8XC151SA and 8XC151SB Hardware Description

June 1996

Order Number: 272832-001

Information in this document is provided in connection with Intel products. Intel assumes no liability whatsoever, including infringement of any patent or copyright, for sale and use of Intel products except as provided in Intel's Terms and Conditions of Sale for such products.

Intel retains the right to make changes to these specifications at any time, without notice. Microcomputer Products may have minor variations to this specification known as errata.

*Other brands and names are the property of their respective owners.

†Since publication of documents referenced in this document, registration of the Pentium, OverDrive and iCOMP trademarks has been issued to Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

COPYRIGHT © INTEL CORPORATION, 1996

HARDWARE DESCRIPTION OF THE 8XC151SX

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

CONTENTS

PAGE

1.0 INTRODUCTION

The 8XC151SA/SB is a highly integrated CMOS 8-bit microcontroller which is instruction set compatible with the MCS®51 microcontroller. It comes in 40-lead PDIP and 44-lead PLCC maintaining pin compatibility with MCS 51 microcontrollers. The 8XC151SA/SB has 256 bytes of on-chip RAM and is available with 8/16 Kbytes of on-chip ROM/OTPROM or ROMless. Several new features like programmable wait states, page mode and Extended ALE can be selected by using the new user programmable configuration. Key features like instruction pipeline, minimum of 2 clocks per instruction and code fetch in page mode are available.

Functional characteristics of 8XC151SA/SB are listed below.

- Pipeline Instruction Execution Unit
- MCS 51 Microcontroller 44-pin PLCC and 40-pin PDIP Compatibility
- MCS 51 Microcontroller Compatible Instruction Set
- Static Standby to at least 16 MHz Operation
- 64K External Code Memory Space
- 64K External Data Memory Space
- 256 Bytes On-chip Data RAM
- 8/16K On-chip Code Memory (OTPROM/ROM) or ROMless Options
- 3 16-Bit Flexible Timer/Counters
- 32 Programmable I/O Lines
- 7 Maskable Interrupt Sources with 4 Programmable Priority Levels
- Programmable Counter Array with:

— High Speed Output
— Compare/Capture
— Pulse Width Modulator
— Watchdog Timer Capabilities

- Programmable Serial I/O Port

— Framing Error Detection
— Automatic Address Recognition

- Hardware Watchdog Timer
- External WAIT Pin
- 16-Bit Internal Code Bus
- Page Mode and Wait States Configuration Options
- Minimum 2-Clock External Code Fetch in Page Mode
- User-Selectable Configurations:

— External Wait States (0–3 Wait States)
— Page Mode

- Power-Saving Features

— Idle Mode
— Power Down Mode

Table 1 summarizes the product names and memory differences of the various 8XC151SA/SB products currently available. Throughout this document, the products will generally be referred to as the C151SX.

Table 1. 8XC151 Family of Microcontrollers

ROM Version	OTPROM Version	ROMless Version	OTPROM ROM Size	RAM Bytes
83C151SA	87C151SA	80C151SB	8K	256
83C151SB	87C151SB	80C151SB	16K	256

2.0 MEMORY ORGANIZATION

The 8XC151SA/SB devices have a separate address space for Program and Data Memory. Up to 64 Kbytes each of external Program and Data Memory can be addressed.

2.1 Program Memory

If the EA# pin is connected to V_{SS}, all program fetches are directed to external memory. On the 83C151SA (or 87C151SA), if the EA# pin is connected to V_{CC}, then program fetches to addresses 0000H through 1FFFH are directed to internal ROM and fetches to addresses 2000H through FFFFH are to external memory.

On the 83C151SB (or 87C151SB) if EA# is connected to V_{CC}, program fetches to addresses 0000H through 3FFFH are directed to internal ROM, and fetches to addresses 4000H through FFFFH are to external memory.

2.2 Data Memory

The C151SX has internal data memory that is mapped into three separate segments: the lower 128 bytes of RAM, upper 128 bytes of RAM and 128 bytes special function register (SFR). Refer to Figure 2.

The three segments are:

1. The Lower 128 bytes of RAM (addresses 00H to 7FH) are directly and indirectly addressable.
2. The Upper 128 bytes of RAM (addresses 80H to FFH) are indirectly addressable only.
3. The Special Function Registers, SFRs, (addresses 80H to FFH) are directly addressable only.

The Lower 128 bytes can be accessed by either direct or indirect addressing. The Upper 128 bytes can be accessed by indirect addressing only. The Upper 128 bytes occupy the same address space as the SFR. That means they have the same address, but are physically separate from SFR space.

When an instruction accesses an internal location above address 7FH, the CPU knows whether the access is to the upper 128 bytes of data RAM or to SFR space by the addressing mode used in the instruction. Instructions that use direct addressing access SFR space. For example:

```
MOV 0A0H, #data
```

accesses the SFR at location 0A0H (which is P2). Instructions that use indirect addressing access the Upper 128 bytes of data RAM. For example:

```
MOV @R0, #data
```

where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

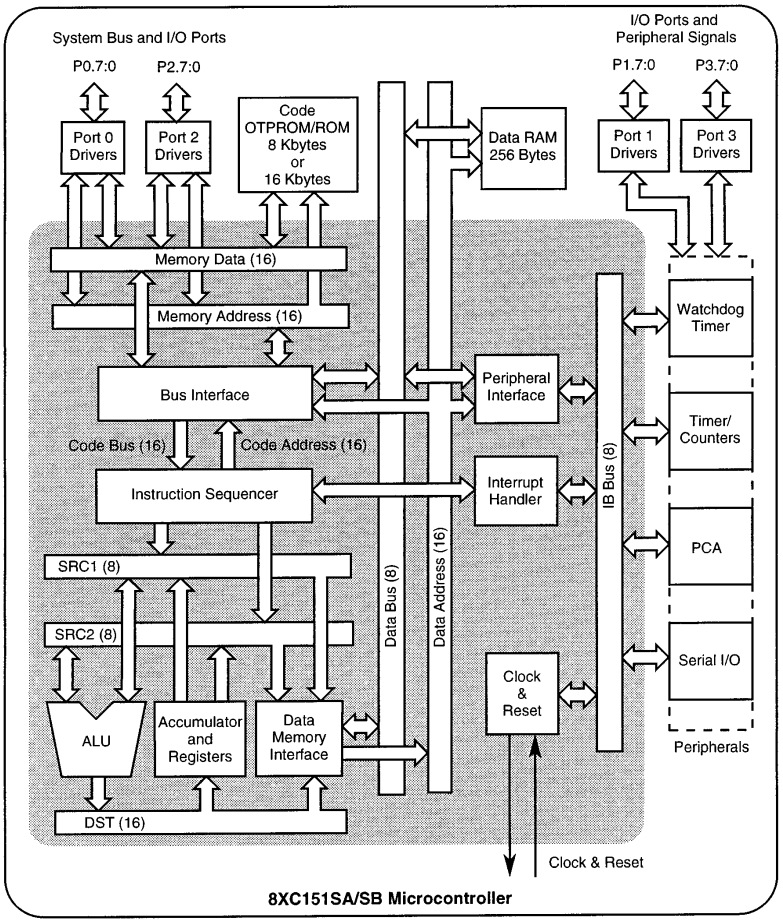


Figure 1. 8XC151SA/SB Functional Block Diagram

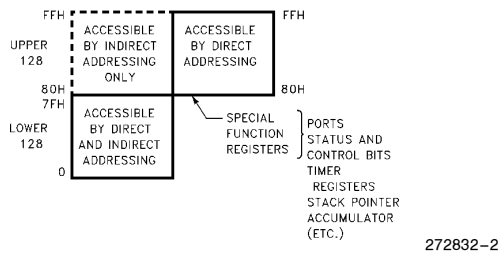


Figure 2. Internal Data Memory

3.0 SPECIAL FUNCTION REGISTERS

The special function registers (SFRs) reside in their associated on-chip peripherals or in the core. Table 2 shows the SFR address space with the SFR mnemonics and reset values. Unoccupied locations in the SFR space (the shaded locations in Table 2) are unimplemented, i.e., no register exists. If an instruction attempts to write to an unimplemented SFR location, the instruction executes, but nothing is actually written. If an unimplemented SFR location is read, it returns an unspecified value.

Table 2. 8XC151SA/SB Map and Reset Values

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
F8		CH 00000000	CCAP0H xxxxxxxx	CCAP1H xxxxxxxx	CCAP2H xxxxxxxx	CCAP3H xxxxxxxx	CCAP4H xxxxxxxx		FF
F0	B 00000000								F7
E8		CL 00000000	CCAP0L xxxxxxxx	CCAP1L xxxxxxxx	CCAP2L xxxxxxxx	CCAP3L xxxxxxxx	CCAP4L xxxxxxxx		EF
E0	ACC 00000000								E7
D8	CCON 00x00000	CMOD 00xxx000	CCAPM0 x0000000	CCAPM1 x0000000	CCAPM2 x0000000	CCAPM3 x0000000	CCAPM4 x0000000		DF
D0	PSW 00000000	PSW 1 00000000							D7
C8	T2CON 00000000	T2MOD xxxxxx00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000			CF
C0									C7
B8	IPL0 x0000000	SADEN 00000000							BF
B0	P3 11111111							IPH0 x0000000	B7
A8	IE0 00000000	SADDR 00000000							AF
A0	P2 11111111						WDTRST xxxxxxxx		A7
98	SCON 00000000	SBUF xxxxxxxx							9F
90	P1 11111111								97
88	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8F
80	P0 11111111	SP 00000111	DPL 00000000	DPH 00000000				PCON 00xx0000	87
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	

The following tables list the mnemonics, names, and addresses of the SFRs:

Table 3—Core SFRs

Table 4—I/O Port SFRs

Table 5—Serial I/O SFRs

Table 6—Timer/Counter and Watchdog Timer SFRs

Table 7—Programmable Counter Array (PCA) SFRs

Table 3. Core SFRs

Mnemonic	Name	Address
ACC	Accumulator	E0H
B	B Register	F0H
PSW	Program Status Word	D0H
PSW1	Program Status Word 1	D1H
SP	Stack Pointer	81H
DPTR	Data Pointer (2 bytes)	—
DPL	Low Byte of DPTR	82H
DPH	High Byte of DPTR	83H
PCON	Power Control	87H
IE0	Interrupt Enable Control 0	A8H
IPH0	Interrupt Priority Control High 0	B7H
IPL0	Interrupt Priority Control Low 0	B8H

Table 4. I/O Port SFRs

Mnemonic	Name	Address
P0	Port 0	80H
P1	Port 1	90H
P2	Port 2	A0H
P3	Port 3	B0H

Table 5. Serial I/O SFRs

Mnemonic	Name	Address
SCON	Serial Control	98H
SBUF	Serial Data Buffer	99H
SADEN	Serial Address Mask	B9H
SADDR	Serial Address	A9H

Table 6. Timer/Counter and Watchdog Timer SFRs

Mnemonic	Name	Address
TL0	Timer/Counter 0 Low Byte	8AH
TH0	Timer/Counter 0 High Byte	8CH
TL1	Timer/Counter 1 Low Byte	8BH
TH1	Timer/Counter 1 High Byte	8DH
TL2	Timer/Counter 2 Low Byte	CCH
TH2	Timer/Counter 2 High Byte	CDH
TCON	Timer/Counter 0 and 1 Control	88H
TMOD	Timer/Counter 0 and 1 Mode Control	89H
T2CON	Timer/Counter 2 Control	C8H
T2MOD	Timer/Counter 2 Mode Control	C9H
RCAP2L	Timer 2 Reload/Capture Low Byte	CAH
RCAP2H	Timer 2 Reload/Capture High Byte	CBH
WDTRST	Watchdog Timer Reset	A6H

Table 7. Programmable Counter Array (PCA) SFRs

Mnemonic	Name	Address
CCON	PCA Timer/Counter Control	D8H
CMOD	PCA Timer/Counter Mode	D9H
CCAPM0	PCA Timer/Counter Mode 0	DAH
CCAPM1	PCA Timer/Counter Mode 1	DBH
CCAPM2	PCA Timer/Counter Mode 2	DCH
CCAPM3	PCA Timer/Counter Mode 3	DDH
CCAPM4	PCA Timer/Counter Mode 4	DEH
CL	PCA Timer/Counter Low Byte	E9H
CH	PCA Timer/Counter High Byte	F9H
CCAP0L	PCA Compare/Capture Module 0 Low Byte	EAH
CCAP1L	PCA Compare/Capture Module 1 Low Byte	EBH
CCAP2L	PCA Compare/Capture Module 2 Low Byte	ECH
CCAP3L	PCA Compare/Capture Module 3 Low Byte	EDH
CCAP4L	PCA Compare/Capture Module 4 Low Byte	EEH
CCAP0H	PCA Compare/Capture Module 0 High Byte	FAH
CCAP1H	PCA Compare/Capture Module 1 High Byte	FBH
CCAP2H	PCA Compare/Capture Module 2 High Byte	FCH
CCAP3H	PCA Compare/Capture Module 3 High Byte	FDH
CCAP4H	PCA Compare/Capture Module 4 High Byte	FEH

Accumulator: ACC is the Accumulator register. The mnemonics for Accumulator-Specific instructions, however, refer to the Accumulator simply as A.

B Register: The B register is used during multiply and divide operations. For other instructions it can be treated as another scratch pad register.

Stack Pointer (SP): The Stack Pointer Register is 8 bits wide. It is incremented before data is stored during PUSH and CALL executions. The stack may reside anywhere in on-chip RAM. On reset, the Stack Pointer is initialized to 07H causing the stack to begin at location 08H.

Data Pointer: The Data Pointer (DPTR) consists of a high byte (DPH) and a low byte (DPL). Its intended function is to hold a 16-bit address, but it may be manipulated as a 16-bit register or as two independent 8-bit registers.

Program Status Word: The PSW and PSW1 registers contain program status information as detailed in PSW: Program Status Word Register on the following page.

Ports 0 to 3 Registers: P0, P1, P2, and P3 are the SFR latches of Port 0, Port 1, Port 2, and Port 3, respectively.

Timer Registers: Register pairs (TH0, TL0), (TH1, TL1), and (TH2, TL2) are the 16-bit count registers for Timer/Counters 0, 1, and 2 respectively. Control and status bits are contained in registers TCON and TMOD for Timers 0 and 1 and in registers T2CON and T2MOD for Timer 2. The register pair (RCAP2H, RCAP2L) are the capture/reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

Programmable Counter Array (PCA) Registers: The 16-bit PCA timer/counter consists of registers CH and CL. Registers CCON and CMOD contain the control and status bits for the PCA. The CCAPMn (n = 0, 1, 2, 3, or 4) registers control the mode for each of the five PCA modules. The register pairs (CCAPnH, CCAPnL) are the 16-bit compare/capture registers for each PCA module.

Serial Port Registers: The Serial Data Buffer, SBUF, is actually two separate registers: a transmit buffer and a receive buffer register. When data is moved to SBUF, it goes to the transmit buffer where it is held for serial transmission. (Moving a byte to SBUF initiates the transmission). When data is moved from SBUF, it comes from the receive buffer. Register SCON contains the control and status bits for the Serial Port. Registers SADDR and SADEN are used to define the Given and the Broadcast addresses for the Automatic Address Recognition feature.

Interrupt Registers: The individual interrupt enable bits are in the IE0 register. Priorities of interrupt can be set in the IPH0 and IPL0 registers.

Power Control Register: PCON controls the Power Reduction Modes. Idle and Power Down Modes.

Watchdog Timer Reset: WDTRST register is used to keep the watchdog timer from periodically resetting the part.

PSW: Program Status Word Register

PSW

Address: D0H
Reset State: 0000 0000B

Program Status Word. PSW contains bits that reflect the results of operations, bits that select the register bank for registers R0–R7, and two general-purpose flags that are available to the user.

7							0
CY	AC	F0	RS1	RS0	OV	UD	P

Bit Mnemonic	Function
CY	Carry Flag: The carry flag is set by an addition instruction (ADD, ADDC) if there is a carry out of the MSB. It is set by a subtraction (SUB, SUBB) if a borrow is needed for the MSB. The carry flag is also affected by some rotate and shift instructions, logical bit instructions, bit move instructions, and the multiply (MUL) and decimal adjust (DA) instructions.
AC	Auxillary Carry Flag: The auxillary carry flag is affected only by instructions that address 8-bit operands. The AC flag is set if an arithmetic instruction with an 8-bit operand produces a carry out of bit 13 (from addition) or a borrow into bit 3 (from subtraction). Otherwise it is cleared. This flag is useful for BCD arithmetic.
F0	Flag 0: This general-purpose flag is available to the user.
RS1:0	Register Bank Select Bits 1 and 0: These bits select the memory locations that comprise the active bank of the register file (registers R0–R7). **RS1 RS0 Bank Address** 0 0 0 00H–07H 0 1 1 08H–0FH 1 0 2 10H–17H 1 1 3 18H–1FH
OV	Overflow Flag: This bit is set if an addition or subtraction of signed variables results in an overflow error (i.e., if the magnitude of the sum or difference is too great for the seven LSBs in 2's-complement representation). The overflow flag is also set if a multiplication product overflows one byte or if a division by zero is attempted.
UD	User-defined Flag: This general-purpose flag is available to the user.
P	Parity Bit: This bit indicates the parity if the accumulator. It is set if an odd number of bits in the accumulator are set. Otherwise, it is cleared. Not all instructions update the parity bit. The parity bit is set or cleared by instructions that change the contents of the accumulator (ACC, Register R11).

PSW1 Address: 0D1H
Reset State: 0000 0000B

7							0
CY	AC	F0	RS1	RS0	OV	Z	—

Bit Mnemonic	Function
CY	Carry Flag: Identical to the CY bit in the PSW register.
AC	Auxillary Carry Flag: Identical to the AC bit in the PSW register.
N	Negative Flag: This bit is set if the result of the last logical or arithmetic operation was negative, i.e., bit 15 = 1. Otherwise it is cleared.
RS1:0	Register Bank Select Bits 0 and 1: Identical to the RS1:0 bits in the PSW register.
OV	Overflow Flag: Identical to the OV bit in the PSW register.
Z	Zero Flag: This flag is set if the result of the last logical or arithmetic operation is zero. Otherwise it is cleared.
—	Reserved: The value read from this bit is indeterminate. Do not write a ‘‘1’’ to this bit.

4.0 PORT STRUCTURES AND OPERATION

The C151SX uses input/output (I/O) ports to exchange data with external devices. In addition to performing general-purpose I/O, some ports are capable of external memory operations; others allow for alternate functions. All four C151SX I/O ports are bidirectional. Each port contains a latch, an output driver, and an input buffer. Port 0 and port 2 output drivers and input buffers facilitate external memory operations. Port 0 drives the lower address byte onto the parallel address bus, and port 2 drives the upper address byte onto the bus. In nonpage mode, the data is multiplexed with the lower address byte on port 0. In page mode, the data is multiplexed with the upper address byte on port 2. All port 1 and port 3 pins serve for both general-purpose I/O and alternative functions.

4.1 I/O Configurations

Each port SFR operates via type-D latches, as illustrated in Figure 3 for ports 1 and 3. A CPU "write to latch" signal initiates transfer of internal bus data into the type-D latch. A CPU "read latch" signal transfers the latched Q output onto the internal bus. Similarly, a "read pin" signal transfers the logical level of the port pin. Some port data instructions activate the "read latch" signal while others activate the "read pin" signal. Latch instructions are referred to as read-modify-write instructions. Each I/O line may be independently programmed as input or output.

4.2 Port 1 and Port 3

Figure 3 shows the structure of ports 1 and 3, which have internal pullups. An external source can pull the pin low. Each port pin can be configured either for general-purpose I/O or for its alternate input or output function.

To use a pin for general-purpose output, set or clear the corresponding bit in the P*x* register ($x = 1, 3$). To use a pin for general-purpose input, set the bit in the P*x* register. This turns off the output driver FET.

To configure a pin for its alternate function, set the bit in the P*x* register. When the latch is set, the "alternate output function" signal controls the output level (Figure 3). The operation of ports 1 and 3 is discussed further in the "Quasi-bidirectional Port Operation" section.

4.3 Port 0 and Port 2

Ports 0 and 2 are used for general-purpose I/O or as the external address/data bus. Port 0, shown in Figure 4, differs from the other ports in not having internal pullups. Figure 5 shows the structure of port 2. An external source can pull a port 2 pin low.

To use a pin for general-purpose output, set or clear the corresponding bit in the P*x* register ($x = 0, 2$). To use a pin for general-purpose input set the bit in the P*x* register to turn off the output driver FET.

