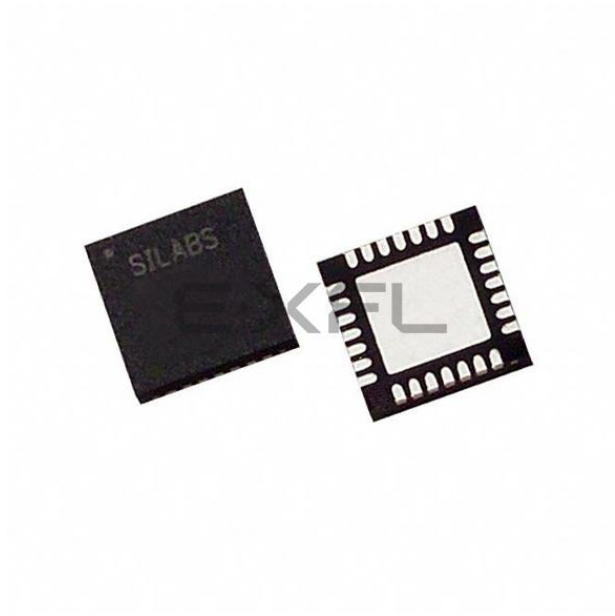


Welcome to E-XFL.COM

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	Not For New Designs
Core Processor	8051
Core Size	8-Bit
Speed	48 MIPS
Connectivity	I²C, SPI, UART/USART, USB
Peripherals	POR, PWM, WDT
Number of I/O	15
Program Memory Size	16KB (16K x 8)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	1.25K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.25V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VFQFN Exposed Pad
Supplier Device Package	28-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/c8051t327-gm

2. Ordering Information

Table 2.1. Product Selection Guide

Ordering Part Number	MIPS (Peak)	EPROM Code Memory (Bytes)	RAM (Bytes)	Calibrated Internal 48 MHz Oscillator	Internal 80 kHz Oscillator	USB with 256 Bytes Endpoint RAM	Supply Voltage Regulator	SMBus/I^2C	Enhanced SPI	UARTs	Timers (16-bit)	Programmable Counter Array	Digital Port I/Os	Separate Port I/O Supply (VIO)	Lead-free (RoHS Compliant)	Package
C8051T622-GM	48	16k[1]	1280	Y	Y	Y	Y	Y	Y	2	4	Y	16	Y	Y	QFN24
C8051T623-GM	48	8k[1]	1280	Y	Y	Y	Y	Y	Y	2	4	Y	16	Y	Y	QFN24
C8051T326-GM[2]	48	16k[1]	1280	Y	Y	Y	Y	Y	Y	2	4	Y	15	Y	Y	QFN28
C8051T327-GM[3]	48	16k[1]	1280	Y	Y	Y	Y	Y	Y	2	4	Y	15	N	Y	QFN28

Notes:
1. 512 Bytes Reserved for Factory use.
2. Pin compatible with the C8051F326-GM.
3. Pin compatible with the C8051F327-GM.

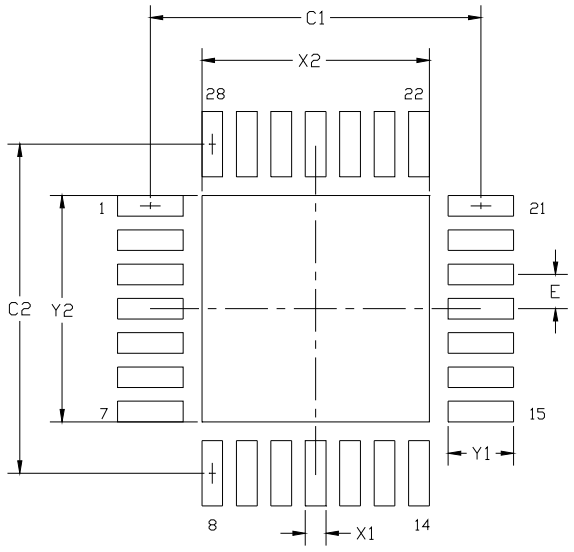


Figure 5.2. QFN-28 Recommended PCB Land Pattern

Table 5.2. QFN-28 PCB Land Pattern Dimensions

Dimension	Min	Max		Dimension	Min	Max
C1	4.80			X2	3.20	3.30
C2	4.80			Y1	0.85	0.95
E	0.50			Y2	3.20	3.30
X1	0.20	0.30				

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 3x3 array of 0.90 mm openings on a 1.1mm pitch should be used for the center pad to assure the proper paste volume (67% 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-020 specification for Small Body Components.

