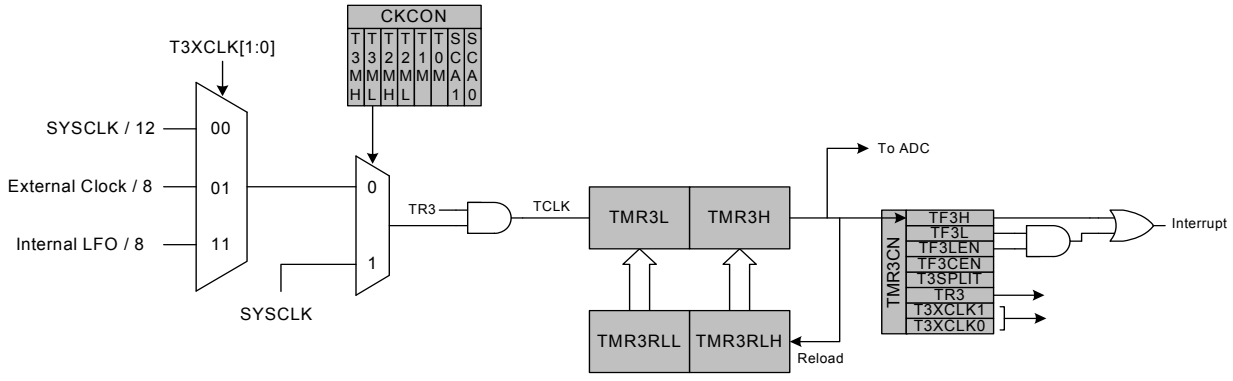


Figure 31.7. Timer 3 16-Bit Mode Block Diagram

31.3.2. 8-bit Timers with Auto-Reload

When T3SPLIT is set, Timer 3 operates as two 8-bit timers (TMR3H and TMR3L). Both 8-bit timers operate in auto-reload mode as shown in Figure 31.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, the external oscillator clock source divided by 8, or the internal Low-frequency Oscillator. The Timer 3 Clock Select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK[1:0] in TMR3CN), as follows:

T3MH	T3XCLK[1:0]	TMR3H Clock Source
0	00	SYSCLK / 12
0	01	External Clock / 8
0	10	Reserved
0	11	Internal LFO
1	X	SYSCLK

T3ML	T3XCLK[1:0]	TMR3L Clock Source
0	00	SYSCLK / 12
0	01	External Clock / 8
0	10	Reserved
0	11	Internal LFO
1	X	SYSCLK

The TF3H bit is set when TMR3H overflows from 0xFF to 0x00; the TF3L bit is set when TMR3L overflows from 0xFF to 0x00. When Timer 3 interrupts are enabled, an interrupt is generated each time TMR3H overflows. If Timer 3 interrupts are enabled and TF3LEN (TMR3CN.5) is set, an interrupt is generated each time either TMR3L or TMR3H overflows. When TF3LEN is enabled, software must check the TF3H and TF3L flags to determine the source of the Timer 3 interrupt. The TF3H and TF3L interrupt flags are not cleared by hardware and must be manually cleared by software.

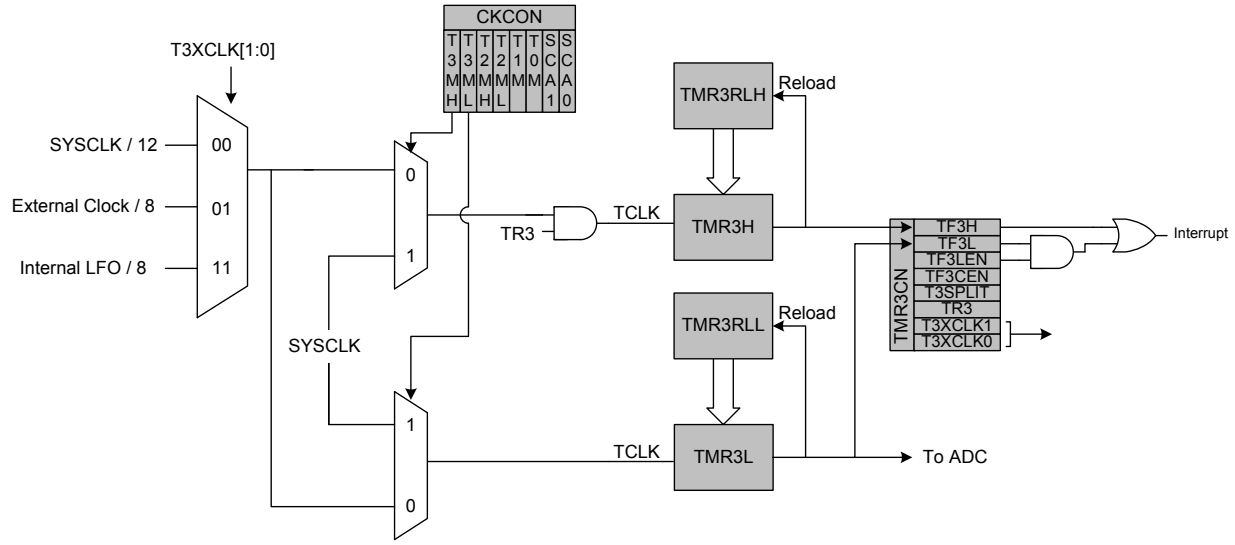


Figure 31.8. Timer 3 8-Bit Mode Block Diagram

31.3.3. Low-Frequency Oscillator (LFO) Capture Mode

The Low-Frequency Oscillator Capture Mode allows the LFO clock to be measured against the system clock or an external oscillator source. Timer 3 can be clocked from the system clock, the system clock divided by 12, or the external oscillator divided by 8, depending on the T3ML (CKCON.6), and T3XCLK[1:0] settings.

Setting TF3CEN to 1 enables the LFO Capture Mode for Timer 3. In this mode, T3SPLIT should be set to 0, as the full 16-bit timer is used. Upon a falling edge of the low-frequency oscillator, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set. By recording the difference between two successive timer capture values, the LFO clock frequency can be determined with respect to the Timer 3 clock. The Timer 3 clock should be much faster than the LFO to achieve an accurate reading. This means that the LFO/8 should not be selected as the timer clock source in this mode.

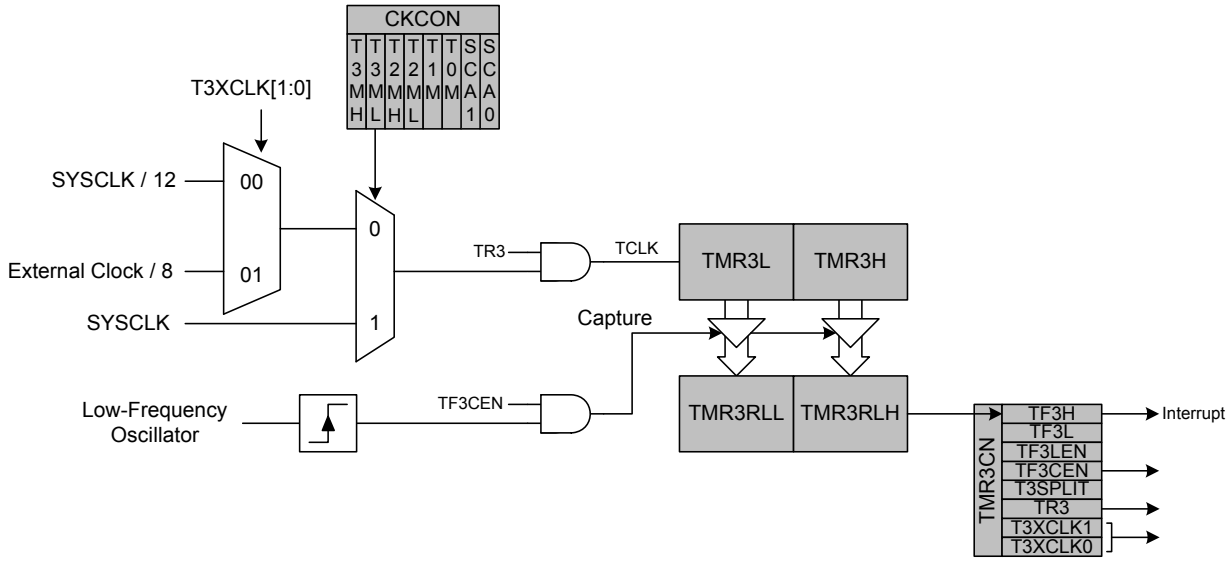


Figure 31.9. Timer 3 Low-Frequency Oscillation Capture Mode Block Diagram

SFR Definition 31.14. TMR3CN: Timer 3 Control

Bit	7	6	5	4	3	2	1	0
Name	TF3H	TF3L	TF3LEN	TF3CEN	T3SPLIT	TR3	T3XCLK[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x91; SFR Page = 0

Bit	Name	Function
7	TF3H	**Timer 3 High Byte Overflow Flag.** Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF3L	**Timer 3 Low Byte Overflow Flag.** Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. TF3L will be set when the low byte overflows regardless of the Timer 3 mode. This bit is not automatically cleared by hardware.
5	TF3LEN	**Timer 3 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows.
4	TF3CEN	**Timer 3 Low-Frequency Oscillator Capture Enable.** When set to 1, this bit enables Timer 3 Low-Frequency Oscillator Capture Mode. If TF3CEN is set and Timer 3 interrupts are enabled, an interrupt will be generated on a falling edge of the low-frequency oscillator output, and the current 16-bit timer value in TMR3H:TMR3L will be copied to TMR3RLH:TMR3RLL.
3	T3SPLIT	**Timer 3 Split Mode Enable.** When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload. 0: Timer 3 operates in 16-bit auto-reload mode. 1: Timer 3 operates as two 8-bit auto-reload timers.
2	TR3	**Timer 3 Run Control.** Timer 3 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in split mode.
1:0	T3XCLK[1:0]	**Timer 3 External Clock Select.** This bit selects the “external” clock source for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the external clock and the system clock for either timer. 00: System clock divided by 12. 01: External clock divided by 8 (synchronized with SYSCLK when not in suspend). 10: Reserved. 11: Internal LFO/8 (synchronized with SYSCLK when not in suspend).

SFR Definition 31.15. TMR3RLL: Timer 3 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x92; SFR Page = 0

Bit	Name	Function
7:0	TMR3RLL[7:0]	**Timer 3 Reload Register Low Byte.** TMR3RLL holds the low byte of the reload value for Timer 3.

SFR Definition 31.16. TMR3RLH: Timer 3 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x93; SFR Page = 0

Bit	Name	Function
7:0	TMR3RLH[7:0]	**Timer 3 Reload Register High Byte.** TMR3RLH holds the high byte of the reload value for Timer 3.

SFR Definition 31.17. TMR3L: Timer 3 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x94; SFR Page = 0

Bit	Name	Function
7:0	TMR3L[7:0]	**Timer 3 Low Byte.** In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value.