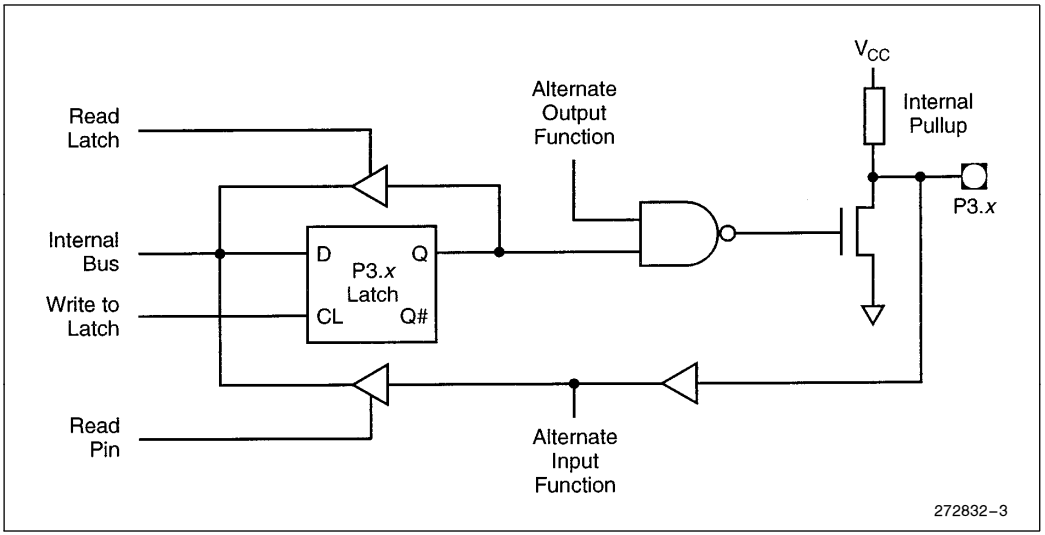


Figure 3. Port 1 and Port 3 Structure

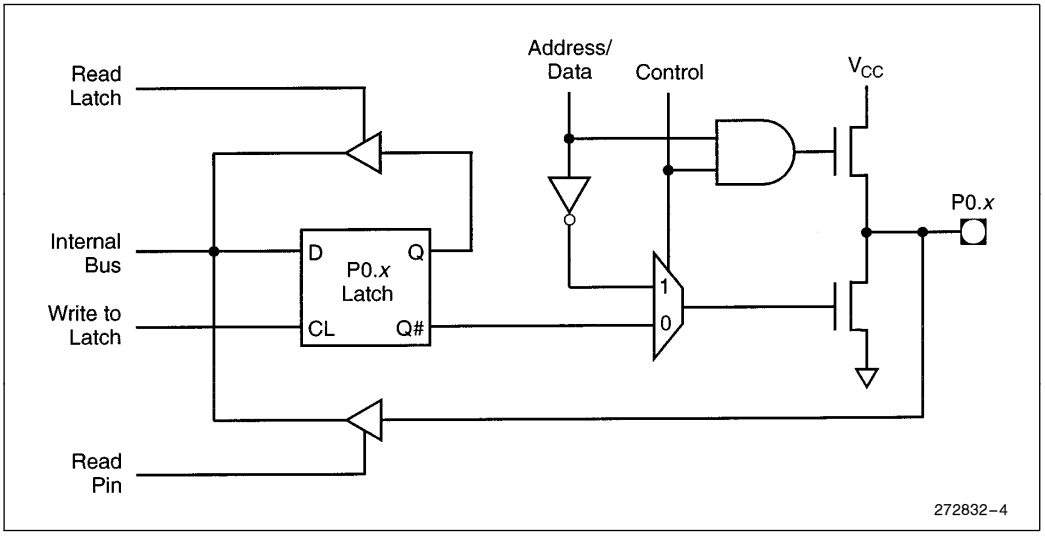


Figure 4. Port 0 Structure

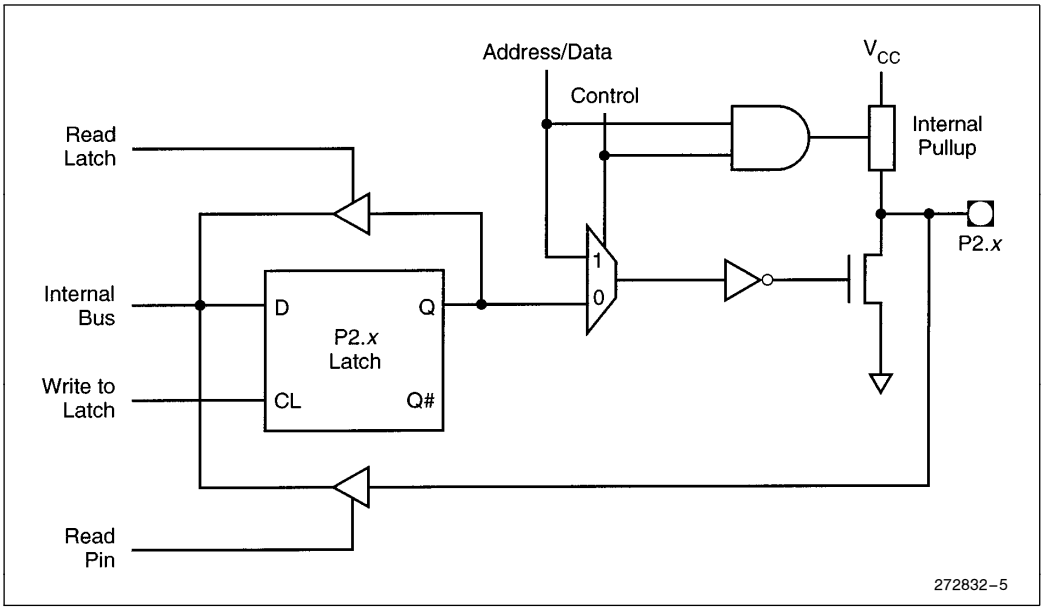


Figure 5. Port 2 Structure

When port 0 and port 2 are used for an external memory cycle, an internal control signal switches the output-driver input from the latch output to the internal address/data line. Port 0 and port 2 are precluded from use as general purpose I/O ports when used as address/data bus drivers.

Port 0 internal pullups assist the logic-one output for memory bus cycles only. Except for these bus cycles, the pullup FET is off. All other port 0 outputs are open drain.

4.4 Read-Modify-Write Instructions

Some instructions read the latch data rather than the pin data. The latch based instructions read the data, modify the data, and then rewrite the latch. These are called "read-modify-write" instructions. Below is a complete list of these special instructions. When the destination operand is a port, or a port bit, these instructions read the latch rather than the pin:

```
ANL             (logical AND, e.g., ANL
                P1, A)

ORL             (logical OR, e.g., ORL
                P2, A)

XRL             (logical EX-OR, e.g.,
                XRL P3, A)

JBC             (jump if bit = 1 and
                clear bit, e.g.,
                JBC P1.1, LABEL)

CPL             (complement bit, e.g.,
                CPL P3.0)

INC             (increment, e.g., INC
                P2)

DEC             (decrement, e.g., DEC
                P2)

DJNZ            (decrement and jump if
                not zero, e.g.,
                DJNZ P3, LABEL)

MOV PX.Y, C     (move carry bit to bit
                Y of port X)

CLR PX.Y        (clear bit Y of port X)

SETB PX.Y       (set bit Y of port X)
```

It is not obvious the last three instructions in this list are read-modify-write instructions. These instructions read the port (all 8 bits), modify the specifically addressed bit, and write the new byte back to the latch. These read-modify-write instructions are directed to the latch rather than the pin in order to avoid possible misinterpretation of voltage (and therefore, logic) levels at the pin. For example, a port bit used to drive the base of an external bipolar transistor cannot rise above the transistor's base-emitter junction voltage (a value lower than V_{IL}). With a logic one written to the bit, attempts by the CPU to read the port at the pin are misinterpreted as logic zero. A read of the latch rather than the pin returns the correct logic-one value.

4.5 Quasi-Bidirectional Port Operation

Port 1, port 2, and port 3 have fixed internal pullups and are referred to as "quasi-bidirectional" ports. When configured as an input, the pin impedance appears as logic one and sources current in response to an external logic-zero condition. Port 0 is a "true bidirectional" pin. The pin floats when configured as input. Resets write logical one to all port latches. If logical zero is subsequently written to a port latch, it can be returned to input conditions by a logical one written to the latch. For additional electrical information, refer to the current 8XC151SA/SB datasheet.

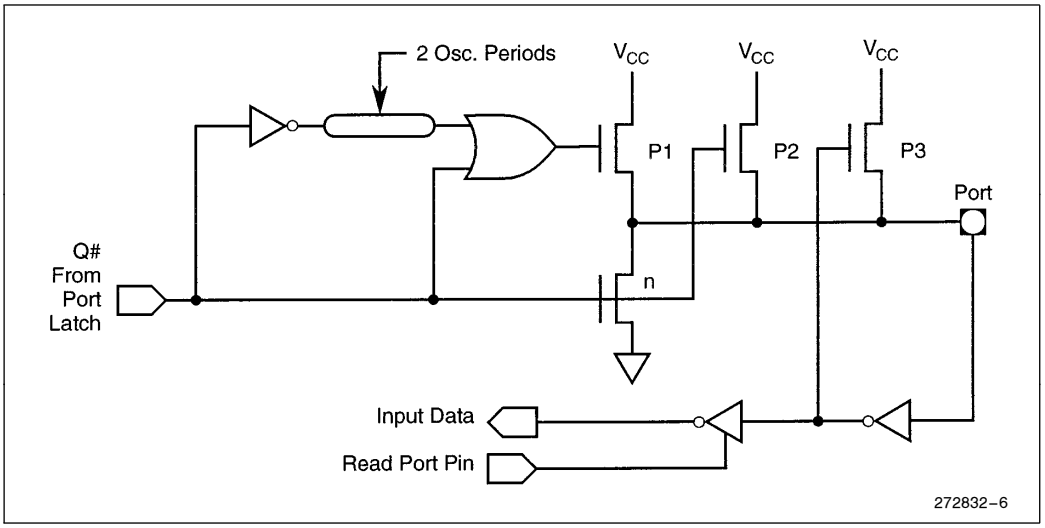


Figure 6. Internal Pullup Configurations

Logical zero-to-one transitions in port 1, port 2, and port 3 utilize an additional pullup to aid this logic transition (see Figure 6). This increases switch speed. The extra pullup briefly sources 100 times normal internal circuit current. The internal pullups are field-effect transistors rather than linear resistors. Pullups consist of three p-channel FET (pFET) devices. A pFET is on when the gate senses logical zero and off when the gate senses logical one. pFET #1 is turned on for two oscillator periods immediately after a zero-to-one transition in the port latch. A logic one at the port pin turns on pFET #3 (a weak pullup) through the inverter. This inverter and pFET pair form a latch to drive logic one. pFET #2 is a very weak pullup switched on whenever the associated nFET is switched off. This is traditional CMOS switch convention. Current strengths are $^1/_{10}$ that of pFET #3.

4.6 Port Loading

Output buffers of port 1, port 2, and port 3 can each sink 1.6 mA at logic zero (see V_{OL} specifications in the 8XC151SA/SB datasheet). These port pins can be driven by open-collector and open-drain devices. Logic zero-to-one transitions occur slowly as limited current pulls the pin to a logic-one condition (Figure 6). A logic-zero input turns off pFET #3. This leaves only pFET #2 weakly in support of the transition. In external bus mode, port 0 output buffers each sink 3.2 mA at logic zero (see V_{OL1} in the 8XC151SA/SB datasheet). However, the port 0 pins require external pullups to drive external gate inputs. See the latest revision of the 8XC151SA/SB datasheet for complete electrical design information. External circuits must be designed to limit current requirements to these conditions.

4.7 Accessing External Memory

The external memory interface comprises the external bus (ports 0 and 2) and the bus control signals. Chip configuration bytes determine several interface options: page mode or nonpage mode for external code fetches; the address ranges for RD#, WR#, and PSEN#; and the number of external wait states. You can use these options to tailor the interface to your application.

The external memory interface operates in either page mode and nonpage mode. Page mode provides increased performance by reducing the time for external code fetches. Page mode does not apply to code fetches from on-chip memory. The reset routine configures the C151SX for operation in page mode or nonpage mode according to bit 1 of configuration bytes UCONFIG0 (refer to Table 8). Figure 7 shows the structure of the external address bus for page and nonpage mode operation. P0 carries address A7:0 while P2 carries address A15:8. Data D7:0 is multiplexed with A7:0 on P0 in nonpage mode and with A15:8 on P2 in page mode.

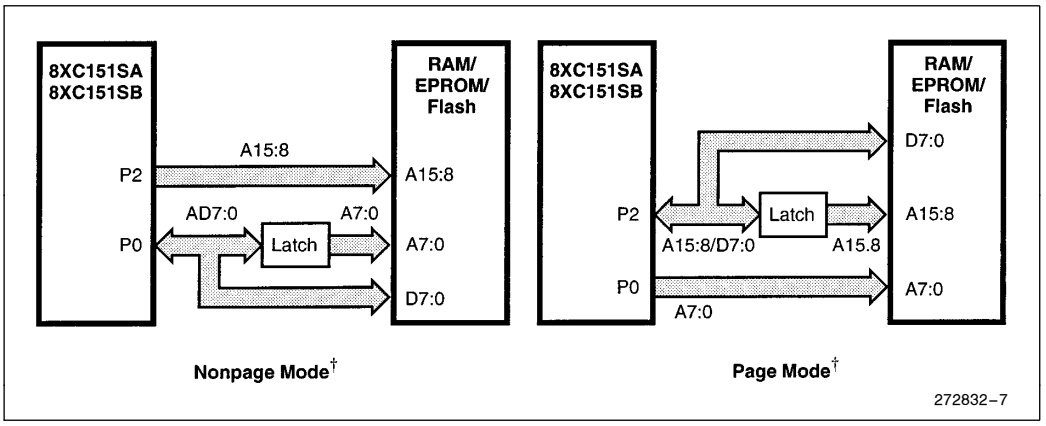


Figure 7. Bus Structure in Nonpage Mode and Page Mode

†The descriptions of A15:8/P2.7:0 and AD7:0/P0.7:0 are for the nonpage-mode chip configuration (compatible with 44-pin PLCC MCS® 51 microcontrollers). If the chip is configured for page-mode operation, port 0 carries the lower address bits (A7:0), and port 2 carries the upper address bits (A15:8) and the data (D7:0).

Table 8. User Configuration Byte UCONFIG0

UConfig0 Address: FFF8H

—	WSa1#	WSa0#	XALE#	—	—	Page#	—
MSB							LSB

Bit Mnemonic	Function
WSa1#	Wait State Select for External Code **WSa1#** **WSa0#** **Description** 1 1 No Wait State 1 0 Insert 1 Code Wait State 0 1 Insert 2 Code Wait State 0 0 Insert 3 Code Wait State
XALE#	Extended ALE **XALE#** **Description** 1 ALE pulse = 1 Clock Period 0 ALE pulse = 3 Clock Period (Additional 1 Wait State)
Page#	Page Mode Select **Page#** **Description** 1 Non-Page Mode (A15:8 on Port 2, A7:0/D7:0 on Port 0) 0 Page-Mode (A15:8/D7:0 on Port 2, A7:0 on Port 0)

Table 9. User Configuration Byte UCONFIG1

UConfig1 Address: FFF9H

—	—	—	—	—	WSb1#	WSb0#	—
MSB							LSB

Bit Mnemonic	Function
WSb1#	Wait State Select for External Data **WSb1#** **WSb0#** **Description** 1 1 No Wait State 1 0 Insert 1 Data Wait State 0 1 Insert 2 Data Wait State 0 0 Insert 3 Data Wait State

Table 10. Bus Cycle Definitions (No Wait States)

Mode	Bus Cycle	Bus Activity		
		State 1	State 2	State 3
Nonpage Mode	Code Read	ALE	RD#/PSEN#, Code In	
	Data Read (Note 2)	ALE	RD#/PSEN#	Data In
	Data Write (Note 2)	ALE	WR#	WR# High, Data Out
Page Mode	Code Read, Page Miss	ALE	RD#/PSEN#, Code In	
	Code Read, Page Hit (Note 3)	PSEN#, Code In		
	Data Read (Note 2)	ALE	RD#/PSEN#	Data In
	Data Write (Note 2)	ALE	WR#	WR# High, Data Out

NOTES:
1. Signal timing implied by this table is approximate (idealized).
2. Data read (page mode) = data read (nonpage mode) and write (page mode) = write (nonpage mode) except that in page mode data appears on P2 (multiplexed with A15:0), whereas in nonpage mode data appears on P0 (multiplexed with A7:0).
3. The initial code read page hit bus cycle can execute only following a code read page miss cycle.

4.7.1 BUS CYCLE DEFINITION

Table 10 lists the types of external bus cycles. It also shows the activity on the bus for nonpage mode and page mode bus cycles with no wait states. There are three types of nonpage mode bus cycles: code read, data read, and data write. There are four types of page mode bus cycles: code read (page miss), code read (page hit), data read, and data write. The data read and data write cycles are the same for page mode and nonpage mode (except the multiplexing of D7:0 on ports 0 and 2).

In nonpage mode, the external bus structure is the same as for MCS 51 microcontrollers. The upper address bits (A15:8) are on port 2, and the lower address bits (A7:0) are multiplexed with the data (D7:0) on port 0. External code read bus cycles execute in approximately two state times. See Table 10 and Figure 8. External data read bus cycles (Figure 9) and external write bus cycles (Figure 10) execute in approximately three state times. For the write cycle (Figure 10), a third state is appended to provide recovery time for the bus.

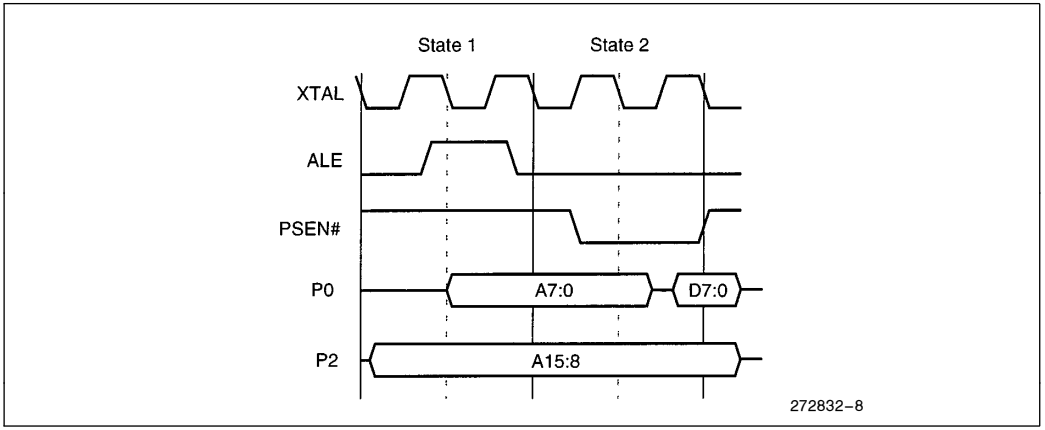


Figure 8. External Bus Cycle: Code Fetch, Nonpage Mode

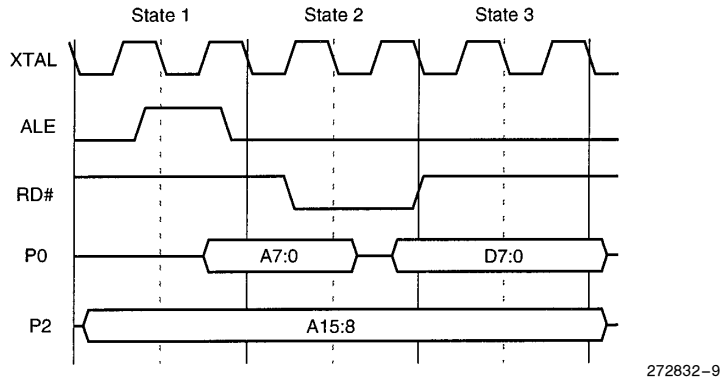


Figure 9. External Bus Cycle: Data Read, Nonpage Mode

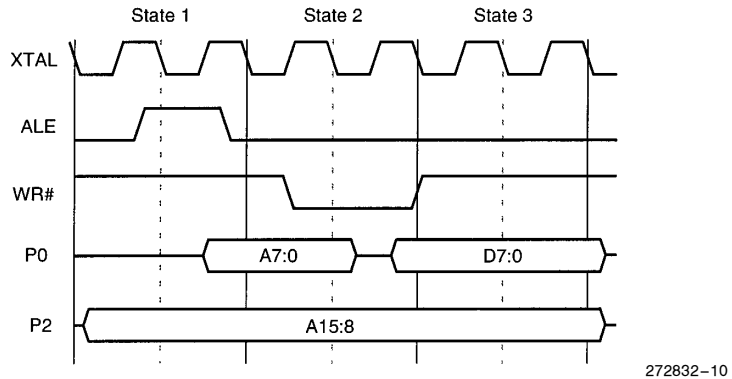


Figure 10. External Bus Cycle: Data Write, Nonpage Mode

Page mode increases performance by reducing the time for external code fetches. Under certain conditions the controller fetches an instruction from external memory in one state time instead of two. Page mode does not affect internal code fetches.

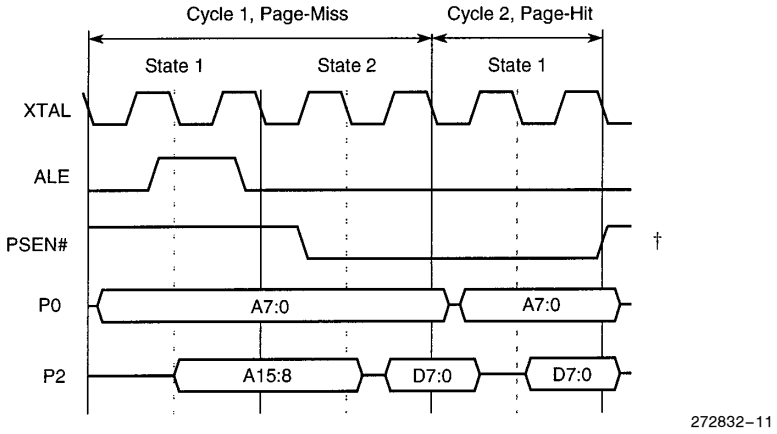
The first code fetch to a 256-byte "page" of memory always uses a two-state bus cycle. Subsequent successive code fetches to the same page (*page hits*) require only a one-state bus cycle. When a subsequent fetch is to a different page (a *page miss*) it again requires a two-state bus cycle. The following external code fetches are always page-miss cycles:

- the first external code fetch after a page rollover†
- the first external code fetch after an external data bus cycle
- the first external code fetch after powerdown or idle mode
- the first external code fetch after a branch, return, interrupt, etc.

Figure 11 shows the two types of external bus cycles for code fetches in page mode. The *page-miss* cycle is the same as a code fetch cycle in nonpage mode (except D7:0 is multiplexed with A15:8 on P2). For the *page-hit* cycle, the upper eight address bits are the same as for the preceding cycle. Therefore, ALE is not asserted, and the values of A15:8 are retained in the address latches. In a single state, the new values of A7:0 are placed on port 0, and memory places the instruction byte on port 2. Notice that a page hit reduces the available address access time by one state. Therefore, faster memories may be required to support page mode.

The bus cycles for data reads and data writes in page mode are identical to those for nonpage mode, except for the different signals on ports 0 and 2.

†A page rollover occurs when the address increments from the top of one 256-byte page to the bottom of the next (e.g., from FAFFH to FB00H).



†During a sequence of page hits, PSEN# remains low until the end of the last page-hit cycle.

