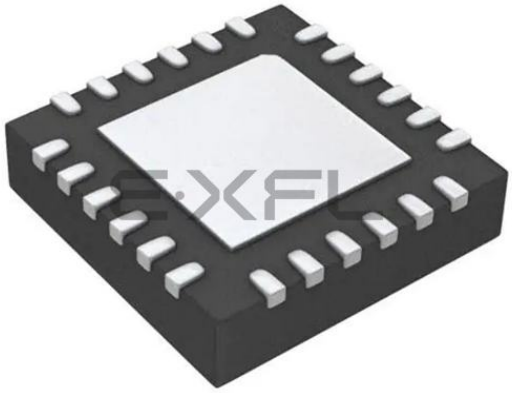


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	8051
Core Size	8-Bit
Speed	50MHz
Connectivity	SMBus (2-Wire/I²C), LINbus, SPI, UART/USART
Peripherals	POR, PWM, Temp Sensor, WDT
Number of I/O	18
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.25V
Data Converters	A/D 18x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	24-WFQFN Exposed Pad
Supplier Device Package	24-QFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051f542-imr

Table 3.1. Pin Definitions for the C8051F54x (Continued)

Name	Pin ‘F540/1/4/5 (32-pin)	Pin ‘F542/3/6/7 (24-pin)	Type	Description
P1.3	23	14	D I/O or A In	Port 1.3.
P1.4	22	13	D I/O or A In	Port 1.4.
P1.5	21	12	D I/O or A In	Port 1.5.
P1.6	20	11	D I/O or A In	Port 1.6.
P1.7	19	10	D I/O or A In	Port 1.7.
P2.0	18	9	D I/O or A In	Port 2.0. See SFR Definition 18.20 for a description.
P2.1	17	—	D I/O or A In	Port 2.1.
P2.2	16	—	D I/O or A In	Port 2.2.
P2.3	15	—	D I/O or A In	Port 2.3.
P2.4	14	—	D I/O or A In	Port 2.4.
P2.5	13	—	D I/O or A In	Port 2.5.
P2.6	12	—	D I/O or A In	Port 2.6.
P2.7	11	—	D I/O or A In	Port 2.7.