10. 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 C8051T622/3 and C8051T326/7 device family is shown in Figure 10.1

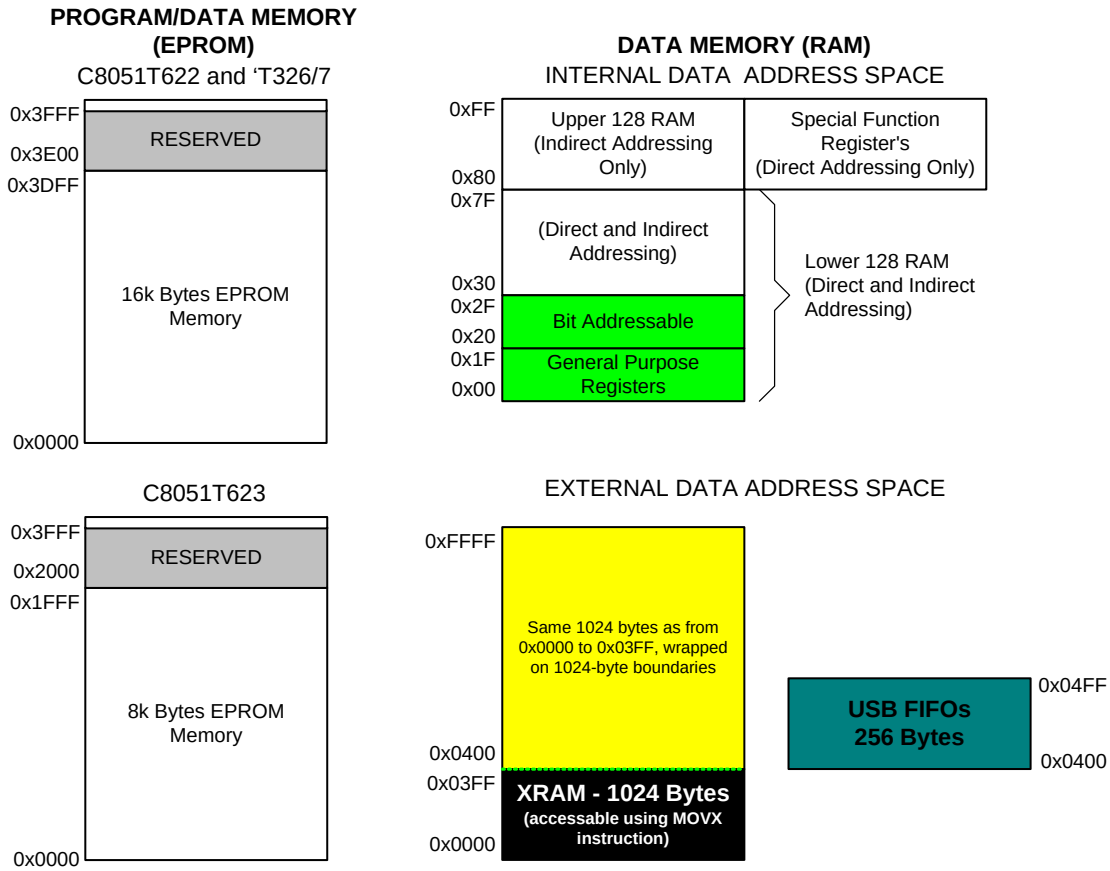


Figure 10.1. Memory Map

10.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051T622/3 and C8051T326/7 implements 16384 or 8192 bytes of this program memory space as in-system byte-programmable EPROM organized in a contiguous block from addresses 0x0000 to 0x3FFF or 0x0000 to 0x1FFF.

Note: 512 bytes (0x3E00 – 0x3FFF) of this memory are reserved for factory use and are not available for user program storage. C2 Register Definition 10.2 shows the program memory maps for C8051T622/3 and C8051T326/7 devices.

