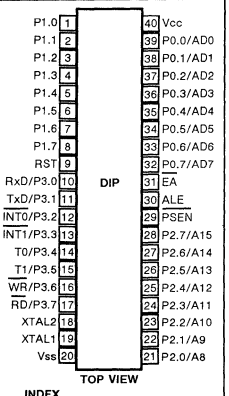
INTEGRATED CIRCUITS

Signetics Microcontroller Users' Guide

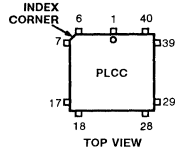
Signetics
Philips Components

PHILIPS

SCN8031AH/SCN8051AH



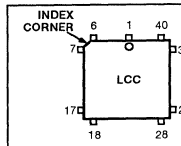
TOP VIEW



TOP VIEW

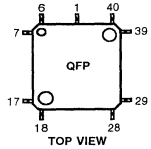
Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

SC80C31B/SC80C51B



TOP VIEW

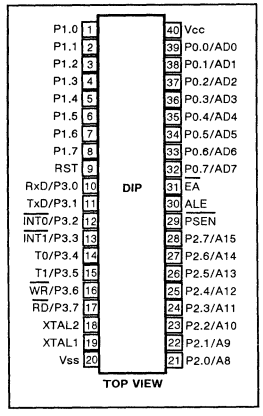
Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	Vss
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

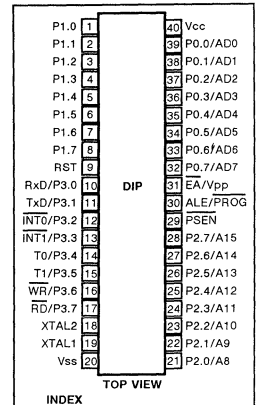
SC80C31B/SC80C51B



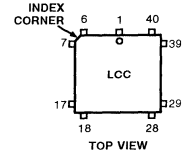
TOP VIEW

See inside of back cover for additional pins.

SC87C51



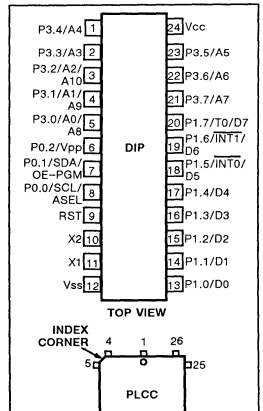
TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE/PROG
12	NC	34	NC
13	TxD/P3.1	35	EA/Vpp
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

S87C751

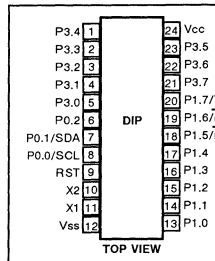


TOP VIEW

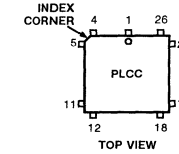
TOP VIEW

Pin	Function	Pin	Function
1	P3.4/A4	15	P1.0/D0
2	P3.3/A3	16	P1.1/D1
3	P3.2/A2/A10	17	P1.2/D2
4	P3.1/A1/A9	18	P1.3/D3
5	N.C.	19	P1.4/D4
6	P3.0/A0/A8	20	P1.5/INT0/D5
7	P0.2/Vpp	21	N.C.
8	P0.1/SDA/ OE-PGM	22	N.C.
9	P0.0/SCL/ ASEL	23	P1.6/INT1/D6
10	N.C.	24	P1.7/T0/D7
11	RST	25	P3.7/A7
12	X2	26	P3.6/A6
13	X1	27	P3.5/A5
14	Vss	28	Vcc

S83C751



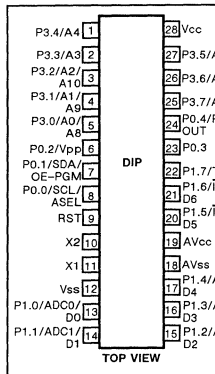
TOP VIEW



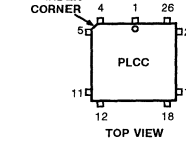
TOP VIEW

Pin	Function	Pin	Function
1	P3.4	15	P1.0
2	P3.3	16	P1.1
3	P3.2	17	P1.2
4	P3.1	18	P1.3
5	N.C.	19	P1.4
6	P3.0	20	P1.5/INT0
7	P0.2	21	N.C.
8	P0.1/SDA	22	N.C.
9	P0.0/SCL	23	P1.6/INT1
10	N.C.	24	P1.7/T0
11	RST	25	P3.7
12	X2	26	P3.6
13	X1	27	P3.5
14	Vss	28	Vcc

S87C752



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	P3.4/A4	15	P1.2/ADC2/D2
2	P3.3/A3	16	P1.3/ADC3/D3
3	P3.2/A2/A10	17	P1.4/ADC4/D4
4	P3.1/A1/A9	18	AVss
5	P3.0/A0/A8	19	AVcc
6	P0.2/Vpp	20	P1.5/INT0/D5
7	P0.1/SDA/ OE-PGM	21	P1.6/INT1/D6
8	P0.0/SCL/ ASEL	22	P1.7/T0/D7
9	RST	23	P0.3
10	X2	24	P0.4/PWM OUT
11	X1	25	P3.7/A7
12	Vss	26	P3.6/A6
13	P1.0/ADC0/D0	27	P3.5/A5
14	P1.1/ADC1/D1	28	Vcc

NOTE:
A0–A10 and D0–D7 available for EPROM verify only.

Signetics

Microprocessor Products

Microcontroller Users' Guide

Signetics reserves the right to make changes, without notice, in the products, including circuits, standard cells, and/or software, described or contained herein in order to improve design and/or performance. Signetics assumes no responsibility or liability for the use of any of these products, conveys no license or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified. Applications that are described herein for any of these products are for illustrative purposes only. Signetics makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Portions of this users' guide are printed under a license from Intel Corporation.

LIFE SUPPORT APPLICATIONS

Signetics Products are not designed for use in life support appliances, devices, or systems where malfunction of a Signetics Product can reasonably be expected to result in a personal injury. Signetics customers using or selling Signetics' Products for use in such applications do so at their own risk and agree to fully indemnify Signetics for any damages resulting from such improper use or sale.

Signetics registers eligible circuits under
the Semiconductor Chip Protection Act.

Signetics

Contents

Microprocessor Products

Signetics

Microprocessor Products

Section 1
8051 Family

INDEX

INDEX (Continued)

Section 1
8051 FAMILY OVERVIEW

The Signetics 8051 family of products is based on the industry standard for 8-bit high performance microcontrollers. The architecture for the family has been optimized for sequential real time control applications. The 8051 family of products are used in a wide range of applications from those that are relatively simple to applications in medical instrumentation and automobile control systems. All of the devices included in the family are available in versions that have either internal ROM, EPROM, or CPU only. With the exception of the 83C751 and 752 (which are limited to on-board memory) all of the devices in the family can address up to 64k bytes of both program and data memory.

The 8051 family of microcontrollers includes the devices listed in Table 1. The basic architecture of these devices is shown in Figure 1.

8051

The 8051 is the original member of the family. Among the features of the 8051 are:
- 8-bit CPU optimized for control applications
- Extensive Boolean processing (single bit logic) capabilities
- 32 bi-directional and individual addressable I/O lines
- 128 bytes of on-chip data RAM
- Two 16-bit timer/counters
- Full duplex UART
- 5-source interrupt structure with 2 priority levels
- On-chip clock oscillator
- 4K bytes of on-chip program memory
- 64K bytes program memory address space
- 64K bytes data memory address space
- 40-pin DIP and 44-pin PLCC packages

The 8031 is a CPU only version of the 8051 and only differs from the 8051 in that it does not have on-chip ROM. The 8031 fetches all instructions from external memory.

8051AH

The 8051AH is identical to the 8051, but is fabricated with HMOS II technology. It is pin-for-pin compatible with the 8051. The ROMless version of the 8051AH is the 8031AH.

80C51BH

The 80C51BH is the CHMOS version of the 8051. Functionally it is fully compatible with the 8051, but being CMOS it draws less current than its HMOS counterpart.

The ROMless version of the 80C51BH is the 80C31BH. The EPROM version is the 87C51.

8052AH

The 8052AH is an enhanced version of the 8051. It is fabricated with HMOS II technology, and is upwards compatible with the 8051. Its enhancements over the 8051 include:

- 256 bytes of on-chip data RAM
- Three counter/timers
- A 6-source interrupt structure
- 8K bytes of on-chip program memory
- 40-pin DIP and 44-pin PLCC packages

The ROMless version of the 8052AH is the 8032AH.

83C451

The 83C451 is an extended I/O version of the 80C51 with the following features:
- Seven 8-bit quasi-bidirectional I/O ports (PLCC version).
- Six 8-bit and one 4-bit quasi-bidirectional I/O ports (DIP version).
- Mailbox port (port 6) features:
  - Operation as normal quasi-bidirectional I/O port
  - 4 handshake control pins
  - Control status register
  - Input and output buffer registers making port 6 suitable for:
    - direct MPU interface
    - parallel printer interface
- 64-pin DIP and 68-pin packages.

All other aspects of the 83C451 are identical to the 80C51. The 87C451 is the EPROM version of this device.

Table 1. 8051 Family of Microcontrollers

Device Name	ROMless Version	EPROM Version	ROM Bytes	RAM Bytes	16-Bit Timers	Circuit Type
8051	8031	–	4K	128	2	HMOS
8051AH	8031AH	–	4K	128	2	HMOS
80C51BH	80C31BH	87C51BH	4K	128	2	CHMOS
8052AH	8032AH	–	8K	256	3	HMOS
83C451	80C451	87C451	4K	128	2	CHMOS
83C552	80C552	87C552	8K	256	3	CHMOS
83C652	80C652	87C652	8K	256	2	CHMOS
83C751	–	87C751	2K	64	1	CHMOS
83C752	–	87C752	2K	64	1	CHMOS

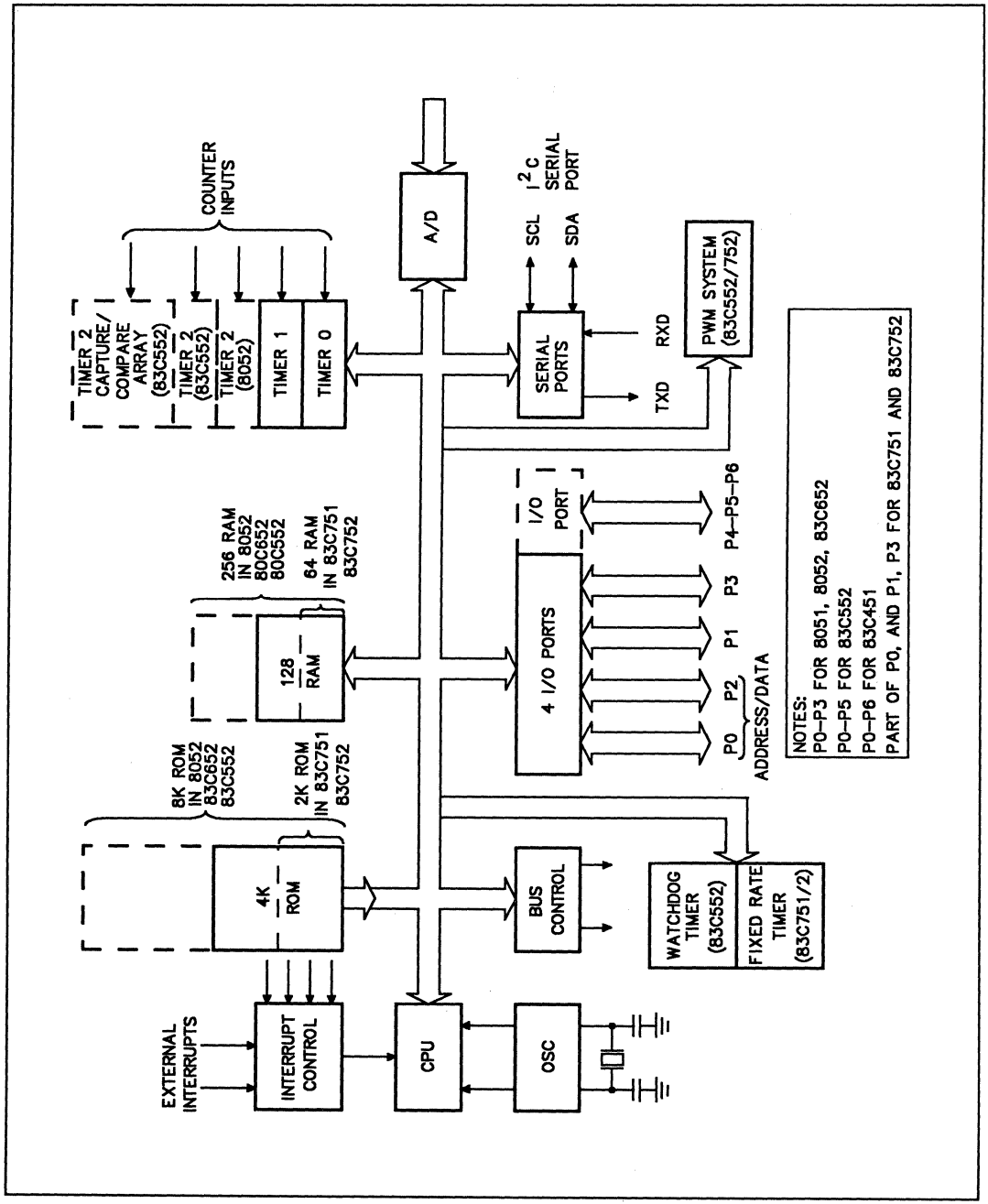


Figure 1. 8051 Family Block Diagram

83C552

The 83C552 is an extended function 80C51 with the following features:

- 8k bytes of ROM
- 256 bytes of RAM
- 10-bit 8 channel A/D
- Counter/timer array with high speed outputs and capture inputs
- 4 counter/timers (including a watchdog timer)
- 2 PWM outputs
- 2 serial ports
- 6 8-bit I/O ports
- I2C serial port
- 68-pin PLCC package

The 83C552 is 100% code compatible with the 80C51. The ROMless version of the 83C552 is the 80C552 and the EPROM version is the 87C552.

83C652

The 83C652 is an 80C51 with the following additions: an 8k ROM, 256 bytes RAM and I2C serial port.

The 83C652 is pin-for-pin compatible with the 80C51 except for minor DC specifications on the I2C serial port pins.

The ROMless version of the 83C652 is the 80C652.

83C751

The 83C751 is a 24-pin version of the 80C51, where small size and cost are of prime consideration. The 83C751 is packaged in a 24-pin "skinnydip" (.300 wide) and in a 28-pin PLCC package.

The following differences exist between the 83C751 and the 80C51. The 83C751 has:

- 2K bytes ROM
- 64 bytes RAM
- I2C serial port (no UART)
- 19 I/O lines
- Single level interrupt structure
- One counter/timer with 16-bit autoload
- No external memory expandability (data memory can be expanded using the I2C serial port and I2C compatible memory devices).

Note that since there is no external expandability, the external memory addressing signals: $\overline{WR}$, $\overline{RD}$, $\overline{PSEN}$, $\overline{EA}$, and ALE are not present. Because of these differences, instructions LJMP, LCALL, and MOVX have no meaning.

For all other instructions the 83C751 is 100% code compatible with the 80C51 and operates at full 80C51 speed.

The EPROM version of the 83C751 is the 87C751.

83C752

The 83C752 is a derivative version of the 80C51 that is intended for use in automotive, electro-mechanical, and consumer applications. The emphasis of the device is on high integration and small packaging. The 83C752 contains most of the features of the 80C51 with the following exceptions:

- 2K bytes ROM
- 64 bytes RAM
- Single level interrupt structure
- One 16-bit counter/timer (mode 2 only) with 16-bit autoload
- Two 8-bit and one 5-bit bidirectional I/O ports
- I2C serial interface
- One PWM with timer, including overflow interrupt capability
- 5 channels of 8-bit A/D
- 28-pin packages, both DIP and SMD

The 83C752 does not have external memory expandability. The EPROM versions of the 83C752 is the 87C752.

Table 2. 80C51 Derivative Comparisons

Device	A/D	Ports	PWM	Timers	UART
80C51	–	4	–	Two standard	Standard
83C451	–	7	–	Two standard	Standard
83C552	8 channel/ 10 bit	6	2	Two standard, timer 2, watchdog (4 total)	Standard, I2C
83C652	–	4	–	Two standard	Standard, I2C
83C751	–	2-3/8	–	One standard, extended to 16-bit autoload	I2C
83C752	5 channel/ 8 bit	2-5/8	1	One standard, extended to 16-bit autoload	I2C

8051 ARCHITECTURE

MEMBERS OF THE FAMILY

The basic architecture of the 8051 devices is as follows:

8051

The 8051 is the original member of the family (see Figure 2). Among the features of the 8051 are:

- 8-bit CPU optimized for control applications
- Extensive Boolean processing (single-bit logic) capabilities
- 32 bi-directional and individually addressable I/O lines
- 128 bytes of on-chip data RAM
- Two 16-bit timer/counters
- Full duplex UART
- 5 source interrupt structure with 2 priority levels
- On-chip clock oscillator
- 4K bytes of on-chip program memory
- 64K program memory address space
- 64K data memory address space

The 8031 differs from the 8051 in not having the on-chip program ROM. Instead, the 8031 fetches all instructions from external memory.