4. Package Specifications

4.1. QFP-32 Package Specifications

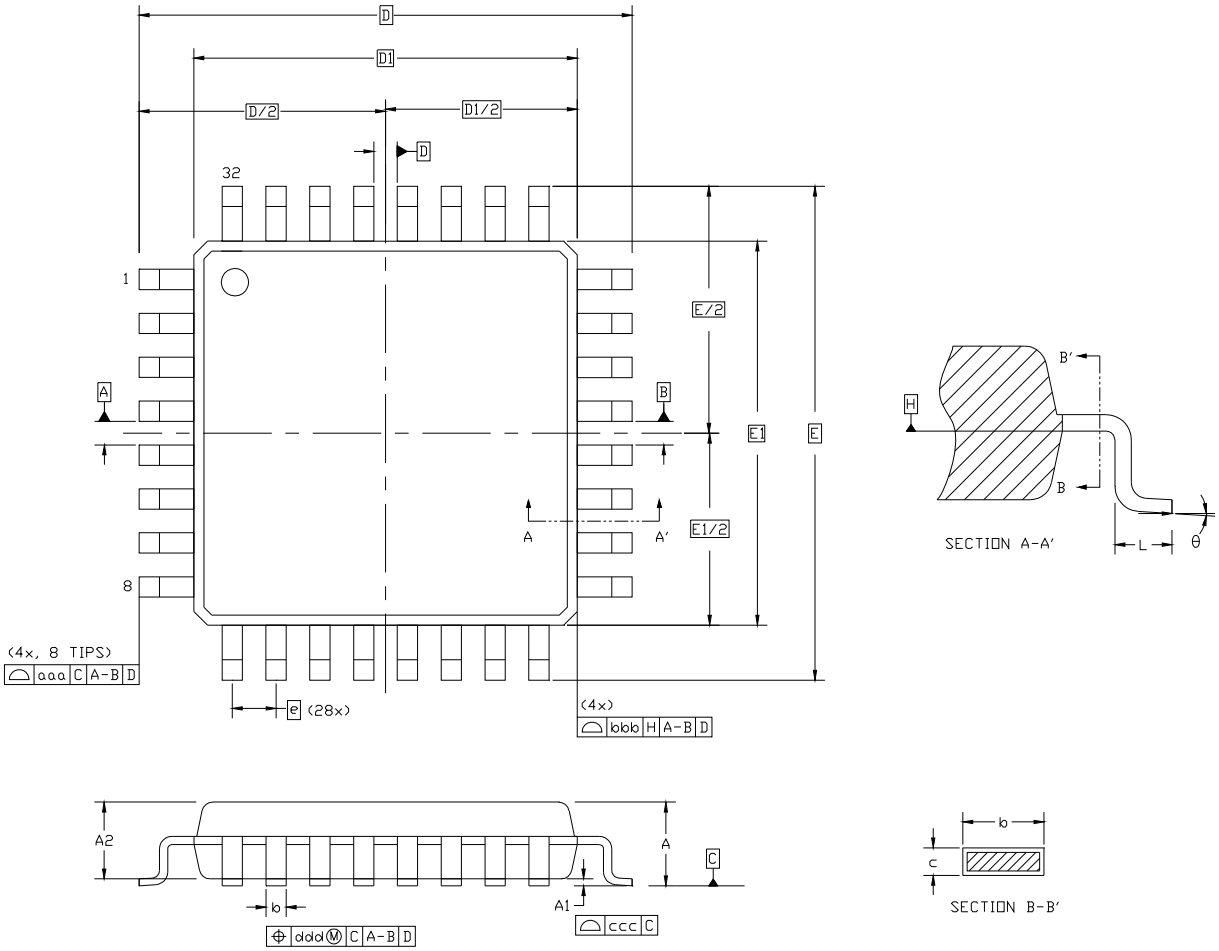


Figure 4.1. QFP-32 Package Drawing

Table 4.1. QFP-32 Package Dimensions

Dimension	Min	Typ	Max
A	—	—	1.60
A1	0.05	—	0.15
A2	1.35	1.40	1.45
b	0.30	0.37	0.45
c	0.09	—	0.20
D	9.00 BSC.		
D1	7.00 BSC.		
e	0.80 BSC.		

Dimension	Min	Typ	Max
E	9.00 BSC.		
E1	7.00 BSC.		
L	0.45	0.60	0.75
aaa	0.20		
bbb	0.20		
ccc	0.10		
ddd	0.20		
θ	0°	3.5°	7°

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 outline MS-026, variation BBA.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

6.2. Temperature Sensor

An on-chip temperature sensor is included on the C8051F54x devices which can be directly accessed via the ADC multiplexer in single-ended configuration. To use the ADC to measure the temperature sensor, the ADC multiplexer channel should be configured to connect to the temperature sensor. The temperature sensor transfer function is shown in Figure 6.3. The output voltage (V_{TEMP}) is the positive ADC input is selected by bits AD0MX[4:0] in register ADC0MX. The TEMPE bit in register REF0CN enables/disables the temperature sensor, as described in SFR Definition 7.1. 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 6.10 for the slope and offset parameters of the temperature sensor.