SFR Definition 31.18. TMR3H Timer 3 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x95; SFR Page = 0

Bit	Name	Function
7:0	TMR3H[7:0]	**Timer 3 High Byte.** In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8-bit mode, TMR3H contains the 8-bit high byte timer value.

31.4. Timer 4

Timer 4 is a 16-bit timer formed by two 8-bit SFRs: TMR4L (low byte) and TMR4H (high byte). Timer 4 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T4SPLIT bit (TMR4CN.3) defines

Timer 4 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. Note that the external oscillator source divided by 8 is synchronized with the system clock.

31.4.1. 16-bit Timer with Auto-Reload

When T4SPLIT (TMR4CN.3) is zero, Timer 4 operates as a 16-bit timer with auto-reload. Timer 4 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 4 reload registers (TMR4RLH and TMR4RLL) is loaded into the Timer 4 register as shown in Figure 31.10, and the Timer 4 High Byte Overflow Flag (TMR4CN.7) is set. If Timer 4 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 4 overflow. Additionally, if Timer 4 interrupts are enabled and the TF4LEN bit is set (TMR4CN.5), an interrupt will be generated each time the lower 8 bits (TMR4L) overflow from 0xFF to 0x00.

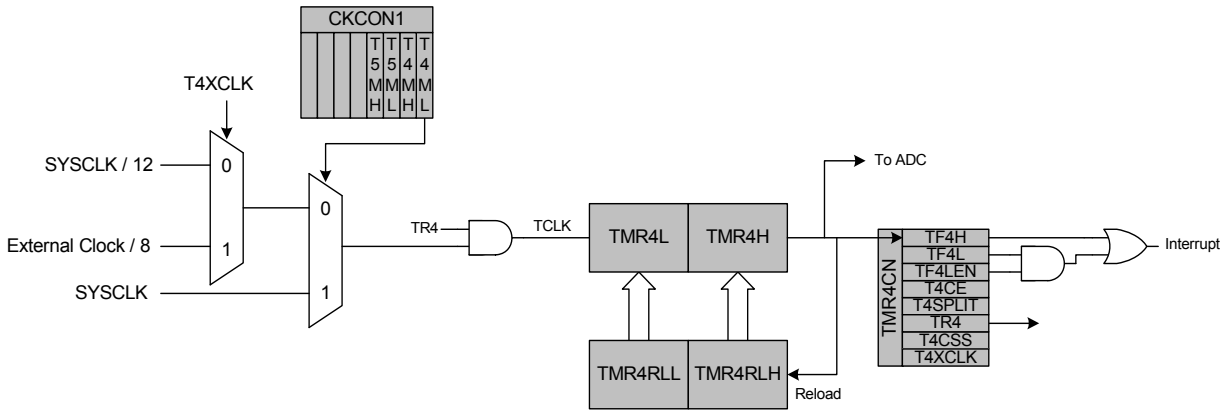


Figure 31.10. Timer 4 16-Bit Mode Block Diagram

31.4.2. 8-bit Timers with Auto-Reload

When T4SPLIT is 1 and T4CE = 0, Timer 4 operates as two 8-bit timers (TMR4H and TMR4L). Both 8-bit timers operate in auto-reload mode as shown in Figure 31.11. TMR4RLL holds the reload value for TMR4L; TMR4RLH holds the reload value for TMR4H. The TR4 bit in TMR4CN handles the run control for TMR4H. TMR4L 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 4 Clock Select bits (T4MH and T4ML in CKCON1) select either SYSCLK or the clock defined by the Timer 4 External Clock Select bit (T4XCLK in TMR4CN), as follows:

T4MH	T4XCLK	TMR4H Clock Source
0	0	SYSCLK/12
0	1	External Clock/8
1	X	SYSCLK

T4ML	T4XCLK	TMR4L Clock Source
0	0	SYSCLK/12
0	1	External Clock/8
1	X	SYSCLK

The TF4H bit is set when TMR4H overflows from 0xFF to 0x00; the TF4L bit is set when TMR4L overflows from 0xFF to 0x00. When Timer 4 interrupts are enabled, an interrupt is generated each time TMR4H overflows. If Timer 4 interrupts are enabled and TF4LEN (TMR4CN.5) is set, an interrupt is generated each time either TMR4L or TMR4H overflows. When TF4LEN is enabled, software must check the TF4H and TF4L flags to determine the source of the Timer 4 interrupt. The TF4H and TF4L interrupt flags are not cleared by hardware and must be manually cleared by software.

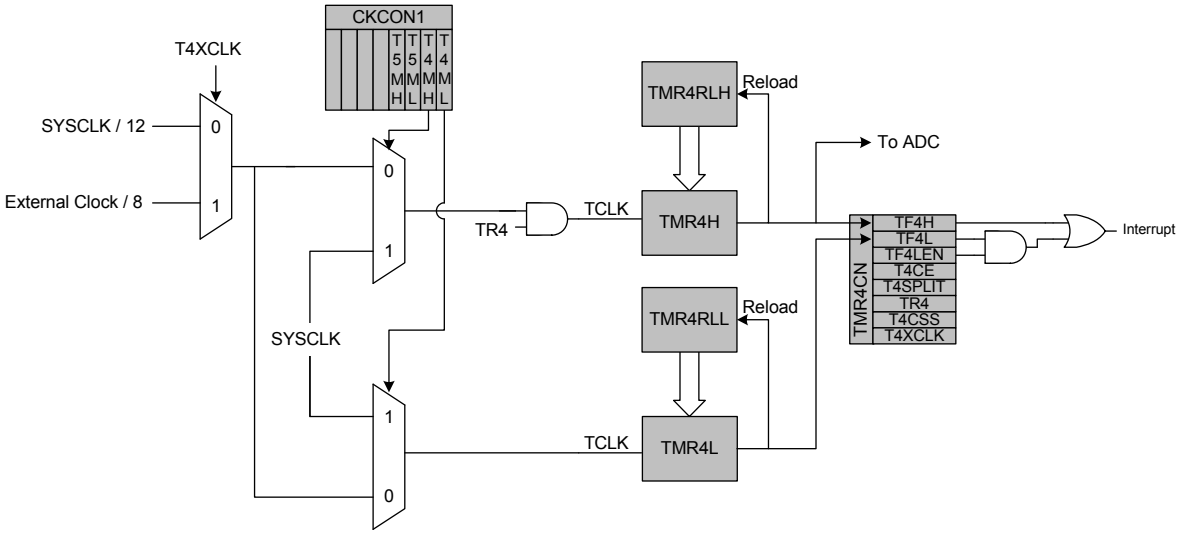


Figure 31.11. Timer 4 8-Bit Mode Block Diagram

SFR Definition 31.19. TMR4CN: Timer 4 Control

Bit	7	6	5	4	3	2	1	0
Name	TF4H	TF4L	TF4LEN		T4SPLIT	TR4		T4XCLK
Type	R/W	R/W	R/W	R	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x91; SFR Page = F

Bit	Name	Function
7	TF4H	**Timer 4 High Byte Overflow Flag.** Set by hardware when the Timer 4 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 4 overflows from 0xFFFF to 0x0000. When the Timer 4 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 4 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF4L	**Timer 4 Low Byte Overflow Flag.** Set by hardware when the Timer 4 low byte overflows from 0xFF to 0x00. TF4L will be set when the low byte overflows regardless of the Timer 4 mode. This bit is not automatically cleared by hardware.
5	TF4LEN	**Timer 4 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 4 Low Byte interrupts. If Timer 4 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 4 overflows.
4	Unused	Read = 0b; Write = don't care.
3	T4SPLIT	**Timer 4 Split Mode Enable.** When this bit is set, Timer 4 operates as two 8-bit timers with auto-reload. 0: Timer 4 operates in 16-bit auto-reload mode. 1: Timer 4 operates as two 8-bit auto-reload timers.
2	TR4	**Timer 4 Run Control.** Timer 4 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR4H only; TMR4L is always enabled in split mode.
1	Unused	Read = 0b; Write = don't care.
0	T4XCLK	**Timer 4 External Clock Select.** This bit selects the external clock source for Timer 4. However, the Timer 4 Clock Select bits (T4MH and T4ML in register CKCON1) may still be used to select between the external clock and the system clock for either timer. 0: Timer 4 clock is the system clock divided by 12. 1: Timer 4 clock is the external clock divided by 8 (synchronized with SYSCLK).

SFR Definition 31.20. TMR4RLL: Timer 4 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x92; SFR Page = F

Bit	Name	Function
7:0	TMR4RLL[7:0]	**Timer 4 Reload Register Low Byte.** TMR4RLL holds the low byte of the reload value for Timer 4.

SFR Definition 31.21. TMR4RLH: Timer 4 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x93; SFR Page = F

Bit	Name	Function
7:0	TMR4RLH[7:0]	**Timer 4 Reload Register High Byte.** TMR4RLH holds the high byte of the reload value for Timer 4.

SFR Definition 31.22. TMR4L: Timer 4 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x94; SFR Page = F

Bit	Name	Function
7:0	TMR4L[7:0]	**Timer 4 Low Byte.** In 16-bit mode, the TMR4L register contains the low byte of the 16-bit Timer 4. In 8-bit mode, TMR4L contains the 8-bit low byte timer value.

SFR Definition 31.23. TMR4H Timer 4 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR4H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x95; SFR Page = F

Bit	Name	Function
7:0	TMR4H[7:0]	**Timer 4 High Byte.** In 16-bit mode, the TMR4H register contains the high byte of the 16-bit Timer 4. In 8-bit mode, TMR4H contains the 8-bit high byte timer value.

31.5. Timer 5

Timer 5 is a 16-bit timer formed by two 8-bit SFRs: TMR5L (low byte) and TMR5H (high byte). Timer 5 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T5SPLIT bit (TMR5CN.3) defines

Timer 5 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. Note that the external oscillator source divided by 8 is synchronized with the system clock.

31.5.1. 16-bit Timer with Auto-Reload

When T5SPLIT (TMR5CN.3) is zero, Timer 5 operates as a 16-bit timer with auto-reload. Timer 5 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 5 reload registers (TMR5RLH and TMR5RLL) is loaded into the Timer 5 register as shown in Figure 31.12, and the Timer 5 High Byte Overflow Flag (TMR5CN.7) is set. If Timer 5 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 5 overflow. Additionally, if Timer 5 interrupts are enabled and the TF5LEN bit is set (TMR5CN.5), an interrupt will be generated each time the lower 8 bits (TMR5L) overflow from 0xFF to 0x00.