Figure 11. External Bus Cycle: Code Fetch, Page Mode

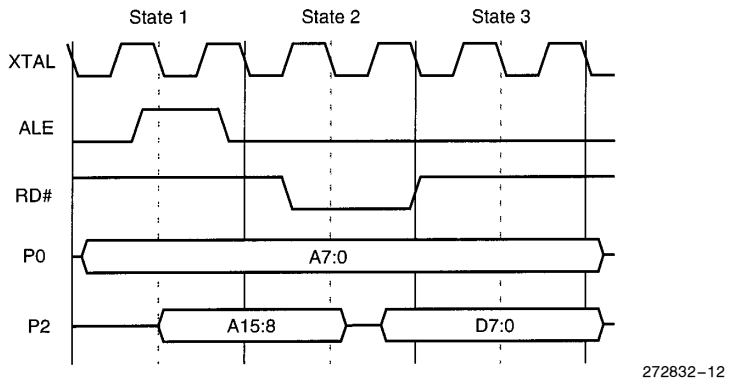


Figure 12. External Bus Cycle: Data Read, Page Mode

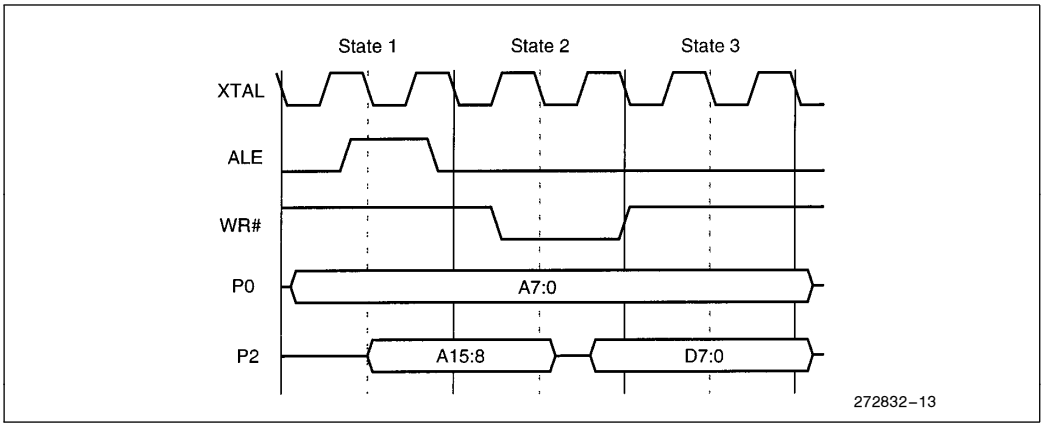


Figure 13. External Bus Cycle: Data Write, Page Mode

4.7.2 EXTERNAL BUS CYCLES WITH WAIT STATES

The C151SX can be configured to add wait states to the external bus cycles by extending the RD#/WR#/PSEN# pulses or by extending the ALE pulse. Configuration bites WAS1:0# and WSB1:0# specify 0, 1, 2, or 3 wait states for RD#/WR#/PSEN#. The XALE# configuration bit specifies 0 or 1 wait state for ALE. You can also configure the chip to use both types of wait states. Accesses to on-chip code and data memory always use zero wait states.

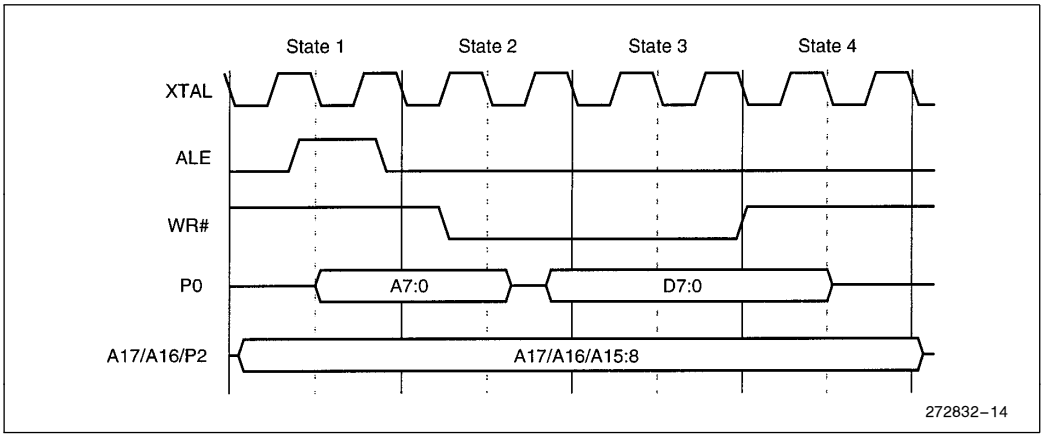


Figure 14. External Bus Cycle: Data Write with One WR# Wait State (Nonpage Mode)

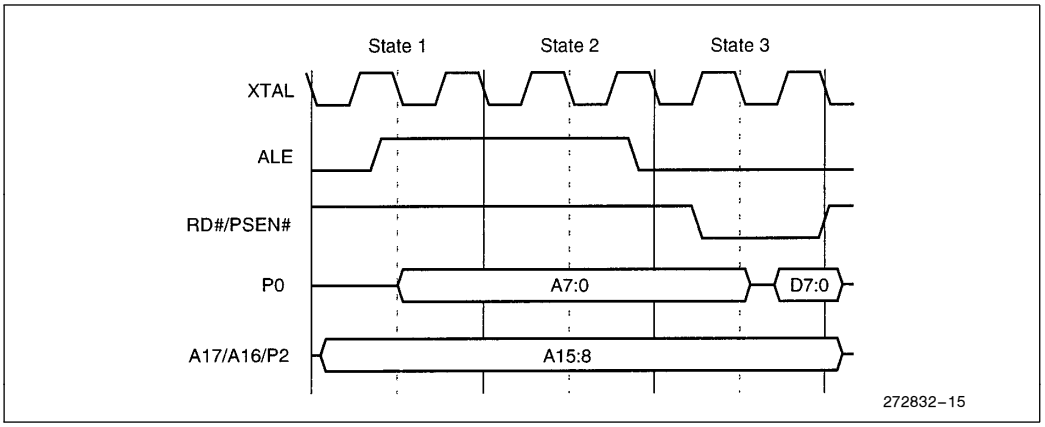


Figure 15. External Bus Cycle: Code Fetch with One ALE Wait State (Nonpage Mode)

Figure 15 shows the nonpage mode code fetch external bus cycle with ALE extended. The wait state extends the bus cycle from two states to three. For read and write external bus cycles, the extended ALE extends the bus cycle from three states to four.

4.7.3 PORT 0 AND PORT 2 FOR STATES IN NON-PAGE AND PAGE MODE

In nonpage mode the port pins have the same signals as those on the 8XC51FX. For an external memory instruction using a 16-bit address, the port pins carry address and data bits during the bus cycle. However, if the instruction uses an 8-bit address (e.g., MOVX@Ri), the contents of P2 are driven onto the pins. These pin signals can be used to select 256-bit pages in external memory.

During a bus cycle, the CPU always writes FFH to P0, and the former contents of P0 are lost. A bus cycle does not change the contents of P2. When the bus is idle, the port 0 pins are held at high impedance, and the contents of P2 are driven onto the port 2 pins.

In a page-mode bus cycle, the data is multiplexed with the upper address byte on port 2. However, if the instruction uses an 8-bit address (e.g., MOVX@Ri), the contents of P2 are driven onto the pins when data is not on the pins. These logic levels can be used to select 256-bit pages in external memory. During bus idle, the port 0 and port 2 pins are held at high impedance.

5.0 TIMERS/COUNTERS

The C151SX has three 16-bit Timer/Counters: Timer 0, Timer 1, and Timer 2. Each consists of two 8-bit registers, THx and TLx, (x = 0, 1, and 2). All three can be configured to operate either as timers or event counters.

In the Timer function, the TLx register is incremented every machine cycle. Thus one can think of it as counting machine cycles. Since a machine cycle consists of 12 oscillator periods, the count rate is $1/12$ of the oscillator frequency.

In the Counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin—T0, T1, or T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since it takes 2 machine cycles (24 oscillator periods) to recognize a 1-to-0 transition, the maximum count rate is $1/24$ of the oscillator frequency. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it should be held for at least one full machine cycle.

In addition to the Timer or Counter selection, Timer 0 and Timer 1 have four operating modes from which to select: Modes 0 – 3. Timer 2 has three modes of operation: Capture, Auto-Reload, and Baud Rate Generator.

5.1 Timer 0 and Timer 1

The Timer or Counter function is selected by control bits C/T# in the Special Function Register TMOD. These two Timer/Counters have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD. Modes 0, 1, and 2 are the same for both Timer/Counters. Mode 3 operation is different for the two timers.

MODE 0

Either Timer 0 or Timer 1 in Mode 0 is an 8-bit Counter with a divide-by-32 prescaler. Figure 16 shows the Mode 0 operation for either timer.

In this mode, the Timer register is configured as a 13-bit register. As the count rolls over from all 1s to all 0s, it sets the Timer interrupt flag TFx. The counted input is enabled to the Timer when TRx = 1 and either GATE = 0 or INTx# = 1. (Setting GATE = 1 allows the Timer to be controlled by external input INTx#, to facilitate pulse width measurements). TRx and TFx are control bits in SFR TCON. The GATE bit is in TMOD. There are two different GATE bits, one for Timer 1 (TMOD.7) and one for Timer 0 (TMOD.3).

The 13-bit register consists of all 8 bits of THx and the lower 5 bits of TLx. The upper 3 bits of TLx are indeterminate and should be ignored. Setting the run flag (TRx) does not clear these registers.

MODE 1

Mode 1 is the same as Mode 0, except that the Timer register uses all 16 bits. Refer to Figure 17. In this mode, THx and TLx are cascaded; there is no prescaler.

MODE 2

Mode 2 configures the Timer register as an 8-bit Counter (TLx) with automatic reload, as shown in Figure 18. Overflow from TLx not only sets TFx, but also reloads TLx with the contents of THx, which is preset by software. The reload leaves THx unchanged.

TMOD: Timer/Counter Mode Control Register

TMOD Address: 089H
Reset State: 0000 0000B

Not Bit Addressable

7			TIMER 1				Timer 0	0
GATE	C/T#	M1	M0	GATE	C/T#	M1	M0	

Symbol	Function
GATE	Gating control when set. Timer/Counter 0 or 1 is enabled only while INT0# or INT1# pin is high and TR0 or TR1 control pin is set. When cleared, Timer 0 or 1 is enabled whenever TR0 or TR1 control bit is set.
C/T#	Timer or Counter Selector. Clear for Timer operation (input from internal system clock). Set for Counter operation (input from T0 or T1 input pin).

M1	M0	Operating Mode
0	0	8-bit Timer/Counter. THx with Tlx as 5-bit prescaler.
0	1	16-bit Timer/Counter. THx and TLx are cascaded; there is no prescaler.
1	0	8-bit auto-reload Timer/Counter. THx holds a value which is to be reloaded into TLx each time it overflows.
1	1	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits.
1	1	(Timer 1) Timer/Counter stopped.

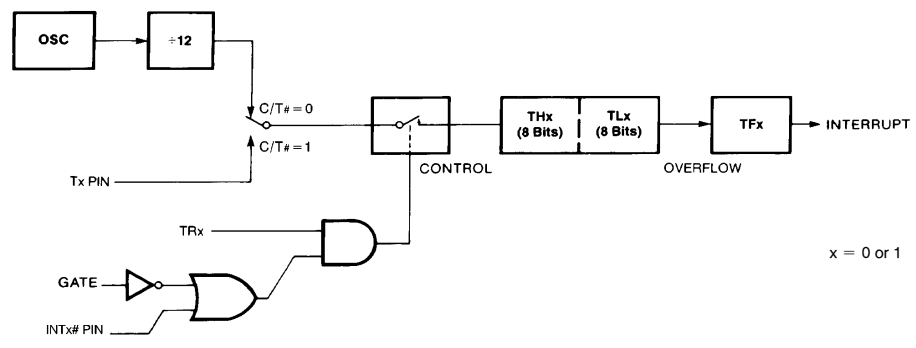


Figure 16. Timer/Counter 0 or 1 in Mode 0: 13-Bit Counter

TCON: Timer/Counter Control Register

TCON Address: 088H
Reset State: 0000 0000B

Bit Addressable

7							0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Symbol	Function
TF1	Timer 1 overflow Flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine.
TR1	Timer 1 Run control bit. Set/cleared by software to turn Timer/Counter 1 on/off.
TF0	Timer 0 overflow Flag. Set by hardware on Timer/Counter 0 overflow. Cleared by hardware when processor vectors to interrupt routine.
TR0	Timer 0 Run control bit. Set/cleared by software to turn Timer/Counter 0 on/off.
IE1	Interrupt 1 flag. Set by hardware when external interrupt 1 edge is detected (transmitted or level-activated). Cleared when interrupt processed only if transition-activated.
IT1	Interrupt 1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupt 1.
IE0	Interrupt 0 flag. Set by hardware when external interrupt 0 edge is detected (transmitted or level-activated). Cleared when interrupt processed only if transition-activated.
IT0	Interrupt 0 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupt 0.

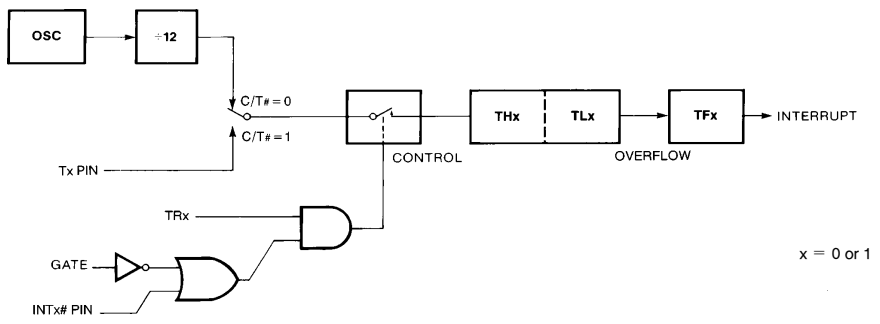


Figure 17. Timer/Counter 0 or 1 in Mode 1: 16-Bit Counter

MODE 3

Timer 1 in Mode 3 simply holds its count. The effect is the same as setting TR1 = 0.

Timer 0 in Mode 3 establishes TL0 and TH0 as two separate counters. The logic for Mode 3 on Timer 0 is shown in Figure 19. TL0 uses the Timer 0 control bits: C/T#, GATE, TR0, INT0#, and TF0. TH0 is locked

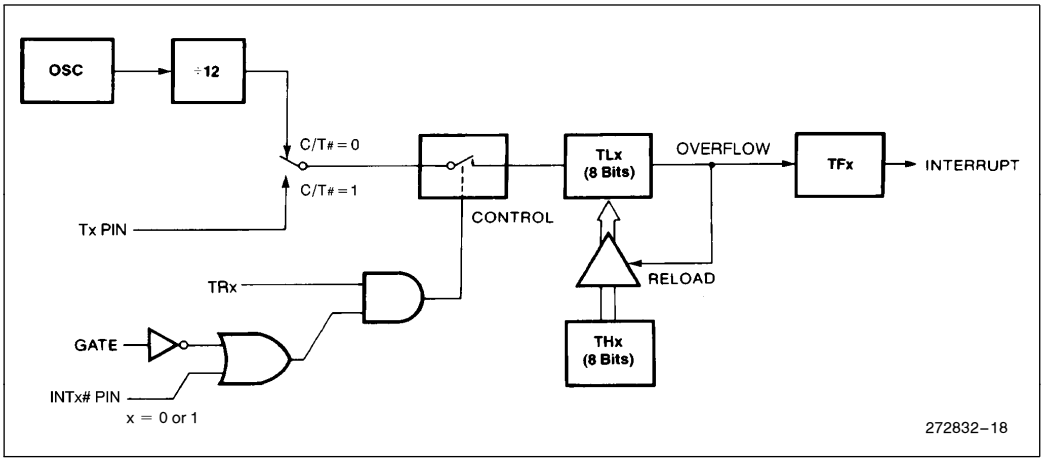


Figure 18. Timer/Counter 1 Mode 2: 8-Bit Auto-Reload

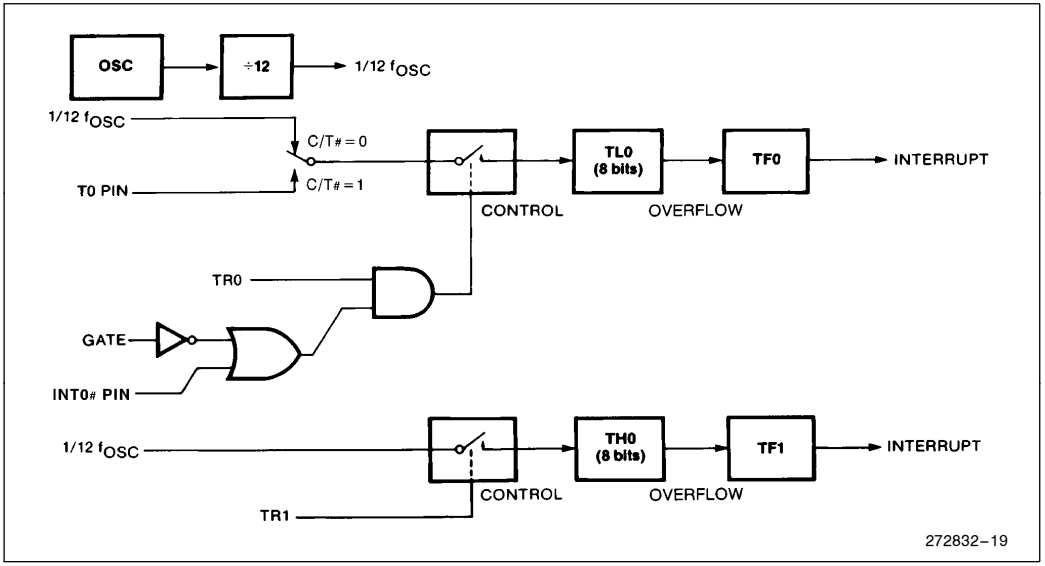


Figure 19. Timer/Counter 0 Mode 3: Two 8-Bit Counters

into a timer function (counting machine cycles) and takes over the use of TR1 and TF1 from Timer 1. Thus TH0 now controls the Timer 1 interrupt.

Mode 3 is provided for applications requiring an extra 8-bit timer or counter. When Timer 0 is in Mode 3, Timer 1 can be turned on and off by switching it out of and into its own Mode 3, or can still be used by the serial port as a baud rate generator, or in any application not requiring an interrupt.

5.2 Timer 2

Timer 2 is a 16-bit Timer/Counter which can operate either as a timer or as an event counter. This is selected by bit C/T2# in the Special Function Register T2CON. It has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON as shown in Table 11.

Table 11. Timer 2 Operating Modes

RCLK + TCLK	CP/RL2#	T2*OE	TR2	Mode
0	0	0	1	16-Bit Auto-Reload
0	1	0	1	16-Bit Capture
1	X	X	1	Baud__Rate Generator
X	0	1	1	Clock-Out on P1.0
X	X	X	0	Timer Off

T2CON: Timer/Counter 2 Control Register

T2CON Address: 0C8H
Reset State: 0000 0000B

Bit Addressable

7							0
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2#	CP/RL2#

Symbol	Function
TF2	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
EXF2	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
RCLK	Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	Timer 2 external enable flag. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	Start/stop control for Timer 2. A logic 1 starts the timer.
C/T2#	Timer or counter select. (Timer 2) 0 = Internal timer (OSC/12 or OSC/2 in baud rate generator mode). 1 = External event counter (falling edge triggered).
CP/RL2#	Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. When cleared, auto-reloads will occur either with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

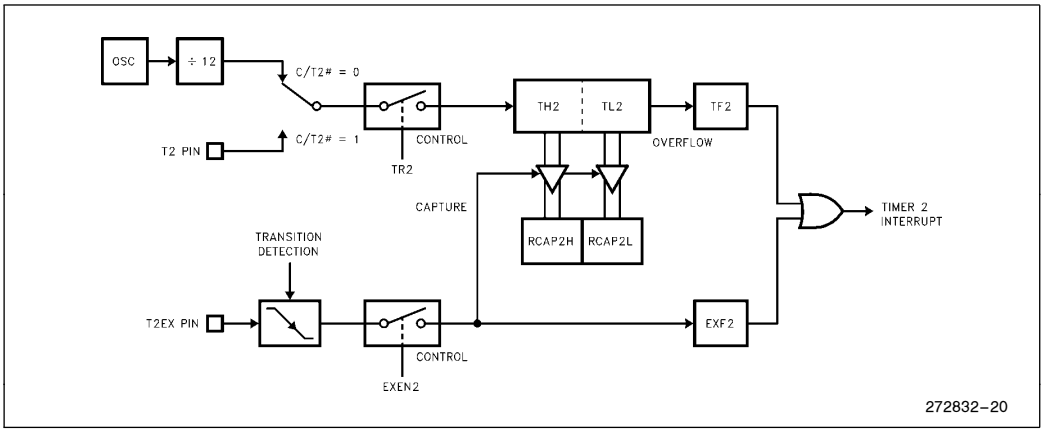


Figure 20. Timer 2 in Capture Mode

CAPTURE MODE

In the capture mode there are two options selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16-bit timer or counter which upon overflow sets bit TF2 in T2CON. This bit can then be used to generate an interrupt. If EXEN2 = 1, Timer 2 still does the above, but with the added feature that a 1-to-0 transition at external input T2EX causes the current value in the Timer 2 registers, TH2 and TL2, to be captured into registers RCAP2H and RCAP2L, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. The EXF2 bit, like TF2, can generate an interrupt. The capture mode is illustrated in Figure 20.

AUTO-RELOAD MODE (UP OR DOWN COUNTER)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by a bit named DCEN (Down Counter Enable) located in the SFR T2MOD. Upon reset the DCEN bit is set to 0 so that Timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down depending on the value of the T2EX pin.