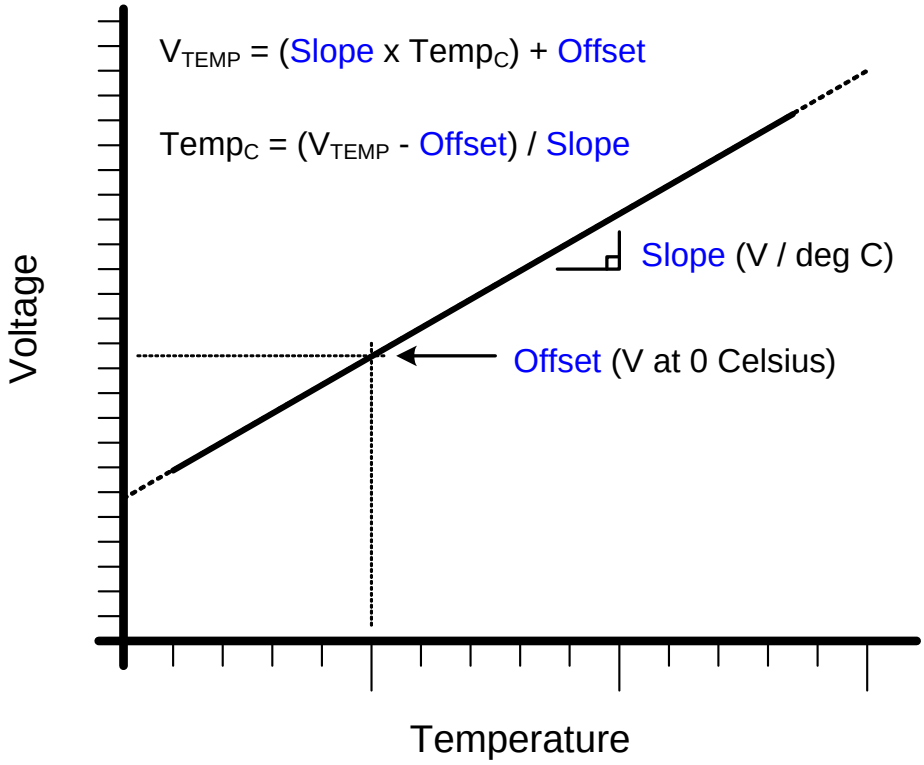


Figure 6.3. Temperature Sensor Transfer Function

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 8.1. 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 = 0x9A; SFR Page = 0x00

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.

9. Voltage Regulator (REG0)

C8051F54x devices include an on-chip low dropout voltage regulator (REG0). The input to REG0 at the V_{REGIN} pin can be as high as 5.25 V. The output can be selected by software to 2.1 V or 2.6 V. When enabled, the output of REG0 appears on the V_{DD} pin, powers the microcontroller core, and can be used to power external devices. On reset, REG0 is enabled and can be disabled by software.

The Voltage regulator can generate an interrupt (if enabled by EREG0, EIE2.0) that is triggered whenever the V_{REGIN} input voltage drops below the dropout threshold voltage. This dropout interrupt has no pending flag and the recommended procedure to use it is as follows:

1. Wait enough time to ensure the V_{REGIN} input voltage is stable
2. Enable the dropout interrupt (EREG0, EIE2.0) and select the proper priority (PREG0, EIP2.0)
3. If triggered, inside the interrupt disable it (clear EREG0, EIE2.0), execute all procedures necessary to protect your application (put it in a safe mode and leave the interrupt now disabled).
4. In the main application, now running in the safe mode, regularly check the DROPOUT bit (REG0CN.0). Once it is cleared by the regulator hardware, the application can enable the interrupt again (EREG0, EIE1.6) and return to the normal mode operation.

The input (V_{REGIN}) and output (V_{DD}) of the voltage regulator should both be bypassed with a large capacitor (4.7 µF + 0.1 µF) to ground as shown in Figure 9.1 below. This capacitor will eliminate power spikes and provide any immediate power required by the microcontroller. The settling time associated with the voltage regulator is shown in Table 6.8 on page 54.