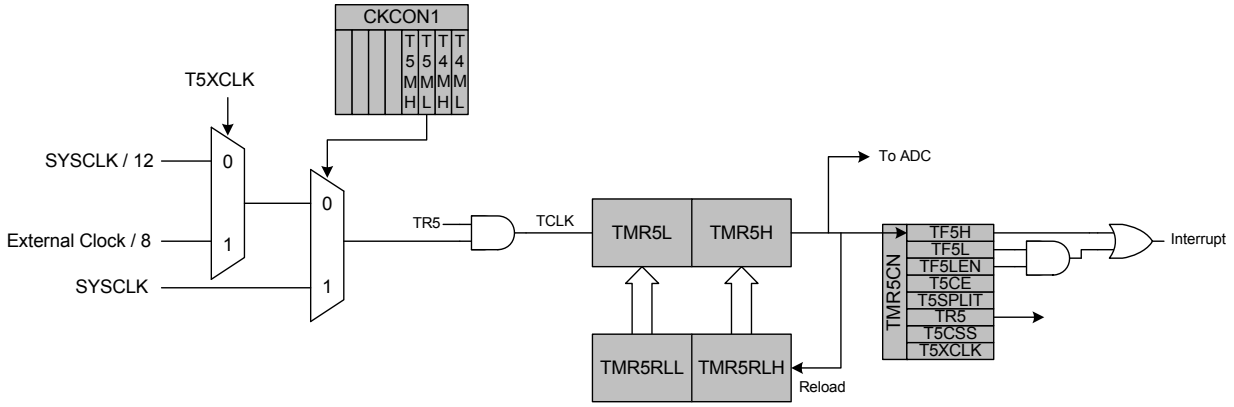


Figure 31.12. Timer 5 16-Bit Mode Block Diagram

31.5.2. 8-bit Timers with Auto-Reload

When T5SPLIT is 1 and T5CE = 0, Timer 5 operates as two 8-bit timers (TMR5H and TMR5L). Both 8-bit timers operate in auto-reload mode as shown in Figure 31.13. TMR5RLL holds the reload value for TMR5L; TMR5RLH holds the reload value for TMR5H. The TR5 bit in TMR5CN handles the run control for TMR5H. TMR5L 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 5 Clock Select bits (T5MH and T5ML in CKCON1) select either SYSCLK or the clock defined by the Timer 5 External Clock Select bit (T5XCLK in TMR5CN), as follows:

T5MH	T5XCLK	TMR5H Clock Source
0	0	SYSCLK/12
0	1	External Clock/8
1	X	SYSCLK

T5ML	T5XCLK	TMR5L Clock Source
0	0	SYSCLK/12
0	1	External Clock/8
1	X	SYSCLK

The TF5H bit is set when TMR5H overflows from 0xFF to 0x00; the TF5L bit is set when TMR5L overflows from 0xFF to 0x00. When Timer 5 interrupts are enabled, an interrupt is generated each time TMR5H overflows. If Timer 5 interrupts are enabled and TF5LEN (TMR5CN.5) is set, an interrupt is generated each time either TMR5L or TMR5H overflows. When TF5LEN is enabled, software must check the TF5H and TF5L flags to determine the source of the Timer 5 interrupt. The TF5H and TF5L interrupt flags are not cleared by hardware and must be manually cleared by software.

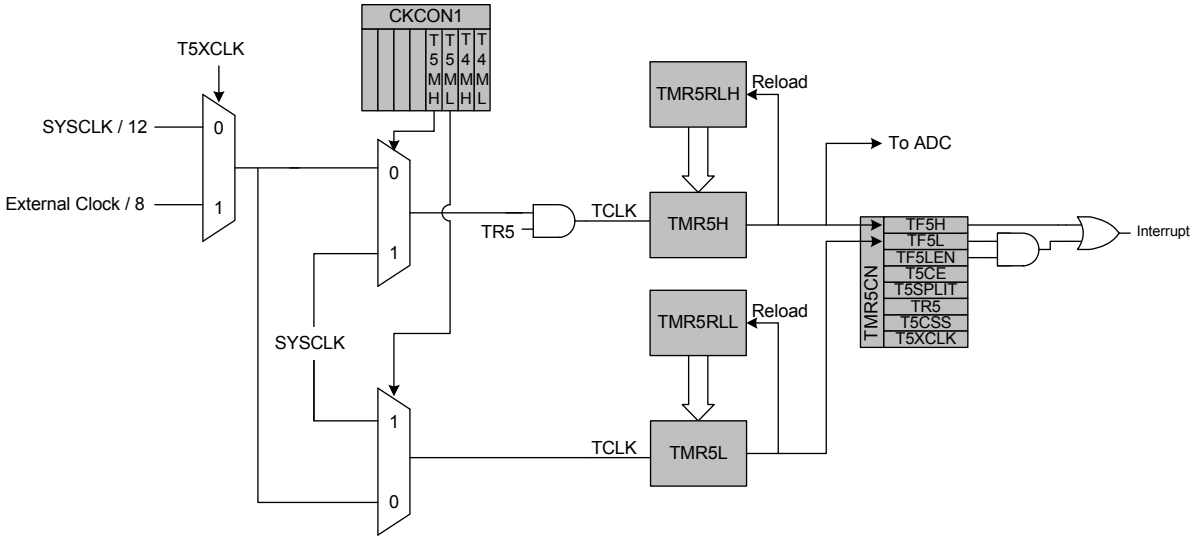


Figure 31.13. Timer 5 8-Bit Mode Block Diagram

SFR Definition 31.24. TMR5CN: Timer 5 Control

Bit	7	6	5	4	3	2	1	0
Name	TF5H	TF5L	TF5LEN		T5SPLIT	TR5		T5XCLK
Type	R/W	R/W	R/W	R	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

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

Bit	Name	Function
7	TF5H	**Timer 5 High Byte Overflow Flag.** Set by hardware when the Timer 5 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 5 overflows from 0xFFFF to 0x0000. When the Timer 5 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 5 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF5L	**Timer 5 Low Byte Overflow Flag.** Set by hardware when the Timer 5 low byte overflows from 0xFF to 0x00. TF5L will be set when the low byte overflows regardless of the Timer 5 mode. This bit is not automatically cleared by hardware.
5	TF5LEN	**Timer 5 Low Byte Interrupt Enable.** When set to 1, this bit enables Timer 5 Low Byte interrupts. If Timer 5 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 5 overflows.
4	Unused	Read = 0b; Write = don't care.
3	T5SPLIT	**Timer 5 Split Mode Enable.** When this bit is set, Timer 5 operates as two 8-bit timers with auto-reload. 0: Timer 5 operates in 16-bit auto-reload mode. 1: Timer 5 operates as two 8-bit auto-reload timers.
2	TR5	**Timer 5 Run Control.** Timer 5 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR5H only; TMR5L is always enabled in split mode.
1	Unused	Read = 0b; Write = don't care.
0	T5XCLK	**Timer 5 External Clock Select.** This bit selects the external clock source for Timer 5. However, the Timer 5 Clock Select bits (T5MH and T5ML in register CKCON1) may still be used to select between the external clock and the system clock for either timer. 0: Timer 5 clock is the system clock divided by 12. 1: Timer 5 clock is the external clock divided by 8 (synchronized with SYSCLK).

SFR Definition 31.25. TMR5RLL: Timer 5 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCA; SFR Page = F

Bit	Name	Function
7:0	TMR5RLL[7:0]	**Timer 5 Reload Register Low Byte.** TMR5RLL holds the low byte of the reload value for Timer 5.

SFR Definition 31.26. TMR5RLH: Timer 5 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCB; SFR Page = F

Bit	Name	Function
7:0	TMR5RLH[7:0]	**Timer 5 Reload Register High Byte.** TMR5RLH holds the high byte of the reload value for Timer 5.

SFR Definition 31.27. TMR5L: Timer 5 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCC; SFR Page = F

Bit	Name	Function
7:0	TMR5L[7:0]	**Timer 5 Low Byte.** In 16-bit mode, the TMR5L register contains the low byte of the 16-bit Timer 5. In 8-bit mode, TMR5L contains the 8-bit low byte timer value.

SFR Definition 31.28. TMR5H Timer 5 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR5H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCD; SFR Page = F

Bit	Name	Function
7:0	TMR5H[7:0]	**Timer 5 High Byte.** In 16-bit mode, the TMR5H register contains the high byte of the 16-bit Timer 5. In 8-bit mode, TMR5H contains the 8-bit high byte timer value.

32. 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. The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, low frequency oscillator divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section “32.3. Capture/Compare Modules” on page 278). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 32.1

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 32.4 for details.

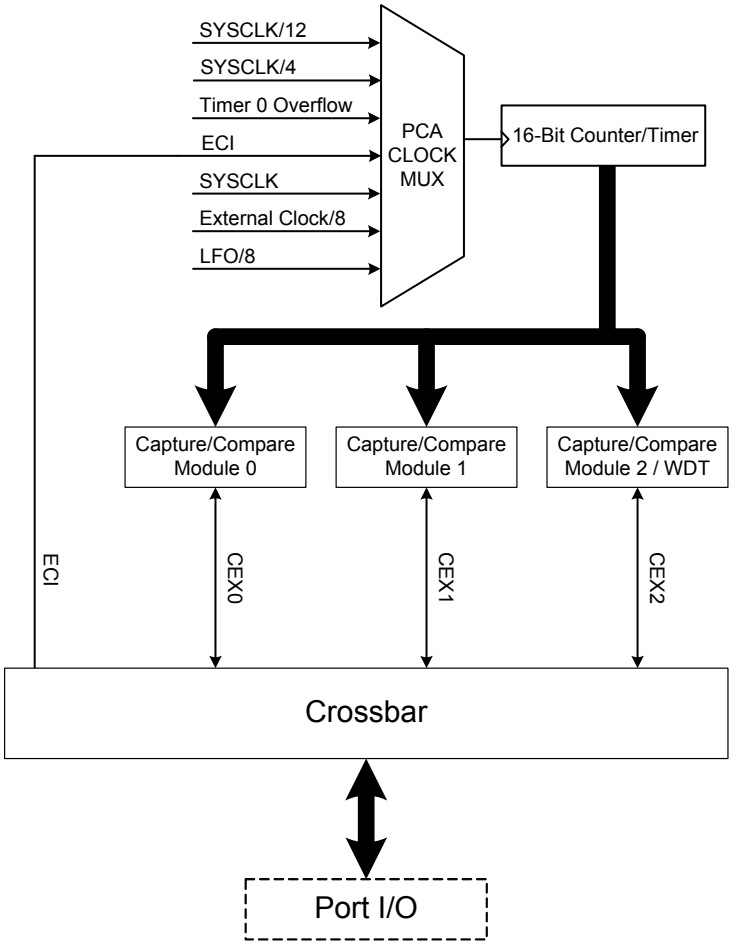


Figure 32.1. PCA Block Diagram

32.1. 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 32.1.

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. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

Table 32.1. 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*
1	1	0	Low frequency oscillator divided by 8*
1	1	1	Reserved
Note: Synchronized with the system clock.			