T2MOD: Timer 2 Mode Control Register

T2MOD Address: 0C9H
Reset State: XXXX XX00B

Not Bit Addressable

7							0
—	—	—	—	—	—	T2OE	DCEN

Symbol	Function
—	Not implemented, reserved for future use.*
T2OE	Timer 2 Output Enable bit.
DCEN	Down Count Enable bit. When set, this allows Timer 2 to be configured as an up/down counter.

NOTE:
*User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

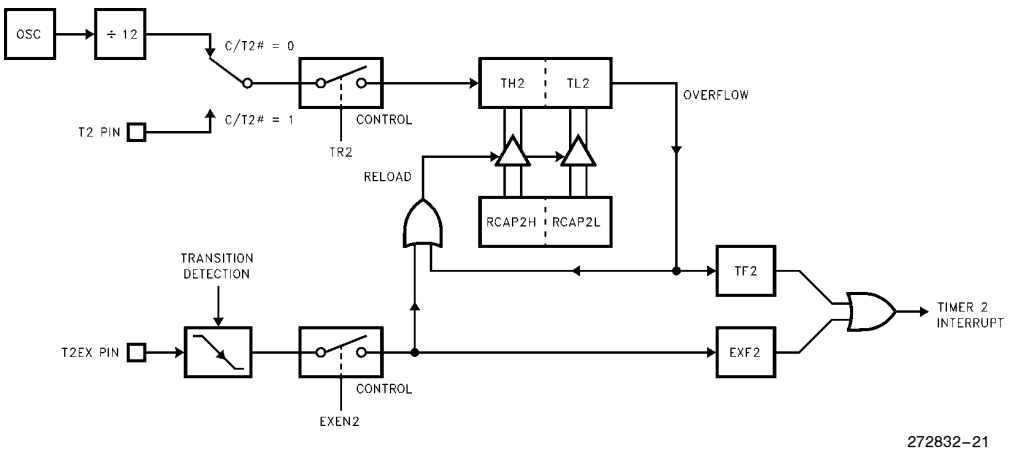


Figure 21. Timer 2 Auto Reload Mode (DCEN = 0)

Figure 21 shows Timer 2 automatically counting up when DCEN = 0. In this mode there are two options selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H and RCAP2L. The values in RCAP2H and RCAP2L are preset by software. If EXEN2 = 1, a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at external input T2EX. This transition also sets the EXF2 bit. Either the TF2 or EXF2 bit can generate the Timer 2 interrupt if it is enabled.

Setting the DCEN bit enables Timer 2 to count up or down as shown in Figure 22. In this mode the T2EX pin controls the direction of count. A logic 1 at T2EX makes Timer 2 count up. The timer will overflow at 0FFFFH and set the TF2 bit which can then generate an interrupt if it is enabled. This overflow also causes the 16-bit value in RCAP2H and RCAP2L to be reloaded into the timer registers, TH2 and TL2, respectively.

A logic 0 at T2EX makes Timer 2 count down. Now the timer underflows when TH2 and TL2 equal the values stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and causes 0FFFFH to be reloaded into the timer registers.

The EXF2 bit toggles whenever Timer 2 overflows or underflows. This bit can be used as a 17th bit of resolution if desired. In this operating mode, EXF2 does not generate an interrupt.

BAUD RATE GENERATOR MODE

The baud rate generator mode is selected by setting the RCLK and/or TCLK bits in T2CON. Timer 2 in this mode will be described in conjunction with the serial port.

PROGRAMMABLE CLOCK OUT

A 50% duty cycle clock can be programmed to come out on P1.0. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed (1) to input the external clock for Timer/Counter 2 or (2) to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz at a 16 MHz operating frequency.

To configure the Timer/Counter 2 as a clock generator, bit C/T2# (in T2CON) must be cleared and bit T2OE in T2MOD must be set. Bit TR2 (T2CON.2) also must be set to start the timer.

The Clock-out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L) as shown in this equation:

$$\text{Clock-out Frequency} = \frac{\text{Oscillator Frequency}}{4 \times (65536 - \text{RCAP2H, RCAP2L})}$$

In the Clock-Out mode Timer 2 roll-overs will not generate an interrupt. This is similar to when Timer 2 is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and Clock-out frequencies cannot be determined independently of one another since they both use the values in RCAP2H and RCAP2L.

6.0 PROGRAMMABLE COUNTER ARRAY

The Programmable Counter Array (PCA) consists of a 16-bit timer/counter and five 16-bit compare/capture modules as shown in Figure 24. The PCA timer/counter serves as a common time base for the five modules and is the only timer which can service the PCA. Its clock input can be programmed to count any one of the following signals:

- oscillator frequency ÷ 12
- oscillator frequency ÷ 4
- Timer 0 overflow
- external input on ECI (P1.2).

Each compare/capture module can be programmed in any one of the following modes:

- rising and/or falling edge capture
- software timer
- high speed output
- pulse width modulator.

Module 4 can also be programmed as a watchdog timer.

When the compare/capture modules are programmed in the capture mode, software timer, or high speed output mode, an interrupt can be generated when the module executes its function. All five modules plus the PCA timer overflow share one interrupt vector (more about this in the PCA Interrupt section).

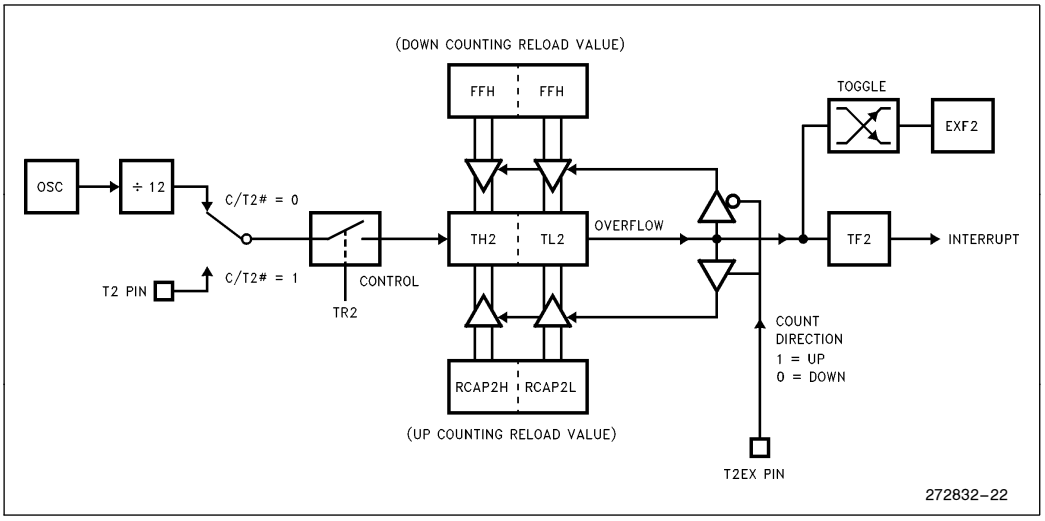


Figure 22. Timer 2 Auto Reload Mode (DCEN = 1)

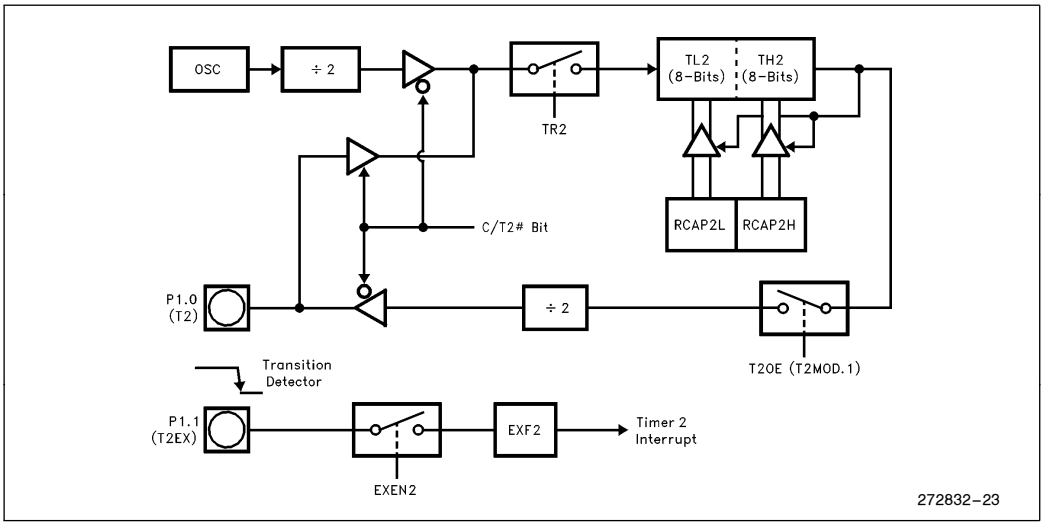


Figure 23. Timer 2 in Clock-Out Mode

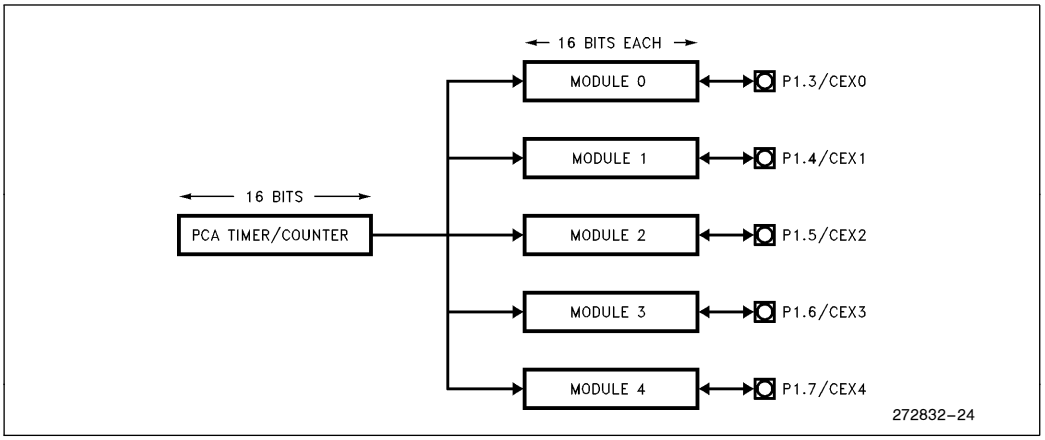


Figure 24. Programmable Counter Array

The PCA timer/counter and compare/capture modules share Port 1 pins for external I/O. These pins are listed below. If the port pin is not used for the PCA, it can still be used for standard I/O.

PCA Component	External I/O Pin
16-bit Counter	P1.2 / ECI
16-bit Module 0	P1.3 / CEX0
16-bit Module 1	P1.4 / CEX1
16-bit Module 2	P1.5 / CEX2
16-bit Module 3	P1.6 / CEX3
16-bit Module 4	P1.7 / CEX4

6.1 PCA 16-Bit Timer/Counter

The PCA has a free-running 16-bit timer/counter consisting of registers CH and CL (the high and low bytes of the count value). These two registers can be read or written to at any time. Figure 25 shows a block diagram of this timer. The clock input can be selected from the following four modes:

- Oscillator frequency ÷ 12
  The CL register is incremented at S5P2 of every machine cycle. With a 16 MHz crystal, the timer increments every 750 nanoseconds.
- Oscillator frequency ÷ 4
  The CL register is incremented at S1P2, S3P2 and S5P2 of every machine cycle. With a 16 MHz crystal, the timer increments every 250 nanoseconds.
- Timer 0 overflows
  The CL register is incremented at S5P2 of the machine cycle when Timer 0 overflows. This mode allows a programmable input frequency to the PCA.
- External input
  The CL register is incremented at the first one of S1P2, S3P2 and S5P2 after a 1-to-0 transition is detected on the ECI pin (P1.2). P1.2 is sampled at S1P2, S3P2 and S5P2 of every machine cycle. The maximum input frequency in this mode is oscillator frequency ÷ 8.

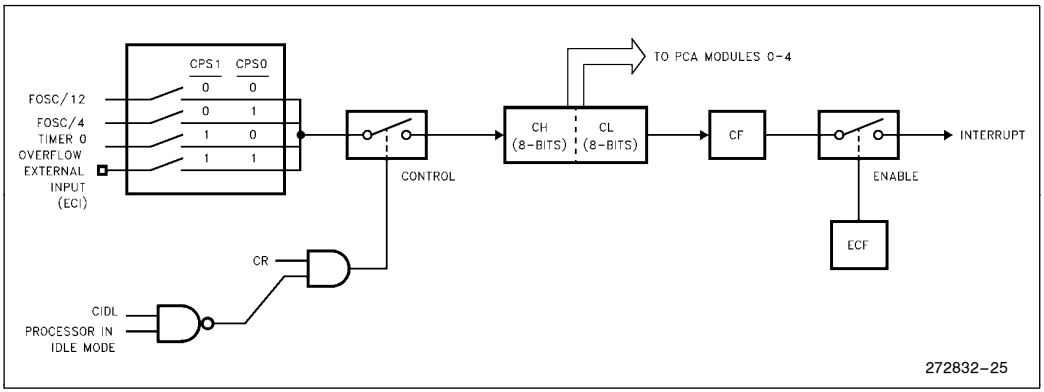


Figure 25. PCA Timer/Counter

CH is incremented after two oscillator periods when CL overflows.

The mode register CMOD contains the Count Pulse Select bits (CPS1 and CPS0) to specify the clock input. This register also contains the ECF bit which enables the PCA counter overflow to generate the PCA interrupt. In addition, the user has the option of turning off the PCA timer during Idle Mode by setting the Counter Idle bit (CIDL). The Watchdog Timer Enable bit (WDTE) will be discussed in a later section.

The CCON register contains two more bits which are associated with the PCA timer/counter. The CF bit gets set by hardware when the counter overflows, and the CR bit is set or cleared to turn the counter on or off. The other five bits in this register are the event flags for the compare/capture modules and will be discussed in the next section.

CMOD: PCA Counter Mode Register

CMOD Address: 0D9H
Reset State: 00XX X000B

Not Bit Addressable

7							0
CIDL	WDTE	—	—	—	CPS1	CPS0	ECF

Symbol	Function
CIDL	Counter idle control: CIDL = 0 programs the PCA Counter to continue functioning during idle Mode. CIDL = 1 programs it to be gated off during idle.
WDTE	Watchdog Timer Enable: WDTE = 0 disables Watchdog Timer function on PCA Module 4. WDTE = 1 enables it.
—	Not implemented, reserved for future use.*
CPS1	PCA Count Pulse Select bit 1.
CPS0	PCA Count Pulse Select bit 0. **CPS1 CPS0 Selected PCA Input**** 0 0 Internal clock, Fosc ÷ 12 0 1 Internal clock, Fosc ÷ 4 1 0 Timer 0 overflow 1 1 External clock at ECI/P1.2 pin (max. rate = Fosc ÷8)
ECF	PCA Enable Counter Overflow interrupt: ECF = 1 enables CF bit in CCON to generate an interrupt. ECF = 0 disables that function of CF.

NOTE:
*User software should not write 1s to reserved bits. These bits may be used in future C151 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.
**Fosc = oscillator frequency

6.2 Capture/Compare Modules

Each of the five compare/capture modules has six possible functions it can perform:

— 16-bit Capture, positive-edge triggered
— 16-bit Capture, negative-edge triggered
— 16-bit Capture, both positive and negative-edge triggered
— 16-bit Software Timer
— 16-bit High Speed Output
— 8-bit Pulse Width Modulator.

In addition, module 4 can be used as a Watchdog Timer. The modules can be programmed in any combination of the different modes.

Each module has a mode register called CCAPMn (n = 0, 1, 2, 3, or 4) to select which function it will perform. Note the ECCFn bit which enables the PCA interrupt when a module's event flag is set. The event flags (CCFn) are located in the CCON register and get set when a capture event, software timer, or high speed output event occurs for a given module.

CCON: PCA Counter Control Register

CCON Address: 0D8H
Reset State: 00X0 0000B

Bit Addressable

7							0
CF	CR	—	CCF4	CCF3	CCF2	CCF1	CCF0

Symbol	Function
CF	PCA Counter Overflow flag. Set by hardware when the counter rolls over. CD flags an interrupt if bit ECF in CMOD is set. CF may be set by either hardware or software but can only be cleared by software.
CR	PCA Counter Run control bit. Set by software to turn the PCA counter on. Must be cleared by software to turn the PCA counter off.
—	Not implemented, reserved for future use.*
CCF4	PCA Module 4 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF3	PCA Module 3 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF2	PCA Module 2 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF1	PCA Module 1 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF0	PCA Module 0 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.

NOTE:
*User software should not write 1s to reserved bits. The value read from a reserved bit is indeterminate.

PCA Module Modes (CCAPMn Register) shows the combinations of bits in the CCAPMn register that are valid and have a defined function. Invalid combinations will produce undefined results.

Each module also has a pair of 8-bit compare/capture registers (CCAPnH and CCAPnL) associated with it. These registers store the time when a capture event occurred or when a compare event should occur. For the PWM mode, the high byte regiser CCAPnH controls the duty cycle of the waveform.

The next five sections describe each of the compare/capture modes in detail.

CCAPMn: PCA Modules Compare/Capture Registers

CCAPMn
(n = 0–4)

Address CCAPM0: 0DAH
CCAPM1: 0DBH
CCAPM2: 0DCH
CCAPM3: 0DDH
CCAPM3: 0DEH
Reset State: X000 0000B

Not Bit Addressable

7							0
—	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn

Symbol	Function
—	Note implemented, reserved for future use.*
ECOMn	Enable Comparator. ECOMn = 1 enables the comparator function.
CAPPn	Capture Positive, CAPPn = 1 enables positive edge capture.
CAPNn	Capture Negative, CAPNn = 1 enables negative edge capture.
MATn	Match. When MATn = 1, a match of the PCA counter with this module's compare/capture register causes the CCFn bit in CCON to be set, flagging an interrupt.
TOGn	Toggle. When TOGn = 1, a match of the PCA counter with this module's compare/ capture register causes the CEXn pin to toggle.
PWMn	Pulse Width Modulation Mode. PWMn = 1 enables the CEXn pin to be used as a pulse width modulated output.
ECCFn	Enable CCF Interrupt. Enables compare/capture flag CCFn in the CCON register to generate an interrupt.

NOTE:
*User software should not write 1s to reserved bits. The value read from a reserved bit is indeterminate.

PCA Module Modes (CCAPMn Register)

—	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	Module Function
X	0	0	0	0	0	0	0	No operation
X	X	1	0	0	0	0	X	16-bit capture by a postive-edge trigger on CEXn
X	X	0	1	0	0	0	X	16-bit capture by a negative-edge trigger on CEXn
X	X	1	1	0	0	0	X	16-bit capture by a transition on CEXn
X	1	0	0	1	0	0	X	16-bit Software Timer
X	1	0	0	1	1	0	X	16-bit High Speed Output
X	1	0	0	0	0	1	0	8-bit PWM
X	1	0	0	1	x	0	x	Watchdog Timer

X = Don't Care

6.3 16-Bit Capture Mode

Both positive and negative transitions can trigger a capture with the PCA. This gives the PCA the flexibility to measure periods, pulse widths, duty cycles, and phase differences on up to five separate inputs. Setting the CAPPn and/or CAPNn in the CCAPMn mode register select the input trigger—positive and/or negative transition—for module n. Refer to Figure 26.

The external input pins CEX0 through CEX4 are sampled for a transition. When a valid transition is detected (positive and/or negative edge), hardware loads the 16-bit value of the PCA timer (CH, CL) into the module's capture registers (CCAPnH, CCAPnL). The resulting value in the capture registers reflects the PCA timer value at the time a transition was detected on the CEXn pin.

Upon a capture, the module's event flag (CCFn) in CCON is set, and an interrupt is flagged if the ECCFn bit in the mode register CCAPMn is set. The PCA interrupt will then be generated if it is enabled. Since the hardware does not clear an event flag when the interrupt is vectored to, the flag must be cleared in software. In the interrupt service routine, the 16-bit capture value must be saved in RAM before the next capture event occurs. A subsequent capture on the same CEXn pin will write over the first capture value in CCAPnH and CCAPnL.

6.4 16-Bit Software Timer Mode

In the compare mode, the 16-bit value of the PCA timer is compared with a 16-bit value pre-loaded in the module's compare registers (CCAPnH, CCAPnL). The comparison occurs three times per machine cycle in order to recognize the fastest possible clock input (i.e. ¼ x oscillator frequency). Setting the ECOMn bit in the mode register CCAPMn enables the comparator function as shown in Figure 27.

For the Software Timer mode, the MATn bit also needs to be set. When a match occurs between the PCA timer and the compare registers, a match signal is generated and the module's event flag (CCFn) is set. An interrupt is then flagged if the ECCFn bit is set. The PCA interrupt is generated only if it has been properly enabled. Software must clear the event flag before the next interrupt will be flagged.

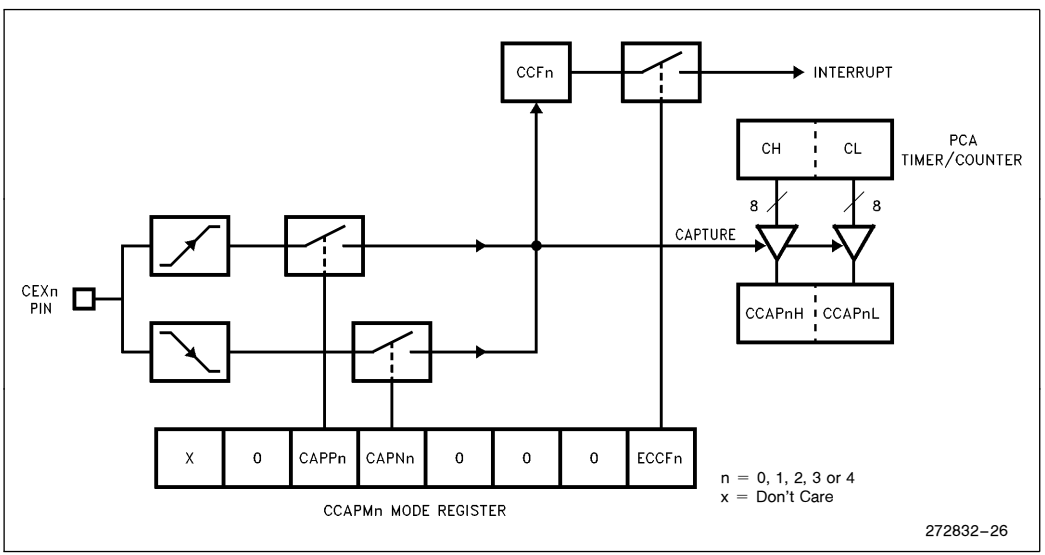


Figure 26. PCA 16-Bit Capture Mode

During the interrupt routine, a new 16-bit compare value can be written to the compare registers (CCAPnH and CCAPnL). *Notice, however, that a write to CCAPnL clears the ECOMn bit which temporarily disables the comparator function while these registers are being updated so an invalid match does not occur.* A write to CCAPnH sets the ECOMn bit and re-enables the comparator. For this reason, user software should write to CCAPnL first, then CCAPnH.