8051AH

The 8051AH is identical to the 8051, but is fabricated with HMOS II technology. It is pin-for-pin compatible with the 8051. The ROMless version of the 8051AH is the 8031AH.

80C51BH

The 80C51BH is the CHMOS version of the 8051. Functionally, it is fully compatible with the 8051, but being CMOS it draws less current than the HMOS counterpart. To further exploit the power savings available in CMOS circuitry, two reduced power modes are added:

1. Software-invoked idle mode, during which the CPU is turned off while the RAM and other on-chip peripherals continue operating. In this mode, current draw is reduced to about 15% of the current drawn when the device is fully active.
2. Software-invoked power down mode, during which all on-chip activities are suspended. The on-chip RAM continues to hold its data. In this mode the device typically draws less than 10μA.

Although the 80C51BH is functionally compatible with its HMOS counterpart, specific differences between the two types of devices must be considered in the design of an application circuit if one wishes to ensure complete interchangeability between the HMOS and CHMOS devices. The ROMless version of the 80C51BH is the 80C31BH. The EPROM version is the 87C51.

8051 FAMILY DEVICES MEMORY ORGANIZATION

All 8051 devices have separate address spaces for program and data memory, as shown in Figure 3. The logical separation of program and data memory allows the data memory to be accessed by 8-bit addresses, which can be more quickly stored and manipulated by an 8-bit CPU. Nevertheless, 16-bit data memory addresses can also be generated through the DPTR register.

Program memory can only be read, not written to. There can be up to 64K bytes of program memory. In the 8051, 8051AH, 80C51BH, and their EPROM versions, the lowest 4K bytes of program memory are on-chip. In the ROMless versions (8031, 8031AH, 80C31BH) all program memory is external. The read strobe for external program memory is the $\overline{PSEN}$ (program store enable).

Data Memory occupies a separate address space from Program Memory. Up to 64K bytes of external RAM can be addressed in the external Data Memory space. The CPU generates read and write signals, $\overline{RD}$ and $\overline{WR}$, as needed during external Data Memory accesses.

External Program Memory and external Data Memory may be combined if desired by applying the $\overline{RD}$ and $\overline{PSEN}$ signals to the inputs of an AND gate and using the output of the gate as the read strobe to the external Program/Data memory.

PROGRAM MEMORY

Figure 4 shows a map of the lower part of the Program Memory. After reset, the CPU begins execution from location 0000H. As shown in Figure 4, each interrupt is assigned a fixed location in Program Memory. The interrupt causes the CPU to jump to that location, where it commences execution of the service routine. External Interrupt 0, for example, is assigned to location 0003H. If External Interrupt 0 is going to be used, its service routine must begin at location 0003H. If the interrupt is not going to be used, its service location is available as general purpose Program Memory.

The interrupt service locations are spaced at 8-byte intervals: 0003H for External Interrupt 0, 000BH for Timer 0, 0013H for External Interrupt 1, 001BH for Timer 1, etc. If an interrupt service routine is short enough (as is often the case in control applications), it can reside entirely within that 8-byte interval. Longer service routines can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use.

The lowest 4K bytes of Program Memory can either be in the on-chip ROM or in an external ROM. This selection is made by strapping the $\overline{EA}$ (External Access) pin to either V_{CC}, or V_{SS}.

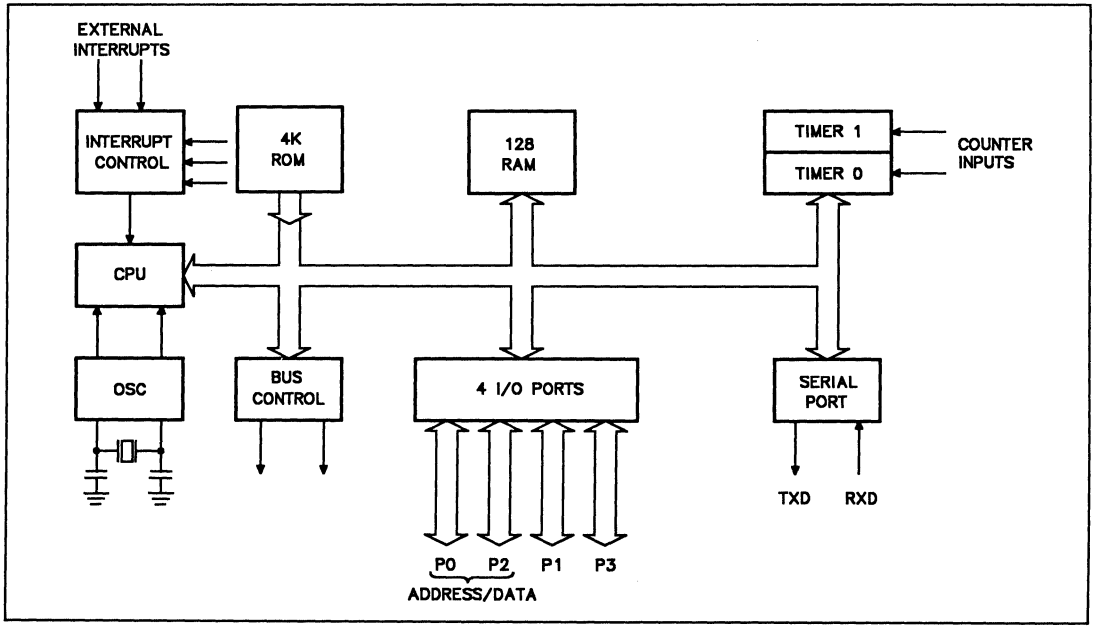


Figure 2. 8051 Block Diagram

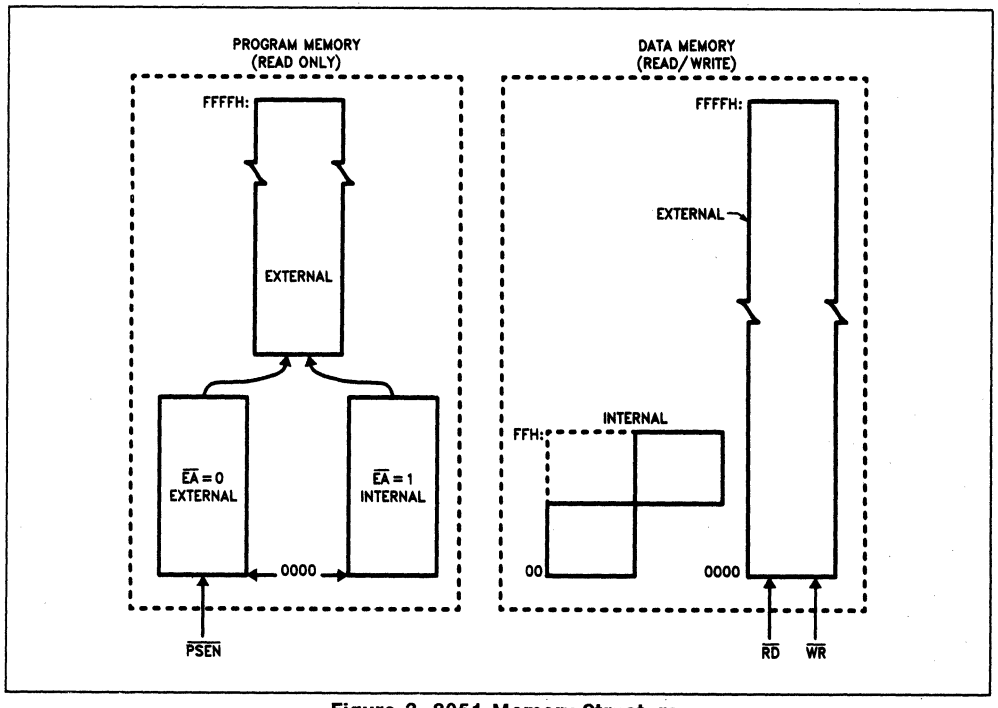


Figure 3. 8051 Memory Structure

In the 8051, if the $\overline{EA}$ pin is strapped to Vcc, then the program fetches to addresses 0000H through 0FFFH are directed to the internal ROM. Program fetches to addresses 1000H through FFFFH are directed to external ROM.

If the $\overline{EA}$ pin is strapped to Vss, then all program fetches are directed to external ROM. The ROMless parts (8031, 80C31, etc..) must have this pin externally strapped to Vss to enable them to execute from external Program Memory.

The read strobe to external ROM, $\overline{PSEN}$, is used for all external program fetches. $\overline{PSEN}$ is not activated for internal program fetches.

The hardware configuration for external program execution is shown in Figure 5. Note that 16 I/O lines (Ports 0 and 2) are dedicated to bus functions during external Program Memory fetches. Port 0 (P0 in Figure 5) serves as a multiplexed address/data bus. It emits the low byte of the Program Counter (PCL) as an address, and then goes into a float state awaiting the arrival of the code byte from the Program Memory. During the time that the low byte of the Program Counter is valid on Port 0, the signal ALE (Address Latch Enable) clocks this byte into an address latch. Meanwhile, Port 2 (P2 in Figure 5) emits the high byte of the Program Counter (PCH). Then $\overline{PSEN}$ strobes the EPROM and the code byte is read into the microcontroller.

Program Memory addresses are always 16 bits wide, even though the actual amount of Program Memory used may be less than 64K bytes. External program execution sacrifices two of the 8-bit ports, P0 and P2, to the function of addressing the Program Memory.

DATA MEMORY

The right half of Figure 3 shows the internal and external Data Memory spaces available to the 8051 user. Figure 6 shows a hardware configuration for accessing up to 2K bytes of external RAM. The CPU in this case is executing from internal ROM. Port 0 serves as a multiplexed address/data bus to the RAM, and 3 lines of Port 2 are being used to page the RAM. The CPU generates $\overline{RD}$ and $\overline{WR}$ signals as needed during external RAM accesses. There can be up to 64K bytes of external Data Memory. External Data Memory addresses can be either 1 or 2 bytes wide. One-byte addresses are often used in conjunction with one or more other I/O lines to page the RAM, as shown in Figure 6.

Two-byte addresses can also be used, in which case the high address byte is emitted at Port 2.

Internal Data Memory is mapped in Figure 7. The memory space is shown divided into three blocks, which are generally referred to as the Lower 128, the Upper 128, and SFR space.

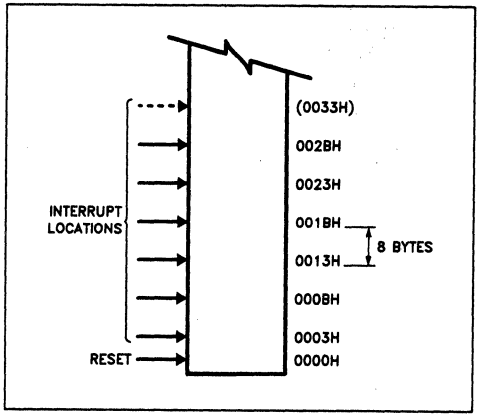


Figure 4. 8051 Program Memory

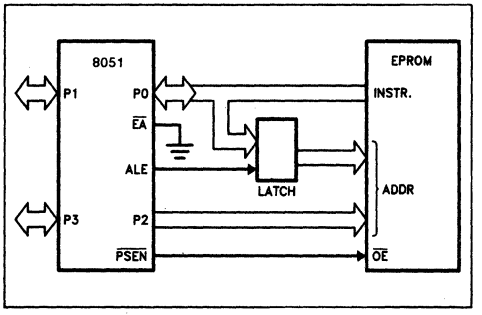


Figure 5. Executing from External Program Memory

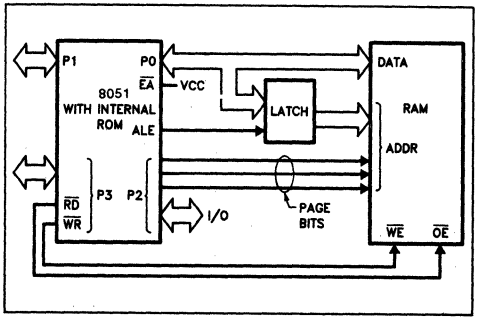


Figure 6. Accessing External Data Memory. If the Program Memory is Internal, the other Bits of P2 are Available as I/O

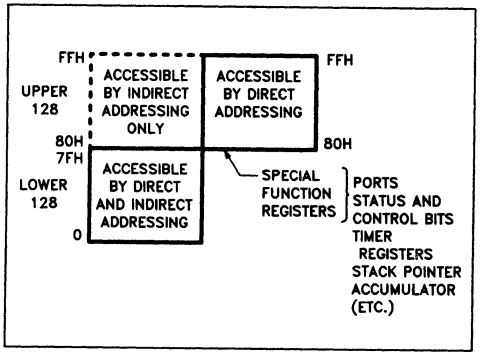


Figure 7. Internal Data Memory

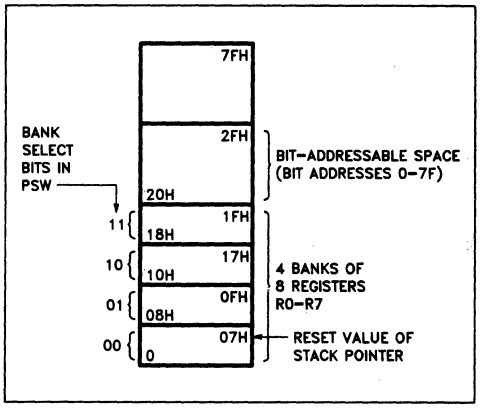


Figure 8. Lower 128 Bytes of Internal RAM

Internal Data Memory addresses are always one byte wide, which implies an address space of only 256 bytes. However, the addressing modes for internal RAM can in fact accommodate 384 bytes, using a simple trick. Direct addresses higher than 7FH access one memory space, and indirect addresses higher than 7FH access a different memory space. Thus Figure 7 shows the Upper 128 and SFR space occupying the same block of addresses, 80H through FFH, although they are physically separate entities.

The Lower 128 bytes of RAM are present in all 8051 devices as mapped in Figure 8. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The 8051 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

All of the bytes in the Lower 128 can be accessed by either direct or indirect addressing. The Upper 128 (Figure 9) can only be accessed by indirect addressing.

Figure 10 gives a brief look at the Special Function Register (SFR) space. SFRs include the Port latches, timers, peripherals controls, etc. These registers can only be accessed by direct addressing. Sixteen addresses in SFR space are both byte- and bit-addressable. The bit-addressable SFRs are those whose address ends in 0H or 8H. The bit addresses in this area are 80H through FFH.

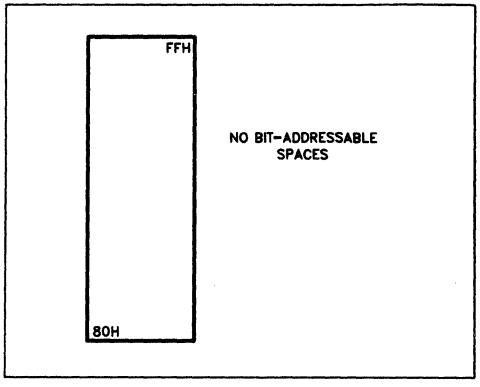


Figure 9. Upper 128 Bytes of Internal RAM

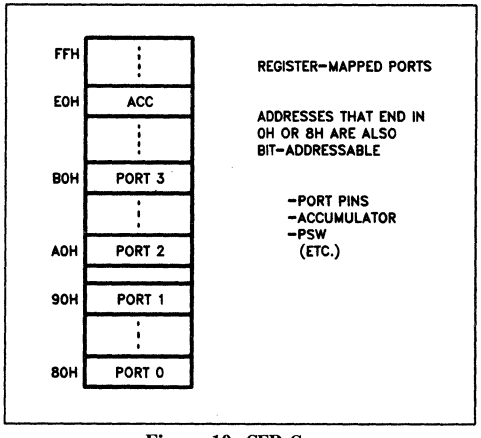


Figure 10. SFR Space

8051 FAMILY INSTRUCTION SET

The 8051 instruction set is optimized for 8-bit control applications. It provides a variety of fast addressing modes for accessing the internal RAM to facilitate byte operations on small data structures. The instruction set provides extensive support for one-bit variables as a separate data type, allowing direct bit manipulation in control and logic systems that require Boolean processing.

PROGRAM STATUS WORD

The Program Status Word (PSW) contains several status bits that reflect the current state of the CPU. The PSW, shown in Figure 11, resides in SFR space. It contains the Carry bit, the Auxiliary Carry (for BCD operations), the two register bank select bits, the Overflow flag, a Parity bit, and two user-definable status flags.

The Carry bit, other than serving the function of a Carry bit in arithmetic operations, also serves as the "Accumulator" for a number of Boolean operations.

The bits RS0 and RS1 are used to select one of the four register banks shown in Figure 8. A number of instructions refer to these RAM locations as R0 through R7. The selection of which of the four is being referred to is made on the basis of the RS0 and RS1 at execution time.

The Parity bit reflects the number of 1s in the Accumulator: P = 1 if the Accumulator contains an odd number of 1s, and P = 0 if the Accumulator contains an even number of 1s. Thus the number of 1s in the Accumulator plus P is always even. Two bits in the PSW are uncommitted and may be used as general purpose status flags.

ADDRESSING MODES

The addressing modes in the 8051 instruction set are as follows:

DIRECT ADDRESSING

In direct addressing the operand is specified by an 8-bit address field in the instruction. Only internal Data RAM and SFRs can be directly addressed.

INDIRECT ADDRESSING

In indirect addressing the instruction specifies a register which contains the address of the operand. Both internal and external RAM can be indirectly addressed.