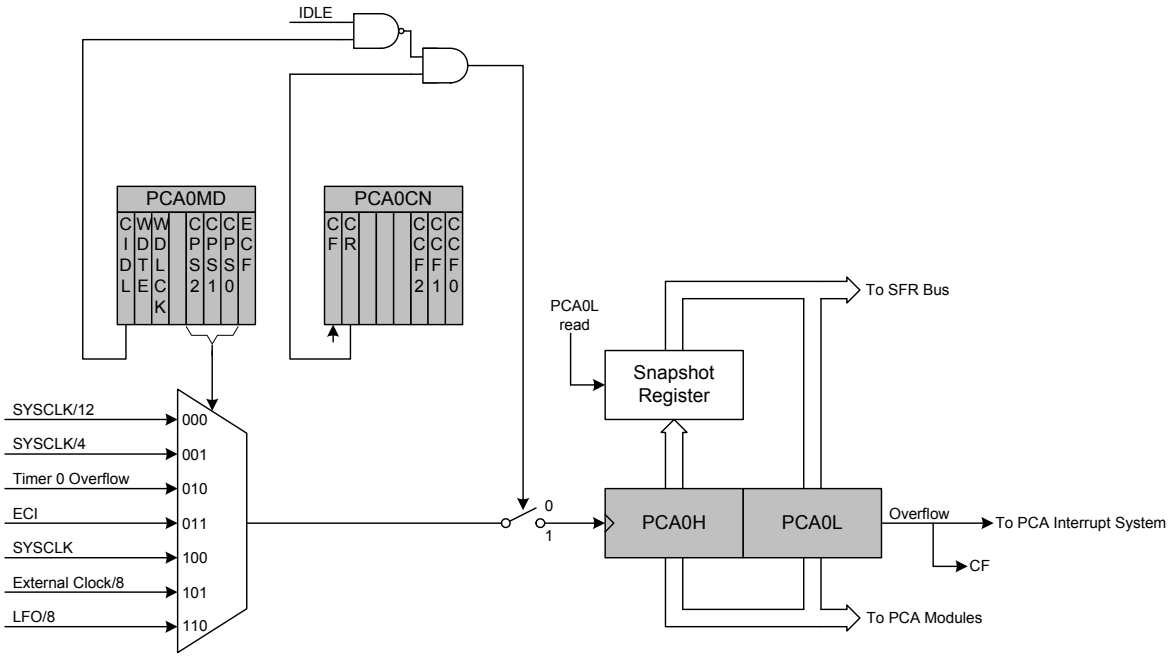


Figure 32.2. PCA Counter/Timer Block Diagram

32.2. PCA0 Interrupt Sources

Figure 32.3 shows a diagram of the PCA interrupt tree. There are five independent event flags that can be used to generate a PCA0 interrupt. They are: the main PCA counter overflow flag (CF), which is set upon a 16-bit overflow of the PCA0 counter, an intermediate overflow flag (COVF), which can be set on an overflow from the 8th, 9th, 10th, or 11th bit of the PCA0 counter, and the individual flags for each PCA channel (CCF0, CCF1, and CCF2), which are set according to the operation mode of that module. These event flags are always set when the trigger condition occurs. Each of these flags can be individually selected to generate a PCA0 interrupt, using the corresponding interrupt enable flag (ECF for CF, ECOV for COVF, and ECCFn for each CCFn). PCA0 interrupts must be globally enabled before any individual interrupt sources are recognized by the processor. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1.

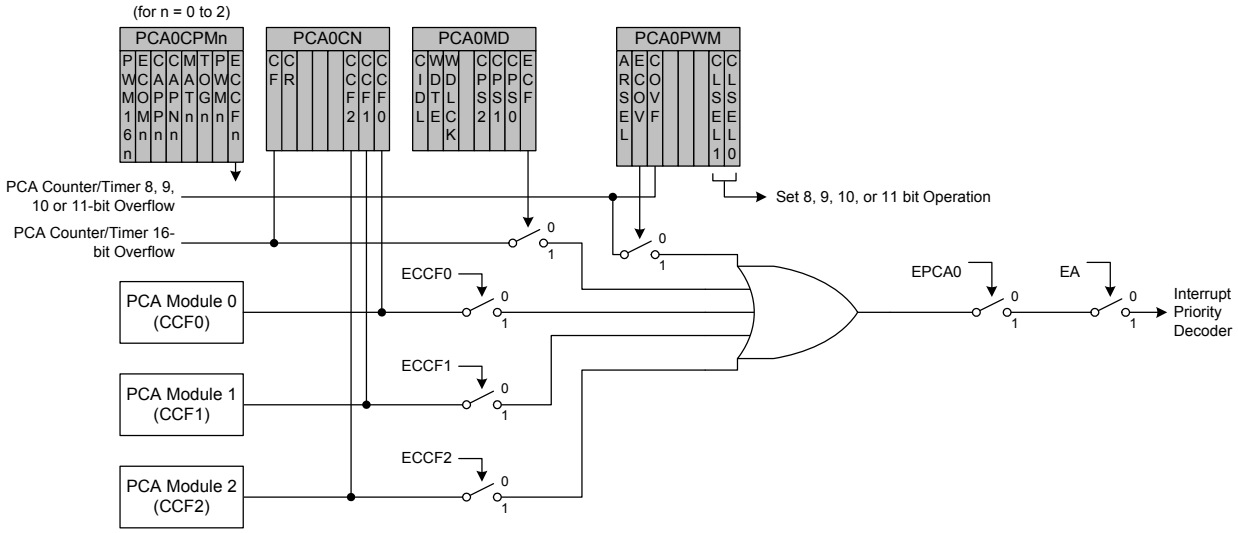


Figure 32.3. PCA Interrupt Block Diagram

32.3. 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 to 11-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 32.2 summarizes the bit settings in the PCA0CPMn and PCA0PWM registers used to select the PCA capture/compare module's operating mode. Note that all modules set to use 8, 9, 10, or 11-bit PWM mode must use the same cycle length (8–11 bits). Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt.

Table 32.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules

Operational Mode	PCA0CPMn								PCA0PWM				
Bit Number	7	6	5	4	3	2	1	0	7	6	5	4–2	1–0
Capture triggered by positive edge on CEXn	X	X	1	0	0	0	0	A	0	X	B	XXX	XX
Capture triggered by negative edge on CEXn	X	X	0	1	0	0	0	A	0	X	B	XXX	XX
Capture triggered by any transition on CEXn	X	X	1	1	0	0	0	A	0	X	B	XXX	XX
Software Timer	X	C	0	0	1	0	0	A	0	X	B	XXX	XX
High Speed Output	X	C	0	0	1	1	0	A	0	X	B	XXX	XX
Frequency Output	X	C	0	0	0	1	1	A	0	X	B	XXX	XX
8-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	0	X	B	XXX	00
9-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	D	X	B	XXX	01
10-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	D	X	B	XXX	10
11-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	D	X	B	XXX	11
16-Bit Pulse Width Modulator	1	C	0	0	E	0	1	A	0	X	B	XXX	XX

Notes:

1. X = Don't Care (no functional difference for individual module if 1 or 0).
2. A = Enable interrupts for this module (PCA interrupt triggered on CCFn set to 1).
3. B = Enable 8th, 9th, 10th or 11th bit overflow interrupt (Depends on setting of CLSEL[1:0]).
4. C = When set to 0, the digital comparator is off. For high speed and frequency output modes, the associated pin will not toggle. In any of the PWM modes, this generates a 0% duty cycle (output = 0).
5. D = Selects whether the Capture/Compare register (0) or the Auto-Reload register (1) for the associated channel is accessed via addresses PCA0CPHn and PCA0CPLn.
6. E = When set, a match event will cause the CCFn flag for the associated channel to be set.
7. All modules set to 8, 9, 10 or 11-bit PWM mode use the same cycle length setting.

32.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. 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.

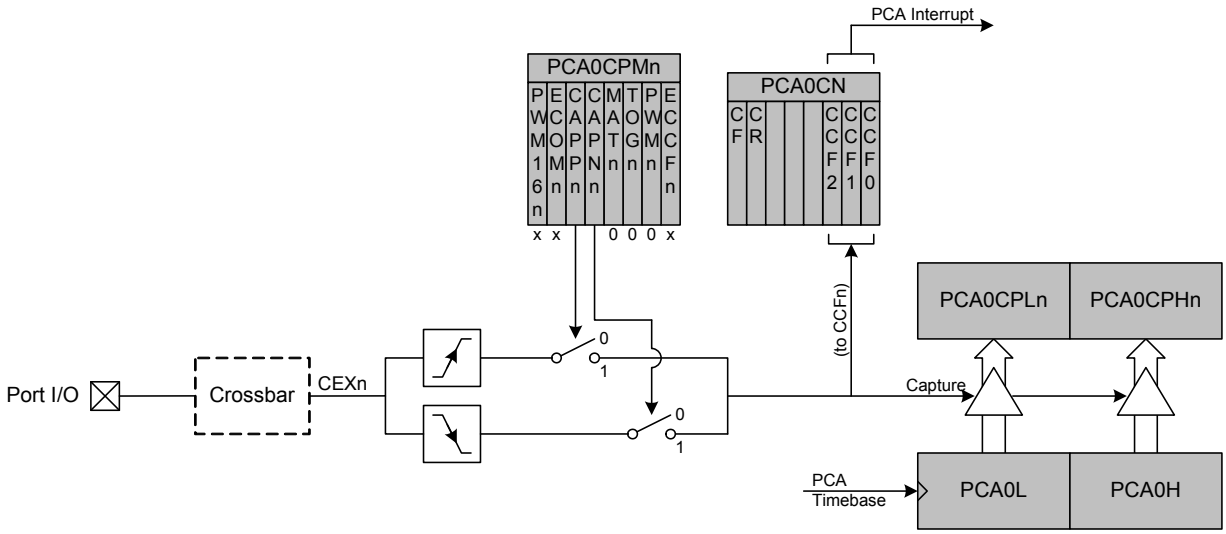


Figure 32.4. PCA Capture Mode Diagram

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

32.3.2. 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. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the 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.