6.5 High Speed Output Mode

The High Speed Output (HSO) mode toggles a CEXn pin when a match occurs between the PCA timer and a pre-loaded value in a module's compare registers. For this mode, the TOGn bit needs to be set in addition to the ECOMn and MATn bits as seen in Figure 27. By setting or clearing the pin in software, the user can select whether the CEXn pin will change from a logical 0 to a logical 1 or vice versa. The user also has the option of flagging an interrupt when a match event occurs by setting the ECCFn bit.

The HSO mode is more accurate than toggling port pins in software because the toggle occurs *before* branching to an interrupt. That is, interrupt latency will not effect the accuracy of the output. If the user does not change the compare registers in an interrupt routine, the next toggle will occur when the PCA timer rolls over and matches the last compare value.

6.6 Watchdog Timer Mode

A Watchdog Timer is a circuit that automatically invokes a reset unless the system being watched sends regular hold-off signals to the Watchdog. These circuits are used in applications that are subject to electrical noise, power glitches, electrostatic discharges, etc., or where high reliability is required.

The Watchdog Timer function is only available on PCA module 4. In this mode, every time the count in the PCA timer matches the value stored in module 4's compare registers, an internal reset is generated. (See Figure 28.) The bit that selects this mode is WDTE in the CMOD register. Module 4 must be set up in either compare mode as a Software Timer or High Speed Output.

When the PCA Watchdog Timer times out, it resets the chip just like a hardware reset, except that it does not drive the reset pin high.

To hold off the reset, the user has three options:

(1) periodically change the compare value so it will never match the PCA timer,

(2) periodically change the PCA timer value so it will never match the compare value,

(3) disable the Watchdog by clearing the WDTE bit before a match occurs and then later re-enable it.

The first two options are more reliable because the Watchdog Timer is never disabled as in option #3. The second option is not recommended if other PCA modules are being used since this timer is the time base for all five modules. Thus, in most applications the first solution is the best option.

If a Watchdog Timer is not needed, module 4 can still be used in other modes.

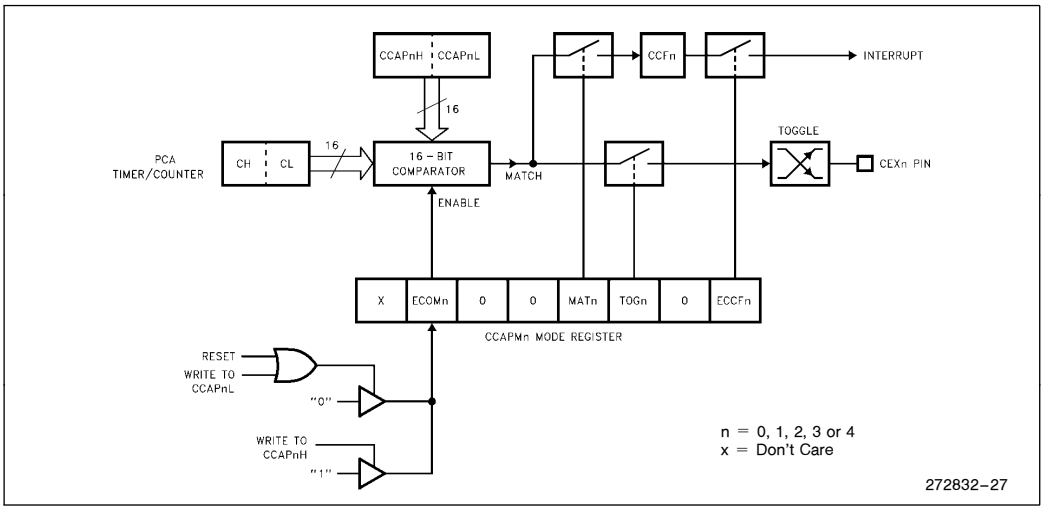


Figure 27. PCA 16-Bit Comparator Mode: Software Timer and High Speed Output

6.7 Pulse Width Modulator Mode

Any or all of the five PCA modules can be programmed to be a Pulse Width Modulator. The PWM output can be used to convert digital data to an analog signal by simple external circuitry. The frequency of the PWM depends on the clock sources for the PCA timer. With a 16 MHz crystal the maximum frequency of the PWM waveform is 15.6 KHz.

The PCA generates 8-bit PWMs by comparing the low byte of the PCA timer (CL) with the low byte of the module's compare registers (CCAPnL). Refer to Figure 29. When CL $<$ CCAPnL the output is low. When CL $\geq$ CCAPnL the output is high. The value in CCAPnL controls the duty cycle of the waveform. To change the value in CCAPnL without output glitches, the user must write to the high byte register (CCAPnH). This value is then shifted by hardware into CCAPnL when CL rolls over from 0FFH to 00H which corresponds to the next period of the output.

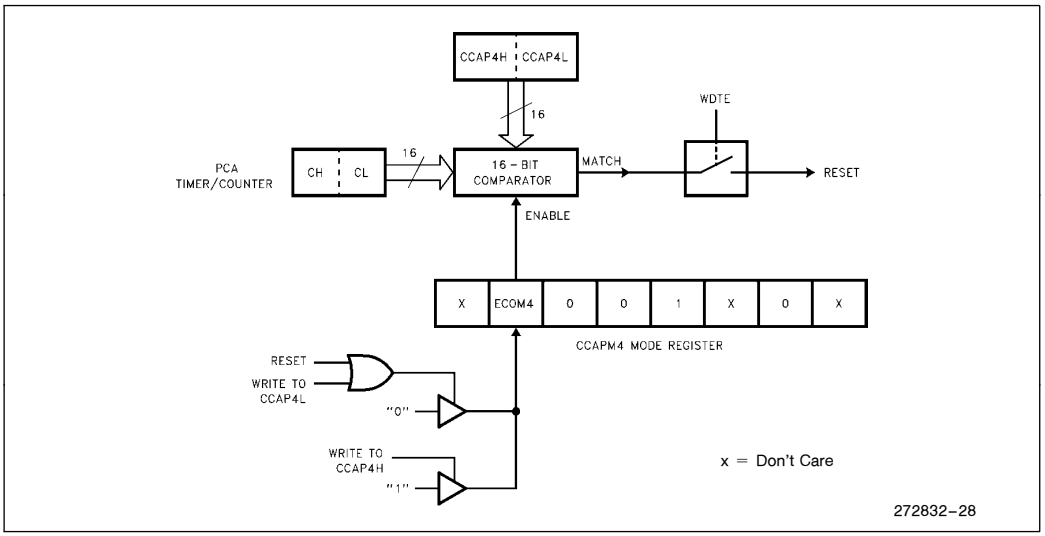


Figure 28. Watchdog Timer Mode

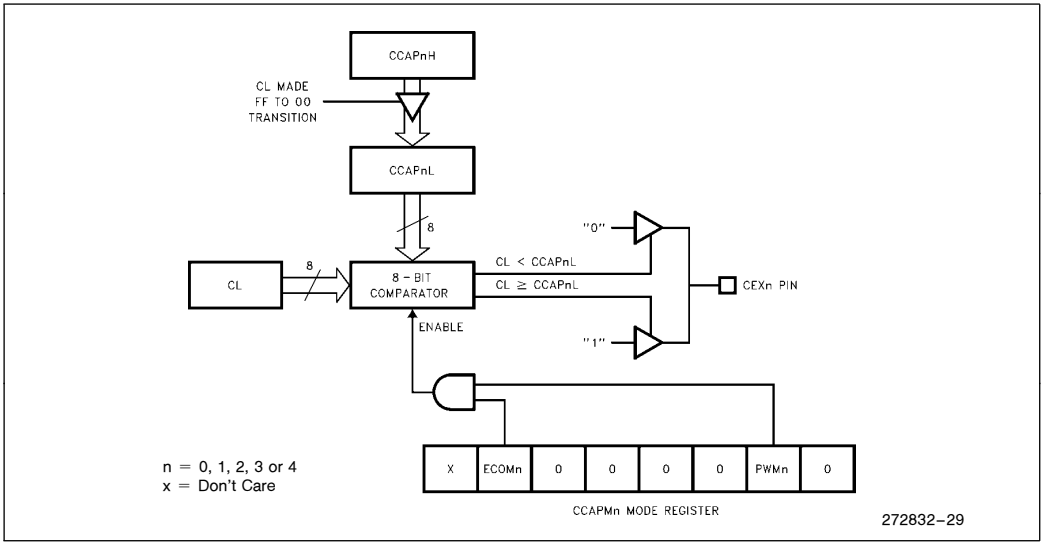


Figure 29. PCA 8-Bit PWM Mode

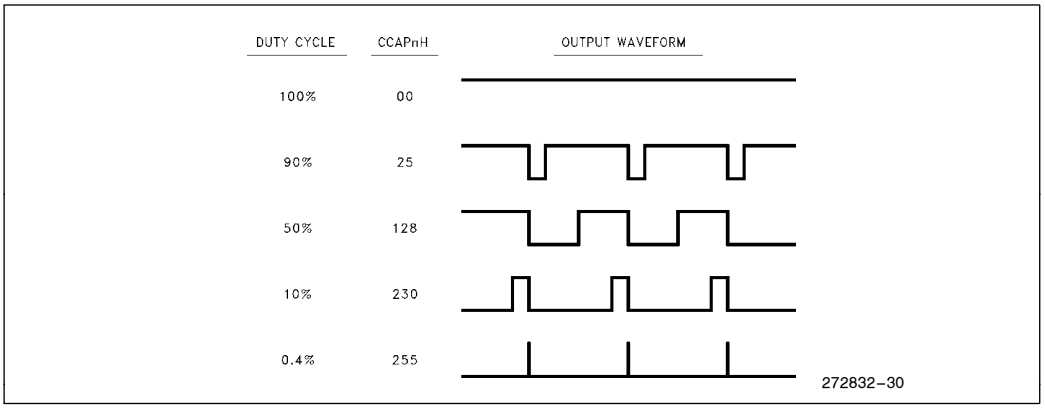


Figure 30. CCAPnH Varies Duty Cycle

CCAPnH can contain any integer from 0 to 255 to vary the duty cycle from a 100% to 0.4% (see Figure 30).

7.0 SERIAL INTERFACE

The serial port is full duplex, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the receive register. (However, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost). The serial port receive and transmit registers are both accessed through Special Function Register SBUF. Actually, SBUF is two separate registers, a transmit buffer and a receive buffer. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The serial port control and status register is the Special Function Register SCON. This register contains the mode selection bits (SM0 and SM1); the SM2 bit for the multiprocessor modes (see Multiprocessor Communications section); the Receive Enable bit (REN); the 9th data bit for transmit and receive (TB8 and RB8); and the serial port interrupt bits (TI and RI).

The serial port can operate in 4 modes:

Mode 0: Serial data enters and exits through RXD. TXD outputs the shift clock. 8 bits are transmitted/received: 8 data bits (LSB first). The baud rate is fixed at $^1/_{12}$ the oscillator frequency.

Mode 1: 10 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in Special Function Register SCON. The baud rate is variable.

Mode 2: 11 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). Refer to Figure 31. On Transmit, the 9th data bit (TB8 in SCON) can be assigned the value of 0 or 1. Or, for example, the parity bit (P in the PSW) could be moved into TB8. On receive, the 9th data bit goes into RB8 in SCON, while the stop bit is ignored. (The validity of the stop bit can be checked with Framing Error Detection.) The baud rate is programmable to either $^1/_{32}$ or $^1/_{64}$ the oscillator frequency.

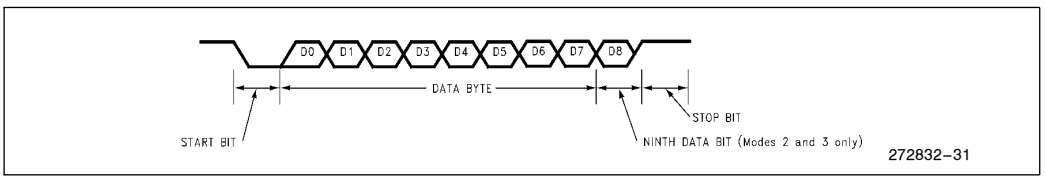


Figure 31. Data Frame: Modes 1, 2 and 3

Mode 3: 11 bits are transmitted (through TXD) or received (through RXD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit and a stop bit (1). In fact, Mode 3 is the same as Mode 2 in all respects except the baud rate. The baud rate in Mode 3 is variable.

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming start bit if REN = 1. For more detailed information on each serial port mode, refer to the "Hardware Description of the 8051, 8052, and 80C51."

7.1 Framing Error Detection

Framing Error Detection allows the serial port to check for valid stop bits in modes 1, 2, or 3. A missing stop bit can be caused, for example, by noise on the serial lines, or transmission by two CPUs simultaneously.

If a stop bit is missing, a Framing Error bit FE is set. The FE bit can be checked in software after each reception to detect communication errors. Once set, the FE bit must be cleared in software. A valid stop bit will not clear FE.

The FE bit is located in SCON and shares the same bit address as SM0. Control bit SMOD0 in the PCON register determines whether the SM0 or FE bit is accessed. If SMOD0 = 0, then accesses to SCON.7 are to SM0. If SMOD0 = 1, then accesses to SCON.7 are to FE.

7.2 Multiprocessor Communications

Modes 2 and 3 provide a 9-bit mode to facilitate multiprocessor communication. The 9th bit allows the controller to distinguish between address and data bytes. The 9th bit is set to 1 for address bytes and set to 0 for data bytes. When receiving, the 9th bit goes into RB8 in SCON. When transmitting, TB8 is set or cleared in software.

The serial port can be programmed such that when the stop bit is received the serial port interrupt will be activated only if the received byte is an address byte (RB8 = 1). This feature is enabled by setting the SM2 bit in SCON. A way to use this feature in multiprocessor systems is as follows.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. Remember, an address byte has its 9th bit set to 1, whereas a data byte has its 9th bit set to 0. All the slave processors should have their SM2 bits set to 1 so they will only be interrupted by an address byte. In fact, the C151SX has an Automatic Address Recognition feature which allows only the addressed slave to be interrupted. That is, the address comparison occurs in hardware, not software.

The addressed slave's software then clears its SM2 bit and prepares to receive the data bytes that will be coming. The other slaves are unaffected by these data bytes. They are still waiting to be addressed since their SM2 bits are all set.

7.3 Automatic Address Recognition

Automatic Address Recognition reduces the CPU time required to service the serial port. Since the CPU is only interrupted when it receives its own address, the software overhead to compare addresses is eliminated. With this feature enabled in one of the 9-bit modes, the Receive Interrupt (RI) flag will only get set when the received byte corresponds to either a Given or Broadcast address.

The feature works the same way in the 8-bit mode (Mode 1) as in the 9-bit modes, except that the stop bit takes the place of the 9th data bit. If SM2 is set, the RI flag is set only if the received byte matches the Given or Broadcast Address and is terminated by a valid stop bit. Setting the SM2 bit has no effect in Mode 0.

The master can selectively communicate with groups of slaves by using the Given Address. Addressing all slaves at once is possible with the Broadcast Address. These addresses are defined for each slave by two Special Function Registers: SADDR and SADEN.

A slave's individual address is specified in SADDR. SADEN is a mask byte that defines don't-cares to form the Given Address. These don't-cares allow flexibility in the user-defined protocol to address one or more slaves at a time. The following is an example of how the user could define Given Addresses to selectively address different slaves.

SCON: Serial Port Control Register

SCON Address: 098H
Reset State: 0000 0000B

Bit Addressable

7							0
SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI

(SMOD0 = 0/1)*

Symbol	Function
FE	Framing Error bit. This bit is set by the receiver when an invalid stop bit is detected. The FE bit is not cleared by valid frames but should be cleared by software. The SMOD0* bit must be set to enable access to the FE bit.
SM0	Serial Port Mode Bit 0, (SMOD0 must = 0 to access bit SM0)
SM1	Serial Port Mode Bit 1 **SM0 SM1 Mode Description Baud Rate**** 0 0 0 Shift Register $F_{OSC}/12$ 0 1 1 8-bit UART Variable 1 0 2 9-bit UART $F_{OSC}/64$ or $F_{OSC}/32$ 1 1 3 9-bit UART Variable
SM2	Enables the Automatic Address Recognition feature in Modes 2 or 3. If SM2 = 1 then RI will not be set unless the received 9th data bit (RB8) is 1, indicating an address, and the received byte is a Given or Broadcast Address. In Mode 1, if SM2 = 1 then RI will not be activated unless a valid stop bit was received, and the received byte is a Given or Broadcast Address. In Mode 0, SM2 should be 0.
REN	Enables serial reception. Set by software to enable reception. Clear by software to disable reception.
TB8	The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.
RB8	In modes 2 and 3, the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used.
TI	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.
RI	Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

NOTE:
*SMOD0 is located at PCON6.
**F_{OSC} = oscillator frequency

Slave 1:

SADDR	=	1111 0001
SADEN	=	1111 1010
GIVEN	=	1111 0X0X

Slave 2:

SADDR	=	1111 0011
SADEN	=	1111 1001
GIVEN	=	1111 0XX1

The SADEN byte are selected such that each slave can be addressed separately. Notice that bit 0 (LSB) is a don't-care for Slave 1's Given Address, but bit 1 = 1 for Slave 2. Thus, to selectively communicate with just Slave 1 the master must send an address with bit 1 = 0 (e.g. 1111 0000).

Similarly, bit 2 = 0 for Slave 1, but is a don't-care for Slave 2. Now to communicate with just Slave 2 an address with bit 2 = 1 must be used (e.g. 1111 0111).

Finally, for a master to communicate with both slaves at once the address must have bit 1 = 1 and bit 2 = 0.

Notice, however, that bit 3 is a don't-care for both slaves. This allows two different addresses to select both slaves (1111 0001 or 1111 0101). If a third slave was added that required its bit 3 = 0, then the latter address could be used to communicate with Slave 1 and 2 but not Slave 3.

The master can also communicate with all slaves at once with the Broadcast Address. It is formed from the logical OR of the SADDR and SADEN registers with zeros defined as don't-cares. The don't-cares also allow flexibility in defining the Broadcast Address, but in most applications a Broadcast Address will be 0FFH.

SADDR and SADEN are located at address A9H and B9H, respectively. On reset, the SADDR and SADEN registers are initialized to 00H which defines the Given and Broadcast Addresses as XXXX XXXX (all don't-cares). This assures the C151SX serial port to be backwards compatibility with other MCS®-51 products which do not implement Automatic Addressing.

7.4 Baud Rates

The baud rate in Mode 0 is fixed:

$$\text{Mode 0 Baud Rate} = \frac{\text{Oscillator Frequency}}{12}$$

The baud rate in Mode 2 depends on the value of bit SMOD1 in Special Function Register PCON. If SMOD1 = 0 (which is the value on reset), the baud rate is $1/64$ the oscillator frequency. If SMOD1 = 1, the baud rate is $1/32$ the oscillator frequency.

$$\text{Mode 2 Baud Rate} = 2^{\text{SMOD1}} \times \frac{\text{Oscillator Frequency}}{64}$$

The baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate, or by Timer 2 overflow rate, or by both (one for transmit and the other for receive).

7.5 Using Timer 1 to Generate Baud Rates

When Timer 1 is used as the baud rate generator, the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD1 as follows:

$$\text{Modes 1 and 3 Baud Rate} = 2^{\text{SMOD1}} \times \frac{\text{Timer 1 Overflow Rate}}{32}$$

Table 12. Timer 1 Generated Commonly Used Baud Rates

Baud Rate	f_{OSC}	SMOD	Timer 1		
			C/T#	Mode	Reload Value
Mode 0 Max: 1 MHz	12 MHz	X	X	X	X
Mode 2 Max: 375K	12 MHz	1	X	X	X
Modes 1, 3: 62.5K	12 MHz	1	0	2	FFH
19.2K	11.059 MHz	1	0	2	FDH
9.6K	11.059 MHz	0	0	2	FDH
4.8K	11.059 MHz	0	0	2	FAH
2.4K	11.059 MHz	0	0	2	F4H
1.2K	11.059 MHz	0	0	2	E8H
137.5	11.986 MHz	0	0	2	1DH
110	6 MHz	0	0	2	72H
110	12 MHz	0	0	1	FEEBH

The Timer 1 interrupt should be disabled in this application. The Timer itself can be configured for either "timer" or "counter" operation, and in any of its 3 running modes. In most applications, it is configured for "timer" operation in the auto-reload mode (high nibble of TMOD = 0010B). In this case, the baud rate is given by the formula:

$$\text{Modes 1 and 3 Baud Rate} = \frac{2^{\text{SMOD1}} \times \text{Oscillator Frequency}}{32 \times 12 \times [256 - (\text{TH1})]}$$

One can achieve very low baud rates with Timer 1 by leaving the Timer 1 interrupt enabled, and configuring the Timer to run as a 16-bit timer (high nibble of TMOD = 0001B), and using the Timer 1 interrupt to do a 16-bit software reload.

Table 12 lists various commonly used baud rates and how they can be obtained from Timer 1.

7.6 Using Timer 2 to Generate Baud Rates

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 11). Note that the baud rates for transmit and receive can be simultaneously different. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 32.