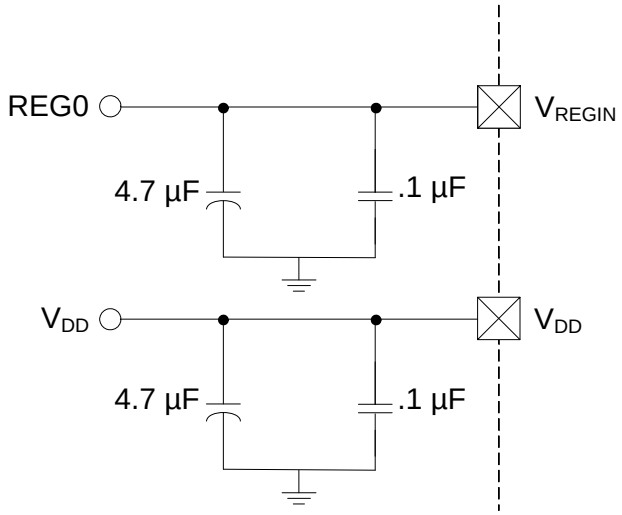


Figure 9.1. External Capacitors for Voltage Regulator Input/Output—Regulator Enabled

If the internal voltage regulator is not used, the V_{REGIN} input should be tied to V_{DD}, as shown in Figure 9.2.

10.2. 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.

10.2.1. 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 10.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

14. 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 Table 6.5 for complete Flash memory electrical characteristics.

14.1. 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 “25. C2 Interface” on page 269.

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 14.4 for more details. Before performing any Flash write or erase procedure, set the FLEWT bit in Flash Scale register (FLSCL) to ‘1’. Also, note that 8-bit MOVX instructions cannot be used to erase or write to Flash memory at addresses higher than 0x00FF

14.1.1. 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 14.2.

14.1.2. 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 doing the following: (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 the PSEE 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.

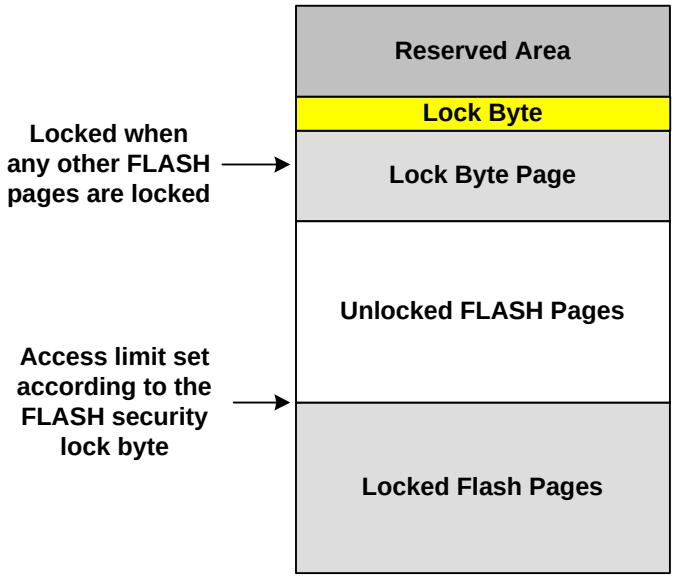
14.2. Non-volatile Data Storage

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.

14.3. Security Options

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

A Security Lock Byte located at the last byte of Flash user space offers protection of the Flash program memory from access (reads, writes, or erases) by unprotected code or the C2 interface. The Flash security mechanism allows the user to lock *n* 512-byte Flash pages, starting at page 0 (addresses 0x0000 to 0x01FF), where *n* is the ones 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).** See example in Figure 14.1.



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

Figure 14.1. Flash Program Memory Map

17.4. 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. For a crystal or ceramic resonator configuration, the crystal/resonator must be wired across the XTAL1 and XTAL2 pins as shown in Option 1 of Figure 17.1. A 10 MΩ resistor also must be wired across the XTAL2 and XTAL1 pins for the crystal/resonator configuration. In RC, capacitor, or CMOS clock configuration, the clock source should be wired to the XTAL2 pin as shown in Option 2, 3, or 4 of Figure 17.1. The type of external oscillator must be selected in the OSCXCN register, and the frequency control bits (XFCN) must be selected appropriately (see SFR Definition 17.6).