The address register for 8-bit addresses can be R0 or R1 of the selected bank, or the Stack Pointer. The address register for 16-bit addresses can only be the 16-bit "data pointer" register, DPTR.

REGISTER INSTRUCTIONS

The register banks, containing registers R0 through R7, can be accessed by certain instructions which carry a 3-bit register specification within the opcode of the instruction. Instructions that access the registers this way are code efficient, since this mode eliminates an address byte. When the instruction is executed, one of the eight registers in the selected bank is accessed. One of four banks is selected at execution time by the two bank select bits in the PSW.

REGISTER-SPECIFIC INSTRUCTIONS

Some instructions are specific to a certain register. For example, some instructions always operate on the Accumulator, or Data Pointer, etc., so no address byte is needed to point to it. The opcode itself does that. Instructions that refer to the Accumulator as A assemble as accumulator specific opcodes.

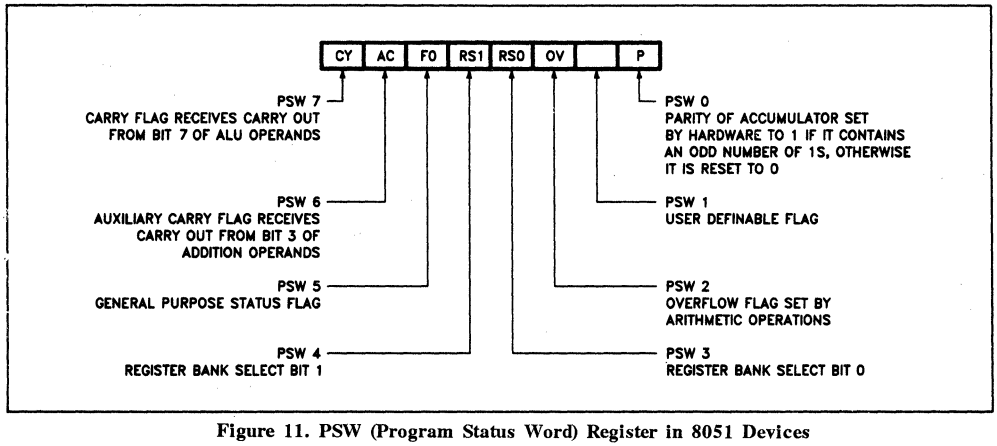


Figure 11. PSW (Program Status Word) Register in 8051 Devices

IMMEDIATE CONSTANTS

The value of a constant can follow the opcode in Program Memory. For example,

MOV A, #100

loads the Accumulator with the decimal number 100. The same number could be specified in hex digits as 64H.

INDEXED ADDRESSING

Only program Memory can be accessed with indexed addressing, and it can only be read. This addressing mode is intended for reading look-up tables in Program Memory. A 16-bit base register (either DPTR or the Program Counter) points to the base of the table, and the Accumulator is set up with the table entry number. The address of the table entry in Program Memory is formed by adding the Accumulator data to the base pointer.

Another type of indexed addressing is used in the "case jump" instruction. In this case the destination address of a jump instruction is computed as the sum of the base pointer and the Accumulator data.

ARITHMETIC INSTRUCTIONS

The menu of arithmetic instructions is listed in Table 3. The table indicates the addressing modes that can be used with each instruction to access the <byte> operand. For example, the ADD A,<byte> instruction can be written as:

ADD A, 7FH (direct addressing)
ADD A, @R0 (indirect addressing)
ADD A, R7 (register addressing)
ADD A, #127 (immediate constant)

The execution times listed in Table 3 assume a 12MHz clock frequency. All of the arithmetic instructions execute in 1μs except the INC DPTR instruction, which takes 2μs, and the Multiply and Divide instructions, which take 4μs.

Note that any byte in the internal Data Memory space can be incremented without going through the Accumulator.

One of the INC instructions operates on the 16-bit Data Pointer. The Data Pointer is used to generate 16-bit addresses for external memory, so being able to increment it in one 16-bit operation is a useful feature.

Table 3. 8051 Arithmetic Instructions

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
ADD A,<byte>	A = A + <byte>	X	X	X	X	1
ADDC A,<byte>	A = A + <byte> + C	X	X	X	X	1
SUBB A,<byte>	A = A − <byte> − C	X	X	X	X	1
INC A	A = A + 1	Accumulator only				1
INC <byte>	<byte> = <byte> + 1	X	X	X		1
INC DPTR	DPTR = DPTR + 1	Data Pointer only				2
DEC A	A = A − 1	Accumulator only				1
DEC <byte>	<byte> = <byte> − 1	X	X	X		1
MUL AB	B:A = B x A	ACC and B only				4
DIV AB	A = Int [A/B] B = Mod [A/B]	ACC and B only				4
DA A	Decimal Adjust	Accumulator only				1

The MUL AB instruction multiplies the Accumulator by the data in the B register and puts the 16-bit product into the concatenated B and Accumulator registers.

The DIV AB instruction divides the Accumulator by the data in the B register and leaves the 8-bit quotient in the Accumulator, and the 8-bit remainder in the B register.

Oddly enough, DIV AB finds less use in arithmetic "divide" routines than in radix conversions and programmable shift operations. An example of the use of DIV AB in a radix conversion will be given later. In shift operations, dividing a number by 2n shifts its n bits to the right. Using DIV AB to perform the division completes the shift in 4μs and leaves the B register holding the bits that were shifted out. The DA A instruction is for BCD arithmetic operations. In BCD arithmetic, ADD and ADDC instructions should always be followed by a DA A operation, to ensure that the result is also in BCD. Note that DA A will not convert a binary number to BCD. The DA A operation produces a meaningful result only as the second step in the addition of two BCD bytes.

LOGICAL INSTRUCTIONS

Table 4 shows the list of 8051 logical instructions. The instructions that perform Boolean operations (AND, OR, Exclusive OR, NOT) on bytes perform the operation on a bit-by-bit basis. That is, if the Accumulator contains 00110101B and byte contains 01010011B, then:

ANL A, <byte>

will leave the Accumulator holding 00010001B.

The addressing modes that can be used to access the <byte> operand are listed in Table 4.

The ANL A, <byte> instruction may take any of the forms:

ANL A,7FH (direct addressing)
ANL A,@R1 (indirect addressing)
ANL A,R6 (register addressing)
ANL A,#53H (immediate constant)

All of the logical instructions that are Accumulator-specific execute in 1μs (using a 12MHz clock). The others take 2μs.

Note that Boolean operations can be performed on any byte in the internal Data Memory space without going through the Accumulator. The XRL <byte>, #data instruction, for example, offers a quick and easy way to invert port bits, as in XRL P1, #0FFH.

If the operation is in response to an interrupt, not using the Accumulator saves the time and effort to stack it in the service routine.

The Rotate instructions (RL A, RLC A, etc.) shift the Accumulator 1 bit to the left or right. For a left rotation, the MSB rolls into the LSB position. For a right rotation, the LSB rolls into the MSB position.

Table 4. 8051 Logical Instructions

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
ANL A,<byte>	A = A .AND. <byte>	X	X	X	X	1
ANL <byte>,A	<byte> = <byte> .AND. A	X				1
ANL <byte>,#data	<byte> = <byte> .AND. #data	X				2
ORL A,<byte>	A = A .OR. <byte>	X	X	X	X	1
ORL <byte>,A	<byte> = <byte> .OR. A	X				1
ORL <byte>,#data	<byte> = <byte> .OR. #data	X				2
XRL A,<byte>	A = A .XOR. <byte>	X	X	X	X	1
XRL <byte>,A	<byte> = <byte> .XOR. A	X				1
XRL <byte>,#data	<byte> = <byte> .XOR. #data	X				2
CRL A	A = 00H	Accumulator only				1
CPL A	A = .NOT. A	Accumulator only				1
RL A	Rotate ACC Left 1 bit	Accumulator only				1
RLC A	Rotate Left through Carry	Accumulator only				1
RR A	Rotate ACC Right 1 bit	Accumulator only				1
RRC A	Rotate Right through Carry	Accumulator only				1
SWAP A	Swap Nibbles in A	Accumulator only				1

The SWAP A instruction interchanges the high and low nibbles within the Accumulator. This is a useful operation in BCD manipulations. For example, if the Accumulator contains a binary number which is known to be less than 100, it can be quickly converted to BCD by the following code:

```
MOV   B,#10
DIV   AB
SWAP  A
ADD   A,B
```

Dividing the number by 10 leaves the tens digit in the low nibble of the Accumulator, and the ones digit in the B register. The SWAP and ADD instructions move the tens digit to the high nibble of the Accumulator, and the ones digit to the low nibble.

DATA TRANSFERS

INTERNAL RAM

Table 5 shows the menu of instructions that are available for moving data around within the internal memory spaces, and the addressing modes that can be used with each one. With a 12MHz clock, all of these instructions execute in either 1 or 2μs.

The MOV <dest>, <src> instruction allows data to be transferred between any two internal RAM or SFR locations without going through the Accumulator. Remember the Upper 128 byes of data RAM can be accessed only by indirect addressing, and SFR space only by direct addressing.

Note that in all 8051 devices, the stack resides in on-chip RAM, and grows upwards. The PUSH instruction first increments the Stack Pointer (SP), then copies the byte into the stack. PUSH and POP use only direct addressing to identify the byte being saved or restored, but the stack itself is accessed by indirect addressing using the SP register. This means the stack can go into the Upper 128, if they are implemented, but not into SFR space.

The Upper 128 are not implemented in the 8051, 8051AH, or 80C51BH, nor in their ROMless or EPROM counterparts. With these devices, if the SP points to the Upper 128, PUSHed bytes are lost, and POPed bytes are indeterminate.

The Data Transfer instructions include a 16-bit MOV that can be used to initialize the Data Pointer (DPTR) for look-up tables in Program Memory, or for 16-bit external Data Memory accesses.

The XCH A, <byte> instruction causes the Accumulator and addressed byte to exchange data. The XCHD A, @Ri instruction is similar, but only the low nibbles are involved in the exchange.

To see how XCH and XCHD can be used to facilitate data manipulations, consider first the problem of shifting an 8-digit BCD number two digits to the right. Figure 12 shows how this can be done using direct MOVs, and for comparison how it can be done using XCH instructions. To aid in understanding how the code works, the contents of the registers that are holding the BCD number and the content of the Accumulator are shown alongside each instruction to indicate their status after the instruction has been executed.

After the routine has been executed, the Accumulator contains the two digits that were shifted out on the right. Doing the routine with direct MOVs uses 14 code bytes and 9μs of execution time (assuming a 12MHz clock). The same operation with XCHs uses less code and executes almost twice as fast.

To right-shift by an odd number of digits, a one-digit shift must be executed.

Figure 13 shows a sample of code that will right-shift a BCD number one digit, using the XCHD instruction. Again, the contents of the registers holding the number and of the Accumulator are shown alongside each instruction.

Table 5. Data Transfer Instructions that Access Internal Data Memory Space

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
MOV A,<src>	A = <src>	X	X	X	X	1
MOV <dest>,A	<dest> = A	X	X	X		1
MOV <dest>, <src>	<dest> = <src>	X	X	X	X	2
MOV DPTR,#data16	DPTR = 16-bit immediate constant.				X	2
PUSH <src>	INC SP : MOV "@SP",<src>	X				2
POP <dest>	MOV <dest>, "@SP" : DEC SP	X				2
XCH A,<byte>	ACC and <byte> exchange data	X	X	X		1
XCHD A,@Ri	ACC and @Ri exchange low nibbles		X			1

		2A	2B	2C	2D	2E	ACC
MOV	A,2EH	00	12	34	56	78	78
MOV	2EH,2DH	00	12	34	56	56	78
MOV	2DH,2CH	00	12	34	34	56	78
MOV	2CH,2BH	00	12	12	34	56	78
MOV	2BH,#0	00	00	12	34	56	78

(a) Using direct MOVs: 14 bytes, 9 μs

		2A	2B	2C	2D	2E	ACC
CLR	A	00	12	34	56	78	00
XCH	A,2BH	00	00	34	56	78	12
XCH	A,2CH	00	00	12	56	78	34
XCH	A,2DH	00	00	12	34	78	56
XCH	A,2EH	00	00	12	34	56	78

(b) Using XCHs: 9 bytes, 5 μs

Figure 12. Shifting a BCD Number Two Digits to the Right

		2A	2B	2C	2D	2E	ACC
	MOV R1,#2EH	00	12	34	56	78	XX
	MOV R0,#2DH	00	12	34	56	78	XX
	loop for R1 = 2EH:						
LOOP:	MOV A,@R1	00	12	34	56	78	78
	XCHD A,@R0	00	12	34	58	78	76
	SWAP A	00	12	34	58	78	67
	MOV @R1,A	00	12	34	58	67	67
	DEC R1	00	12	34	58	67	67
	DEC R0	00	12	34	58	67	67
	CJNE R1,#2AH,LOOP						
	loop for R1 = 2DH:	00	12	38	45	67	45
	loop for R1 = 2CH:	00	18	23	45	67	23
	loop for R1 = 2BH:	08	01	23	45	67	01
	CLR A	08	01	23	45	67	00
	XCH A,2AH	00	01	23	45	67	08

Figure 13. Shifting a BCD Number One Digit to the Right

First, pointers R1 and R0 are set up to point to the two bytes containing the last four BCD digits. Then a loop is executed which leaves the last byte, location 2EH, holding the last two digits of the shifted number. The pointers are decremented, and the loop is repeated for location 2DH. The CJNE instruction (Compare and Jump if Not Equal) is a loop control that will be described later. The loop executed from LOOP to CJNE for R1 = 2EH, 2DH, 2CH and 2BH. At that point the digit that was originally shifted out on the right has propagated to location 2AH. Since that location should be left with 0s, the lost digit is moved to the Accumulator.

EXTERNAL RAM

Table 6 shows a list of the Data Transfer instructions that access external Data Memory. Only indirect addressing can be used. The choice is whether to use a one-byte address, @Ri, where Ri can be either R0 or R1 of the selected register bank, or a two-byte address, @DPTR. The disadvantage to using 16-bit addresses if only a few K bytes of external RAM are involved is that 16-bit addresses use all 8 bits of Port 2 as address bus. On the other hand, 8-bit addresses allow one to address a few bytes of RAM, as shown in Figure 6, without having to sacrifice all of Port 2. All of these instructions execute in 2μs, with a 12MHz clock.

Table 6. 8051 Data Transfer Instructions that Access External Data Memory Space

Address Width	Mnemonic	Operation	Execution Time (μs)
8 bits	MOVX A,@Ri	Read external RAM @Ri	2
8 bits	MOVX @Ri,A	Write external RAM @Ri	2
16 bits	MOVX A,@DPTR	Read external RAM @DPTR	2
16 bits	MOVX @DPTR,A	Write external RAM @DPTR	2

Note that in all external Data RAM accesses, the Accumulator is always either the destination or source of the data.

The read and write strobes to external RAM are activated only during the execution of a MOVX instruction. Normally these signals are inactive, and in fact if they're not going to be used at all, their pins are available as extra I/O lines. More about that later.

LOOKUP TABLES

Table 7 shows the two instructions that are available for reading lookup tables in Program Memory. Since these instructions access only Program Memory, the lookup tables can only be read, not updated.

If the table access is to external Program Memory, then the read strobe is PSEN.

Table 7. 8051 Lookup Table Read Instructions

Mnemonic	Operation	Execution Time (μs)
MOVC A,@A+DPTR	Read Pgm Memory at (A+DPTR)	2
MOVC A,@A+PC	Read Pgm Memory at (A+PC)	2

The mnemonic is MOVC for "move constant". The first MOVC instruction in Table 7 can accommodate a table of up to 256 entries numbered 0 through 255. The number of the desired entry is loaded into the Accumulator, and the Data Pointer is set up to point to beginning of the table. Then:

```
MOVC    A,@A+DPTR
```

copies the desired table entry into the Accumulator.

The other MOVC instruction works the same way, except the Program Counter (PC) is used as the table base, and the table is accessed through a subroutine. First the number of the desired entry is loaded into the Accumulator, and the subroutine is called:

```
MOV     A,ENTRY NUMBER
CALL    TABLE
```

The subroutine "TABLE" would look like this:

```
TABLE: MOVC     A,@A+PC
       RET
```

The table itself immediately follows the RET (return) instruction in Program Memory. This type of table can have up to 255 entries, numbered 1 through 255. Number 0 cannot be used, because at the time the MOVC instruction is executed, the PC contains the address of the RET instruction. An entry numbered 0 would be the RET opcode itself.

BOOLEAN INSTRUCTIONS

8051 devices contain a complete Boolean (single-bit) processor. The internal RAM contains 128 addressable bits, and the SFR space can support up to 128 other addressable bits. All of the port lines are bit-addressable, and each one can be treated as a separate single-bit port. The instructions that access these bits are not just conditional branches, but a complete menu of move, set, clear, complement, OR, and AND instructions. These kinds of bit operations are not easily obtained in other architectures with any amount of byte-oriented software.

The instruction set for the Boolean processor is shown in Table 8. All bit accesses are by direct addressing.

Bit addresses 00H through 7FH are in the Lower 128, and bit addresses 80H through FFH are in SFR space.

Note how easily an internal flag can be moved to a port pin:

```
MOV     C,FLAG
MOV     P1.0,C
```

In this example, FLAG is the name of any addressable bit in the Lower 128 or SFR space. An I/O line (the LSB of Port 1, in this case) is set or cleared depending on whether the flag bit is 1 or 0.