13. Program Memory (EPROM)

C8051T622/3 and C8051T326/7 devices include 16 or 8 kB of on-chip byte-programmable EPROM for program code storage. The EPROM memory can be programmed via the C2 debug and programming interface when a special programming voltage is applied to the V_{PP} pin. Additionally, EPROM bytes can be programmed in system using an external capacitor on the V_{PP} pin. Each location in EPROM memory is programmable only once (i.e. non-erasable). Table 6.6 on page 31 shows the EPROM specifications.

13.1. Programming the EPROM Memory

13.1.1. EPROM Programming over the C2 Interface

Programming of the EPROM memory is accomplished through the C2 programming and debug interface. When creating hardware to program the EPROM, it is necessary to follow the programming steps listed below. Please refer to the “C2 Interface Specification” available at http://www.silabs.com for details on communicating via the C2 interface. Section “25. C2 Interface” on page 244 has information about C2 register addresses for the C8051T622/3 and C8051T326/7.

1. **Reset the device using the $\overline{\text{RST}}$ pin**.
2. **Wait at least 20 ms** before sending the first C2 command.
3. Place the device in core reset: **Write 0x04 to the DEVCTL register**.
4. Set the device to program mode (1st step): **Write 0x40 to the EPCTL register**.
5. Set the device to program mode (2nd step): **Write 0x4A to the EPCTL register**.

Note: Devices with a Date Code prior to 1111 should write 0x58 to the EPCTL register.

6. **Apply the V_{PP} programming Voltage**.
7. **Write the first EPROM address for programming to EPADDRH and EPADDRL**.
8. **Write a data byte to EPDAT**. EPADDRH:L will increment by 1 after this write.
9. **Poll the EPBusy bit** using a C2 Address Read command. Note: If EPError is set at this time, the write operation failed.
10. If programming is not finished, return to Step 8 to write the next address in sequence, or return to Step 7 to program a new address.
11. **Remove the V_{PP} programming Voltage**.
12. Remove program mode (1st step): **Write 0x40 to the EPCTL register**.
13. Remove program mode (2nd step): **Write 0x00 to the EPCTL register**.
14. Reset the device: **Write 0x02 and then 0x00 to the DEVCTL register**.

Important Note: There is a finite amount of time which V_{PP} can be applied without damaging the device, which is cumulative over the life of the device. Refer to Table 6.1 on page 28 for the V_{PP} timing specification.

SFR Definition 14.1. 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

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.)

SFR Definition 15.2. RSTSRC: Reset Source

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

SFR Address = 0xEF

Bit	Name	Description	Write	Read
7	USBRSF	**USB Reset Flag**	Writing a 1 enables USB as a reset source.	Set to 1 if USB caused the last reset.
6	MEMERR	**EPROM Error Reset Flag.**	N/A	Set to 1 if EPROM read/write error caused the last reset.
5	UNUSED	Unused. Read = 0b. Write = don't care		
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

17.1. Port I/O Modes of Operation

Port pins use the Port I/O cell shown in Figure 17.2. 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).

17.1.1. Port Pins Configured for Analog I/O

Any pins to be used as an external oscillator input/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 inputs may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

17.1.2. Port Pins Configured For Digital I/O

Any pins to be used by digital peripherals (UART, SPI, SMBus, etc.), external digital event capture 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 V_{IO} 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 and 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 V_{DD} 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 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.