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 pins used by the oscillator circuit; see Section “18.3. Priority Crossbar Decoder” on page 150 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 “18.4. Port I/O Initialization” on page 152 for details on Port input mode selection.

SFR Definition 18.1. XBR0: Port I/O Crossbar Register 0

Bit	7	6	5	4	3	2	1	0
Name	CP1AE	CP1E	CP0AE	CP0E	SMB0E	SPI0E	Reserved	URT0E
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 = 0xE1; SFR Page = 0x0F

Bit	Name	Function
7	CP1AE	**Comparator1 Asynchronous Output Enable.** 0: Asynchronous CP1 unavailable at Port pin. 1: Asynchronous CP1 routed to Port pin.
6	CP1E	**Comparator1 Output Enable.** 0: CP1 unavailable at Port pin. 1: CP1 routed to Port pin.
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	SMB0E	**SMBus I/O Enable.** 0: SMBus I/O unavailable at Port pins. 1: SMBus I/O routed to Port pins.
2	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.
1	Reserved	Always Write to 0.
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 18.27. P3SKIP: Port 3Skip

Bit	7	6	5	4	3	2	1	0
Name								P3SKIP[0]
Type	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD7; SFR Page = 0x0F

Bit	Name	Function
7:1	Unused	Read = 0000000b; Write = Don't Care.
0	P3SKIP[0]	**Port 3 Crossbar Skip Enable Bits.** These bits select Port 3 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 P3.n pin is not skipped by the Crossbar. 1: Corresponding P3.n pin is skipped by the Crossbar.
Note: Port P3.0 is only available on the 32-pin packages.		

20.1. 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.

20.2. SMBus Configuration

Figure 20.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 V and 5.0 V; different devices on the bus may operate at different voltage levels. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

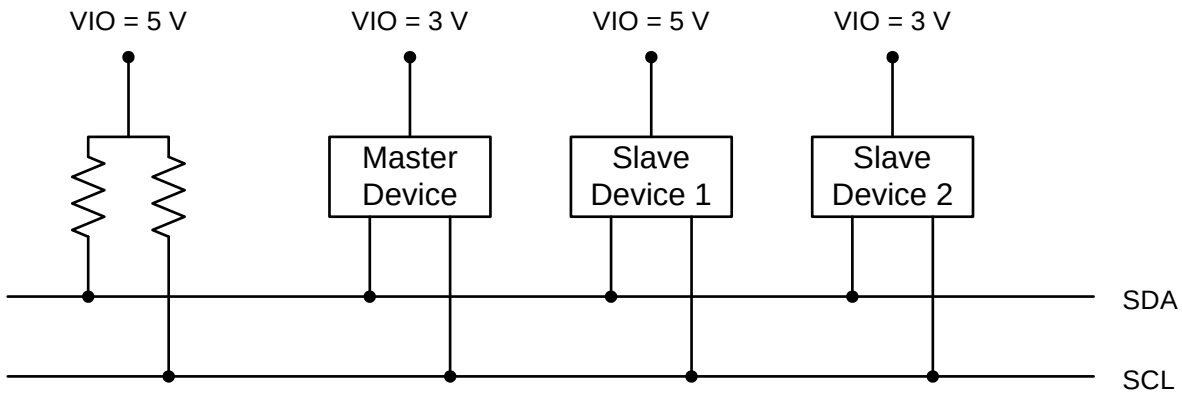


Figure 20.2. Typical SMBus Configuration

20.3. 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 20.3). 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.

Table 20.1. SMBus Clock Source Selection

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

The SMBCS1–0 bits select the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 20.1. Note that the selected clock source may be shared by other peripherals so long as the timer is left running at all times. For example, Timer 1 overflows may generate the SMBus and UART baud rates simultaneously. Timer configuration is covered in Section “23. Timers” on page 227.

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

Equation 20.1. 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 20.1. 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 20.2.