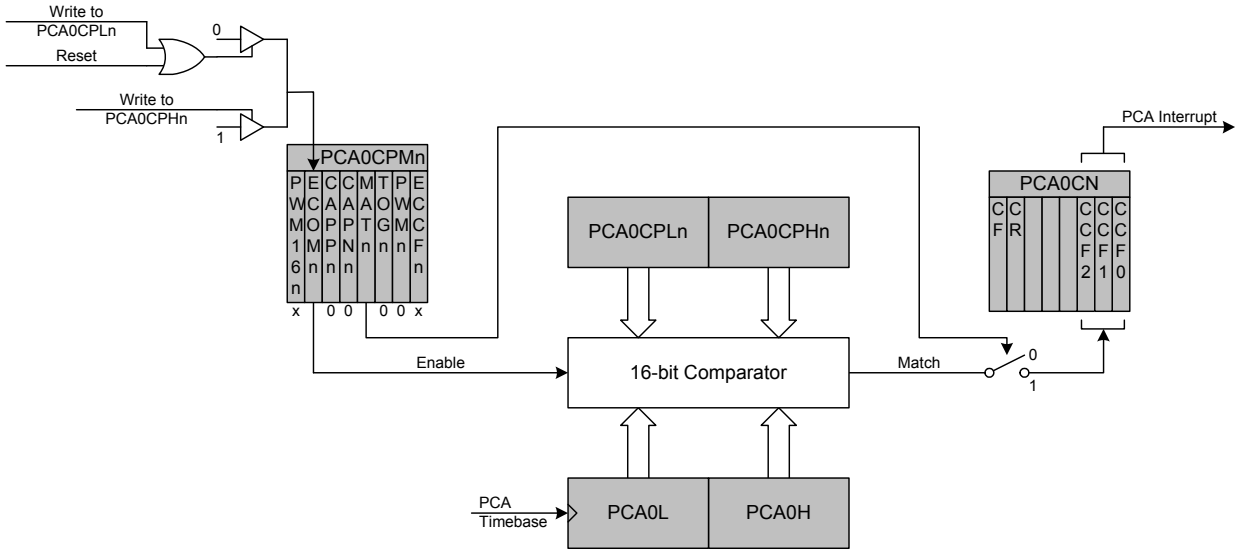


Figure 32.5. PCA Software Timer Mode Diagram

32.3.3. High-Speed Output Mode

In High-Speed Output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

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

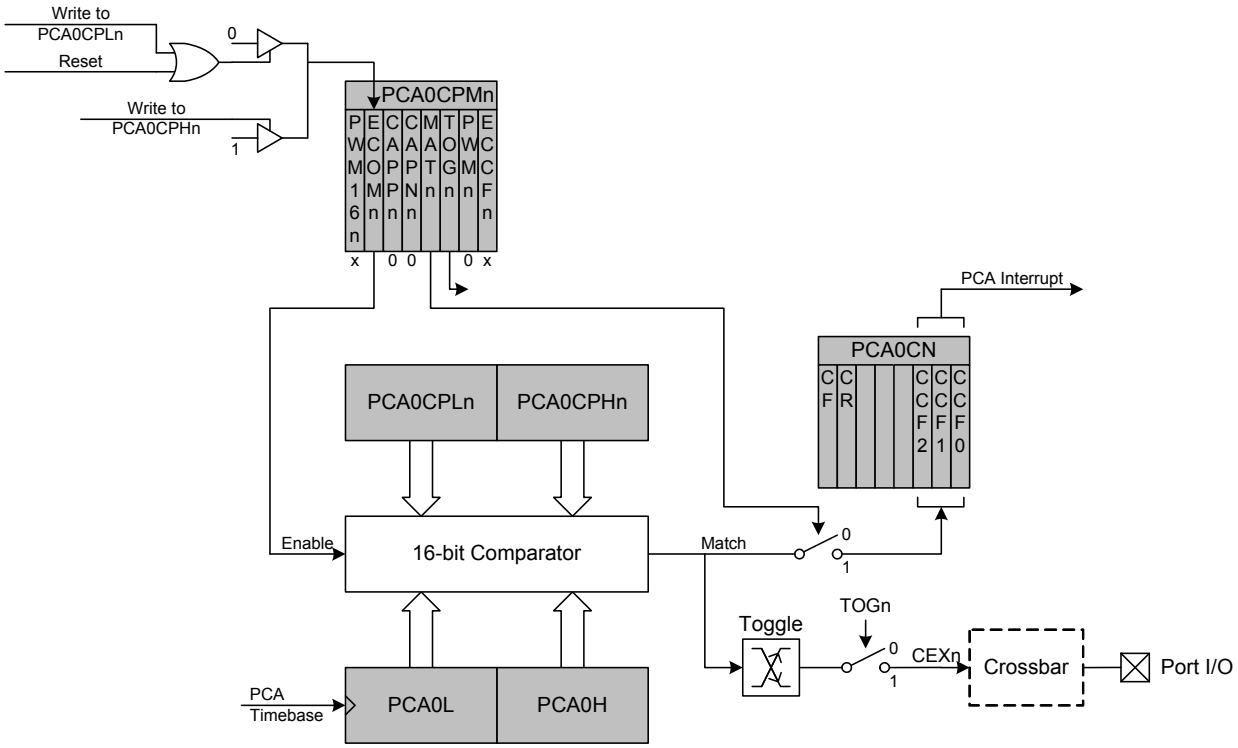


Figure 32.6. PCA High-Speed Output Mode Diagram

32.3.4. 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 32.1.

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

Note: A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

Equation 32.1. 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, n 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. Note that the MATn bit should normally be set to 0 in this mode. If the MATn bit is set to 1, the CCFn flag for the channel will be set when the 16-bit PCA0 counter and the 16-bit capture/compare register for the channel are equal.

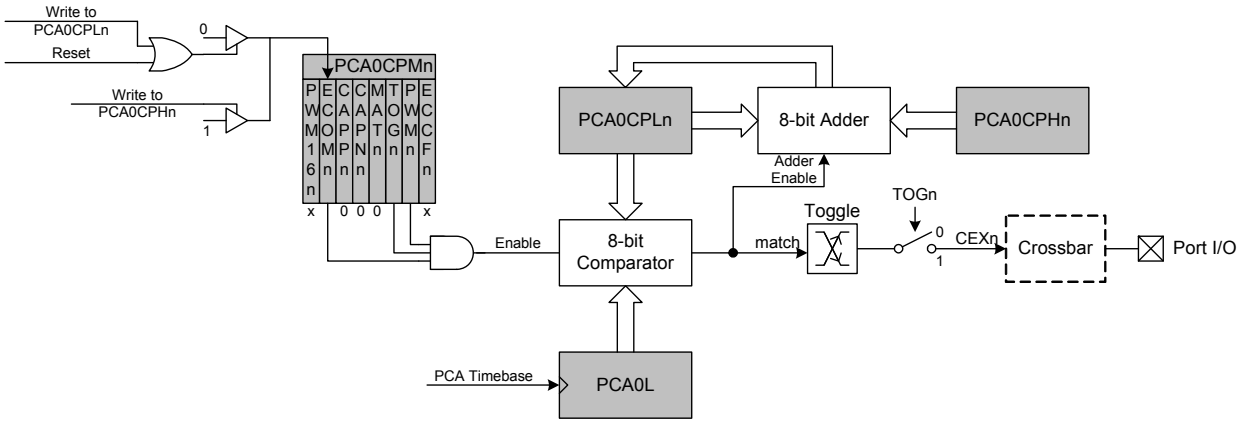


Figure 32.7. PCA Frequency Output Mode

32.3.5. 8-bit, 9-bit, 10-bit and 11-bit Pulse Width Modulator Modes

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, and the setting of the PWM cycle length (8, 9, 10 or 11-bits). For backwards-compatibility with the 8-bit PWM mode available on other devices, the 8-bit PWM mode operates slightly different than 9, 10 and 11-bit PWM modes. **It is important to note that all channels configured for 8/9/10/11-bit PWM mode will use the same cycle length.** It is not possible to configure one channel for 8-bit PWM mode and another for 11-bit mode (for example). However, other PCA channels can be configured to Pin Capture, High-Speed Output, Software Timer, Frequency Output, or 16-bit PWM mode independently.

32.3.5.1. 8-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 8-bit PWM mode 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. When the count value in PCA0L overflows, the CEXn output will be reset (see Figure 32.8). 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, and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit Pulse Width Modulator mode. If the MATn bit is set to 1, the CCFn flag for the module will be set each time an 8-bit comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 256 PCA clock cycles. The duty cycle for 8-Bit PWM Mode is given in Equation 32.2.

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.

$$\text{Duty Cycle} = \frac{(256 - PCA0CPHn)}{256}$$

Equation 32.2. 8-Bit PWM Duty Cycle

Using Equation 32.2, 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.

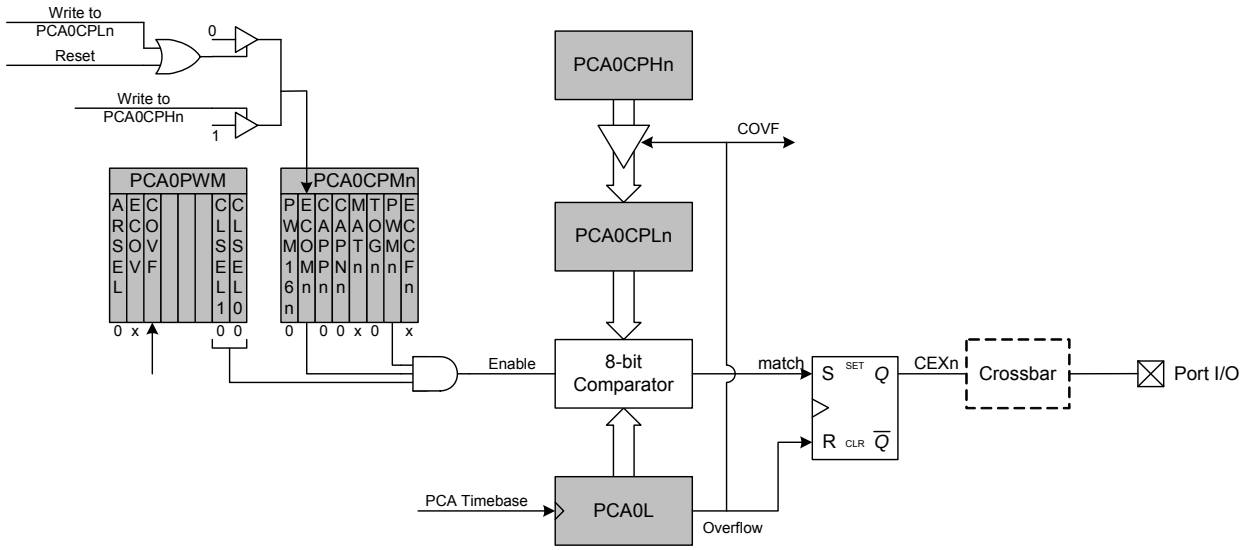


Figure 32.8. PCA 8-Bit PWM Mode Diagram

32.3.5.2. 9/10/11-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 9/10/11-bit PWM mode should be varied by writing to an "Auto-Reload" Register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The data written to define the duty cycle should be right-justified in the registers. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0.

When the least-significant N bits of the PCA0 counter match the value in the associated module's capture/compare register (PCA0CPn), the output on CEXn is asserted high. When the counter overflows from the Nth bit, CEXn is asserted low (see Figure 32.9). Upon an overflow from the Nth bit, the COVF flag is set, and the value stored in the module's auto-reload register is loaded into the capture/compare register. The value of N is determined by the CLSEL bits in register PCA0PWM.

The 9, 10 or 11-bit PWM mode is selected by setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to the desired cycle length (other than 8-bits). If the MATn bit is set to 1, the CCFn flag for the module will be set each time a comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 512 (9-bit), 1024 (10-bit) or 2048 (11-bit) PCA clock cycles. The duty cycle for 9/10/11-Bit PWM Mode is given in Equation 32.2, where N is the number of bits in the PWM cycle.