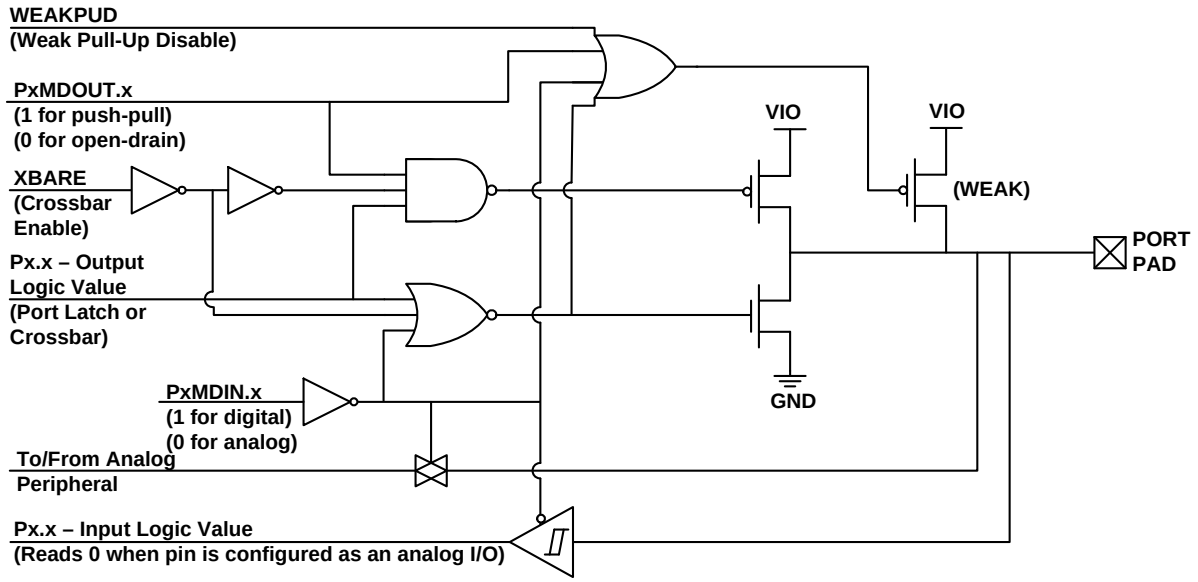
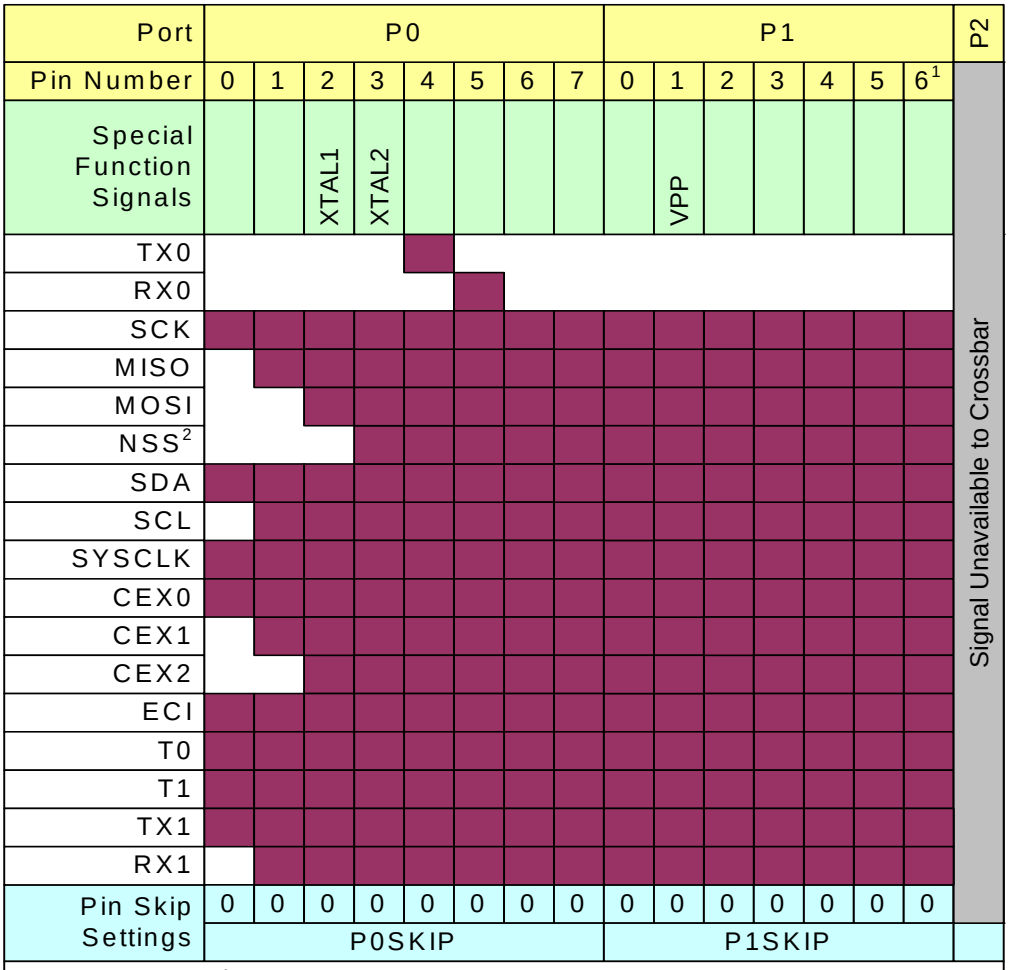