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

Equation 20.2. Typical SMBus Bit Rate

Figure 20.4 shows the typical SCL generation described by Equation 20.2. 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 20.1.

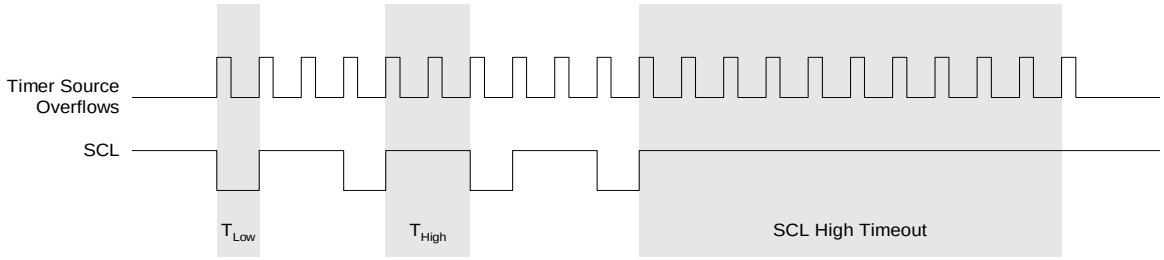


Figure 20.4. Typical SMBus SCL Generation

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

Table 20.2. 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 SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “20.3.4. SCL Low Timeout” on page 189). The SMBus interface will force Timer 3 to reload while SCL is high, and allow Timer 3 to count when SCL is low. The Timer 3 interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods (see Figure 20.4).

20.5.3. 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. 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 the received slave address is ignored, slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received. Software must write the ACK bit at that time to ACK or NACK the received byte.

The interface exits Slave Receiver Mode after receiving a STOP. Note that the interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 20.7 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 **before** the ACK in this mode.