The baud rate generator mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate as follows:

$$\text{Modes 1 and 3 Baud Rates} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

The Timer can be configured for either "timer" or "counter" operation. In most applications, it is configured for "timer" operation (C/T2# = 0). The "Timer" operation is different for Timer 2 when it's being used as a baud rate generator. Normally, as a timer, it increments every machine cycle ($1/12$ the oscillator frequency). As a baud rate generator, however, it increments every state time ($1/2$ the oscillator frequency). The baud rate formula is given below:

$$\text{Modes 1 and 3 Baud Rate} = \frac{\text{Oscillator Frequency}}{32 \times [65536 - (\text{RCAP2H, RCAP2L})]}$$

where (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

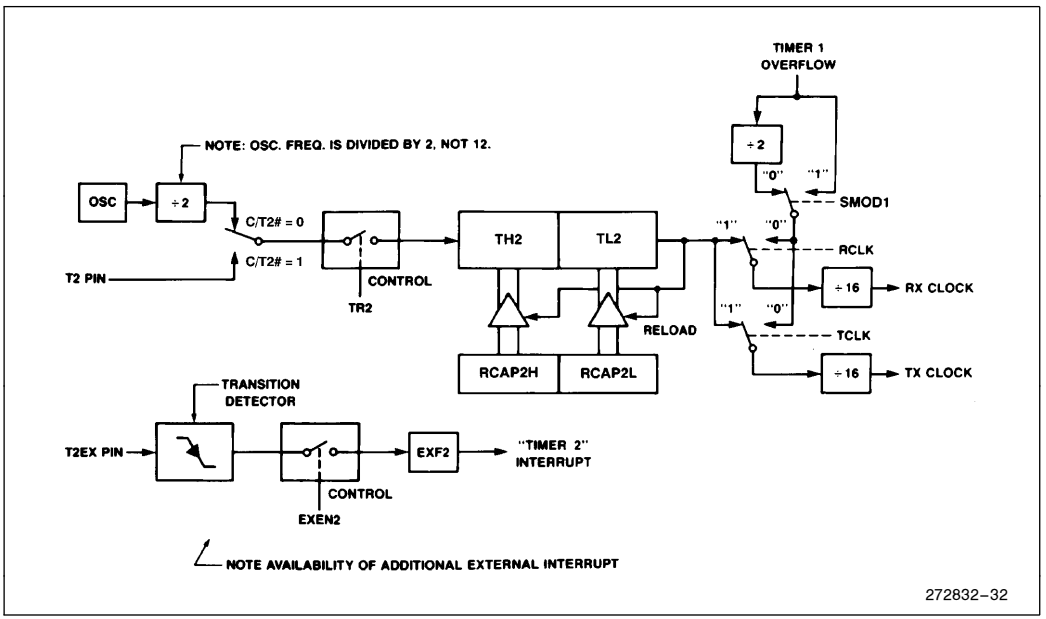


Figure 32. Timer 2 in Baud Rate Generator Mode

Timer 2 as a baud rate generator is shown in Figure 32. This figure is valid only if RCLK and/or TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2, and will not generate an interrupt. Therefore, the Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt, if desired.

It should be noted that when Timer 2 is running (TR2 = 1) in "timer" function in the baud rate generator mode, one should not try to read or write TH2 or TL2. Under these conditions the Timer is being incremented every state time, and the results of a read or write may not be accurate. The RCAP2 registers may be read, but shouldn't be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

Table 13 lists commonly used baud rates and how they can be obtained from Timer 2.

Table 13. Timer 2 Generated Commonly Used Baud Rates

Baud Rate	Osc Freq	Timer 2	
		RCAP2H	RCAP2L
375K	12 MHz	FF	FF
9.6K	12 MHz	FF	D9
4.8K	12 MHz	FF	B2
2.4K	12 MHz	FF	64
1.2K	12 MHz	FE	C8
300	12 MHz	FB	1E
110	12 MHz	F2	AF
300	6 MHz	FD	8F
110	6 MHz	F9	57

8.0 WATCHDOG TIMER

The C151SX contains a dedicated, hardware watchdog timer (WDT) that automatically resets the chip if it is allowed to time out. The WDT provides a means of recovering from routines that do not complete successfully due to software malfunctions. The WDT described in this section is not associated with the PCA watchdog timer, which is implemented in software.

The WDT is a 14-bit counter that counts peripheral cycles, i.e. the system clock divided by twelve (F_{OSC}/12). The WDTRST special function register at address A6H provides control access to the WDT. Two operations control the WDT:

- Device reset clears and disables the WDT
- Writing a specific two-byte sequence to the WDTRST register clears and enables the WDT.

If it is not cleared, the WDT overflows on count 3FFFH + 1. With F_{OSC} = 16 MHz, a peripheral cycle is 750 ns and the WDT overflows in 750 × 16384 = 12.288 ms. The WDTRST is a write-only register. Attempts to read it return FFH. The WDT itself is not read or write accessible. The WDT does **not** drive the external RESET pin.

8.1 Using the WDT

To use the WDT to recover from software malfunctions, the user program should control the WDT as follows:

1. Following device reset, write the two-byte sequence 1EH–E1H to the WDTRST register to enable the WDT. The WDT begins counting from 0.
2. Repeatedly for the duration of program execution, write the two-byte sequence 1EH–E1H to the WDTRST register to clear and enable the WDT before it overflows. The WDT starts over at 0.

If the WDT overflows, it initiates a device reset. Device reset clears the WDT and disables it.

8.2 WDT During Idle Mode and Powerdown

Operation of the WDT during the power reduction modes deserves special attention. The WDT continues to count while the microcontroller is in idle mode. This means the user must service the WDT during idle. One approach is to use a peripheral timer to generate an interrupt request when the timer overflows. The interrupt service routine then clears the WDT, reloads the peripheral timer for the next service period, and puts the microcontroller back into idle.

The powerdown mode stops all phase clocks. This causes the WDT to stop counting and to hold its count. The WDT resumes counting from where it left off if the powerdown mode is terminated by INT0/INT1. To ensure that the WDT does not overflow shortly after exiting the powerdown mode, clear the WDT just before entering powerdown. The WDT is cleared and disabled if the powerdown mode is terminated by a reset.

9.0 INTERRUPTS

Figure 33 illustrates the interrupt control system. The C151SX has eight interrupt sources; seven maskable sources and the TRAP instruction (always enabled). The maskable sources include two external interrupts (INT0# and INT1#), three timer interrupts (timers 0, 1, and 2), one programmable counter array (PCA) interrupt, amd one serial port interrupt. Each interrupt (except TRAP) has an interrupt request flag, which can be set by software as well as by hardware. For some interrupts, hardware clears the request flag when it grants an interrupt. Software can clear any request flag to cancel an impending interrupt.

9.1 External Interrupts

External interrupts INT0# and INT1# (INT*x*#) pins may each be programmed to be level-triggered or edge-triggered, dependent upon bits IT0 and IT1 in the TCON register. If IT*x* = 0, INT*x*# is triggered by a detected low at the pin. If IT*x* = 1, INT*x*# is negative-edge triggered. External interrupts are enabled with bits EX0 and EX1 (EX*x*) in the IE0 register. Events on the external interrupt pins set the interrupt request flags IE*x* in TCON. These request bits are cleared by hardware vectors to service routines only if the interrupt is negative-edge triggered. If the interrupt is level-triggered, the interrupt service routine must clear the request bit. External hardware must deassert INT*x*# before the service routine completes, or an additional interrupt is requested. External interrupt pins must be deasserted for at least four state times prior to a request.

External interrupt pins are sampled once every four state times (a frame length of 666.4 ns at 12 MHz). A level-triggered interrupt pin held low or high for any five-state time period guarantees detection. Edge-triggered external interrupts must hold the request pin low for at least five state times. This ensures edge recognition and sets interrupt request bit EX*x*. The CPU clears EX*x* automatically during service routine fetch cycles for edge-triggered interrupts.

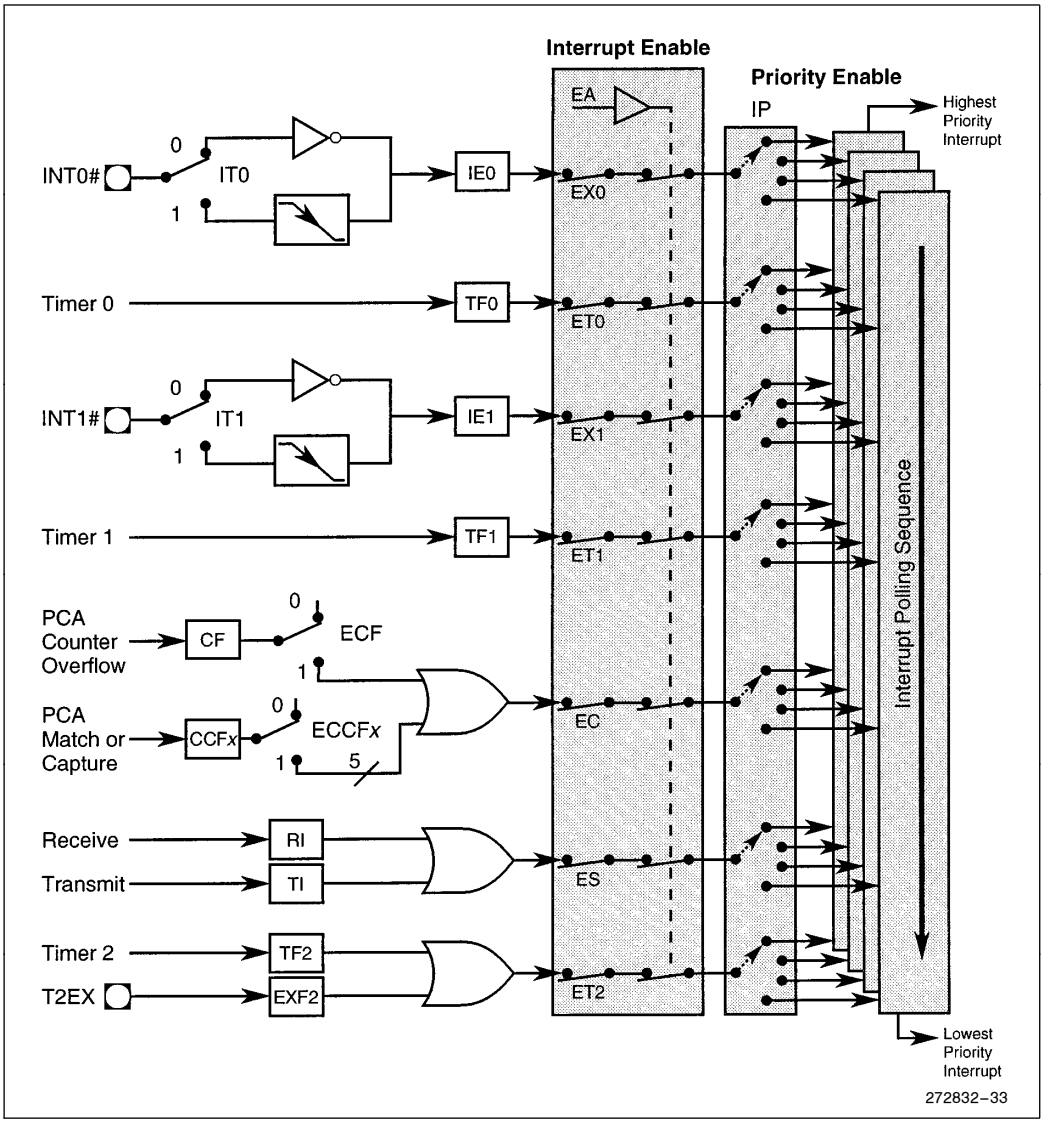


Figure 33. Interrupt Control System

Table 14. Interrupt Control Matrix

Interrupt Name	Global Enable	PCA	Timer 2	Serial Port	Timer 1	INT1#	Timer 0	INT0#
Bit Name in IE0 Register	EA	EC	ET2	ES	ET1	EX1	ET0	EX0
Interrupt Priority-Within-Level (7 = Low Priority, (1 = High Priority)	NA	7	6	5	4	3	2	1
Bit Names in: IPH0 IPL0	Reserved Reserved	IPH0.6 IPL0.6	IPH0.5 IPL0.5	IPH0.4 IPL0.4	IPH0.3 IPL0.3	IPH0.2 IPL0.2	IPH0.1 IPL0.1	IPH0.0 IPL0.0
Programmable for Negative-edge Triggered or Level-Triggered Detect?	NA	Edge	No	No	No	Yes	No	Yes
Interrupt Request Flag in CCON, T2CON, SCON, or TCON Register	NA	CF, CCF*x*	TF2, EXF2	RI, TI	TF1	IE1	TF0	IE0
Interrupt Request Flag Cleared by Hardware?	No	No	No	No	Yes	Edge Yes, Level No	Yes	Edge Yes, Level No
ISR Vector Address	NA	0033H	002BH	0023H	001BH	0013H	000BH	0003H

9.2 Timer Interrupts

Two timer-interrupt request bits TF0 and TF1 (see TCON register) are set by timer overflow (the exception is Timer 0 in Mode 3). When a timer interrupt is generated, the bit is cleared by an on-chip-hardware vector to an interrupt service routine. Timer interrupts are enabled by bits ET0, ET1, and ET2 in the IE0 register.

Timer 2 interrupts are generated by a logical OR of bits TF2 and EXF2 in register T2CON. Neither flag is cleared by a hardware vector to a service routine. In fact, the interrupt service routine must determine if TF2 or EXF2 generated the interrupt, and then clear the bit. Timer 2 interrupt is enabled by ET2 in register IE0.

9.3 Programmable Counter Array (PCA) Interrupt

The programmable counter array (PCA) interrupt is generated by logical OR of five event flags (CCF*x*) and the PCA timer overflow flag (CF) in the CCON register. All PCA interrupts share a common interrupt vector. Bits are not cleared by hardware vectors to service routines. Normally, interrupt service routines resolve interrupt requests and clear flag bits. This allows the user to define the relative priorities of the five PCA interrupts.

The PCA interrupt is enabled by bit EC in the IE0 register. In addition, the CF flag and each of the CCF*x* flags must also be individually enabled by bits ECF and ECCF*x* in registers CMOD and CCAPM*x* respectively for the flag to generate an interrupt.

NOTE:
CCF*x* refers to 5 separate bits, one for each PCA module (CCF0, CCF1, CCF2, CCF3, CCF4).
CCAPM*x* refers to 5 separate registers, one for each PCA module (CCAPM0, CCAPM1, CCAPM2, CCAPM3, CCAPM4).

9.4 Serial Port Interrupt

Serial port interrupts are generated by the logical OR of bits RI and TI in the SCON register. Neither flag is cleared by a hardware vector to the service routine. The service routine resolves RI or TI interrupt generation and clears the serial port request flag. The serial port interrupt is enabled by bit ES in the IE0 register.

9.5 Interrupt Enable

Each interrupt source (with the exception of TRAP) may be individually enabled or disabled by the appropriate interrupt enable bit in the IE0 register. Note IE0 also contains a global disable bit (EA). If EA is set, interrupts are individually enabled or disabled by bits in IE0. If EA is clear, all interrupts are disabled.

Interrupt Enable Register

IE0 Address: 0A8H
Reset State: 0000 0000B

7							0
EA	EC	ET2	ES	ET1	EX1	ET0	EX0

Bit Mnemonic	Function
EA	Global Interrupt Enable: Setting this bit enables all interrupts that are individually enabled by bits 0–6. Clearing this bit disables all interrupts, except the TRAP interrupt, which is always enabled.
EC	PCA Interrupt Enable: Setting this bit enables the PCA interrupt.
ET2	Timer 2 Overflow Interrupt Enable: Setting this bit enables the timer 2 overflow interrupt.
ES	Serial I/O Port Interrupt Enable: Setting this bit enables the serial I/O port interrupt.
ET1	Timer 1 Overflow Interrupt Enable: Setting this bit enables the timer 1 overflow interrupt.
EX1	External Interrupt 1 Enable: Setting this bit enables external interrupt 1.
ET0	Timer 0 Overflow Interrupt Enable: Setting this bit enables the timer 0 overflow interrupt.
EX0	External Interrupt 0 Enable: Setting this bit enables external interrupt 0.

9.6 Interrupt Priorities

Each of the seven C151SX interrupt sources may be individually programmed to one of four priority levels. This is accomplished with the IPH0.*x*/IPL0.*x* bit pairs in the interrupt priority high (IPH0) and interrupt priority low (IPL0) registers. Specify the priority level as shown in Table 15 using IPH0.*x* as the MSB and IPL0.*x* as the LSB.

Table 15. Level of Priority

IPH0.*x* (MSB)	IPL0.*x* (LSB)	Priority Level
0	0	0 Lowest Priority
0	1	1
1	0	2
1	1	3 Highest Priority

A low-priority interrupt is always interrupted by a higher priority interrupt but not by another interrupt of equal or lower priority. The highest priority interrupt is not interrupted by any other interrupt source. Higher priority interrupts are serviced before lower priority interrupts. The response to simultaneous occurrence of equal priority interrupts (i.e., sampled within the same four state interrupt cycle) is determined by a hardware priority-within-level resolver as shown in Table 16.

Table 16. Interrupt Priority Within Level

Priority Number *	Interrupt Name
1 (Highest Priority)	INT0#
2	Timer 0
3	INT1#
4	Timer 1
5	Serial Port
6	Timer 2
7 (Lowest Priority)	PCA

* The C151SX interrupt priority-within-level table differs from MCS® 51 microcontrollers.

Interrupt Priority High Register

IPH0 Address: 0B7H
Reset State: X000 0000B

7							0
—	IPH0.6	IPH0.5	IPH0.4	IPH0.3	IPH0.2	IPH0.1	IPH0.0

Bit Mnemonic	Function
—	Reserved. The value read from this bit is indeterminate. Do not write a ‘‘1’’ to this bit.
IPH0.6	PCA Interrupt Priority Bit High
IPH0.5	Timer 2 Overflow Interrupt Priority Bit High
IPH0.4	Serial I/O Port Interrupt Priority Bit High
IPH0.3	Timer 1 Overflow Interrupt Priority Bit High
IPH0.2	External Interrupt 1 Priority Bit High
IPH0.1	Timer 0 Overflow Interrupt Priority Bit High
IPH0.0	External Interrupt 0 Priority Bit High

Interrupt Priority Low Register

IPL0 Address: 0B8H
Reset State: X000 0000B

7							0
—	IPL0.6	IPL0.5	IPL0.4	IPL0.3	IPL0.2	IPL0.1	IPL0.0

Bit Mnemonic	Function
—	Reserved. The value read from this bit is indeterminate. Do not write a ‘‘1’’ to this bit.
IPL0.6	PCA Interrupt Priority Bit Low
IPL0.5	Timer 2 Overflow Interrupt Priority Bit Low
IPL0.4	Serial I/O Port Interrupt Priority Bit Low
IPL0.3	Timer 1 Overflow Interrupt Priority Bit Low
IPL0.2	External Interrupt 1 Priority Bit Low
IPL0.1	Timer 0 Overflow Interrupt Priority Bit Low
IPL0.0	External Interrupt 0 Priority Bit Low

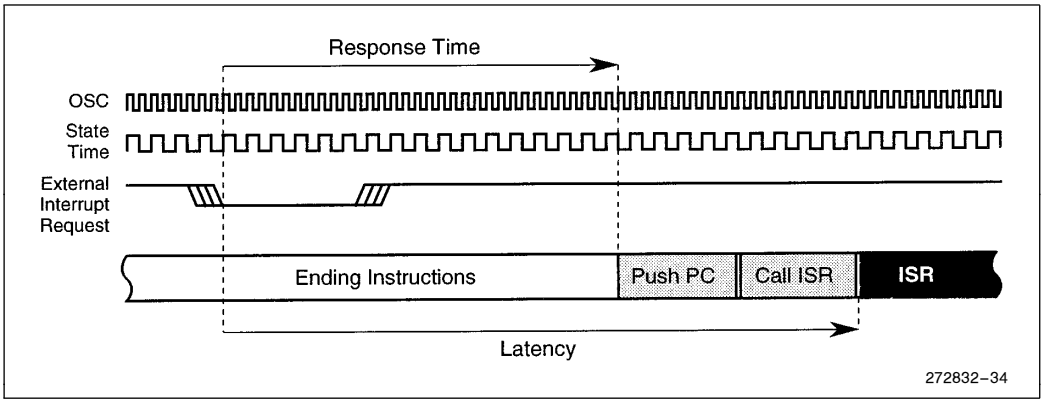


Figure 34. The Interrupt Process

9.7 Interrupt Processing

Interrupt processing is a dynamic operation that begins when a source requests an interrupt and lasts until the execution of the first instruction in the interrupt service routine (see Figure 34). *Response time* is the amount of time between the interrupt request and the resulting break in the current instruction stream. *Latency* is the amount of time between the interrupt request and the execution of the first instruction in the interrupt service routine. These periods are dynamic due to the presence of both fixed-time sequences and several variable conditions. These conditions contribute to total elapsed time.

Both response time and latency begin with the request. The subsequent minimum fixed sequence comprises the interrupt sample, poll, and request operations. The variables consist of (but are not limited to): specific instructions in use at request time, internal versus external interrupt source requests, internal versus external program operation, stack location, presense of wait states, page-mode operation, and branch pointer length.

NOTE:

In the following discussion external interrupt request pins are assumed to be inactive for at least four state times prior to assertion. In this section all external hardware signals maintain some setup period (i.e., less than one state time). Signals must meet V_{IH} and V_{IL} specifications prior to any state time under discussion. This setup state time is not included in examples or calculations for either response or latency.

9.7.1 MINIMUM FIXED INTERRUPT TIME

All interrupts are sampled or polled every four state times (see Figure 34). Two of eight interrupts are latched and polled per state time within any given four-state time window. One additional state time is required for a context switch request. For code branches to jump locations in the current 64-Kbyte memory region (compatible with MCS 51 microcontrollers), the context switch time is 11 states. Therefore, the minimum fixed poll and request time is 16 states (4 poll states + 1 request state + 11 states for context switch = 16 state times).

Therefore, this minimum fixed period rests upon four assumptions:

- The source request is an internal interrupt with high enough priority to take precedence over other potential interrupts,
- The request is coincident with internal execution and needs no instruction completion time,
- The program uses an internal stack location, and
- The ISR is in on-chip OTPROM/ROM.