Figure 17.2. Port I/O Cell Block Diagram



Pins P0.0-P1.6[1] are capable of being assigned to crossbar peripherals.

The crossbar peripherals are assigned in priority order from top to bottom, according to this diagram.

■ These boxes represent Port pins which can potentially be assigned to a peripheral.

□ Special Function Signals are not assigned by the crossbar. When these signals are enabled, the Crossbar should be manually configured to skip the corresponding port pins.

□ Pins can be “skipped” by setting the corresponding bit in PnSKIP to ‘1’.

Notes:
1. P1.6 is not available on all devices.
2. NSS is only pinned out when the SPI is in 4-wire mode.

Figure 17.3. Priority Crossbar Decoder Potential Pin Assignments

SFR Definition 17.4. 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 = 0xAE

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 17.5. 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 = 0x84

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.

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 P2.0, 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 17.8. 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; 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 17.11. 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

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 17.12. P1: Port 1

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

SFR Address = 0x90; Bit-Addressable

Bit	Name	Description	Write	Read
7	Unused	Unused. Read = 1b. Write = don't care.		
6:0	P1[6:0]	**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.

USB Register Definition 18.4. INDEX: USB0 Endpoint Index

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

USB Register Address = 0x0E

Bit	Name	Function
7:4	Unused	Unused. Read = 0000b. Write = don't care.
3:0	EPSEL[3:0]	**Endpoint Select Bits.** These bits select which endpoint is targeted when indexed USB0 registers are accessed. 0000: Endpoint 0 0001: Endpoint 1 0010: Endpoint 2 0011-1111: Reserved.

18.4. USB Clock Configuration

USB0 is capable of communication as a Full or Low Speed USB function. Communication speed is selected via the SPEED bit in SFR USB0XCN. When operating as a Low Speed function, the USB0 clock must be 6 MHz. When operating as a Full Speed function, the USB0 clock must be 48 MHz. Clock options are described in Section "16. Oscillators and Clock Selection" on page 86. The USB0 clock is selected via SFR CLKSEL (see SFR Definition 16.1).

Clock Recovery circuitry uses the incoming USB data stream to adjust the internal oscillator; this allows the internal oscillator to meet the requirements for USB clock tolerance. Clock Recovery should be used in the following configurations:

Communication Speed	USB Clock
Full Speed	Internal Oscillator
Low Speed	Internal Oscillator / 8

When operating USB0 as a Low Speed function with Clock Recovery, software must write 1 to the CRLOW bit to enable Low Speed Clock Recovery. Clock Recovery is typically not necessary in Low Speed mode.

Single Step Mode can be used to help the Clock Recovery circuitry to lock when high noise levels are present on the USB network. This mode is not required (or recommended) in typical USB environments.

Suspend Interrupt Service Routine (ISR) should perform application-specific configuration tasks such as disabling appropriate peripherals and/or configuring clock sources for low power modes. See Section “16.3. Programmable Internal High-Frequency (H-F) Oscillator” on page 89 for more details on internal oscillator configuration, including the Suspend mode feature of the internal oscillator.

USB0 exits Suspend mode when any of the following occur: (1) Resume signaling is detected or generated, (2) Reset signaling is detected, or (3) a device or USB reset occurs. If suspended, the internal oscillator will exit Suspend mode upon any of the above listed events.

Resume Signaling: USB0 will exit Suspend mode if Resume signaling is detected on the bus. A Resume interrupt will be generated upon detection if enabled (RESINTE = 1). Software may force a Remote Wakeup by writing 1 to the RESUME bit (POWER.2). When forcing a Remote Wakeup, software should write RESUME = 0 to end Resume signaling 10-15 ms after the Remote Wakeup is initiated (RESUME = 1).