Table 8. 8051 Boolean Instructions

Mnemonic		Operation	Execution Time (μs)
ANL	C,bit	C = C .AND. bit	2
ANL	C,/bit	C = C .AND. .NOT. bit	2
ORL	C,bit	C = C .OR. bit	2
ORL	C,/bit	C = C .OR. .NOT. bit	2
MOV	C,bit	C = bit	1
MOV	bit,C	bit = C	2
CLR	C	C = 0	1
CLR	bit	bit = 0	1
SETB	C	C = 1	1
SETB	bit	bit = 1	1
CPL	C	C = .NOT. C	1
CPL	bit	bit = .NOT. bit	1
JC	rel	Jump if C = 1	2
JNC	rel	Jump if C = 0	2
JB	bit,rel	Jump if bit = 1	2
JNB	bit,rel	Jump if bit = 0	2
JBC	bit,rel	Jump if bit = 1; CLR bit	2

The Carry bit in the PSW is used as the single-bit Accumulator of the Boolean processor. Bit instructions that refer to the Carry bit as C assemble as Carry-specific instructions (CLR C, etc). The Carry bit also has a direct address, since it resides in the PSW register, which is bit-addressable.

Note that the Boolean instruction set includes ANL and ORL operations, but not the XRL (Exclusive OR) operation. An XRL operation is simple to implement in software. Suppose, for example, it is required to form the Exclusive OR of two bits:

```
C = bit1 .XRL. bit2
```

The software to do that could be as follows:

```
        MOV     C,bit1
        JNB     bit2,OVER
        CPL     C
OVER:   (continue)
```

First, bit1 is moved to the Carry. If bit2 = 0, then C now contains the correct result. That is, bit1 .XRL. bit2 = bit1 if bit2 = 0. On the other hand, if bit2 = 1, C now contains the complement of the correct result. It need only be inverted (CPL C) to complete the operation.

This code uses the JNB instruction, one of a series of bit-test instructions which execute a jump if the addressed bit is set (JC, JB, JBC) or if the addressed bit is not set (JNC, JNB). In the above case, bit2 is being tested, and if bit2 = 0 the CPL C instruction is jumped over.

JBC executes the jump if the addressed bit is set, and also clears the bit. Thus a flag can be tested and cleared in one operation.

All the PSW bits are directly addressable, so the Parity bit, or the general purpose flags, for example, are also available to the bit-test instructions.

RELATIVE OFFSET

The destination address for these jumps is specified to the assembler by a label or by an actual address in Program memory. However, the destination address assembles to a relative offset byte. This is a signed (two's complement) offset byte which is added to the PC in two's complement arithmetic if the jump is executed.

The range of the jump is therefore −128 to + 127 Program Memory bytes relative to the first byte following the instruction.

JUMP INSTRUCTIONS

Table 9 shows the list of unconditional jumps with execution time for a 12MHz clock.

Table 9. Unconditional Jumps in 8051 Devices

Mnemonic	Operation	Execution Time (μs)
JMP addr	Jump to addr	2
JMP @A + DPTR	Jump to A + DPTR	2
CALL addr	Call subroutine at addr	2
RET	Return from subroutine	2
RETI	Return from interrupt	2
NOP	No operation	1

The Table lists a single "JMP addr" instruction, but in fact there are three SJMP, LJMP and AJMP, which differ in the format of the destination address. JMP is a generic mnemonic which can be used if the programmer does not care which way the jump is encoded.

The SJMP instruction encodes the destination address as a relative offset, as described above. The instruction is 2 bytes long, consisting of the opcode and the relative offset byte. The jump distance is limited to a range of −128 to +127 bytes relative to the instruction following the SJMP.

The LJMP instruction encodes the destination address as a 16-bit constant. The instruction is 3 bytes long, consisting of the opcode and two address bytes. The destination address can be anywhere in the 64K Program Memory space.

The AJMP instruction encodes the destination address as an 11-bit constant. The instruction is 2 bytes long, consisting of the opcode, which itself contains 3 of the 11 address bits, followed by another byte containing the low 8 bits of the destination address. When the instruction is executed, these 11 bits are simply substituted for the low 11 bits in the PC. The high 5 bits stay the same. Hence the destination has to be within the same 2K block as the instruction following the AJMP.

In all cases the programmer specifies the destination address to the assembler in the same way: as a label or as a 16-bit constant. The assembler will put the destination address into the correct format for the given instruction. If the format required by the instruction will not support the distance to the specified destination address, a "Destination out of range" message is written into the List file.

The JMP @A+DPTR instruction supports case jumps. The destination address is computed at execution time as the sum of the 16-bit DPTR register and the Accumulator. Typically, DPTR is set up with the address of a jump table, and the Accumulator is given an index to the table. In a 5-way branch, for example, an integer 0 through 4 is loaded into the Accumulator. The code to be executed might be as follows:

```
MOV     DPTR,#JUMP TABLE
MOV     A,INDEX_NUMBER
RL      A
JMP     @A+DPTR
```

The RL A instruction converts the index number (0 through 4) to an even number on the range 0 through 8, because each entry in the jump table is 2 bytes long:

```
JUMP TABLE:
        AJMP    CASE 0
        AJMP    CASE 1
        AJMP    CASE 2
        AJMP    CASE 3
        AJMP    CASE 4
```

Table 9 shows a single "CALL addr" instruction, but there are two of them, LCALL and ACALL, which differ in the format in which the subroutine address is given to the CPU. CALL is a generic mnemonic which can be used if the programmer does not care which way the address is encoded.

The LCALL instruction uses the 16-bit address format, and the subroutine can be anywhere in the 64K Program Memory space. The ACALL instruction uses the 11-bit format, and the subroutine must be in the same 2K block as the instruction following the ACALL.

In any case the programmer specifies the subroutine address to the assembler in the same way: as a label or as a 16-bit constant. The assembler will put the address into the correct format for the given instructions.

Subroutines should end with a RET instruction, which returns execution to the instruction following the CALL.

RETI is used to return from an interrupt service routine. The only difference between RET and RETI is that RETI tells the interrupt control system that the interrupt in progress is done. If there is no interrupt in progress at the time RETI is executed, then the RETI is functionally identical to RET.

Table 10 shows the list of conditional jumps available to the 8051 user. All of these jumps specify the destination address by the relative offset method, and so are limited to a jump distance of −128 to + 127 bytes from the instruction following the conditional jump instruction. Important to note, however, the user specifies to the assembler the actual destination address the same way as the other jumps: as a label or a 16-bit constant.

There is no Zero bit in the PSW. The JZ and JNZ instructions test the Accumulator data for that condition.

The DJNZ instruction (Decrement and Jump if Not Zero) is for loop control. To execute a loop N times, load a counter byte with N and terminate the loop with a DJNZ to the beginning of the loop, as shown below for N = 10:

```
        MOV     COUNTER,#10
LOOP:   (begin loop)
          •
          •
          •
        (end loop)
        DJNZ    COUNTER,LOOP
        (continue)
```

The CJNE instruction (Compare and Jump if Not Equal) can also be used for loop control as in Figure 13. Two bytes are specified in the operand field of the instruction. The jump is executed only if the two bytes are not equal. In the example of Figure 13, the two bytes were data in R1 and the constant 2AH. The initial data in R1 was 2EH. Every time the loop was executed, R1 was decremented, and the looping was to continue until the R1 data reached 2AH.

Another application of this instruction is in "greater than, less than" comparisons. The two bytes in the operand field are taken as unsigned integers. If the first is less than the second, then the Carry bit is set (1). If the first is greater than or equal to the second, then the Carry bit is cleared.

CPU TIMING

All 8051 microcontrollers have an on-chip oscillator which can be used if desired as the clock source for the CPU. To use the on-chip oscillator, connect a crystal or ceramic resonator between the XTAL1 and XTAL2 pins of the microcontroller, and capacitors to ground as shown in Figure 14.

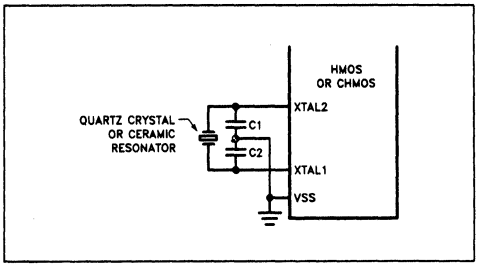


Figure 14. Using the On-Chip Oscillator

Examples of how to drive the clock with an external oscillator are shown in Figure 15. Note that in the HMOS devices (8051, etc.) the signal at the XTAL2 pin actually drives the internal clock generator. In the CHMOS devices (80C51BH, etc.), the signal at the XTAL1 pin drives the internal clock generator. The internal clock generator defines the sequence of states that make up the 8051 machine cycle.

MACHINE CYCLES

A machine cycle consists of a sequence of 6 states, numbered S1 through S6. Each state time lasts for two oscillator periods. Thus a machine cycle takes 12 oscillator periods or 1μs if the oscillator frequency is 12MHz.

Each state is divided into a Phase 1 half and a Phase 2 half. Figure 16 shows that fetch/execute sequences in states and phases for various kinds of instructions. Normally two program fetches are generated during each machine cycle, even if the instruction being executed doesn't require it. If the instruction being executed doesn't need more code bytes, the CPU simply ignores the extra fetch, and the Program Counter is not incremented.

Table 10. Conditional Jumps in 8051 Devices

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
JZ rel	Jump if A = 0	Accumulator only				2
JNZ rel	Jump if A ≠ 0	Accumulator only				2
DJNZ <byte>,rel	Decrement and jump if not zero	X		X		2
CJNE A,<byte>,rel	Jump if A ≠ <byte>	X			X	2
CJNE <byte>,#data,rel	Jump if <byte> ≠ #data		X	X		2

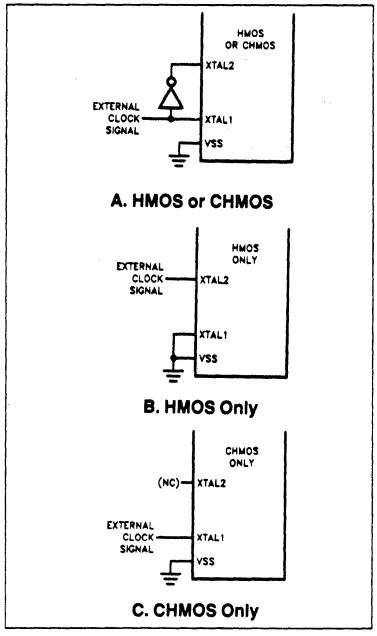


Figure 15. Using an External Clock

Execution of a one-cycle instruction (Figure 16A and B) begins during State 1 of the machine cycle, when the opcode is latched into the Instruction Register. A second fetch occurs during S4 of the same machine cycle. Execution is complete at the end of State 6 of this machine cycle.

The MOVX instructions take two machine cycles to execute. No program fetch is generated during the second cycle of a MOVX instruction. This is the only time program fetches are skipped. The fetch/execute sequence for MOVX instructions is shown in Figure 16D.

The fetch/execute sequences are the same whether the Program Memory is internal or external to the chip. Execution times do not depend on whether the Program Memory is internal or external.

Figure 17 shows the signals and timing involved in program fetches when the Program Memory is external. If Program Memory is external, then the Program Memory read strobe $\overline{\text{PSEN}}$ is normally activated twice per machine cycle, as shown in Figure 17(A).

If an access to external Data Memory occurs, as shown in Figure 17(B), two $\overline{\text{PSEN}}$s are skipped, because the address and data bus are being used for the Data Memory access.

Note that a Data Memory bus cycle takes twice as much time as a Program Memory bus cycle. Figure 17 shows the relative timing of the addresses being emitted at Ports 0 and 2, and of ALE and $\overline{\text{PSEN}}$. ALE is used to latch the low address byte from P0 into the address latch.

When the CPU is executing from internal Program Memory, $\overline{\text{PSEN}}$ is not activated, and program addresses are not emitted. However, ALE continues to be activated twice per machine cycle and so it is available as a clock output signal. Note, however, that one ALE is skipped during the execution of the MOVX instruction.

INTERRUPT STRUCTURE

The 8051, 8051AH, 80C51BH, 83C451, and their ROMless and EPROM versions, provide 5 interrupt sources: 2 external interrupts, 2 timer interrupts, and the serial port interrupt.

What follows is an overview of the interrupt structure for these devices. More detailed information for specific members of the 8051 family is provided in later chapters of this handbook.

INTERRUPT ENABLES

Each of the interrupt sources can be individually enabled or disabled by setting or clearing a bit in the SFR named IE (Interrupt Enable). This register also contains a global disable bit, which can be cleared to disable all interrupts at once. Figure 18 shows the IE register for the 8051 devices.

INTERRUPT PRIORITIES

Each interrupt source can also be individually programmed to one of two priority levels by setting or clearing a bit in the SFR named IP Interrupt Priority). Figure 19 shows the IP register in the 8051.

A low-priority interrupt can be interrupted by a high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

If two interrupt requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence.

Figure 20 shows, for the 8051, how the IE and IP registers and the polling sequence work to determine which if any interrupt will be serviced.

In operation, all the interrupt flags are latched into the interrupt control system during State 5 of every machine cycle. The samples are polled during the following machine cycle. If the flag for an enabled interrupt is found to be set (1), the interrupt system generates an LCALL to the appropriate location in Program Memory, unless some other condition blocks the interrupt. Several con-

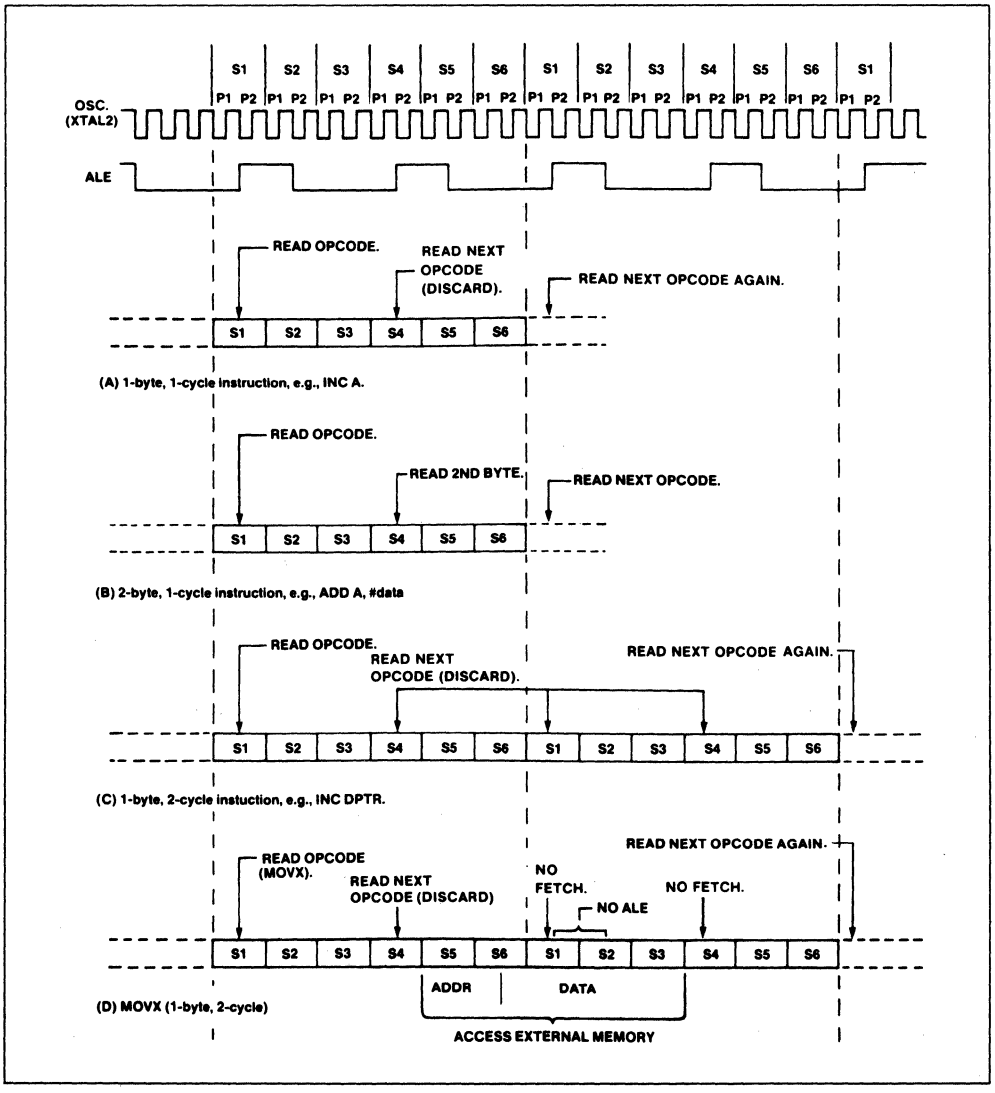


Figure 16. State Sequence in 8051 Family devices

ditions can block an interrupt, among them that an interrupt of equal or higher priority level is already in progress.

The hardware-generated LCALL causes the contents of the Program Counter to be pushed into the stack, and reloads the PC with the beginning address of the service routine. As previously noted (Figure 4), the service routine for each interrupt begins at a fixed location.

Only the Program Counter is automatically pushed onto the stack, not the PSW or any other register. Having only the PC be automatically saved allows the programmer to decide how much time to spend saving which other registers. This enhances the interrupt response time, albeit at the expense of increasing the programmer's burden of responsibility. As a result, many interrupt functions that are typical in control applications toggling a port pin for example, or reloading a timer, or unloading a serial buffer can often be completed in less time than it takes other architectures to complete.