9.7.2 VARIABLE INTERRUPT PARAMETERS

Both response time and latency calculations contain fixed and variable components. By definition, it is often difficult to predict exact timing calculations for real-time requests. One large variable is the completion time of an instruction cycle coincident with the occurrence of an interrupt request. Worst-case predictions typically use the longest-executing instruction in an architecture's code set.

9.7.3 RESPONSE TIME VARIABLES

Response time is defined as the start of a dynamic time period when a source requests an interrupt and lasts until a break in the current instruction execution stream occurs (see Figure 34). Response time (and therefore latency) is affected by two primary factors: the incidence of the request relative to the four-state-time sample window and the completion time of instructions in the response period (i.e., shorter instructions complete earlier than longer instructions).

NOTE:

External interrupt signals require one additional state time in comparison to internal interrupts. This is necessary to sample and latch the pin value prior to a poll of interrupts. The sample occurs in the first half of the state time and the poll/request occurs in the second half of the next state time. Therefore, this sample and poll/request portion of the minimum fixed response and latency time is five states for internal interrupts and six states for external interrupts. External interrupts must remain active for at least five state times to guarantee interrupt recognition when the request occurs immediately after a sample has been taken (i.e., requested in the second half of a sample state time).

If the external interrupt goes active one state after the sample state, the pin is not resampled for another three states. After the second sample is taken and the interrupt request is recognized, the interrupt controller requests the context switch. The programmer must also consider the time to complete the instruction at the moment the context switch request is sent to the execution unit. If 9 states of a 10-state instruction have completed when the context switch is requested, the total response time is 6 states, with a context switch immediately after the final state of the 10-state instruction (see Figure 35).

Conversely, if the external interrupt requests service in the state just prior to the next sample, response is much quicker. One state asserts the request, one state samples, and one state requests the context switch. If at that point the same instruction conditions exist, one additional state time is needed to complete the 10-state instruction prior to the context switch (see Figure 36). The total response time in this case is four state times. The programmer must evaluate all pertinent conditions for accurate predictability.

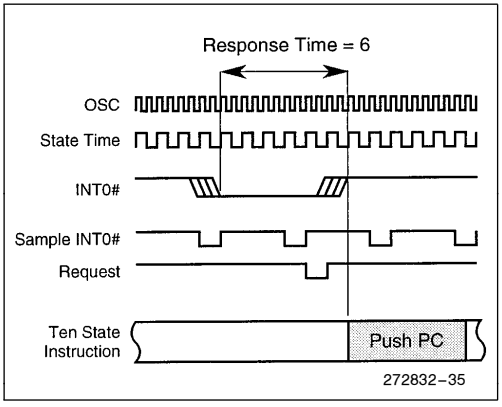


Figure 35. Response Time Example #1

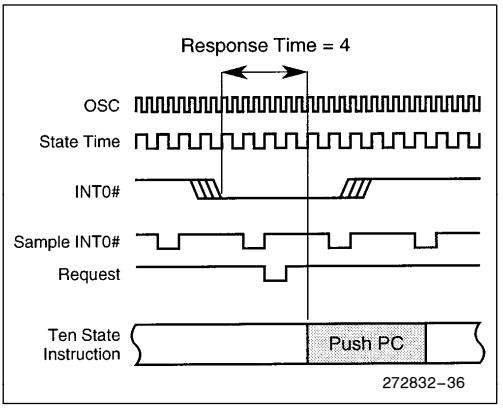


Figure 36. Response Time Example #2

9.7.4 COMPUTATION OF WORST-CASE LATENCY WITH VARIABLES

Worst-case latency calculations assume that the longest C151SX instruction used in the program must fully execute prior to a context switch. The instruction execution time is reduced by one state with the assumption the instruction state overlaps the request state (therefore, DIV is 10 state times − 1 = 9 states for latency calculations). The calculations add fixed and variable interrupt times to this instruction time to predict latency. The worst-case latency (both fixed and variable times included) is expressed by a pseudo-formula:

```
FIXED_TIME + VARIABLES +
LONGEST_INSTRUCTION = MAXIMUM LATENCY
PREDICTION
```

Table 17. Interrupt Latency Variables

Variable	INT0#, INT1#, T2EX	External Execution	Page Mode	External Memory Wait State
Number of States Added	1	2	1	1 per Bus Cycle

NOTES:
Base case fixed time is 16 states and assumes:
—A 2-byte instruction is the first ISE byte
—Internal peripheral interrupt
—Internal execution

9.7.5 BLOCKING CONDITIONS

If all enable and priority requirements have been met, a single prioritized interrupt request at a time generates a vector cycle to an interrupt service routine. There are three causes of blocking conditions with hardware-generated vectors:

1. An interrupt of equal or higher priority level is already in progress (defined as any point after the flag has been set and the RETI of the ISR has not executed).
2. The current polling cycle is not the final cycle of the instruction in progress.
3. The instruction in progress is RETI or any write to the IE0, IPH0, or IPL0 registers.

Any of these conditions blocks calls to interrupt service routines. Condition two ensures the instruction in progress completes before the system vectors to the ISR. Condition three ensures at least one more instruction executes before the system vectors to additional interrupts if the instruction in progress is a RETI or any write to IE0, IPH0, or IPL0. The complete polling cycle is repeated each four state times. If the interrupt flag for a level-triggered external interrupt is set but denied for one of the above conditions and is clear when the blocking condition is removed, then the denied interrupt is ignored. In other words, blocked interrupt requests are not buffered for retention.

9.7.6 INTERRUPT VECTOR CYCLE

When an interrupt vector cycle is initiated, the CPU breaks the instruction stream sequence, resolves all instruction pipeline decisions, and pushes multiple program counter (PC) bytes onto the stack. The CPU then reloads the PC with a start address for the appropriate ISR. The complete sample, poll, request and context switch vector sequence is illustrated in the interrupt latency timing diagram (see Figure 34).

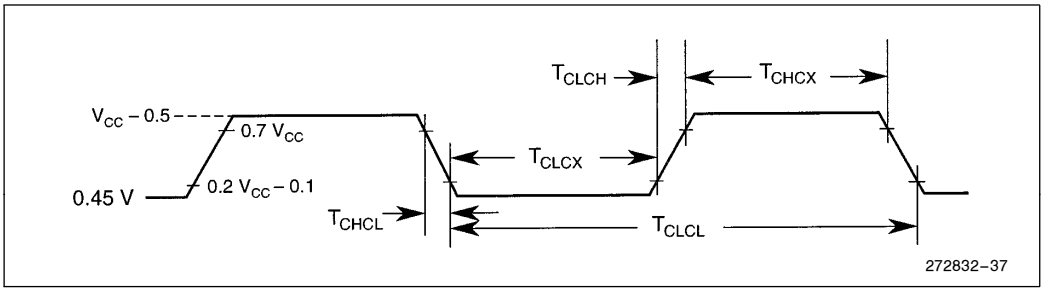


Figure 37. External Clock Drive Waveforms

9.7.7 ISRS IN PROCESS

ISR execution proceeds until the RETI instruction is encountered. The RETI instruction informs the processor the interrupt routine is completed. The RETI instruction in the ISR pops PC address bytes off the stack and execution resumes at the suspended instruction stream.

With the exception of TRAP, the start addresses of consecutive interrupt service routines are eight bytes apart. If consecutive interrupts are used (IE0 and TF0, for example, or TF0 and IE1), the first interrupt routine (if more than seven bytes long) must execute a jump to some other memory location. This prevents overlap of the start address of the following interrupt routine.

For external clock drive requirements, see the device datasheet. Figure 37 shows the clock drive waveform. The external clock source must meet the minimum high and low times (T_{CHCX} and T_{CLCX}) and the maximum rise and fall times (T_{CLCH} and T_{CHCL}) to minimize the effect of external noise on the clock generator circuit. Long rise and fall times increase the chance that external noise will affect the clock circuitry and cause unreliable operation.

The external clock driver may encounter increased capacitance loading at XTAL1 due to the Miller effect of the internal inverter as the clock waveform builds up in amplitude following power on. Once the input waveform requirements are met, the input capacitance remains under 20 pF.

10.0 RESET

A device initializes the C151SX and vectors the CPU to address 0000H. A reset is required after applying power at turn-on. A reset is a means of exiting the idle and powerdown modes or recovering from software malfunctions.

To achieve a valid reset, V_{CC} must be within its normal operating range (see device datasheet) and the reset signal must be maintained for 64 clock cycles (64 T_{OSC}) after the oscillator has stabilized.

Device reset is initiated in two ways:

- externally, by asserting the RST pin
- internally, if the hardware WDT or the PCA WDT expires

The power off flag (POF) in the PCON register indicates whether a reset is a warm start or a cold start. A cold start reset (POF = 1) is a reset that occurs after power has been off or V_{CC} has fallen below 3V, so the contents of volatile memory are indeterminate. POF is set by hardware when V_{CC} rises from less than 3V to its normal operating level. A warm start reset (POF = 0) is a reset that occurs while the chip is at operating voltage, for example, a reset initiated by a WDT overflow or an external reset used to terminate the idle or power-down modes.

10.1 Externally Initiated Resets

To reset the C151SX, hold the RST pin at a logic high for at least 64 clock cycles (64 T_{OSC}) while the oscillator is running. Reset can be accomplished automatically at the time power is applied by capacitively coupling RST to V_{CC} (see Figure 38). The RST pin has a Schmitt trigger input and a pulldown resistor.

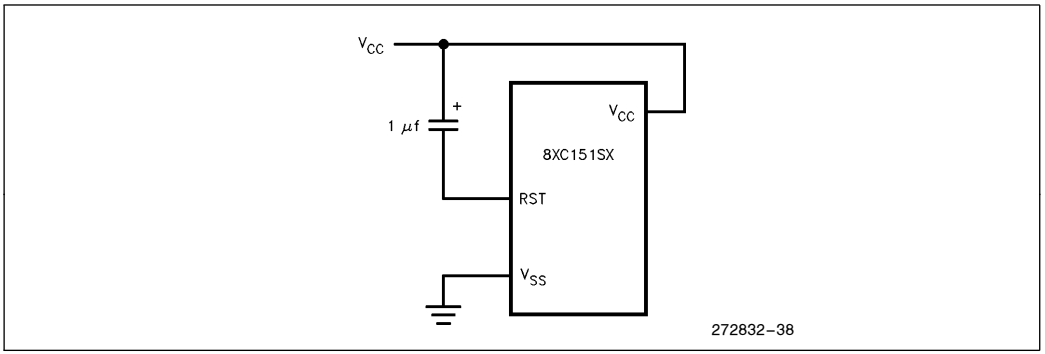


Figure 38. Power on Reset Circuitry

10.2 WDT Initiated Resets

Expiration of the hardware WDT (overflow) or the PCA WDT (comparison match) generates a reset signal. WDT initiated resets have the same effect as an external reset.

10.3 Reset Operation

When a reset is initiated, whether externally or by a WDT, the port pins are immediately forced to their reset condition as a fail-safe precaution, whether the clock is running or not.

The external reset signal and the WDT initiated reset signals are combined internally. For an external reset the voltage on the RST pin must be held high for 64 T_{OSC}. For WDT initiated resets, a 5-bit counter in the reset logic maintains the signal for the required 64 T_{OSC}.

The CPU checks for the presence of the combined reset signal every 2 T_{OSC}. When a reset is detected, the CPU responds by triggering the internal reset routine. The reset routine loads the SFR's with their reset values (refer to Table 2). Reset does not affect on-chip data RAM or the register file. (However following a cold start reset, these are indeterminate because V_{CC} has fallen too low or has been off.) Following a synchronizing operation and the configuration fetch, the CPU vectors to address 0000H. Figure 39 shows the reset timing sequence.

While the RST pin is high ALE, PSEN#, and the port pins are weakly pulled high. The first ALE occurs 32 T_{OSC} after the reset signal goes low. For this reason, other devices can not be synchronized to the internal timings of the C151SX.

NOTE:
Externally driving the ALE and/or PSEN# pins to 0 during the reset routine may cause the device to go into an indetermine state.
Powering up the C151SX without a reset may improperly initialize the program counter and SFRs and cause the CPU to execute instructions from an undetermined memory location.

10.4 Power-on Reset

To automatically generate a reset on power up, connect the RST pin to the V_{CC} pin through a 1 μF capacitor as shown in Figure 38.

When V_{CC} is applied, the RST pin rises to V_{CC}, then decays exponentially as the capacitor charges. The time constant must be such that RST remains high (above the turn-off threshold of the Schmitt trigger) long enough for the oscillator to start and stabilize, plus 64 T_{OSC}. At power up, V_{CC} should rise within approximately 10 ms. Oscillator start-up time is a function the crystal frequency; typical start-up times are 1 ms for 10 MHz crystal and 10 ms for 1 MHz crystal.

During power up, the port pins are in a random state until forced to their reset state by the asynchronous logic.

Reducing V_{CC} quickly to 0 causes the RST pin voltage to momentarily fall below 0V. This voltage is internally limited and does not harm the device.

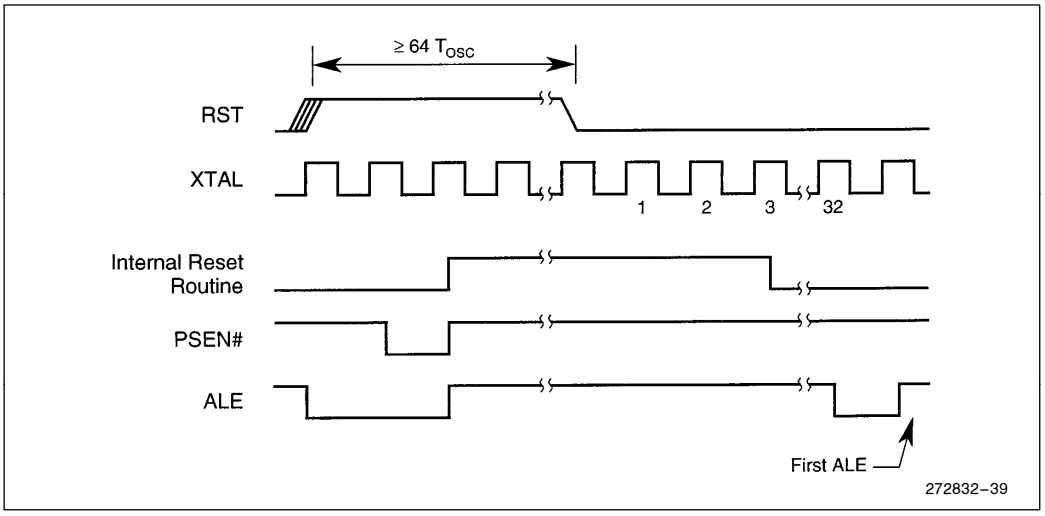


Figure 39. Reset Timing Sequence

11.0 POWER-SAVING MODES OF OPERATION

For applications where power consumption is critical, the C151SX provides two power reducing modes of operation: Idle and Power Down. The input through which backup power is supplied during these operations is V_{CC}. Figure 40 shows the internal circuitry which implements these features. In the Idle mode (IDL = 1), the oscillator continues to run and the Interrupt, Serial Port, PCA, and Timer blocks continue to be clocked, but the clock signal is gated off to the CPU. In Power Down (PD = 1), the oscillator is frozen. The Idle and Power Down modes are activated by setting bits in Special Function Register PCON.

11.1 Idle Mode

An instruction that sets PCON.0 causes that to be the last instruction executed before going into the Idle mode. In the Idle mode, the internal clock signal is gated off to the CPU, but not to the Interrupt, Timer, and Serial Port functions. The PCA can be programmed either to pause or continue operating during Idle (refer to the PCA section for more details). The CPU status is preserved in its entirety: the Stack Pointer, Program Counter, Program Status Word, Accumulator, and all other registers maintain their data during Idle. The port pins hold the logical states they had at the time Idle was activated. ALE and PSEN# hold at logic high levels.

There are two ways to terminate the Idle Mode. Activation of any enabled interrupt will cause PCON.0 to be cleared by hardware, terminating the Idle mode. The interrupt will be serviced, and following RETI the next instruction to be executed will be the one following the instruction that put the device into Idle.

The flag bits (GF0 and GF1) can be used to give an indication if an interrupt occurred during normal operation or during Idle. For example, an instruction that activates Idle can also set one or both flag bits. When Idle is terminated by an interrupt, the interrupt service routine can examine the flag bits.

The other way of terminating the Idle mode is with a hardware reset.

The signal at the RST pin clears the IDL bit directly and asynchronously. At this time the CPU resumes program execution from where it left off; that is, at the instruction following the one that invoked the Idle Mode.

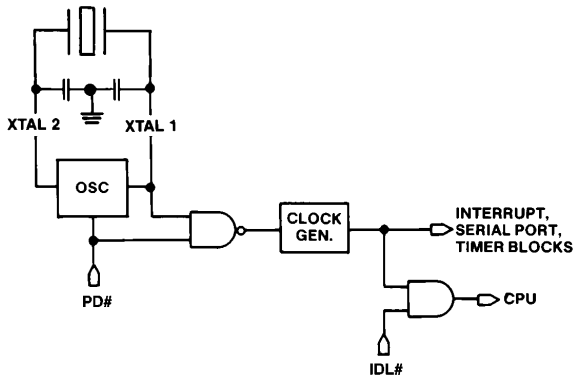


Figure 40. Idle and Power Down Hardware

PCON: Power Control Register

PCON Address: 087H
Reset State: 00XX 0000B

Not Bit Addressable

7							0
SMOD1	SMOD0	—	POF	GF1	GF0	PD	IDL

Symbol	Function
SMOD1	Double Baud rate bit. When set to a 1 and Timer 1 is used to generate baud rates, and the Serial Port is used in modes 1, 2, or 3.
SMOD0	When set, Read/Write accesses to SCON.7 are to the FE bit. When clear, Read/Write accesses to SCON.7 are to the SM0 bit.
—	Not implemented, reserved for future use.*
POF	Power Off Flag. Set by hardware on the rising edge of V_{CC}. Set or cleared by software. This flag allows detection of a power failure caused reset. V_{CC} must remain above 3V to retain this bit.
GF1	General-purpose flag bit.
GF0	General-purpose flag bit.
PD	Power Down bit. Setting this bit activates Power Down operation.
IDL	Idle mode bit. Setting this bit activates idle modes operation. If 1s are written to PD and IDL at the same time, PD takes precedence.

NOTE:
*User software should not write 1s to unimplemented bits. The value read from a reserved bit is indeterminate.

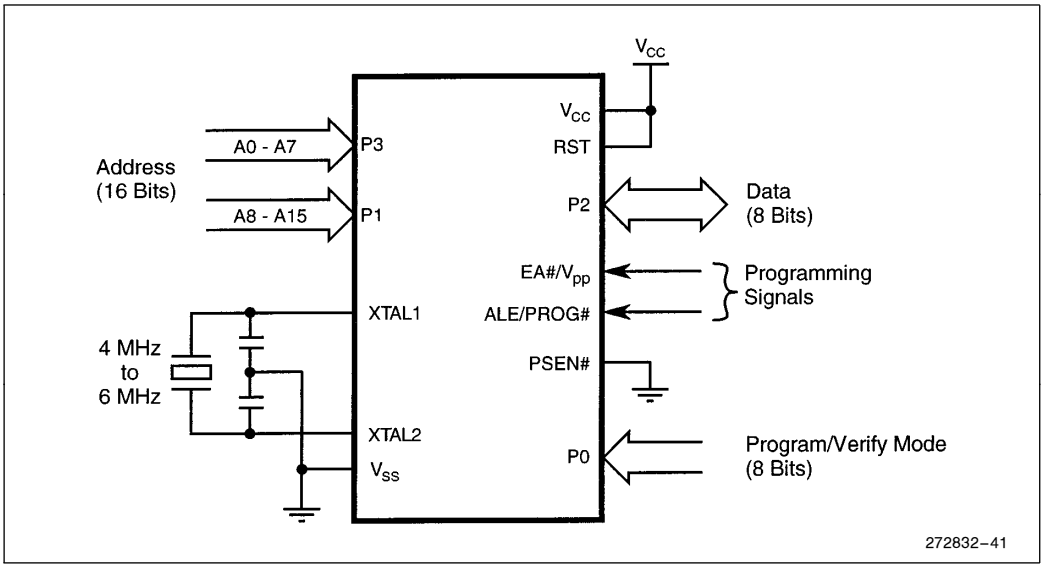


Figure 41. Setup for Programming and Verifying Nonvolatile Memory

12.0 PROGRAMMING AND VERIFYING NONVOLATILE MEMORY

The C151SX is programmed and verified in the same manner as the 87C51FX, using same quick-pulse programming algorithm, which programs at V_{PP} = 12.75V using a series of five 100 μs PROG# pulses per byte. This results in a programming time of approximately 16 seconds for the 16-Kbyte on-chip code memory.

12.1 Programming and Verifying Modes

Table 18 lists the programming and verifying modes and provides details about the setup. The value applied to port 0 determines the mode. The upper digit specifies program or verify and the lower digit selects what memory function is programmed, i.e. on-chip code memory, encryption array, configuration bytes, etc. The addresses applied to port 1 and port 3 address locations in the selected memory function. The encryption array, lock bits, signature bytes reside in nonvolatile memory outside the memory address space. User configuration bytes (UCONFIG0 and UCONFIG1) reside in nonvolatile memory at top of the memory address space for ROM/OTPROM devices and in external memory for devices without ROM/OTPROM.

The controller must be running with an oscillator frequency of 4 MHz to 6 MHz. To program, set up the controller as shown in Table 18 with the mode of operation (program/verify and memory area) specified on port 0, the address with respect to the starting address of the memory area applied to ports 1 and 3, and the data on port 2. Apply a logic high to the RST pin and V_{CC} to EA#/V_{PP}. ALE/PSEN#, normally an output pin, must be held low externally.

To perform the write operation, raise V_{PP} to 12.75V and pulse the PROG# pin per Table 18. Then return V_{PP} to 5V. Verification is performed in a similar manner but without increasing V_{PP} and without pulsing PROG#. The V_{PP} source must be well regulated and free of glitches. The voltage on the V_{PP} pin must not exceed the specified maximum, even under transient conditions.