Important Note About PCA0CPHn and PCA0CPLn Registers: When writing a 16-bit value to the PCA0CPn registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(2^N - PCA0CPn)}{2^N}$$

Equation 32.3. 9, 10, and 11-Bit PWM Duty Cycle

A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

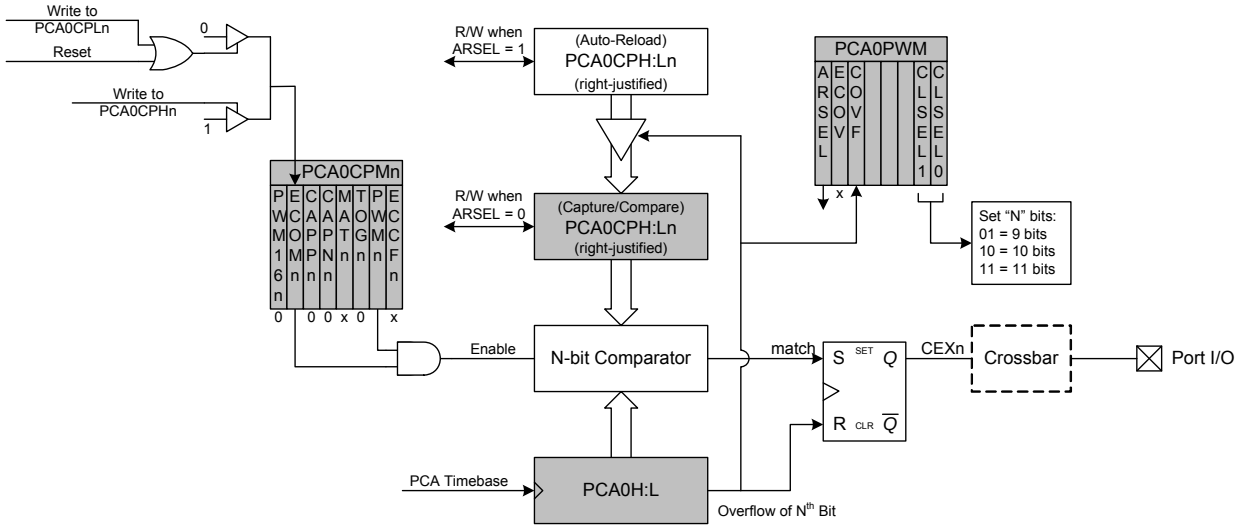


Figure 32.9. PCA 9, 10 and 11-Bit PWM Mode Diagram

32.3.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8/9/10/11-bit) PWM modes. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is asserted high; when the 16-bit counter overflows, CEXn is asserted low. 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. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a 16-bit comparator match (rising edge) occurs. The CF flag in PCA0CN can be used to detect the overflow (falling edge). The duty cycle for 16-Bit PWM Mode is given by Equation 32.4.

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

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

Equation 32.4. 16-Bit PWM Duty Cycle

Using Equation 32.4, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

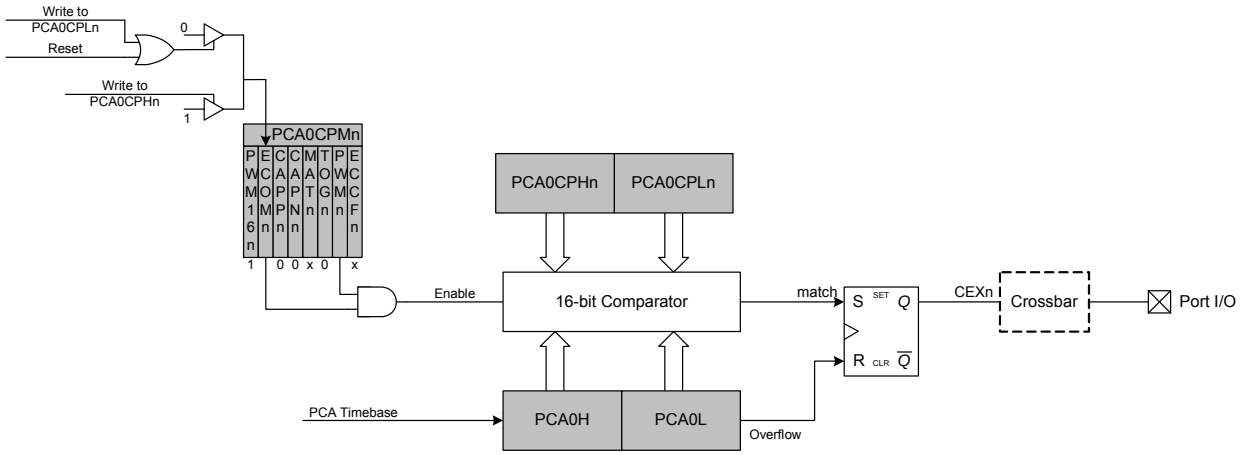


Figure 32.10. PCA 16-Bit PWM Mode

32.4. 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 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.** The WDT will generate a reset shortly after code begins execution. To avoid this reset, the WDT should be explicitly disabled (and optionally re-configured and re-enabled if it is used in the system).

32.4.1. 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 bit (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 32.11).

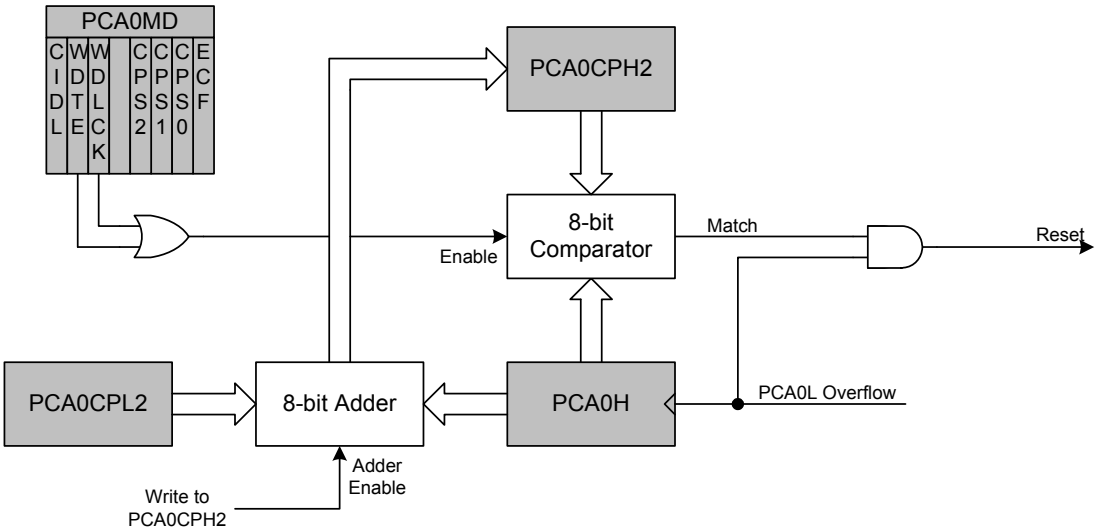


Figure 32.11. PCA Module 2 with Watchdog Timer Enabled

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 32.5, where PCA0L is the value of the PCA0L register at the time of the update.

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

Equation 32.5. 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.

32.4.2. Watchdog Timer Usage

To configure the WDT, perform the following tasks:

1. Disable the WDT by writing a 0 to the WDTE bit.
2. Select the desired PCA clock source (with the CPS2–CPS0 bits).
3. Load PCA0CPL2 with the desired WDT update offset value.
4. Configure the PCA Idle mode (set CIDL if the WDT should be suspended while the CPU is in Idle mode).
5. Enable the WDT by setting the WDTE bit to 1.
6. Reset the WDT timer by writing 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 32.5, this results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. Table 32.3 lists some example timeout intervals for typical system clocks.

Table 32.3. 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
3,062,500[2]	255	257
3,062,500[2]	128	129.5
3,062,500[2]	32	33.1
32,000	255	24576
32,000	128	12384
32,000	32	3168

Notes:
1. Assumes SYSCLK/12 as the PCA clock source, and a PCA0L value of 0x00 at the update time.
2. Internal SYSCLK reset frequency = Internal Oscillator divided by 8.

32.5. Comparator Clear Function

In 8/9/10/11/16-bit PWM modes, the comparator clear function utilizes the Comparator0 output synchronized to the system clock to clear CEXn to logic low for the current PWM cycle. This comparator clear function can be enabled for each PWM channel by setting the CPCEn bits to 1 in the PCA0CLR SFR (see SFR Definition 32.4). When the comparator clear function is disabled, CEXn is unaffected. See Figure 32.12.

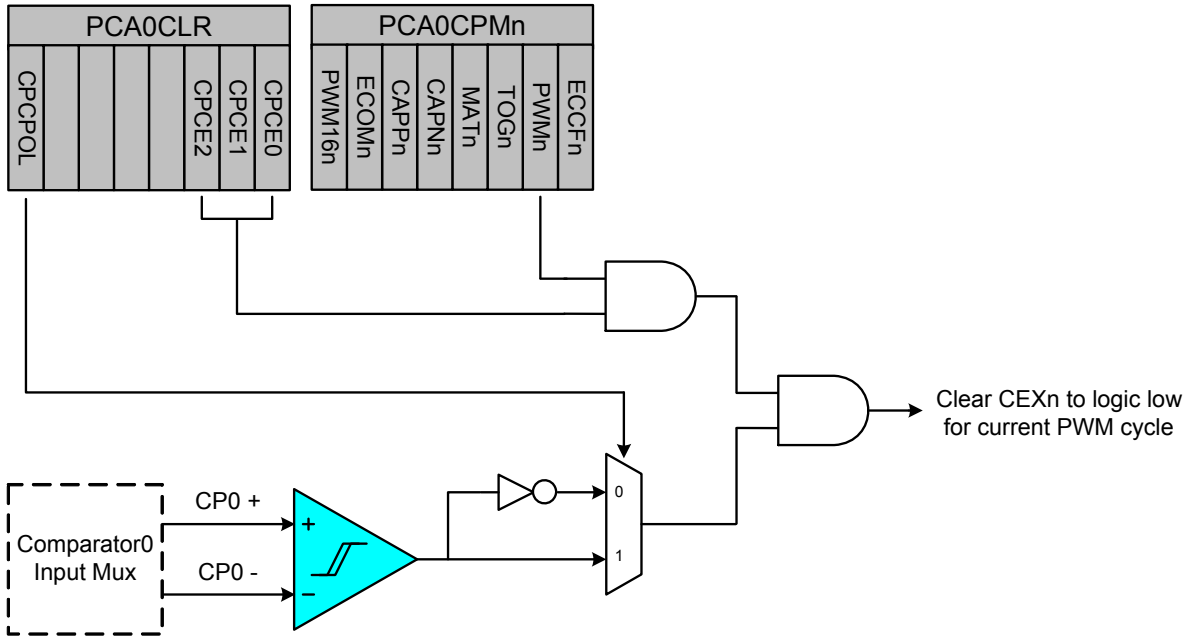


Figure 32.12. Comparator Clear Function Diagram

The asynchronous Comparator0 output is logic high when the voltage of CP0+ is greater than CP0- and logic low when the voltage of CP0+ is less than CP0-. The polarity of the Comparator0 output is used to clear CEXn as follows: when CPCPOL = 0, CEXn is forced to logic low on the falling edge of the Comparator0 output (see Figure 32.13); when CPCPOL = 0, CEXn is forced logic low on the rising edge of the Compartor0 output (see Figure 32.14).