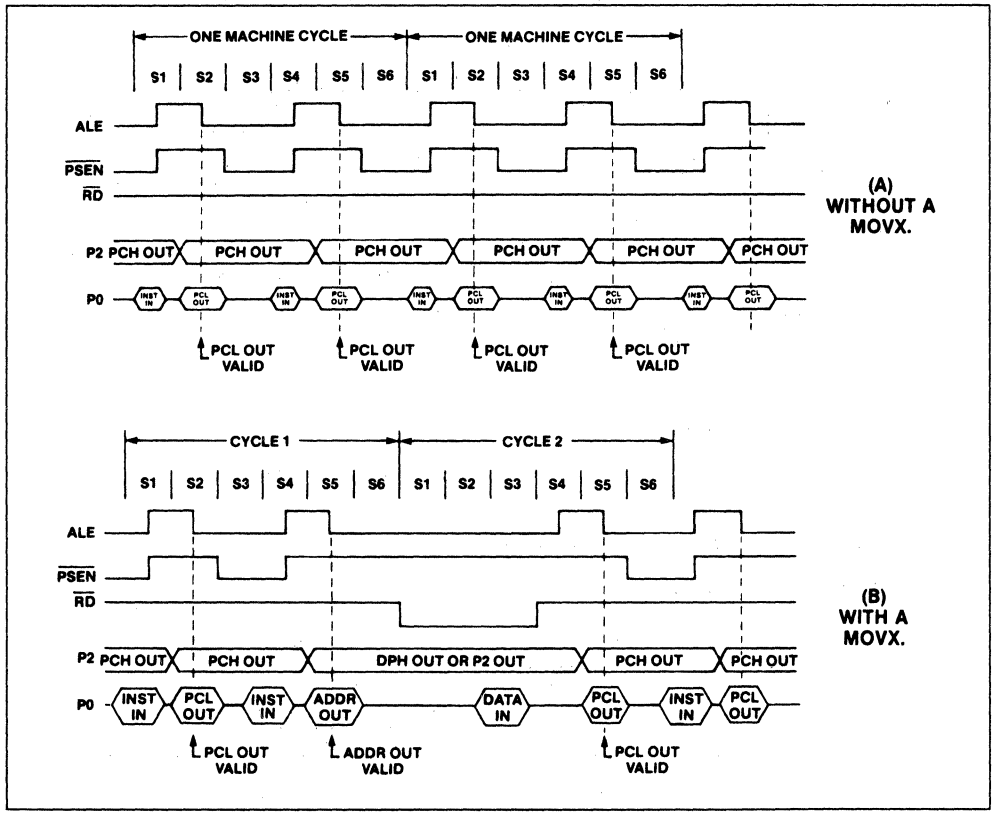


Figure 17. Bus Cycles in 8051 Family Devices Executing from External Program Memory

SIMULATING A THIRD PRIORITY LEVEL IN SOFTWARE

Some applications require more than two priority levels that are provided by on-chip hardware in 8051 devices. In these cases, relatively simple software can be written to produce the same effect as a third priority level. First interrupts that are to have higher priority than 1 are assigned to priority 1 in the Interrupt Priority (IP) register. The service routines for priority 1 interrupts that are supposed to be interruptable by "priority 2" interrupts are written to include the following code:

```
        PUSH    IE
        MOV     IE,#MASK
        CALL    LABEL
           **********
    (execute service routine)
           **********
        POP     IE
        RET
LABEL:  RETI
```

As soon as any priority interrupt is acknowledged, the Interrupt Enable (IE) register is redefined so as to disable all but "priority 2" interrupts. Then a CALL to LABEL executes the RETI instruction, which clears the priority 1 interrupt-in-progress flip-flop. At this point any priority 1 interrupt that is enabled can be serviced, but only "priority 2" interrupts are enabled.

POPing IE restores the original enable byte. Then a normal RET (rather than another RETI) is used to terminate the service routine. The additional software adds 10μs (at 12MHz) to priority 1 interrupts.

(MSB)							(LSB)
$\overline{EA}$	X	X	ES	ET1	EX1	ET0	EX0

Symbol	Position	Function
$\overline{EA}$	IE.7	disables all interrupts. If $\overline{EA}$ = 0, no interrupt will be acknowledged. If $\overline{EA}$ = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
	IE.6	reserved.
	IE.5	reserved.
ES	IE.4	enables or disables the Serial Port interrupt. If ES = 0, the Serial Port interrupt is disabled.
ET1	IE.3	enables or disables the Timer 1 Overflow interrupt. If ET1 = 0, the Timer 1 interrupt is disabled.
EX1	IE.2	enables or disables External Interrupt 1. If EX1 = 0, External Interrupt 1 is disabled.
ET0	IE.1	enables or disables the Timer 0 Overflow interrupt. If ET0 = 0, the Timer 0 interrupt is disabled.
EX0	IE.0	enables or disables External Interrupt 0. If EX0 = 0, External Interrupt 0 is disabled.

Figure 18. Interrupt Enable (IE) Register

(MSB)							(LSB)
X	X	X	PS	PT1	PX1	PT0	PX0

Symbol	Position	Function
	IP.7	reserved
	IP.6	reserved
	IP.5	reserved
PS	IP.4	defines the Serial Port interrupt priority level. PS = 1 programs it to the higher priority level.
PT1	IP.3	defines the Timer 1 interrupt priority level. PT1 = 1 programs it to the higher priority level.
PX1	IP.2	defines the External interrupt 1 priority level. PX1 = 1 programs it to the higher priority level.
PT0	IP.1	defines the Timer 0 interrupt priority level. PT0 = 1 programs it to the higher priority level.
PX0	IP.0	defines the External interrupt 0 priority level. PX0 = 1 programs it to the higher priority level.

Figure 19. Interrupt Priority (IP) Register

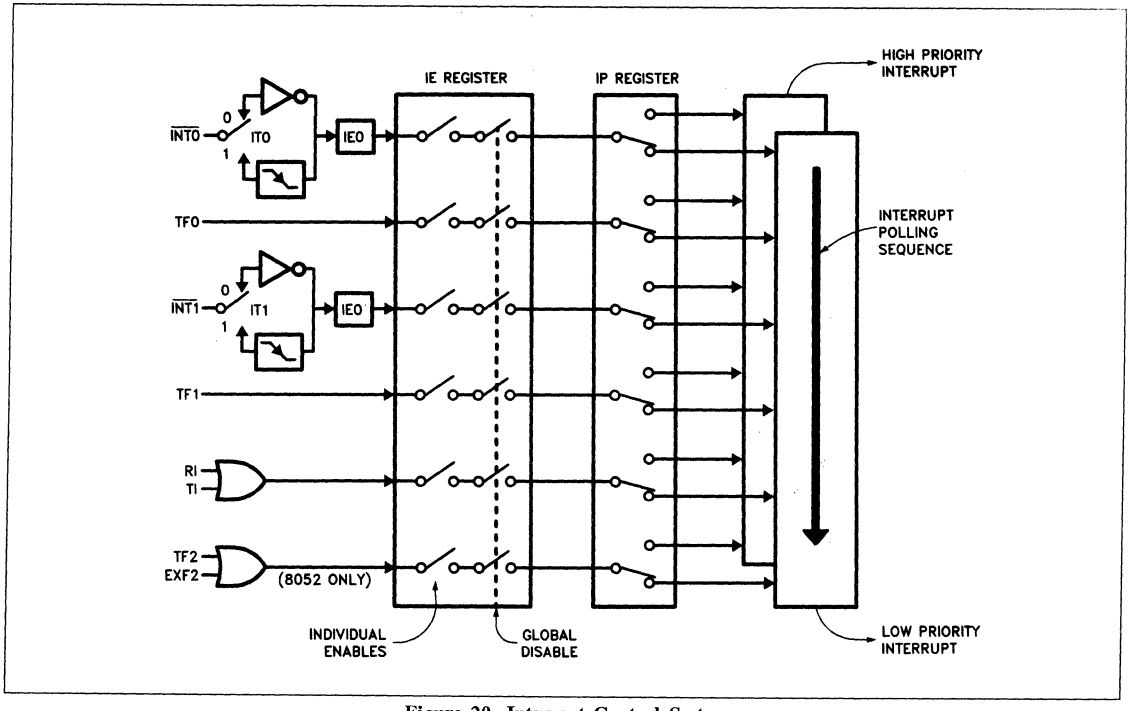
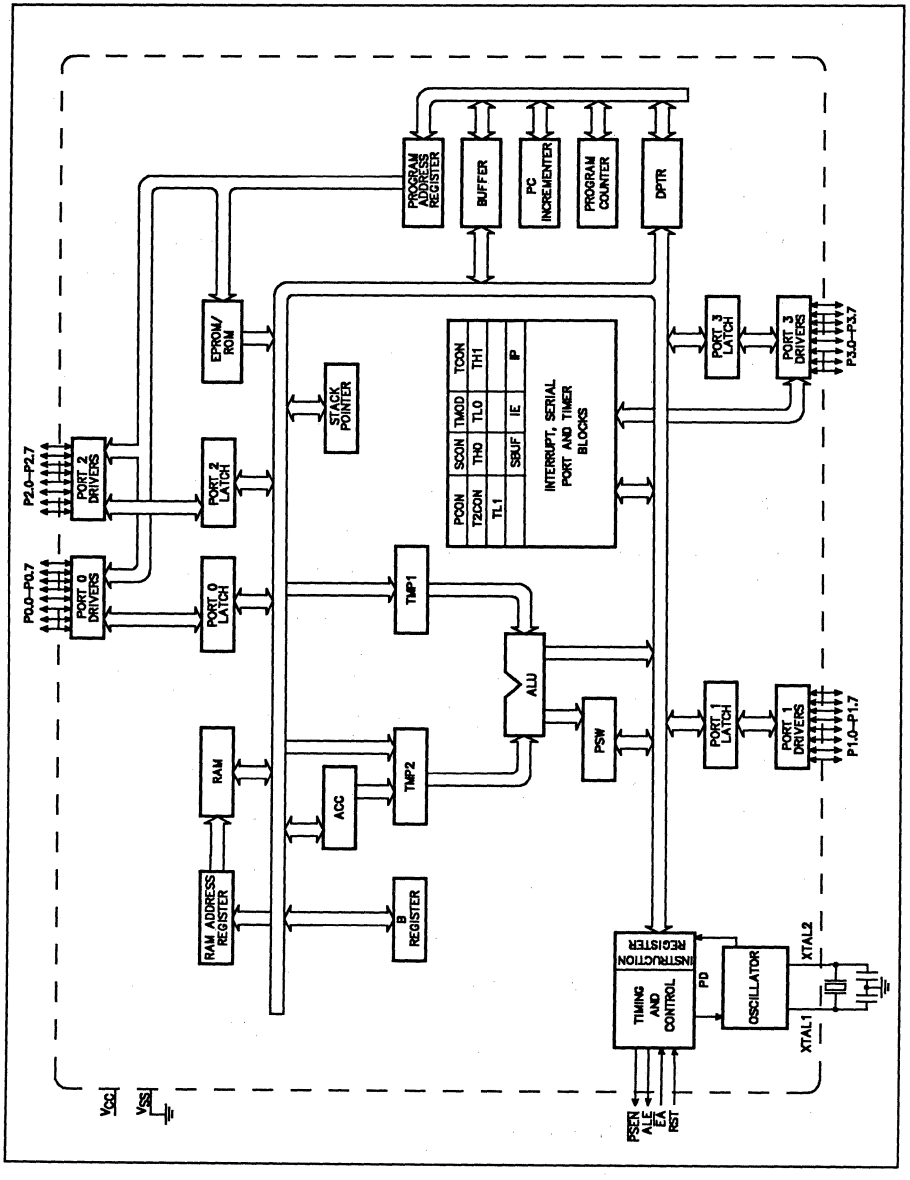


Figure 20. Interrupt Control System

HARDWARE DESCRIPTION

This chapter provides a detailed description of the 8051 microcontrollers (see Figure 21). Included in this description are:

- The port drivers and how they function both as ports and, for Ports 0 and 2, in bus operations
- The Timers/Counters
- The Serial Interface
- The Interrupt System
- Reset
- The Reduced Power Modes in CHMOS devices
- The EPROM version of the 80C51BH



SPECIAL FUNCTION REGISTERS

A Map of the on-chip memory area called Special Function Register (SFR) space is shown in Figure 22.

Note that in the SFRs not all of the addresses are occupied. Unoccupied addresses are not implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have no effect.

User software should not write 1s to these unimplemented locations, since they may be used in other 8051 Family products to invoke new features. The function of the SFRs are described in the text that follows.

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.

PROGRAM STATUS WORD

The PSW register contains program status information as detailed in Figure 23.

STACK POINTER

The Stack Pointer register is 8 bits wide. It is incremented before data is stored during PUSH and CALL executions. While the stack may reside anywhere in on-chip RAM, the Stack Pointer is initialized to 07H after a reset. This causes the stack to begin at locations 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. It may be manipulated as a 16-bit register or as two independent 8-bit registers.

PORTS 0 TO 3

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

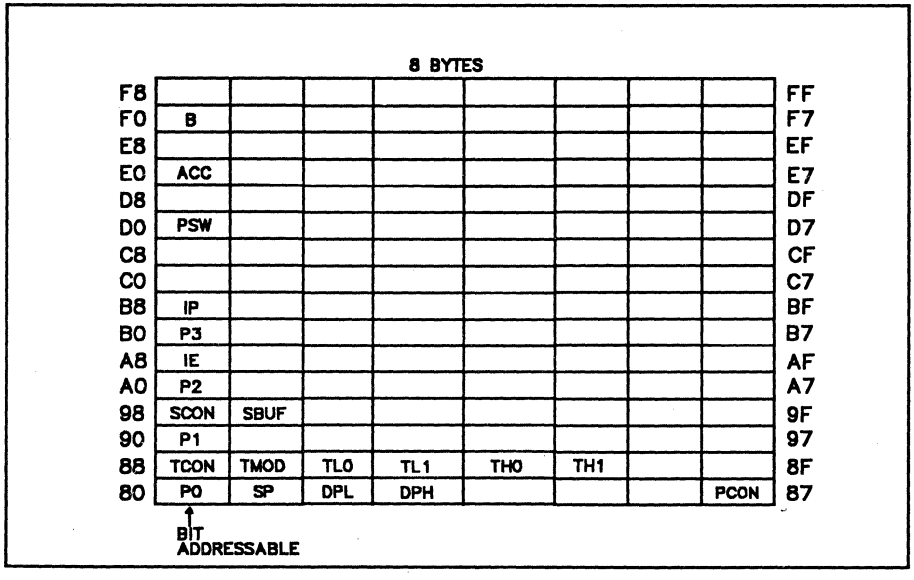
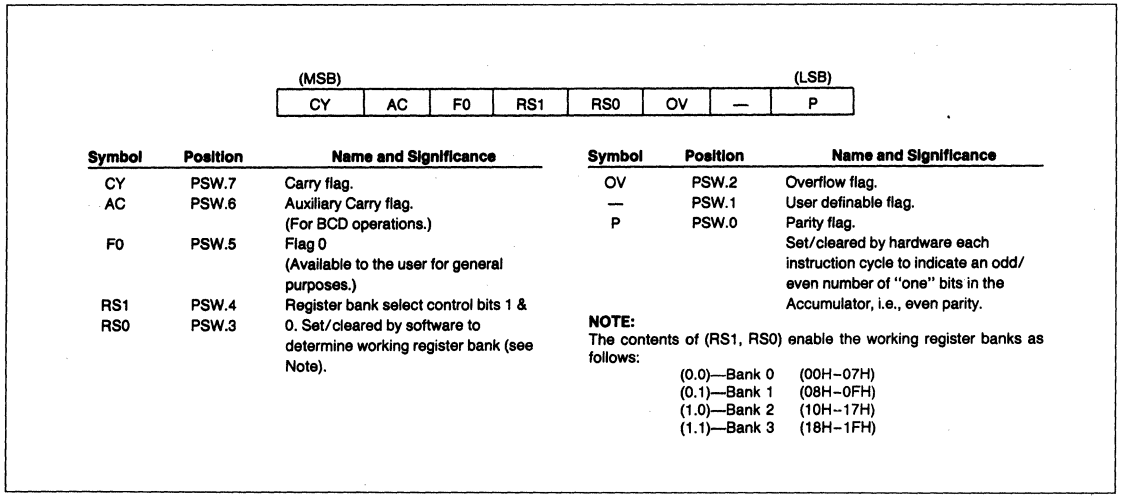


Figure 22. 8051 SFR Memory Map



(MSB)							(LSB)
CY	AC	F0	RS1	RS0	OV	—	P

Symbol	Position	Name and Significance
CY	PSW.7	Carry flag.
AC	PSW.6	Auxiliary Carry flag. (For BCD operations.)
F0	PSW.5	Flag 0 (Available to the user for general purposes.)
RS1	PSW.4	Register bank select control bits 1 &
RS0	PSW.3	0. Set/cleared by software to determine working register bank (see Note).

Symbol	Position	Name and Significance
OV	PSW.2	Overflow flag.
—	PSW.1	User definable flag.
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/ even number of "one" bits in the Accumulator, i.e., even parity.