Table 18. Programming and Verifying Modes

Mode	RST	PSEN#	V_{PP}	PROG#	Port 0	Port 2	Address Port 1 (High) Port 3 (Low)	Notes
Program Mode. On-Chip Code Memory 8K 87C151SA 16K 87C151SB	High	Low	5V, 12.75V	5 Pulses	68H	Data	 0000H–1FFFH 0000H–3FFFH	1, 4
Verify Mode. On-Chip Code Memory 8K 87/83C151SA 16K 87/83C151SB	High	Low	5V	High	28H	Data	 0000H–1FFFH 0000H–3FFFH	4
Program Mode. Configuration Bytes (UCONFIG0, UCONFIG1) 87C151S*x*	High	Low	5V, 12.75V	5 Pulses	69H	Data	FFF8H–FFFFH	1, 4
Verify Mode. Configuration Bytes (UCONFIG0, UCONFIG1) 87C151S*x*	High	Low	5V	High	29H	Data	FFF8H–FFFFH	4
Program Mode. Lock Bits 87C151S*x*	High	Low	5V, 12.75V	25 Pulses	6BH	Data	0001H–0003H	1, 2
Verify Mode. Lock Bits 87C151S*x*, 83C151S*x*	High	Low	5V	High	2BH	Data	0000H	3
Program Mode. Encryption Array 87C151S*x*	High	Low	5V, 12.75V	25 Pulses	6CH	Data	0000H–007FH	1
Verify Mode. Signature Bytes 87C151S*x*, 83C151S*x*	High	Low	5V	High	29H	Data	0030H, 0031H, 0060H, 0061H	

NOTES:
1. To program, raise V_{PP} to 12.75V and pulse the PROG# pin.
2. No data input. Identify the lock bits with the address lines as follows: LB3-0003H, LB2-0002H, LB1-0001H.
3. The three lock bits are verified in a single operation. The states of the lock bits appear simultaneously at port 2 as follows: LB3-P2.3, LB2-P2.2, LB1-P2.1. High = programmed.
4. For these modes, the internal address is XXXXH.

Table 19. Lock Bit Function

	Lock Bits Programmed			Protection Type
	LB3	LB2	LB1	
Level 1	U	U	U	No program lock features are enabled. On-chip user code is encrypted when verified, if encryption array is programmed.
Level 2	U	U	P	External code is prevented from fetching code bytes from on-chip code memory. Further programming of the on-chip OTPROM is disabled.
Level 3	U	P	P	Same as level 2, plus on-chip code memory verify is disabled.
Level 4	P	P	P	Same as level 3, plus external memory execution is disabled.

NOTE:
Other combinations of the lock bits are not defined.

The C151SX stores configuration information in an eight-byte configuration array at FFF8H–FFFFH. UCONFIG0 (FFF8H) and UCONFIG1 (FFF9H) are implemented; the remaining bytes are reserved for future use.

12.2 Lock Bit System

The 87C151SX provides a three-level lock system for protecting user program code stored in the on-chip code memory from unauthorized access. On the 83151SX, only LB1 protection is available. Table 19 describes the levels of protection.

12.3 Encryption Array

The 87C151SX and 83C151SX controllers include a 128-byte encryption array located in nonvolatile memory outside the memory address space. During verification of the on-chip code memory, the seven low-order address bits also address the encryption array. As the byte of the code memory is read, it is exclusive-NOR'ed (XNOR) with the key byte from the encryption array. If the encryption array is not programmed (still all 1s), the user program code is placed on the data bus in its original, unencrypted form. If the encryption array is programmed with key bytes, the user program code is encrypted and can't be used without knowledge of the key byte sequence. If the encryption feature is implemented, the portion of the on-chip code memory that does not contain program code should be filled with "random" byte values other than FFH to prevent the encryption key sequence from being revealed.

To preserve the secrecy of the encryption key byte sequence, the encryption array can not be verified.

12.4 Signature Bytes

The C151SX contains factory-programmed signature bytes. These bytes are located in nonvolatile memory outside the memory address space at 30H, 31H, and 60H. Signature byte values are listed in Table 20.

Table 20. Contents of the Signature Bytes

ADDRESS	CONTENTS	DEVICE TYPE
30H	89H	Indicates Intel Devices
31H	48H	Indicates MCS 151 Core Product
60H	7BH	Indicates 83C151SB Device
60H	FBH	Indicates 87C151SB Device
60H	7AH	Indicates 83C151SA Device
60H	FAH	Indicates 87C151SA Device

13.0 ON-CIRCUIT EMULATION (ONCE) MODE

The on-circuit emulation (ONCE) mode permits external testers to test and debug C151SX based systems without removing the chip from the circuit board. A clamp-on emulator or test CPU is used in place of the C151SX which is electrically isolated from the system.

To enter the ONCE mode:

1. Assert RST to initiate a device reset.
2. While holding RST asserted, apply and hold logic levels to I/O pins as follows: PSEN# = low, P0.7:5 = low, P0.4 = high, P0.3:0 = low (i.e., port 0 = 10H).
3. Deassert RST, then remove the logic levels from PSEN# and port 0.

These actions cause the C151SX to enter the ONCE mode. Port 1, 2, and 3 pins are weakly pulled high and port 0, ALE, and PSEN# pins are floating. Thus the device is electrically isolated from the remainder of the system which can then be tested by an emulator or test CPU. Note that in the ONCE mode the device oscillator remains active.

14.0 ON-CHIP OSCILLATOR

This clock source uses an external quartz crystal connected from XTAL1 to XTAL2 as the frequency-determining element (Figure 42). The crystal operates in its fundamental mode as an inductive reactance in parallel resonance with capacitance external to the crystal. Oscillator design considerations include crystal specifications, operating temperature range, and parasitic board capacitance. Consult the crystal manufacturer's datasheet for parameter values. With high quality components, C1 = C2 = 30 pF is adequate for this application.

Pins XTAL1 and XTAL2 are protected by on-chip electrostatic discharge (ESD) devices, D1 and D2, which are diodes parasitic to the R_F FETs. They serve as clamps to V_{CC} and V_{SS}. Feedback resistor R_F in the inverter circuit, formed from paralled n- and p- channel FETs, permits the PD bit in the PCON register to disable the clock during powerdown.

Noise spikes at XTAL1 and XTAL2 can disrupt microcontroller timing. To minimize coupling between other digital circuits and the oscillator, locate the crystal and the capacitors near the chip and connect to XTAL1, XTAL2, and V_{SS} with short, direct traces. To further reduce the effects of noise, place guard rings around the oscillator circuitry and ground the metal crystal case.

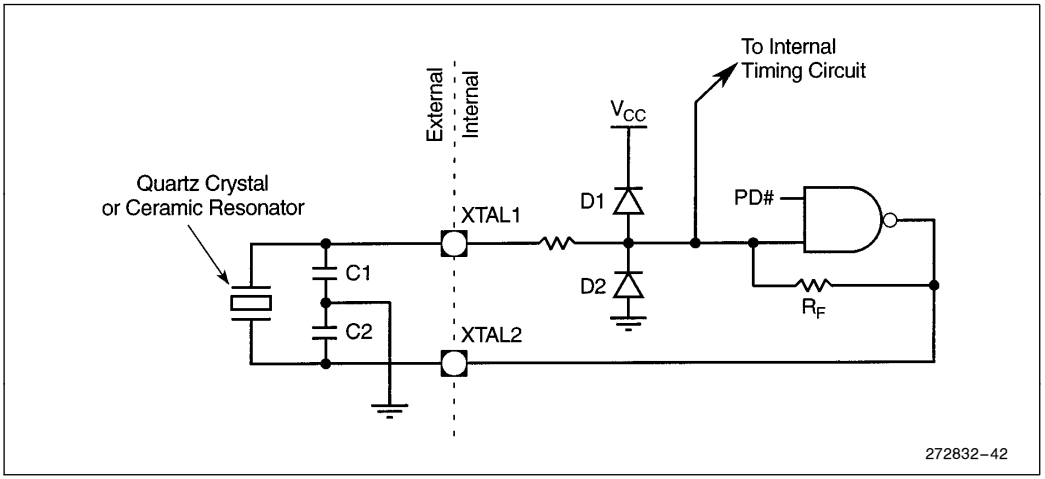
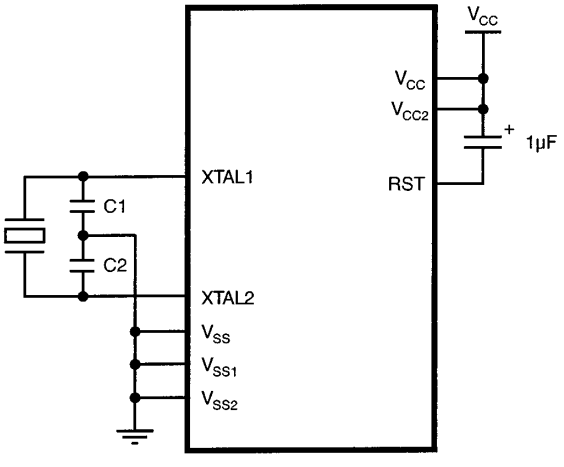


Figure 42. CHMOS On-chip Oscillator

For a more in-depth discussion of crystal specifications, ceramic resonators, and the selection of C1 and C2 see Applications Note AP-155, "Oscillators for Microcontrollers" in the *Embedded Applications* Databook.

In cost-sensitive applications, you may choose a ceramic resonator instead of a crystal. Ceramic resonator applications may require slightly different capacitor values and circuit configuration. Consult the manufacturer's datasheet for specific information.

To operate the C151SX from an external clock, connect the clock source to the XTAL1 pin as shown in Figure 43. Leave the XTAL2 pin floating. The external clock driver can be CMOS gate. If the clock driver is a TTL device, its output must be connected to V_{CC} through a 4.7 KΩ pullup resistor.



NOTE:
V_{CC2} is a secondary power pin that reduces power supply noise. V_{SS1} and V_{SS2} are secondary ground pins that reduce ground bounce and improve power supply by-passing. Connections to these pins are not required for proper device operation.

Figure 43. External Clock Connection

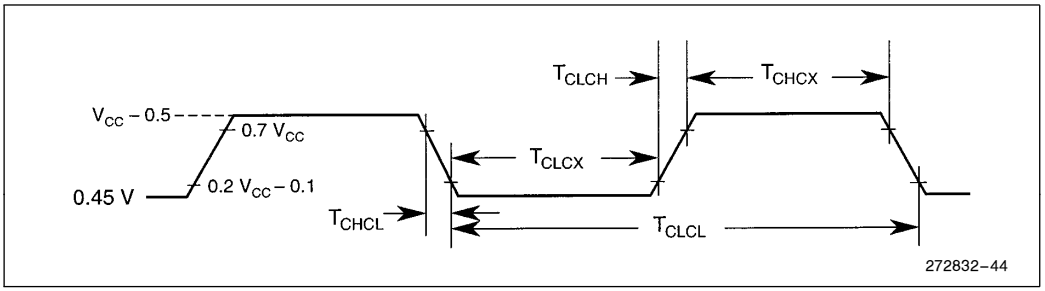


Figure 44. External Clock Drive Waveforms

For external clock drive requirements, see the device datasheet. Figure 44 shows the clock drive waveform. The external clock source must meet the minimum high and low times (T_{CHCX} and T_{CLCX}) and the maximum rise and fall times (T_{CLCH} and T_{CHCL}) to minimize the effect of external noise on the clock generator circuit. Long rise and fall times increase the chance that external noise will affect the clock circuitry and cause unreliable operation.

The external clock driver may encounter increased capacitance loading at XTAL1 due to the Miller effect of the internal inverter as the clock waveform builds up in amplitude following power on. Once the input waveform requirements are met, the input capacitance remains under 20 pF.

ADDITIONAL REFERENCES

The following application notes provide supplemental information to this document and can be found in the *Embedded Applications* Databook.

1. AP-125 "Designing Microcontroller Systems for Electrically Noisy Environments"
2. AP-155 "Oscillators for Microcontrollers"
3. 8XC151SA/SB Datasheet
4. 8XC251SA/SB/SP/SQ Datasheets
5. 8XC251SA/SB/SP/SQ User Manual
6. MCS® 51 Family User Manual

15.0 INSTRUCTION SET REFERENCE

Table 21. Summary of Add and Subtract Instructions

Add	ADD <dest>,<src>	dest opnd ← dest cond + src opnd
Add with Carry	ADDC <dest>,<src>	(A) ← (A) − src opnd + carry bit
Subtract with Borrow	SUBB <dest>,<src>	(A) ← (A) − src opnd − carry bit

Mnemonic	<desc>,<src>	Notes	Bytes	States
ADD	A, Rn	Reg to Acc	1	1
	A, Dir8	Dir Byte to Acc	2	1 (1)
	A, @RI	Indir Addr to Acc	1	2
	A, #data	Immediate Data to Acc	2	1
ADDC; SUBB	A, Rn	Reg to/from Acc with Carry	1	1
	A, Dir8	Dir Byte to/from Acc with Carry	2	1 (1)
	A, @RI	Indir RAM to/from Acc with Carry	1	2
	A, #data	Immediate Data to/from Acc with Carry	2	1

NOTE:
1. If this instruction addresses an I/O port (P*x*, *x* = 3:0), add 1 to the number of the states.

Table 22. Summary of Increment and Decrement Instructions

Increment	INC DPTR	(DPTR) ← (DPTR) + 1
Increment	INC byte	byte ← byte + 1
Increment	INC <dest>,<src>	dest opnd ← dest opnd + src opnd
Decrement	DEC byte	byte ← byte − 1

Mnemonic	<**desc**>,<**src**>	Notes	Bytes	States
INC; DEC	A	Acc	1	1
	Rn	Reg	1	1
	Dir8	Dir Byte	2	2 (1)
	@RI	Indir RAM	1	3
INC	DPTR	Data Pointer	1	1

NOTE:
1. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 2 to the number of the states.

Table 23. Summary of Multiply, Divide, and Decimal-adjust Instructions

Multiply	MUL AB	(B:A) = A × B
Divide	DIV AB	(A) = Quotient; (B) = Remainder
Decimal-adjust ACC for Addition (BCD)	DA A	

Mnemonic	<**desc**>,<**src**>	Notes	Bytes	States
MUL	AB	Multiply A and B	1	5
DIV	AB	Divide A by B	1	10
DA	A	Decimal Adjust Acc	1	1

Table 24. Summary of Logical Instructions

Logical AND	ANL <dest>,<src>	dest opnd ← dest opnd A src opnd
Logical OR	ORL <dest>,<src>	dest opnd ← dest opnd V src opnd
Logical Exclusive OR	XRL <dest>,<src>	dest opnd ← dest opnd V src opnd
Clear	CLR A	(A) ← 0
Complement	CPL A	(AI) ← Ø(AI)
Rotate	RXX A	(1)
SWAP	A	A3:0 ⟷ A7:4

Mnemonic	<**desc**>,<**src**>	Notes	Bytes	States
ANL; ORL; XRL	A, Rn	Reg to Acc	1	1
	A, Dir8	Dir Byte to Acc	2	1 (1)
	A, @RI	Indir Addr to Acc	1	2
	A, #data	Immediate Data to Acc	2	1
	Dir8, A	Acc to Dir Byte	2	2 (2)
	Dir8, #data	Immediate Data to Dir Byte	3	3 (2)
CLR	A	Clear Acc	1	1
CPL	A	Complement Acc	1	1
RL	A	Rotate Acc Left	1	1
RLC	A	Rotate Acc Left Through the Carry	1	1
RR	A	Rotate Acc Right	1	1
RRC	A	Rotate Acc Right Through the Carry	1	1
SWAP	A	Swap Nibbles within the Acc	1	2

NOTES:
1. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 1 to the number of the states.
2. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 2 to the number of the states.

Table 25. Summary of Move Instructions

Move	MOV <dest>,<src>	destination ← src opnd
Move Code Byte	MOVC <dest>,<src>	A ← code byte
Move to External Mem	MOVX <dest>,<src>	external mem ← (A)
Move from External Mem	MOVX <dest>,<src>	A ← source opnd in external mem

Mnemonic	<desc>,<src>	Notes	Bytes	States
MOV	A, Rn	Reg to Acc	1	1
	A, Dir8	Dir Byte to Acc	2	1 (1)
	A, @RI	Indir RAM to Acc	1	2
	A, #data	Immediate Data to Acc	2	1
	Rn, A	Acc to Reg	1	1
	Rn, Dir8	Dir Byte to Reg	2	1 (1)
	Rn, #data	Immediate Data to Reg	2	1
	Dir8, A	Acc to Dir Byte	2	2 (1)
	Dir8, Rn	Reg to Dir Byte	2	2 (1)
	Dir8, Dir8	Dir Byte to Dir Byte	3	3
	Dir8, @RI	Indir RAM to Dir Byte	2	3
	Dir8, #data	Immediate Data to Dir Byte	3	3 (1)
	@RI, A	Acc to Indir RAM	1	3
	@RI, Dir8	Dir Byte to Indir RAM	2	3
	@RI, #data	Immediate Data to Indir RAM	2	3
	DPTR, #data16	Load Data Pointer with a 16-bit Const	3	2
MOVC	A, @A + DPTR	Code Byte Relative to DPTR to Acc	1	6
	A, @A + PC	Code Byte Relative to PC to Acc	1	6
MOVX	A, @RI	External Mem (8-bit Addr) to Acc	1	4
	A, @DPTR	External Mem (16-bit Addr) to Acc	1	5
	@RI, A	Acc to External Mem (8-bit Addr)	1	4
	@DPTR, A	Acc to External Mem (16-bit Addr)	1	5

NOTE:
1. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 1 to the number of the states.

Table 26. Summary of Exchange, Push, and Pop Instructions

Exchange Contents	XCH <dest>,<src>	A ⟷ src opnd
Exchange Digit	XCH0 <dest>,<src>	A3:0 ⟷ on-chip RAM bits 3:0
Push	PUSH <src>	SP ← SP + 1; (SP) ← src
Pop	POP <dest>	dest ← (SP); SP ← SP − 1

Mnemonic	<desc>,<src>	Notes	Bytes	States
XCH	A, Rn	Acc and Reg	1	3
	A, DirB	Acc and Dir Addr	2	3 (1)
	A, @RI	Acc and On-chip RAM (8-bit Addr)	1	4
XCHD	A, @RI	Acc and Low Nibble in On-chip RAM (8-bit Addr)	1	4
PUSH	DirB	Push Dir Byte onto Stack	2	2
POP	Dir	Pop Dir Byte from Stack	2	3/3

NOTE:
1. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 2 to the number of the states.

Table 27. Summary of Bit Instructions

Clear Bit	CLR bit	bit ← 0
Set Bit	SETB bit	bit ← 1
Complement Bit	CPL bit	bit ← Ø bit
AND Carry with Bit	ANL CY, /bit	CY ← CY ∧ bit
AND Carry with Complement of Bit	ANL CY, bit	CY ← CY ∧ Ø bit
OR Carry with Bit	ORL CY, bit	CY ← CY V bit
OR Carry with Complement of Bit	ORL CY, /bit	CY ← CY V Ø bit
Move Bit to Carry	MOV CY, bit	CY ← bit
Move Bit from Carry	MOV bit, CY	bit ← CY

Mnemonic	<desc>,<src>	Notes	Bytes	States
CLR	CY	Clear Carry	1	1
	Bit51	Clear Dir Bit	2	2 (1)
SETB	CY	Set Carry	1	1
	Bit51	Set Dir Bit	2	2 (1)
CPL	CY	Complement Carry	1	1
	Bit51	Complement Dir Bit	2	2 (1)
ANL	CY, Bit51	AND Dir Bit to Carry	2	1 (2)
ANL/	CY, /Bit51	AND Complemented Dir Bit to Carry	2	1 (2)
ORL	CY, Bit51	OR Dir Bit to Carry	2	1 (2)
ORL/	CY, /Bit51	OR Complemented Dir Bit to Carry	2	1 (2)
MOV	CY, Bit51	Move Dir Bit to Carry	2	1 (2)
	Bit51, CY	Move Carry to Dir Bit	2	2 (1)

NOTES:
1. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 2 to the number of the states.
2. If this instruction addresses an I/O port (P*x*, *x* = 0–3), add 1 to the number of the states.

Table 28. Summary of Control Instructions

Mnemonic	<desc>,<src>	Notes	Bytes	States (1)
ACALL	Addr11	Absolute Subroutine Call	2	9
LCALL	Addr16	Long Subroutine Call	3	9
RET		Return from Subroutine	1	6
ERET		Extended Subroutine Return	3	10
RETI		Return from Interrupt	1	6
AJMP	Addr11	Absolute Jump	2	3
LJMP	Addr16	Long Jump	3	4
SJMP	Rel	Short Jump (Relative Addr)	2	3
JMP	@A + DPTR	Jump Indir Relative to the DPTR	1	5
JC	Rel	Jump if Carry is Set	2	1/4
JNC	Rel	Jump if Carry not Set	2	1/4
JB	Bit51, Rel	Jump if Dir Bit is Set	3	2/5
JNB	Bit51, Rel	Jump if Dir Bit is not Set	3	2/5
JBC	Bit51, Rel	Jump if Dir Bit is Set and Clear Bit	3	4/7
JZ	Rel	Jump if Acc is Zero	2	2/5
JNZ	Rel	Jump if Acc not Zero	2	2/5
CJNE	A, Dir8, Rel	Compare Dir Byte to Acc and Jump if not Equal	3	2/5
	A, #data, Rel	Compare Immediate to Acc and Jump if not Equal	3	2/5
	Rn, #data, Rel	Compare Immediate to Reg and Jump if not Equal	3	2/5
	@RI, #data, Rel	Compare Immediate to Indir and Jump if not Equal	3	3/6
DJNZ	Rn, rel	Decrement Reg and Jump if not Zero	2	2/5
	Dir8, Rel	Decrement Dir Byte and Jump if not Zero	3	3/6
NOP	—	No Operation	1	1

NOTE:
1. For conditional jumps, times are given as not-taken/taken.