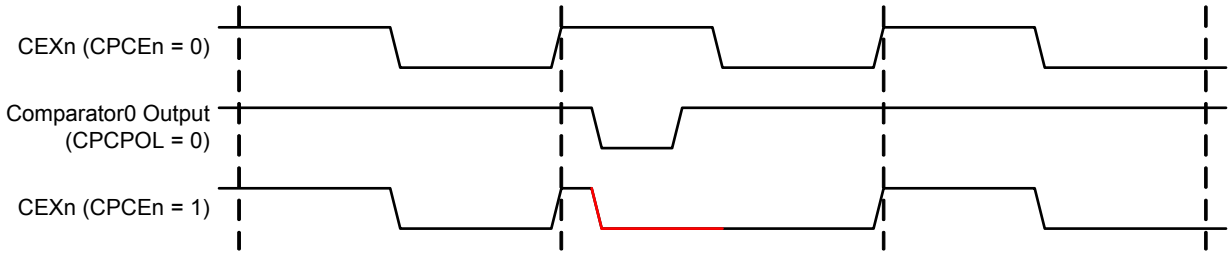


Figure 32.13. CEXn with CPCEn = 1, CPCPOL = 0

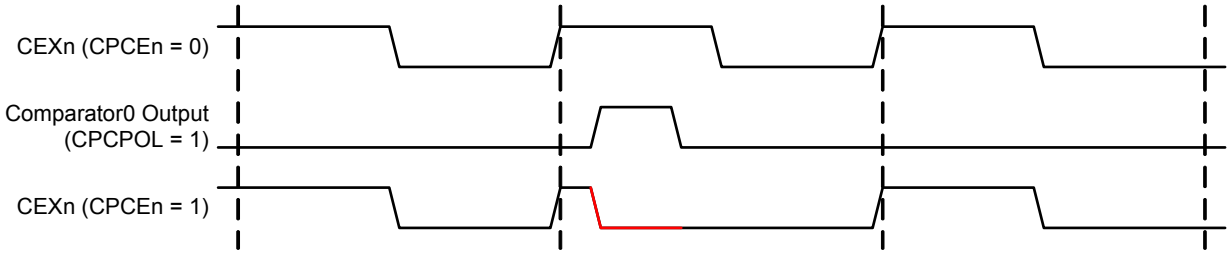


Figure 32.14. CEXn with CPCEn = 1, CPCPOL = 1

In the PWM cycle following the current cycle, should the Comparator0 output remain logic low when CPCPOL = 0 or logic high when CPCPOL = 1, CEXn will continue to be logic low. See Figure 32.15 and Figure 32.16.

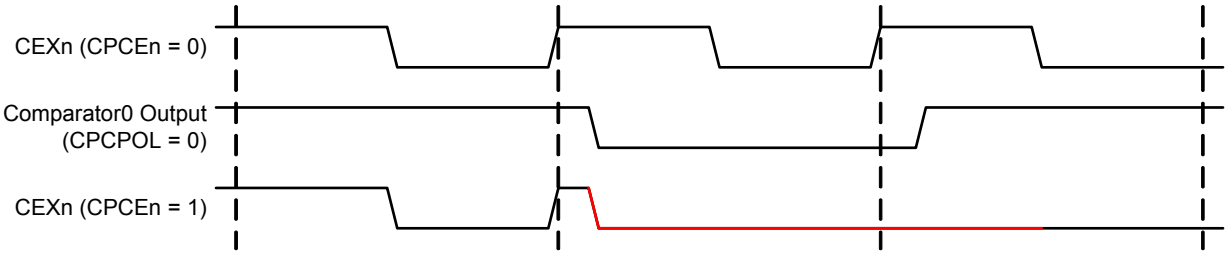


Figure 32.15. CEXn with CPCEn = 1, CPCPOL = 0

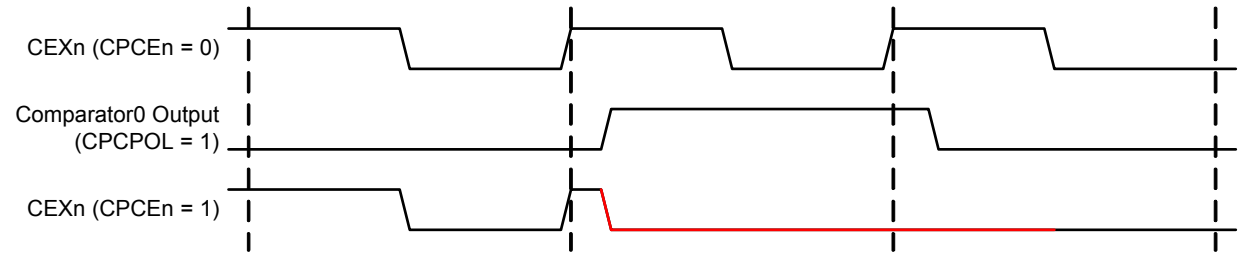


Figure 32.16. CEXn with CPCEn = 1, CPCPOL = 1

32.6. Register Descriptions for PCA0

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

SFR Definition 32.1. PCA0CN: PCA Control

Bit	7	6	5	4	3	2	1	0
Name	CF	CR				CCF2	CCF1	CCF0
Type	R/W	R/W	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	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.
6	CR	**PCA Counter/Timer Run Control.** This bit enables/disables the PCA Counter/Timer. 0: PCA Counter/Timer disabled. 1: PCA Counter/Timer enabled.
5:3	Unused	Unused. Read = 000b, Write = Don't care.
2	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.
1	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.
0	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 32.2. PCA0MD: PCA Mode

Bit	7	6	5	4	3	2	1	0
Name	CIDL	WDTE	WDLCK		CPS2	CPS1	CPS0	ECF
Type	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Reset	0	1	0	0	0	0	0	0

SFR Address = 0xD9; SFR Page = All Pages

Bit	Name	Function
7	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.
6	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.
5	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.
4	Unused	Unused. Read = 0b, Write = Don't care.
3:1	CPS[2:0]	**PCA Counter/Timer Pulse Select.** These bits select the timebase source for the PCA counter 000: System clock divided by 12 001: System clock divided by 4 010: Timer 0 overflow 011: High-to-low transitions on ECI (max rate = system clock divided by 4) 100: System clock 101: External clock divided by 8 (synchronized with the system clock) 110: Low frequency oscillator divided by 8 111: Reserved
0	ECF	**PCA Counter/Timer Overflow Interrupt Enable.** This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt. 0: Disable the CF interrupt. 1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set.
Note: When the WDTE bit is set to 1, the other bits in the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled.		

SFR Definition 32.3. PCA0PWM: PCA PWM Configuration

Bit	7	6	5	4	3	2	1	0
Name	ARSEL	ECOV	COVF				CLSEL[1:0]	
Type	R/W	R/W	R/W	R	R	R	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF7; SFR Page = All Pages

Bit	Name	Function
7	ARSEL	**Auto-Reload Register Select.** This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function. 0: Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn. 1: Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn.
6	ECOV	**Cycle Overflow Interrupt Enable.** This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt. 0: COVF will not generate PCA interrupts. 1: A PCA interrupt will be generated when COVF is set.
5	COVF	**Cycle Overflow Flag.** This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software. 0: No overflow has occurred since the last time this bit was cleared. 1: An overflow has occurred since the last time this bit was cleared.
4:2	Unused	Unused. Read = 000b; Write = Don't care.
1:0	CLSEL[1:0]	**Cycle Length Select.** When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode. 00: 8 bits. 01: 9 bits. 10: 10 bits. 11: 11 bits.

SFR Definition 32.4. PCA0CLR: PCA Comparator Clear Control

Bit	7	6	5	4	3	2	1	0
Name	CPCPOL					CPCE2	CPCE1	CPCE0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCE; SFR Page = All Pages

Bit	Name	Function
7	CPCPOL	**Comparator Clear Polarity.** Selects the polarity of the comparator result that will clear the PCA channel(s). 0: PCA channel(s) will be cleared when comparator result goes logic low 1: PCA channel(s) will be cleared when comparator result goes logic high
6:3	Reserved	Must write 0000b.
2	CPCE2	**Comparator Clear Enable for CEX2.** Enables the comparator clear function on PCA channel 2.
1	CPCE1	**Comparator Clear Enable for CEX1.** Enables the comparator clear function on PCA channel 1.
0	CPCE0	**Comparator Clear Enable for CEX0.** Enables the comparator clear function on PCA channel 0.

SFR Definition 32.5. PCA0CPMn: PCA Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16n	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Addresses: PCA0CPM0 = 0xDA, PCA0CPM1 = 0xDB, PCA0CPM2 = 0xDC

SFR Pages: PCA0CPM0 = All Pages, PCA0CPM1 = All Pages, PCA0CPM2 = All Pages

Bit	Name	Function
7	PWM16n	**16-bit Pulse Width Modulation Enable.** This bit enables 16-bit mode when Pulse Width Modulation mode is enabled. 0: 8 to 11-bit PWM selected. 1: 16-bit PWM selected.
6	ECOMn	**Comparator Function Enable.** This bit enables the comparator function for PCA module n when set to 1.
5	CAPPn	**Capture Positive Function Enable.** This bit enables the positive edge capture for PCA module n when set to 1.
4	CAPNn	**Capture Negative Function Enable.** This bit enables the negative edge capture for PCA module n when set to 1.
3	MATn	**Match Function Enable.** This bit enables the match function for PCA module n when set to 1. 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.
2	TOGn	**Toggle Function Enable.** This bit enables the toggle function for PCA module n when set to 1. 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.
1	PWMn	**Pulse Width Modulation Mode Enable.** This bit enables the PWM function for PCA module n when set to 1. When enabled, a pulse width modulated signal is output on the CEXn pin. 8 to 11-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	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.
Note: When the WDTE bit is set to 1, the PCA0CPM2 register cannot be modified, and module 2 acts as the watchdog timer. To change the contents of the PCA0CPM2 register or the function of module 2, the Watchdog Timer must be disabled.		

SFR Definition 32.6. PCA0L: PCA Counter/Timer Low Byte

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

SFR Address = 0xF9; SFR Page = All Pages

Bit	Name	Function
7:0	PCA0[7:0]	**PCA Counter/Timer Low Byte.** The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.
Note: When the WDTE bit is set to 1, the PCA0L register cannot be modified by software. To change the contents of the PCA0L register, the Watchdog Timer must first be disabled.		