NOTE:
The contents of (RS1, RS0) enable the working register banks as follows:

(0.0)—Bank 0 (00H–07H)
(0.1)—Bank 1 (08H–0FH)
(1.0)—Bank 2 (10H–17H)
(1.1)—Bank 3 (18H–1FH)

Figure 23. Program Status Word (PSW) Register

SERIAL DATA BUFFER

The Serial Buffer 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 is held for serial transmission. (Moving a byte to SBUF is what initiates the transmission.) When data is moved from SBUF, it comes from the receive buffer.

TIMER REGISTERS BASIC TO 8051

Register pairs (TH0, TLO), and (TH1, TL1) are the 16–bit Counting registers for Timer/Counters 0, and 1 respectively.

CONTROL REGISTERS FOR THE 8051

Special Function Registers IP, IE, TMOD, TCON, SCON, and PCON contain control and status bits for the interrupt system, the Timer/Counters, and the serial port. They are described in later sections.

PORT STRUCTURES AND OPERATION

All four ports in the 8051 are bidirectional. Each consists of a latch (Special Function Registers P0 through P3), an output driver, and an input buffer.

The output drivers of Ports 0 and 2, and the input buffers of Port 0, are used in accesses to external memory. In this application, Port 0 outputs the low byte of the external memory address, time-multiplexed with the byte being written or read.

Port 2 outputs the high byte of the external memory address when the address is 16 bits wide. Otherwise, the Port 2 pins continue to emit the P2 SFR content.

All the Port 3 pins are multifunctional. They are not only port pins, but also serve the functions of various special features as listed below:

Port Pin	Alternate Function
P3.0	RxD (serial input port)
P3.1	TxD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt)
P3.3	$\overline{INT1}$ (external interrupt)
P3.4	T0 (Timer/Counter 0 external input)
P3.5	T1 (Timer/Counter 1 external input)
P3.6	$\overline{WR}$ (external Data Memory write strobe)
P3.7	$\overline{RD}$ (external Data Memory read strobe)

The alternate functions can only be activated if the corresponding bit latch in the port SFR contains a 1. Otherwise the port pin remains at 0.

I/O CONFIGURATIONS

Figure 24 shows a functional diagram of a typical bit latch and I/O buffer in each of the four ports. The bit latch (one bit in the port's SFR) is represented as a Type D flip-flop, which will clock in a value from the internal bus in response to a "write to latch" signal from the CPU. The Q output of the flip-flop is placed on the internal bus in response to a "read latch" signal from the CPU. The level of the port pin itself is placed on the internal bus in response to a "read pin" signal from the CPU. Some instructions that read a port activate the "read latch" signal, and others activate the "read pin" signal.

As shown in Figure 24, the output drivers of Port 0 and 2 are switchable to an internal ADDR and ADDR/ DATA bus by an internal CONTROL signal for use in exter-

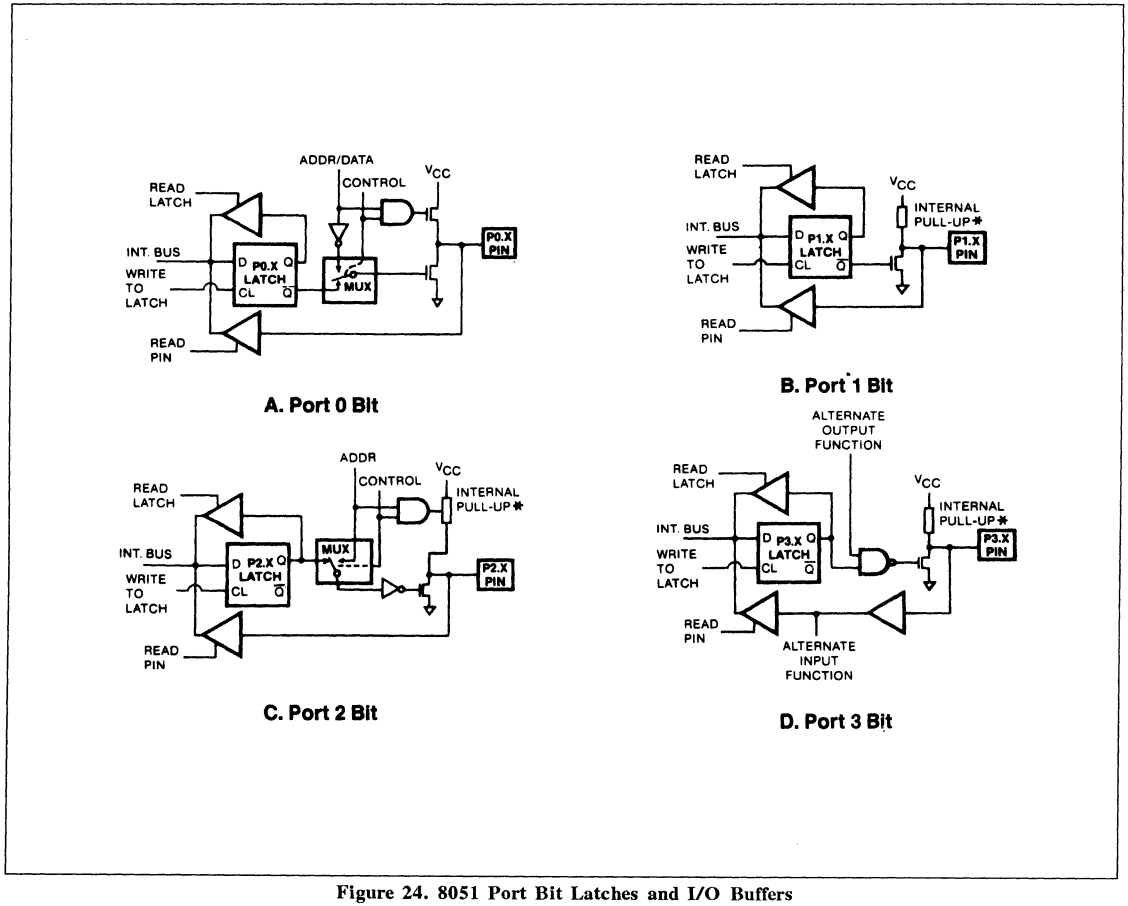


Figure 24. 8051 Port Bit Latches and I/O Buffers
*See Figure 25 for details of the internal pullup.

nal memory accesses. During external memory accesses, the P2 SFR remains unchanged, but the P0 SFR gets 1s written to it.

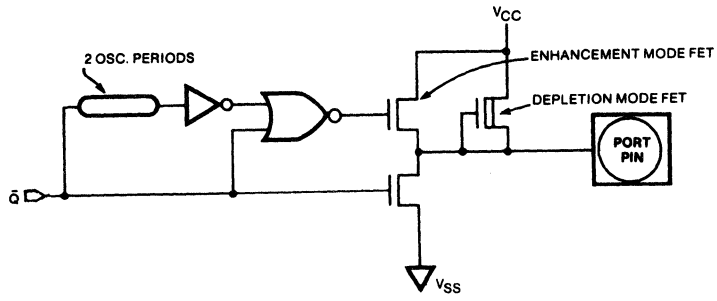
Also shown in Figure 24, is that if a P3 bit latch contains a 1, then the output level is controlled by the signal labeled "alternate output function". The actual P3.X pin level is always available to the pin's alternate input function, if any.

Ports 1, 2, and 3 have internal pullups. Port 0 has open drain outputs. Each I/O line can be independently used as an input or an output. (Port 0 and 2 may not be used as general purpose I/O when being used as the ADDR/DATA BUS). To be used as an input, the port bit latch must contain a 1, which turns off the output driver FET. Then, for Ports 1, 2, and 3, the pin is pulled high by the internal pullup, but can be pulled low by an external source.

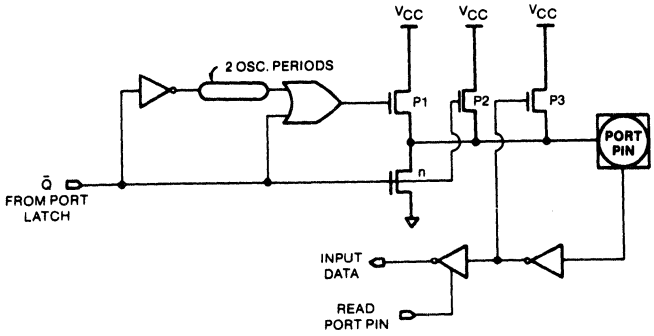
Port 0 differs in not having internal pullups. The pullup FET in the P0 output driver (see Figure 24) is used only when the port is emitting 1s during external memory accesses. Otherwise the pullup FET is off. Consequently P0 lines that are being used as output port lines are open drain. Writing a 1 to the bit latch leaves both output FETs off, so the pin floats. In that condition it can be used a high-impedance input.

Because Ports 1, 2, and 3 have fixed internal pullups they are sometimes called "quasi-bidirectional" ports. When configured as inputs they pull high and will source current (I_{IL}, in the data sheets) when externally pulled low. Port 0, on the other hand, is considered "true" bidirectional, because when configured as an input it floats.

All the port latches in the 8051 have 1s written to them by the reset function. If a 0 is subsequently written to a port latch, it can be reconfigured as an input by writing a 1 to it.



A. HMOS Configuration. The enhancement mode transistor is turned on for 2 osc. periods after Q makes a 1-to-0 transition.



B. CHMOS Configuration. pFET 1 is turned on for 2 osc. periods after Q makes a 1-to-0 transition. During this time, pFET 1 also turns on pFET 3 through the inverter to form a latch which holds the 1. pFET 2 is also on.

Figure 25. Ports 1 and 3 HMOS and CHMOS Internal Pullup Configurations Port 2 is similar except that it holds the strong pullup on while emitting 1s that are address bits. (See Accessing External Memory).

WRITING TO A PORT

In the execution of an instruction that changes the value in a port latch, the new value arrives at the latch during S6P2 of the final cycle of the instruction. However, port latches are in fact sampled by their output buffers only during Phase 1 of any clock period. (During Phase 2 the output buffer holds the value it saw during the previous Phase 1). Consequently, the new value in the port latch won't actually appear at the output pin until the next Phase 1, which will be at S1P1 of the next machine cycle.

If the change requires a 0-to-1 transition in Port 1, 2, or 3, an additional pullup is turned on during S1P1 and S1P2 of the cycle in which the transition occurs. This is done to increase the transition speed. The extra pullup can source about 100 times the current that the normal pullup can. It should be noted that the internal pullups are field-effect transistors, not linear resistors. The pullup arrangements are shown in Figure 25.

In HMOS versions of the 8051, the fixed part of the pullup is a depletion mode transistor with the gate wired to the source. This transistor will allow the pin to source about 0.25mA when shorted to ground. In parallel with the fixed pullup is an enhancement mode transistor, which is activated during S1 whenever the port bit does a 0-to-1 transition. During this interval, if the port pin is shorted to ground, this extra transistor will allow the pin to source an additional 30mA.

In the CHMOS versions, the pullup consists of three pFETs. It should be noted that an n-channel FET (nFET) is turned on when a logical 1 is applied to its gate, and is turned off when a logical 0 is applied to its gate. A p-channel FET (PFET is the opposite: it is on when its gate sees a 0, and off when its gate sees a 1.

pFET1 in Figure 25 is the transistor that is turned on for 2 oscillator periods after a 0-to-1 transition in the port latch. While it's on, it turns on pFET3 (a weak pullup), through the inverter. This inverter and pPET form a latch which hold the 1.

Note that if the pin is emitting a 1, a negative glitch on the pin from some external source can turn off pFET3, causing the pin to go into a float state. pFET is a very weak pullup which is on whenever the nFET is off, in traditional CMOS style. It's only about 1/10 the strength of pFET3. Its function is to restore a 1 to the pin in the event the pin had a 1 and lost it to a glitch.

PORT LOADING AND INTERFACING

The output buffers of Ports 1, 2, and 3 can each drive 4 LS TTL inputs. These ports on HMOS versions can be driven in a normal manner by any TTL or NMOS circuit. Both HMOS and CHMOS pins can be driven by open-collector and open-drain outputs, but note that 0-to-1 transitions will not be fast.

In the HMOS device, if the pin is driven by an open-collector output, a 0-to-1 transition will have to be driven by the relatively weak depletion mode FET in Figure 25(A). In the CHMOS device, an input 0 turns off pullup pFET3, leaving only the very weak pullup pFET2 to drive the transition.

Port 0 output buffers can each drive 8 LS TTL inputs. They do, however, require external pullups to drive NMOS inputs, except when being used as the ADDRESS/DATA bus.

READ-MODIFY-WRITE FEATURE

Some instructions that read a port read the latch and others read the pin. Which ones do which? The instructions that read the latch rather than the pin are the ones that read a value, possibly change it, and then rewrite it to the latch. These are called "read-modify-write" instructions. The instructions listed below are read-modify-write 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)
SET PX.Y	(set bit Y of Port X)

It is not obvious that the last three instructions in this list are read-modify-write instructions, but they are. They read the port byte, all 8 bits, modify the addressed bit, then write the new byte back to the latch.

The reason that read-modify-write instructions are directed to the latch rather than the pin is to avoid a possible misinterpretation of the voltage level at the pin. For example, a port bit might be used to drive the base of a transistor. When a 1 is written to the bit, the transistor is turned on. If the CPU then reads the same port bit at the pin rather than the latch, it will read the base voltage of the transistor and interpret it as a 0. Reading the latch rather than the pin will return the correct value of 1.

ACCESSING EXTERNAL MEMORY

Accesses to external memory are of two types: accesses to external Program Memory and accesses to external Data Memory. Accesses to external Program Memory use signal $\overline{PSEN}$ (program store enable) as the read strobe. Accesses to external Data Memory use $\overline{RD}$ or $\overline{WR}$ (alternate functions of P3.7 and P3.6) to strobe the memory. Fetches from external Program Memory always use a 16-bit address. Accesses to external Data Memory can use either a 16-bit address (MOVX @ DPTR) or an 8-bit address (MOVX @Ri).

Whenever a 16-bit address is used, the high byte of the address comes out on Port 2, where it is held for the duration of the read or write cycle. Note that the Port 2 drivers use the strong pullups during the entire time that they are emitting address bits that are 1s. This is during the execution of a MOVX @DPTR instruction. During this time the Port 2 latch (the Special Function Register) does not have to contain 1s, and the contents of the Port 2 SFR are not modified. If the external memory cycle is not immediately followed by another external memory cycle, the undisturbed contents of the Port 2 SFR will reappear in the next cycle.

If an 8-bit address is being used (MOVX @Ri), the contents of the Port 2 SFR remain at the Port 2 pins throughout the external memory cycle. This will facilitate paging.

In any case, the low byte of the address is time-multiplexed with the data byte on Port 0. The ADDR/DATA signals drives both FETs in the Port 0 output buffers. Thus, in this application the Port 0 pins are not open-drain outputs, and do not require external pullups. ALE (Address Latch Enable) should be used to capture the address byte into an external latch. The address byte is valid at the negative transition of ALE. Then, in a write cycle, the data byte to be written appears on Port 0 just before $\overline{WR}$ is activated, and remains there until after $\overline{WR}$ is deactivated. In a read cycle, the incoming byte is accepted at Port 0 just before the read strobe is deactivated.

During any access to external memory, the CPU writes 0FFH to the Port 0 latch (the Special Function Register), thus obliterating whatever information the Port 0 SFR may have been holding.

External Program Memory is accessed under two conditions: Whenever signal EA is active; or whenever the program counter (PC) contains a number that is larger than 0FFFH (in the 8051).

This requires that the ROMless versions have $\overline{EA}$ wired low to enable the lower 4K program bytes to be fetched from external memory.

When the CPU is executing out of external Program Memory, all 8 bits of Port 2 are dedicated to an output function and may not be used for general purpose I/O. During external program fetches they output the high byte of the PC. During this time the Port 2 drivers use the strong pullups to emit PC bits that are 1s.

TIMER/COUNTERS

The 8051 has two 16-bit Timer/Counter registers: Timer 0 and Timer 1. Both can be configured to operate either as timers or event counters (see Figure 26).

In the "Timer" function, the 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 or T1. 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 cycle. In addition to the "Timer" or "Counter" selection, Timer 0 and Timer 1 have four operating modes from which to select.

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 Timers/Counters. Mode 3 is different. The four operating modes are described in the following text.

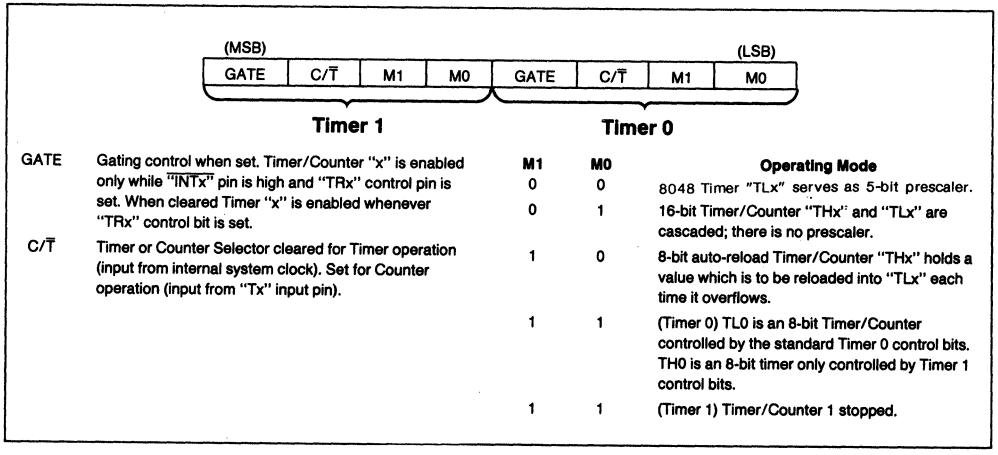


Figure 26. Timer/Counter Mode Control (TMOD) Register

MODE 0

Putting either Timer into Mode 0 makes it look like an 8048 Timer, which is an 8-bit Counter with a divide-by-32 prescaler. Figure 27 shows the Mode 0 operation as it applies to Timer 1.