ISO Update: When software writes 1 to the ISOUP bit (POWER.7), the ISO Update function is enabled. With ISO Update enabled, new packets written to an ISO IN endpoint will not be transmitted until a new Start-Of-Frame (SOF) is received. If the ISO IN endpoint receives an IN token before a SOF, USB0 will transmit a zero-length packet. When ISOUP = 1, ISO Update is enabled for all ISO endpoints.

USB Enable: USB0 is disabled following a Power-On-Reset (POR). USB0 is enabled by clearing the USBINH bit (POWER.4). Once written to 0, the USBINH can only be set to 1 by one of the following: (1) a Power-On-Reset (POR), or (2) an asynchronous USB0 reset generated by writing 1 to the USBRST bit (POWER.3).

Software should perform all USB0 configuration before enabling USB0. The configuration sequence should be performed as follows:

1. Select and enable the USB clock source.
2. Reset USB0 by writing USBRST= 1.
3. Configure and enable the USB Transceiver.
4. Perform any USB0 function configuration (interrupts, Suspend detect).
5. Enable USB0 by writing USBINH = 0.

18.13.1. Endpoints1-2 OUT Interrupt or Bulk Mode

When the ISO bit (EOUTCSRH.6) = 0 the target endpoint operates in Bulk or Interrupt mode. Once an endpoint has been configured to operate in Bulk/Interrupt OUT mode (typically following an Endpoint0 SET_INTERFACE command), hardware will set the OPRDY bit (EOUTCSRL.0) to 1 and generate an interrupt upon reception of an OUT token and data packet. The number of bytes in the current OUT data packet (the packet ready to be unloaded from the FIFO) is given in the EOUTCNTH and EOUTCNTL registers. In response to this interrupt, firmware should unload the data packet from the OUT FIFO and reset the OPRDY bit to 0.

A Bulk or Interrupt pipe can be shut down (or Halted) by writing 1 to the SDSTL bit (EOUTCSRL.5). While SDSTL = 1, hardware will respond to all OUT requests with a STALL condition. Each time hardware generates a STALL condition, an interrupt will be generated and the STSTL bit (EOUTCSRL.6) set to 1. The STSTL bit must be reset to 0 by firmware.

Hardware will automatically set OPRDY when a packet is ready in the OUT FIFO. Note that if double buffering is enabled for the target endpoint, it is possible for two packets to be ready in the OUT FIFO at a time. In this case, hardware will set OPRDY to 1 immediately after firmware unloads the first packet and resets OPRDY to 0. A second interrupt will be generated in this case.

18.13.2. Endpoints1-2 OUT Isochronous Mode

When the ISO bit (EOUTCSRH.6) is set to 1, the target endpoint operates in Isochronous (ISO) mode. Once an endpoint has been configured for ISO OUT mode, the host will send exactly one data per USB frame; the location of the data packet within each frame may vary, however. Because of this, it is recommended that double buffering be enabled for ISO OUT endpoints.

Each time a data packet is received, hardware will load the received data packet into the endpoint FIFO, set the OPRDY bit (EOUTCSRL.0) to 1, and generate an interrupt (if enabled). Firmware would typically use this interrupt to unload the data packet from the endpoint FIFO and reset the OPRDY bit to 0.

If a data packet is received when there is no room in the endpoint FIFO, an interrupt will be generated and the OVRUN bit (EOUTCSRL.2) set to 1. If USB0 receives an ISO data packet with a CRC error, the data packet will be loaded into the endpoint FIFO, OPRDY will be set to 1, an interrupt (if enabled) will be generated, and the DATAERR bit (EOUTCSRL.3) will be set to 1. Software should check the DATAERR bit each time a data packet is unloaded from an ISO OUT endpoint FIFO.

20.2. Operational Modes

UART0 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S0MODE bit (SCON0.7). Typical UART connection options are shown in Figure 20.3.