SFR Definition 32.7. PCA0H: PCA Counter/Timer High Byte

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

SFR Address = 0xFA; SFR Page = All Pages

Bit	Name	Function
7:0	PCA0[15:8]	**PCA Counter/Timer High Byte.** The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a “snapshot” register, whose contents are updated only when the contents of PCA0L are read (see Section 32.1).
Note: When the WDTE bit is set to 1, the PCA0H register cannot be modified by software. To change the contents of the PCA0H register, the Watchdog Timer must first be disabled.		

SFR Definition 32.8. PCA0CPLn: PCA Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPn[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Addresses: PCA0CPL0 = 0xFB, PCA0CPL1 = 0xE9, PCA0CPL2 = 0xEB

SFR Pages: PCA0CPL0 = All Pages, PCA0CPL1 = All Pages, PCA0CPL2 = All Pages

Bit	Name	Function
7:0	PCA0CPn[7:0]	**PCA Capture Module Low Byte.** The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture module n. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
Note: A write to this register will clear the module's ECOMn bit to a 0.		

SFR Definition 32.9. PCA0CPHn: PCA Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPn[15:8]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Addresses: PCA0CPH0 = 0xFC, PCA0CPH1 = 0xEA, PCA0CPH2 = 0xEC

SFR Pages: PCA0CPH0 = All Pages, PCA0CPH1 = All Pages, PCA0CPH2 = All Pages

Bit	Name	Function
7:0	PCA0CPn[15:8]	**PCA Capture Module High Byte.** The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture module n. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
Note: A write to this register will set the module's ECOMn bit to a 1.		

33. C2 Interface

C8051F39x/37x devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow Flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

33.1. C2 Interface Registers

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

C2 Register Definition 33.1. C2ADD: C2 Address

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

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

C2 Register Definition 33.2. DEVICEID: C2 Device ID

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

C2 Address: 0x00

Bit	Name	Function
7:0	DEVICEID[7:0]	**Device ID.** This read-only register returns the 8-bit device ID: 0x2B (C8051F39x/37x).

C2 Register Definition 33.3. REVID: C2 Revision ID

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

C2 Address: 0x01

Bit	Name	Function
7:0	REVID[7:0]	**Revision ID.** This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A.

C2 Register Definition 33.4. FPCTL: C2 Flash Programming Control

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

C2 Address: 0x02

Bit	Name	Function
7:0	FPCTL[7:0]	**Flash Programming Control Register.** This register is used to enable Flash programming via the C2 interface. To enable C2 Flash programming, the following codes must be written in order: 0x02, 0x01. Note that once C2 Flash programming is enabled, a system reset must be issued to resume normal operation.

C2 Register Definition 33.5. FPDAT: C2 Flash Programming Data

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

C2 Address: 0xB4

Bit	Name	Function	
7:0	FPDAT[7:0]	**C2 Flash Programming Data Register.** This register is used to pass Flash commands, addresses, and data during C2 Flash accesses. Valid commands are listed below.	
		Code	**Command**
		0x06	Flash Block Read
		0x07	Flash Block Write
		0x08	Flash Page Erase
		0x03	Device Erase

33.2. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and Flash programming 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 (RST) and C2D pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application. A typical isolation configuration is shown in Figure 33.1.

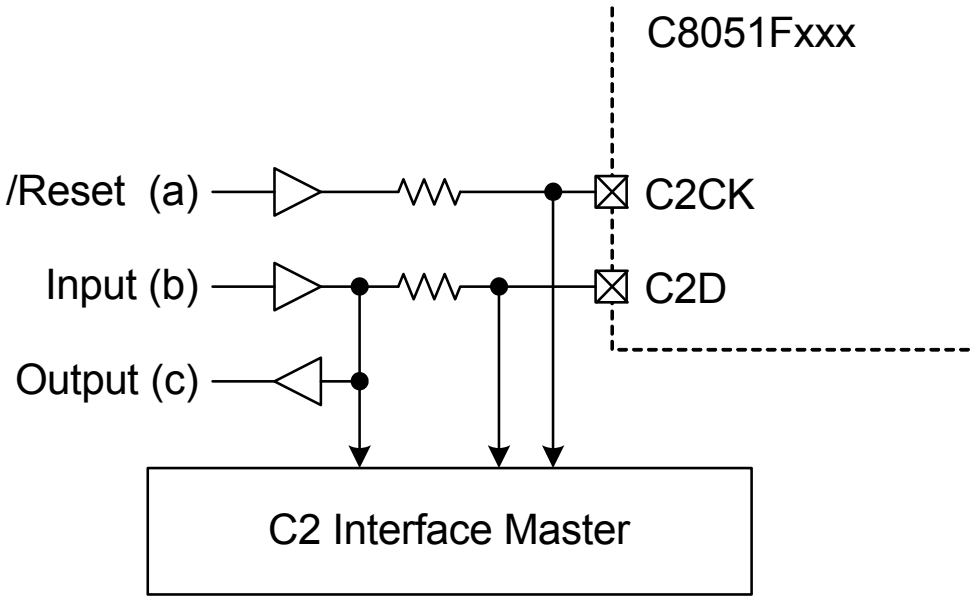


Figure 33.1. Typical C2 Pin Sharing

The configuration in Figure 33.1 assumes the following:

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

Additional resistors may be necessary depending on the specific application.

DOCUMENT CHANGE LIST

Revision 0.1 to Revision 0.7

- Added Section 8.1 “Temperature in Two’s Complement”
- Changed clock cycles for “CJNE A, direct, re” to “4/6” in Section 15. “CIP-15 Microcontroller”
- Changed bit 5 of CRC0CNT to reserved in Section 23. “Cyclic Redundancy Check Unit” (CRC0)
- Changed SFRPGCN reset value to “0x01” in Section 19. “Special Function Registers”
- Added Section 19.2 “Interrupts and Automatic SFR Paging”
- Added Section 19.3 “SFR Page Stack Example”
- Corrected incorrect references to C8501F34x in Section 2. “Ordering Information”
- Removed “The C8051F37x does not include the 4x clock multiplier” bullet point from Section 3.1
- Removed “External Oscillator C and RC Modes” bullet point from Section 3.1
- Updated the block diagram on the front page to show EEPROM and 500 ksps ADC
- Removed references to the REG0MD bit and low power mode in Section 13.1
- Removed REG0MD bit in the REG0CN SFR definition. This bit (bit 2) is now reserved.
- Section 22. “EEPROM” completely rewritten
- Moved the “from IPH, EIPH1 or EIPH2” text from the LSB column to the MSB column in Table 20.1
- Moved the “from IP, EIP1 or EIP2” text from the MSB column to the LSB column in Table 20.1
- Changed Figure 27.4 to show all five footnotes
- Changed Figure 27.5 to show correct SF signals and all five footnotes
- Added 5 V tolerance and lock byte address bullet points to Section 3.1. “Hardware Incompatibilities”

Revision 0.7 to Revision 0.71

- Updated part numbers in Table 2.1 on page 20.
- Updated replacement part numbers in Table 3.1 on page 21 to match Flash sizes.
- Corrected units for normal and active mode IDD (V_{DD} = 3.0 V, F = 80 kHz) in Table 7.2 on page 33.
- Updated maximum normal mode IDD in Table 7.2 on page 33.
- Added EESDA and EESCL DC electrical characteristics to Table 7.3 on page 35.
- Added EEPROM supply current to Table 7.6 on page 37.
- Added maximum EESCL clock frequency to Table 7.6 on page 37.
- Updated typical INL and DNL in Table 7.10 on page 39.
- Updated resolution in Table 7.12 on page 40.
- Updated typical and maximum INL and DNL in Table 7.15 on page 42.
- Updated typical full scale error in Table 7.15 on page 42.
- Updated references to Table 28.3 in the SMB0CN and SMB1CN SFR definitions.

Revision 0.71 to Revision 1.0

- Added typical precision temperature sensor curve Figure 7.3 on page 46.
- Added note to CLKSEL SFR definition.
- Corrected CPCPOL description in the PCA0CLR SFR definition.
- Updated C mode K factors in the OSCXCN SFR definition.
- Updated maximum normal mode IDD (F = 50 MHz, F = 25 MHz) in Table 7.2 on page 33.
- Updated typical suspend and stop mode digital supply current in Table 7.2 on page 33.
- Updated typical precision temperature sensor absolute error in Table 7.12 on page 40.
- Updated maximum precision temperature sensor INL in Table 7.12 on page 40.
- Updated minimum IDAC DNL in Table 7.15 on page 42.
- Updated minimum and maximum IDAC full scale error in Table 7.15 on page 42.
- Updated typical IDAC gain variation in Table 7.15 on page 42.
- Updated maximum comparator supply current at DC in Table 7.16 on page 43.
- Corrected flash security restrictions for erase page containing lock byte (if no pages are locked) in Table 21.1 on page 133.

Revision 0.71 to Revision 1.0

- Corrected EESCL and EESDA pin assignments to P2.2 and P2.3 respectively in Figure 27.3 and Figure 27.4.
- Corrected bit 5 of CRC0CNT to CRC0CNT[5] in Section 23. “Cyclic Redundancy Check Unit” (CRC0).
- Corrected C8051F374/5 flash size to 8kB in Figure 17.1 and Figure 17.2.
- Correct C8051F394/5 flash size to 8kB in

Figure 17.2

Revision 1.0 to Revision 1.1

- Corrected the CRC0CNT size to [5:0] and removed the second note on the CRC0CNT field in the CRC0CNT register (SFR Definition 23.5). The issue where only 31 blocks can be calculated at a time does not apply to this device family.
- Swapped the EESCL (correct pin is P2.2) and EESDA (correct pin is P2.3) in Figure “27.3 Crossbar Priority Decoder - Possible Pin Assignments” on page 178 and Figure “27.4 Crossbar Priority Decoder Example” on page 179.
- Updated Figure “17.1 C8051F39x/37x Memory Map” on page 93 and Figure “17.2 Flash Program Memory Map” on page 94 to correctly list C8051F374/5 with 8 kB of flash.
- Removed the sentence regarding maximum VDD ramp time from “Power-On Reset” on page 156.
- Updated Figure “24.2 Power-On and VDD Monitor Reset Timing” on page 156 to use the V_{RST_LOW} threshold for power-on, since this is the default setting.
- Removed the part numbers and added a “16-byte page” condition on the EEPROM write time in Table 7.6, “EEPROM Electrical Characteristics,” on page 37.
- Clarified throughout “EEPROM (C8051F37x)” on page 140 that writes are 16-byte page aligned.
- Updated the maximum VREF Short-Circuit Current specification in Table 7.13, “Voltage Reference Electrical Characteristics,” on page 41.

CONTACT INFORMATION

Silicon Laboratories Inc.

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
Please visit the Silicon Labs Technical Support web page:
https://www.silabs.com/support/pages/contacttechnicalsupport.aspx
and register to submit a technical support request.