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 TF1. The counted input is enabled to the Timer when TR1 = 1 and either GATE = 0 or INT1 = 1. (Setting GATE = 1 allows the Timer to be controlled by external input INT1, to facilitate pulse width measurements). TR1 is a control bit in the Special Function Register TCON (Figure 28). GATE is in TMOD.

The 13-Bit register consists of all 8 bits of TH1 and the lower 5 bits of TL1. The upper 3 bits of TL1 are indeterminate and should be ignored. Setting the run flag (TR1) does not clear the registers.

Mode 0 operation is the same for the Timer 0 as for Timer 1. Substitute TR0, TF0 and INT0 for the corresponding Timer 1 signals in Figure 27. There are two different GATE bits, one for Timer 1 (TMOD.7) and one for Timer 0 (TMOD.3).

MODE 1

Mode 1 is the same as Mode 0, except that the Timer register is being run with all 16 bits.

MODE 2

Mode 2 configures the Timer register as an 8-bit Counter (TL1) with automatic reload, as shown in Figure 29. Overflow from TL1 not only sets TF1, but also reloads TL1 with the contents of TH1, which is preset by software. The reload leaves TH1 unchanged.

Mode 2 operation is the same for Timer/Counter 0.

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 30. TL0 uses the Timer 0 control bits: C/T, GATE, TR0, INT0, and TF0. TH0 is locked 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 on the counter. With Timer 0 in Mode 3, an 8051 can look like it has three Timer/Counters. 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 fact, in any application not requiring an interrupt.

STANDARD 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 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 at Special Function Register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The serial port can operate in 4 modes:

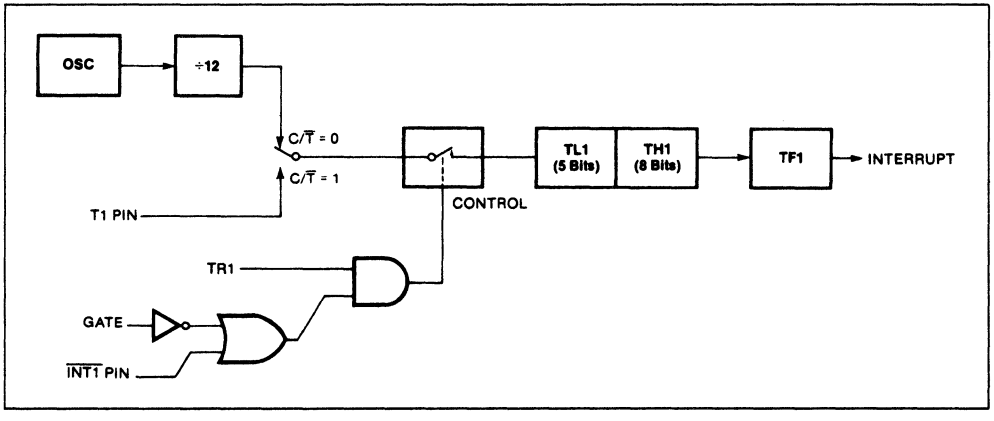


Figure 27. Timer/Counter 1 Mode 0: 13-Bit Counter

(MSB)							(LSB)
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Symbol	Position	Name and Significance
TF1	TCON.7	Timer 1 overflow Flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine.
TR1	TCON.6	Timer 1 Run control bit. Set/cleared by software to turn Timer/Counter on/off.
TF0	TCON.5	Timer 0 overflow Flag. Set by hardware on Timer/Counter overflow. Cleared by hardware when processor vectors to interrupt routine.
TR0	TCON.4	Timer 0 Run control bit. Set/cleared by software to turn Timer/Counter on/off.
IE1	TCON.3	Interrupt 1 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
IT1	TCON.2	Interrupt 1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
IE0	TCON.1	Interrupt 0 Edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
IT0	TCON.0	Interrupt 0 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.

Figure 28. Timer/Counter Control (TCON) Register

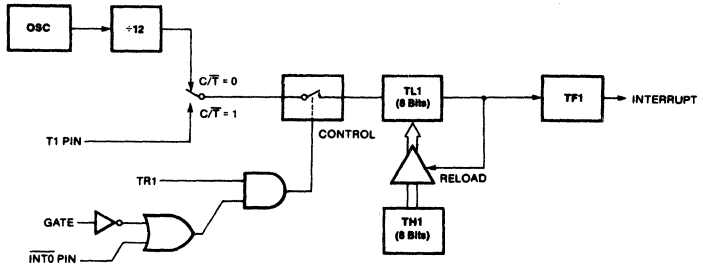


Figure 29. Timer/Counter 1 Mode 2: 8-Bit Auto-Load

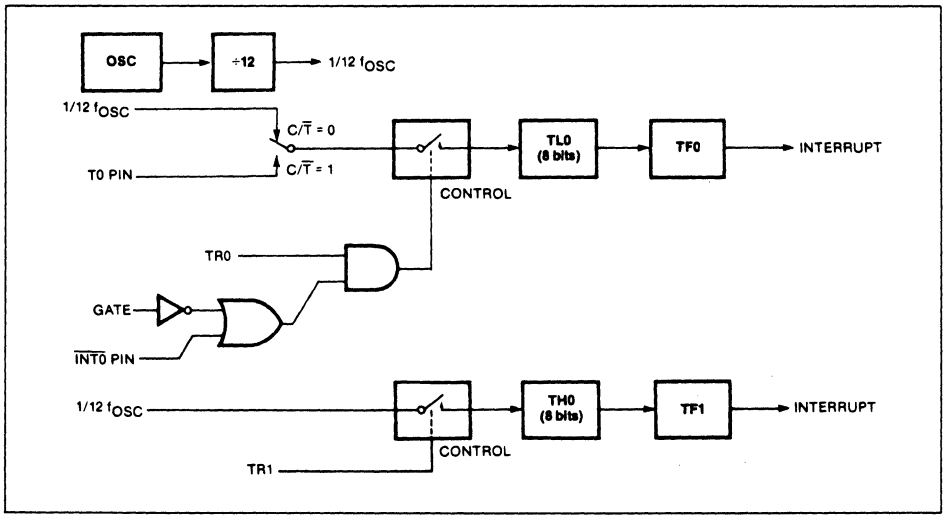


Figure 30. Timer/Counter 0 Mode 3: Two 8-Bit Counters

Mode 0: Serial data enters and exits through RxD. TxD outputs the shift clock. 8 bits are transmitted/received (LSB first). The baud is fixed at 1/12 the oscillator frequency.

Mode 1: 10 bits are transmitted (through TxD) or received (through Rx D): 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): start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). 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 Special Function Register SCON, while the stop bit is ignored. The baud rate is programmable to either 1/32 or 1/62 the oscillator frequency.

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 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.

MULTIPROCESSOR COMMUNICATIONS

Modes 2 and 3 have a special provision for multiprocessor communications. In these modes, 9 data bits are received. The 9th one goes into RB8. Then comes a stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 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. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that weren't being addressed leave their SM2s set and go on about their business, ignoring the coming data bytes.

SM2 has no effect in Mode 0, and in Mode 1 can be used to check the validity of the stop bit. In a Mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

SERIAL PORT CONTROL REGISTER

The serial port control and status register is the Special Function Register SCON, shown in Figure 31. This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (T1 and R1).

(MSB)							(LSB)
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Where SM0, SM1 specify the serial port mode, as follows:

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 multiprocessor communication feature in Modes 2 and 3. In Mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In Mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In Mode 0, SM2 should be 0.
- REN enables serial reception. Set by software to enable reception. Clear by software to disable reception.
- TB8 is 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, is 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 is 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 is 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.

Figure 31. Serial Port Control (SCON) Register

BAUD RATES

The baud rate in Mode 0 is fixed: Mode 0 Baud Rate = Oscillator Frequency / 12. The baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON. If SMOD = 0 (which is the value on reset), the baud rate is 1/64 the oscillator frequency. If SMOD = 1, the baud rate is 1/32 the oscillator frequency.

Mode 2 Baud Rate =

$$\frac{2^{SMOD}}{64} \times (\text{Oscillator Frequency})$$

In the 8051, the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate.

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 SMOD as follows:

Mode 1, 3 Baud Rate =

$$\frac{2^{SMOD}}{32} \times (\text{Timer 1 Overflow Rate})$$

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 the most typical applications, it is configured for "timer" operation, in the auto-reload mode (high nibble of TMOD = 0010B). In that case the baud rate is given by the formula:

Mode 1, 3 Baud Rate =

$$\frac{2^{SMOD}}{32} \times \frac{\text{Oscillator Frequency}}{12 \times [256 - (TH1)]}$$

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. Figure 32 lists various commonly used baud rates and how they can be obtained from Timer 1.

MORE ABOUT MODE 0

Serial data enters and exists 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.

Figure 33 shows a simplified functional diagram of the serial port in Mode 0, and associated timing.

Baud Rate	fosc	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

Figure 32. Timer 1 Generated Commonly Used Baud Rates

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal at S6P2 also loads a 1 into the 9th position of the transmit shift register and tells the TX Control block to commence a transmission. The internal timing is such that one full machine cycle will elapse between "write to SBUF", and activation of SEND.

SEND enables the output of the shift register to the alternate output function line of P3.0 and also enables SHIFT CLOCK to the alternate output function line of P3.1. SHIFT CLOCK is low during S3, S4, and S5 of every machine cycle, and high during S6, S1 and S2. At S6P2 of every machine cycle in which SEND is active, the contents of the transmit shift are shifted to the right one position.

As data bits shift out to the right, zeros come in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position, is just to the left of the MSB, and all positions to the left of that contain zeros. This condition flags the TX Control block to do one last shift and then deactivate SEND and set T1. Both of these actions occur at S1P1 of the 10th machine cycle after "write to SBUF".

Reception is initiated by the condition REN = 1 and R1 = 0. At S6P2 of the next machine cycle, the RX Control unit writes the bits 11111110 to the receive shift register, and in the next clock phase activates RECEIVE.

RECEIVE enables SHIFT CLOCK to the alternate output function line of P3.1. SHIFT CLOCK makes transitions at S3P1 and S6P1 of every machine cycle. At S6P2 of every machine cycle in which RECEIVE is active, the contents of the receive shift register are shifted to the left one position. The value that comes in from the right is the value that was sampled at the P3.0 pin at S5P2 of the same machine cycle.

As data bits come in from the right, 1s shift out to the left. When the 0 that was initially loaded into the rightmost position arrives at the leftmost position in the shift register, it flags the RX Control block to do one last shift and load SBUF. At S1P1 of the 10th machine cycle after the write to SCON that cleared RI, RECEIVE is cleared as RI is set.

MORE ABOUT MODE 1

Ten 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 SCON. In the 8051 the baud rate is determined by the Timer 1 overflow rate.

Figure 34 shows a simplified functional diagram of the serial port in Mode 1, and associated timings for transmit receive.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal also loads a 1 into the 9th bit position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission actually commences at S1P1 of the machine cycle following the next rollover in the divide-by-16 counter. (Thus, the bit times are synchronized to the divide-by-16 counter, not to the "write to SBUF" signal).

The transmission begins with activation of SEND which puts the start bit at TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that.

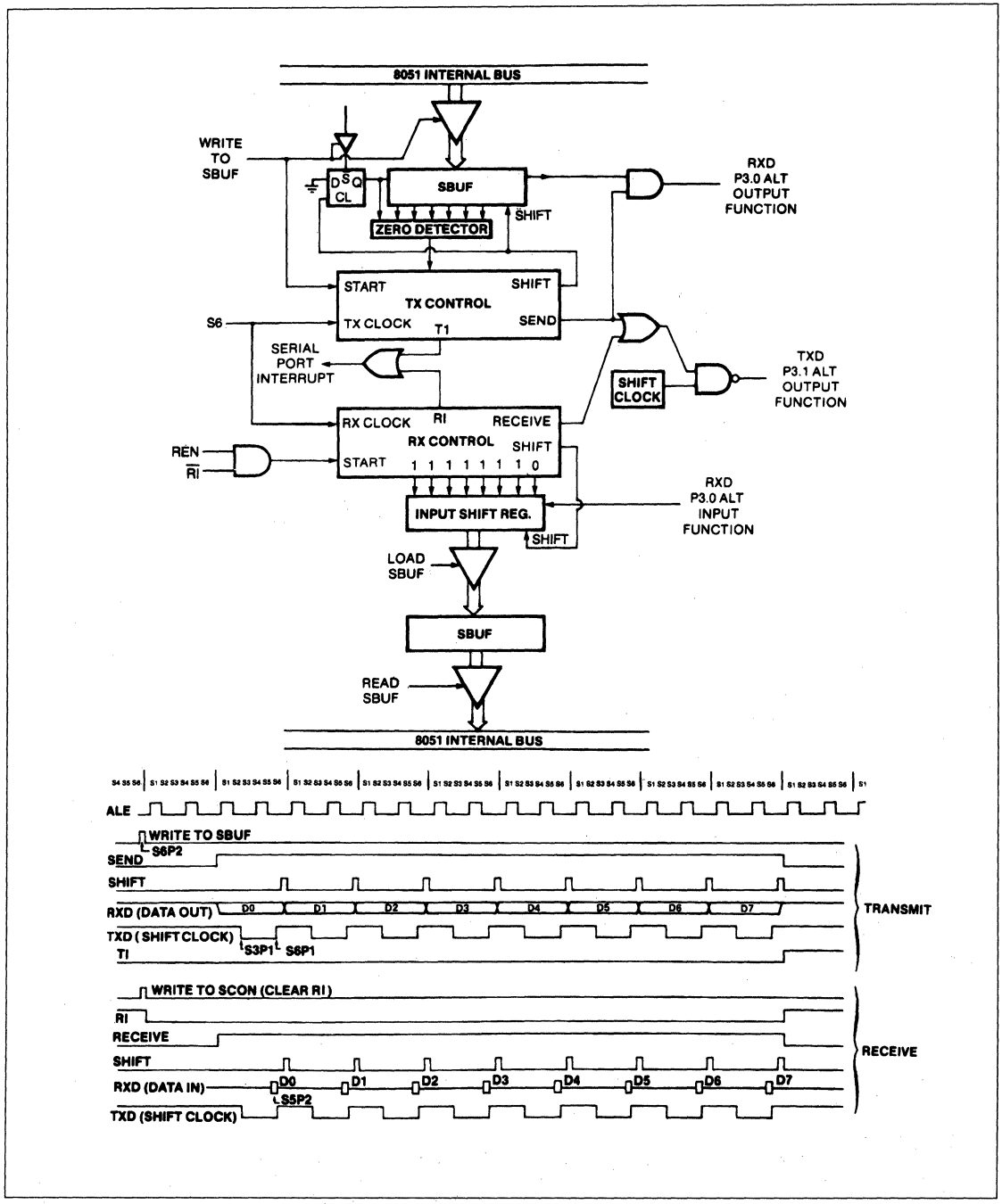


Figure 33. Serial Port Mode 0

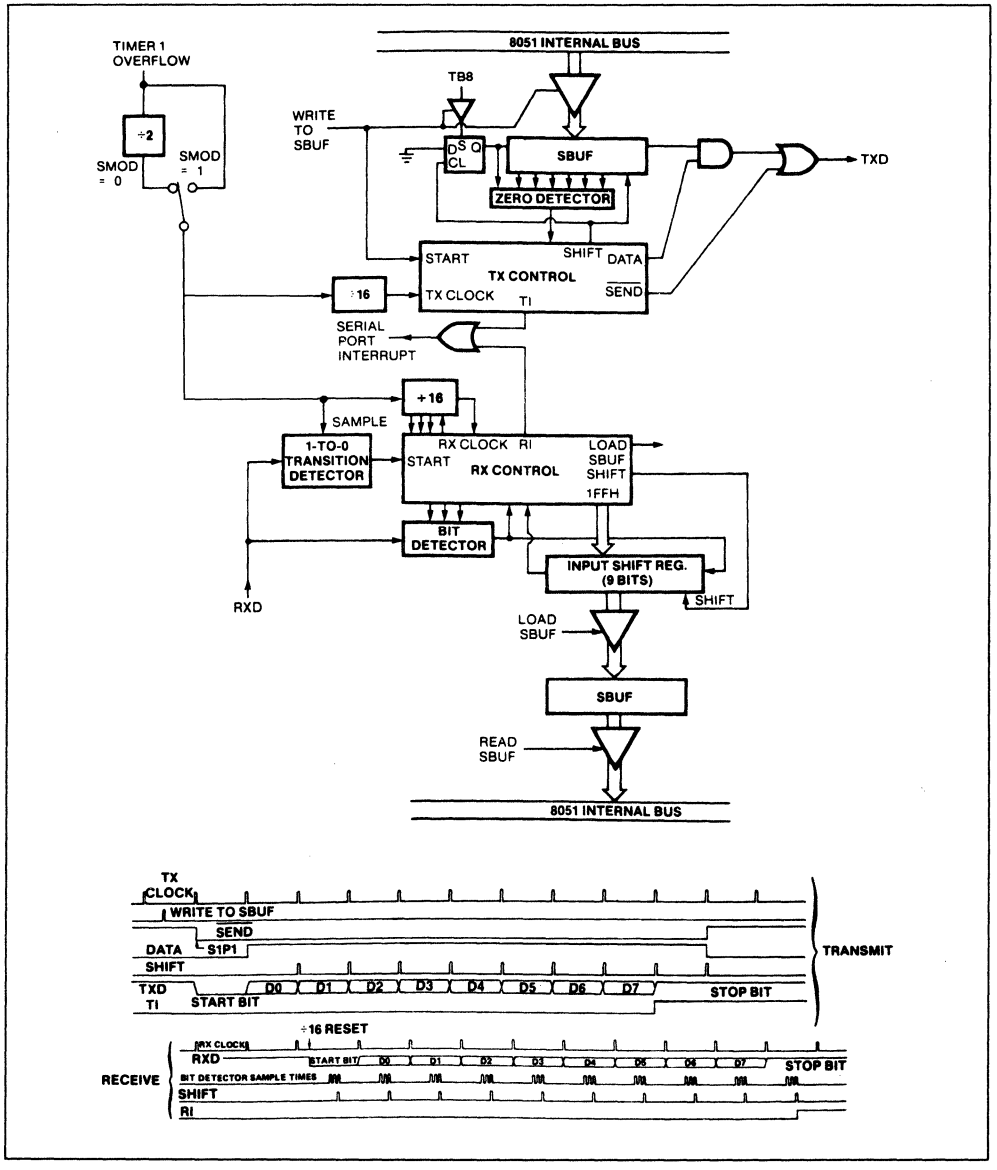


Figure 34. Serial Port Mode 1

As data bits shift out to the right, zeros are clocked in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position is just to the left of the MSB, and all positions to the left of that contain zeros. This condition flags the TX Control unit to do one last shift and then deactivate SEND and set TI. This occurs at the 10th divide-by-16 rollover after "write to SBUF".