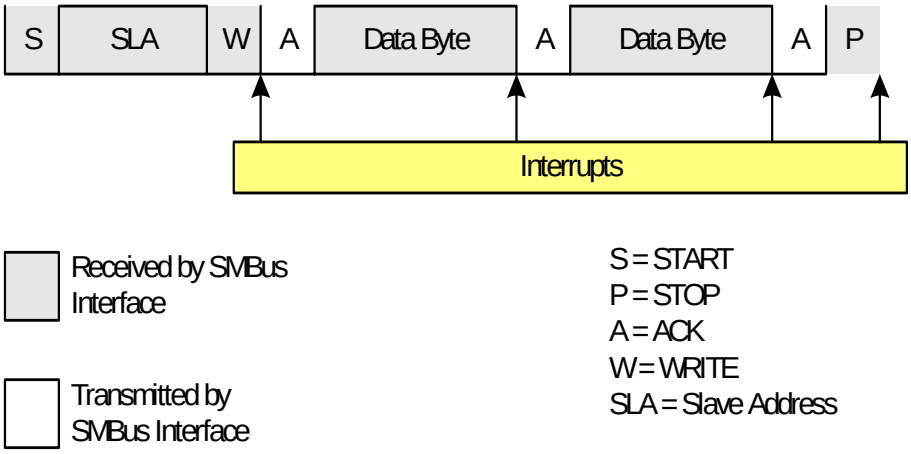
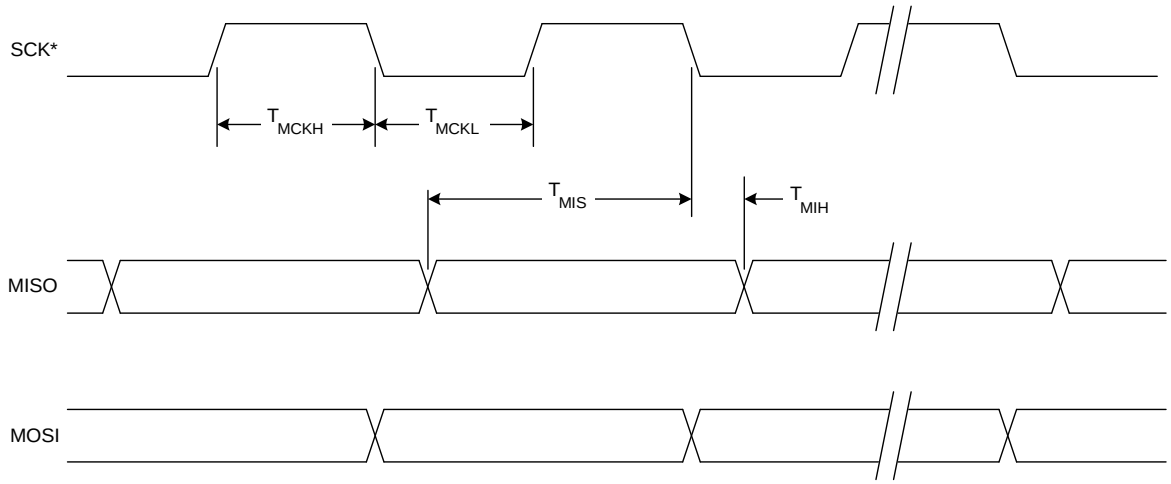
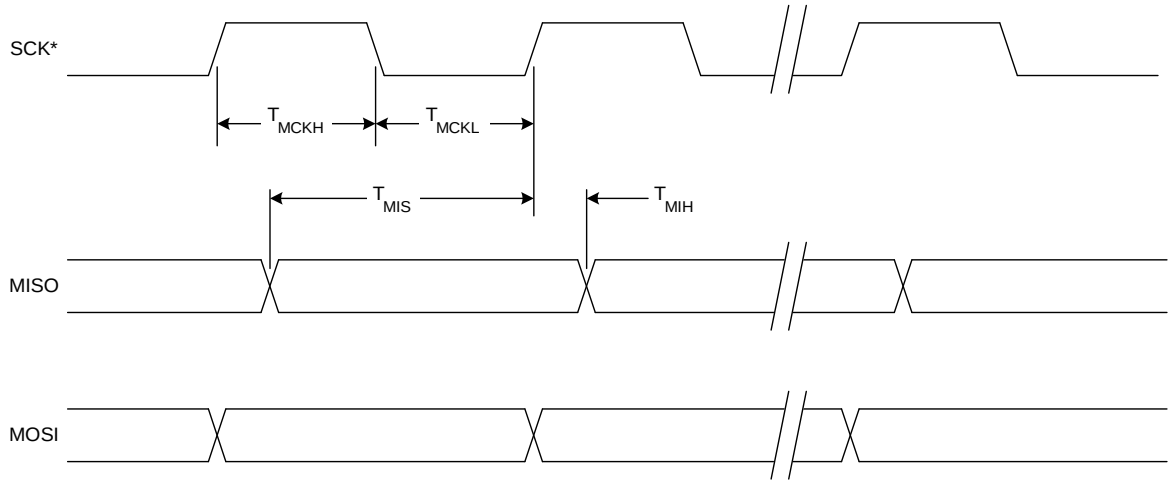


Figure 20.7. Typical Slave Write Sequence



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

Figure 22.8. SPI Master Timing (CKPHA = 0)



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

Figure 22.9. SPI Master Timing (CKPHA = 1)

SFR Definition 23.9. 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 = 0x00

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 23.10. 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 = 0x00

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.

Table 24.3. Watchdog Timer Timeout Intervals[1]

System Clock (Hz)	PCA0CPL5	Timeout Interval (ms)
24,000,000	255	32.8
24,000,000	128	16.5
24,000,000	32	4.2
3,000,000	255	262.1
3,000,000	128	132.1
3,000,000	32	33.8
187,500[2]	255	4194
187,500[2]	128	2114
187,500[2]	32	541

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 128.

C2 Register Definition 25.4. 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 25.5. 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