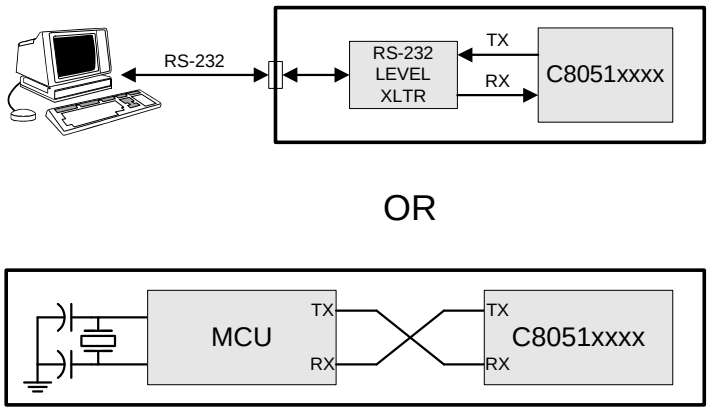


Figure 20.3. UART Interconnect Diagram

20.2.1. 8-Bit UART

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

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

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

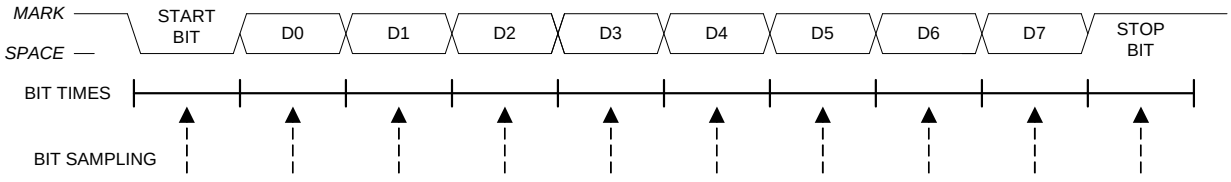


Figure 20.4. 8-Bit UART Timing Diagram

20.2.2. 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.

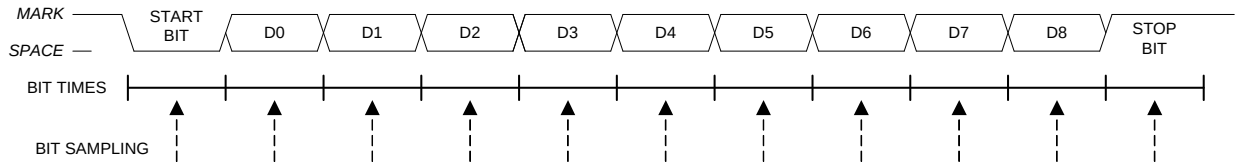


Figure 20.5. 9-Bit UART Timing Diagram

23. Timers

Each MCU includes four counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and two 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 and Timer 3 offer 16-bit and split 8-bit timer functionality with auto-reload. Additionally, Timer 3 offers the ability to be clocked from the external oscillator while the device is in Suspend mode, and can be used as a wake-up source. This allows for implementation of a very low-power system, including RTC capability.

<table>
<tr><th>Timer 0 and Timer 1 Modes:</th><th>Timer 2 Modes:</th><th>Timer 3 Modes:</th></tr>
<tr><td>13-bit counter/timer</td><td rowspan="2">16-bit timer with auto-reload</td><td rowspan="2">16-bit timer with auto-reload</td></tr>
<tr><td>16-bit counter/timer</td></tr>
<tr><td>8-bit counter/timer with auto-reload</td><td rowspan="2">Two 8-bit timers with auto-reload</td><td rowspan="2">Two 8-bit timers with auto-reload</td></tr>
<tr><td>Two 8-bit counter/timers (Timer 0 only)</td></tr>
</table>

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 23.1 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 and Timer 3 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.

C2 Register Definition 25.12. CRC2: CRC Byte 2

Bit	7	6	5	4	3	2	1	0
Name	CRC[23:16]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

C2 Address: 0xAB

Bit	Name	Function
7:0	CRC[23:16]	**CRC Byte 2.** See Section “13.4. Program Memory CRC” on page 74.

C2 Register Definition 25.13. CRC3: CRC Byte 3

Bit	7	6	5	4	3	2	1	0
Name	CRC[31:24]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

C2 Address: 0xAC

Bit	Name	Function
7:0	CRC[31:24]	**CRC Byte 3.** See Section “13.4. Program Memory CRC” on page 74.