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written into the input shift register. Resetting the divide-by-16 counter aligns its rollovers with the boundaries of the incoming bit times.

The 16 states of the counter divide each bit time into 16ths. At the 7th, 8th, and 9th counter states of each bit time, the bit detector samples the value of RxD. The value accepted is the value that was seen in at least 2 of the 3 samples. This is done for noise rejection. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. This is to provide rejection of false start bits. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register, (which in mode 1 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI. The signal to load SBUF and RB8, and to set RI, will generated if, and only if, the following conditions are met at the time the final shift pulse is generated.

1. R1 = 0, and
2. Either SM2 = 0, or the received stop bit = 1

If either of these two conditions is not met, the received frame is irretrievably lost. If both conditions are met, the stop bit goes into RB8, the 8 data bits go into SBUF, and RI is activated. At this time, whether the above conditions are met or not, the unit goes back to looking for a 1-to-0 transition in RxD.

MORE ABOUT MODES 2 AND 3

Eleven 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). On transmit, the 9th data bit (TB8) can be assigned the value of 0 or 1. On receive, the 9th data bit goes into RB8 in SCON. The baud rate is programmable to either 1/32 or 1/64 the oscillator frequency in Mode 2. Mode 3 may have a variable baud rate generated from Timer 1.

Figures 35 and 36 show a functional diagram of the serial port in Modes 2 and 3. The receive portion is exactly the same as in Mode 1. The transmit portion differs from Mode 1 only in the 9th bit of the transmit shift register.

Transmission is initiated by any instruction that uses SBUF as a destination register. The "write to SBUF" signal also loads TB8 into the 9th bit position of the transmit shift register and flags the TX Control unit that a transmission is requested. Transmission commences at S1P1 of the machine cycle following the next roll-over in the divide-by-16 counter (thus, the bit times are synchronized to the divide-by-16 counter, not to the "write to SBUF" signal).

The transmission begins with activation of SEND, which puts the start bit at TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that. The first shift clocks a 1 (the stop bit) into the 9th bit position of the shift register. Thereafter, only zeros are clocked in. Thus, as data bits shift out to the right, zeros are clocked in from the left. When TB8 is at the output position of the shift register, then the stop bit is just to the left of TB8, and all positions to the left of that contain zeros. This condition flags the TX Control unit to do one last shift and then deactivate SEND and set TI. This occurs at the 11th divide-by-16 rollover after "write to SUBF".

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1FFH is written to the input shift register.

At the 7th, 8th and 9th counter states of each bit time, the bit detector samples the value of R-D. The value accepted is the value that was seen in at least 2 of the 3 samples. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back to looking for another 1-to-0 transition. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bits come in from the right, 1s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in Modes 2 and 3 is a 9-bit register), it flags the RX Control block to do one last shift, load SBUF and RB8, and set RI.

The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated.

1. RI = 0, and
2. Either SM2 = 0 or the received 9th data bit = 1

If either of these conditions is not met, the received frame is irretrievably lost, and RI is not set. If both conditions are met, the received 9th data bit goes into RB8, and the first 8 data bits go into SBUF. One bit time later, whether the above conditions were met or not, the unit goes back to looking for a 1-to-0 transition at the RxD input.

INTERRUPTS

The 8051 provides 5 interrupt sources. These are shown in Figure 37. The External Interrupts $\overline{INT0}$ and $\overline{INT1}$ can each be either level-activated or transition-activated, depending on bits IT0 and IT1 in Register TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. When an external interrupt is generated, the flag that generated it is cleared by the hardware when the service routine is vectored to only if the interrupt was transition-activated. If the interrupt was level activated, then the external requesting source is what controls the request flag, rather than the on-chip hardware.

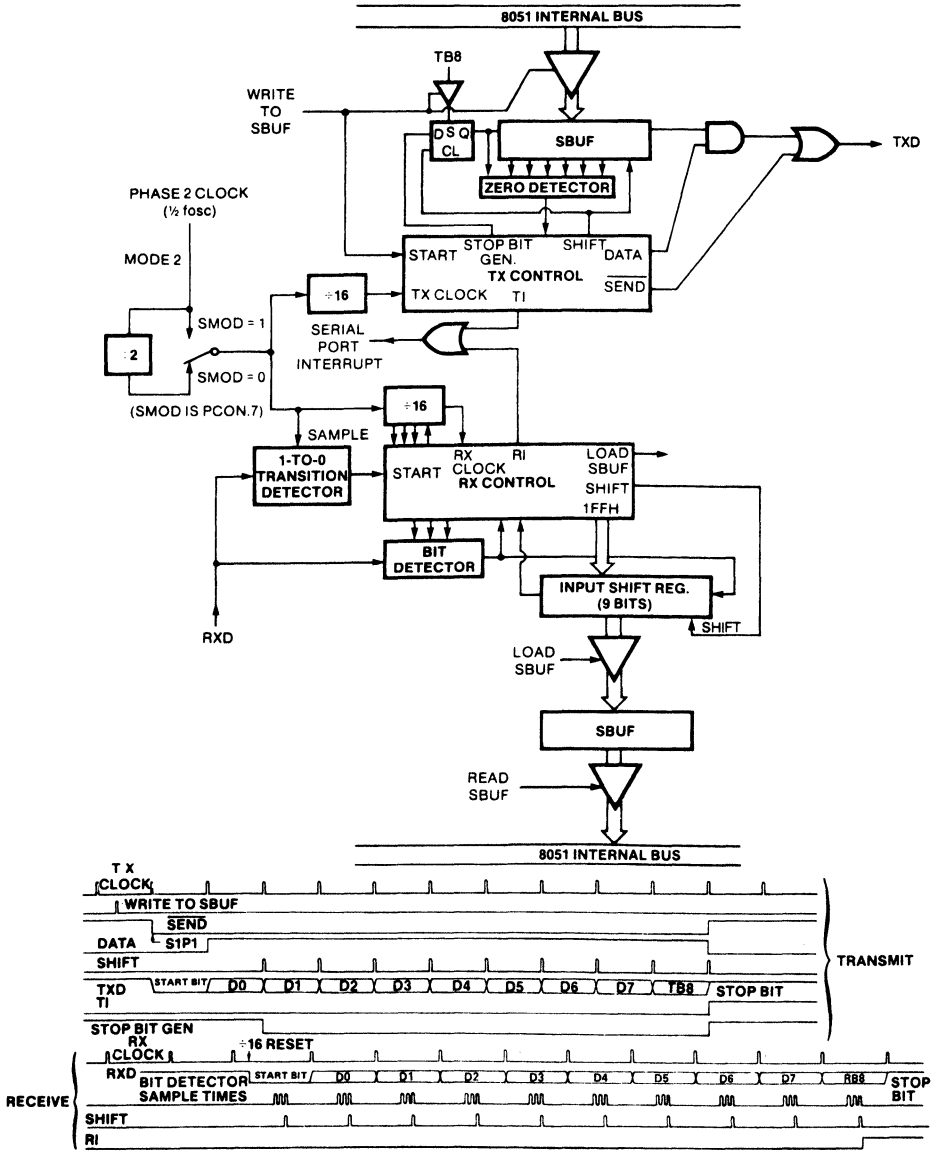


Figure 35. Serial Port Mode 2

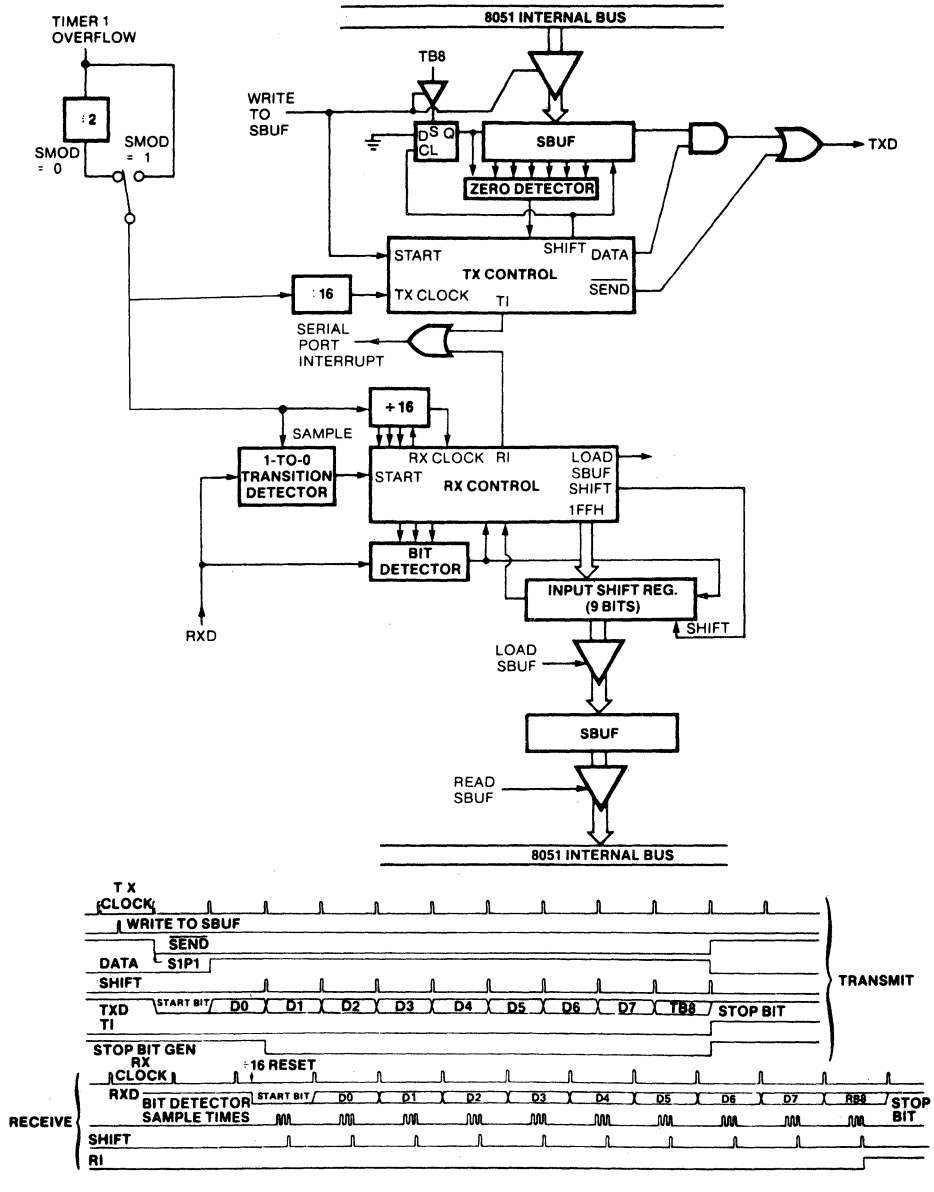


Figure 36. Serial Port Mode 3

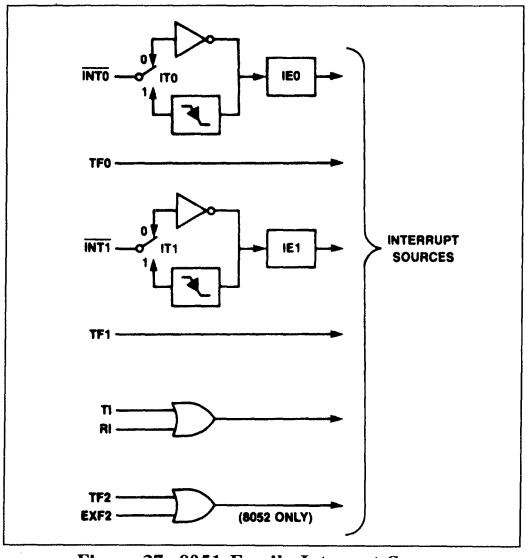


Figure 37. 8051 Family Interrupt Sources

(MSB)							(LSB)
$\overline{EA}$	X	X	ES	ET1	EX1	ET0	EX0

Symbol	Position	Function
$\overline{EA}$	IE.7	disables all interrupts. If $\overline{EA}$ = 0, no interrupt will be acknowledged. If $\overline{EA}$ = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
—	IE.6	reserved
—	IE.5	reserved
ES	IE.4	enables or disables the Serial Port interrupt. If ES = 0, the Serial Port interrupt is disabled.
ET1	IE.3	enables or disables the Timer 1 Overflow interrupt. If ET1 = 0, the Timer 1 interrupt is disabled.
EX1	IE.2	enables or disables External Interrupt 1. If EX1 = 0, External Interrupt 1 is disabled.
ET0	IE.1	enables or disables the Timer 0 Overflow interrupt. If ET0 = 0, the Timer 0 interrupt is disabled.
EX0	IE.0	enables or disables External Interrupt 0. If EX0 = 0, External Interrupt 0 is disabled.

Figure 38. Interrupt Enable Register (IE)

The Timer 0 and Timer 1 Interrupts are generated by TF0 and TF1, which are set by a rollover in their respective Timer/Counter registers (except see Timer 0 in Mode 3). When a timer interrupt is generated, the flag that generated it is cleared by the on-chip hardware when the service routine is vectored to.

The Serial Port Interrupt is generated by the logical OR of RI and TI. Neither of theses flags is cleared by hardware when the service routine is vectored to. In fact, the service routine will normally have to determine whether it was RI or TI that generated the interrupt, and the bit will have to be cleared in software.

All of the bits that generate interrupts can be set or cleared by software, with the same result as though it had been set or cleared by hardware. That is, interrupts can be generated or pending interrupts can be canceled in software.

Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE (Figure 38). IE also contains a global disable bit, $\overline{EA}$, which disables all interrupts at once.

PRIORITY LEVEL STRUCTURE

Each interrupt source can also be individually programmed to one of two priority levels by setting or clearing a bit in Special Function Register IP (Figure 39). A low-priority interrupt can itself be interrupted by a high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt can't be interrupted by any other interrupt source.

(MSB)							(LSB)
X	X	X	PS	PT1	PX1	PT0	PX0

Symbol	Position	Function
–	IP.7	reserved
–	IP.6	reserved
–	IP.5	reserved
PS	IP.4	defines the Serial Port interrupt priority level. PS=1 programs it to the higher priority level.
PT1	IP.3	defines the Timer 1 interrupt priority level. PT1=1 programs it to the higher priority level.
PX1	IP.2	defines the External Interrupt 1 priority level. PX1=1 programs it to the higher priority level.
PT0	IP.1	defines the Timer 0 interrupt priority level. PT0=1 programs it to the higher priority level.
PX0	IP.0	defines the External Interrupt 0 priority level. PX0=1 programs it to the higher priority level.

Figure 39. Interrupt Priority Register (IP)

If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence as follows:

Source	Priority Within Level
1. IE0	(highest)
2. TF0	
3. IE1	
4. TF1	
5. RI+TI	(lowest)

Note that the "priority within level" structure is only used to resolve simultaneous requests of the same priority level.

The IP register contains a number of unimplemented bits. IP.7, IP.6 and IP.5 are reserved in the 8051s. User software should not write 1s to these positions, since they may be used in other 8051 Family products.

HOW INTERRUPTS ARE HANDLED

The interrupt flags are sampled at S5P2 of every machine cycle. The samples are polled during the following machine cycle. If one of the flags was in a set condition at S5P2 of the preceding cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

1. An interrupt of equal or higher priority level is already in progress.
2. The current (polling) cycle is not the final cycle in the execution of the instruction in progress.
3. The instruction in progress is RETI or any write to the IE or IP registers.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress will be completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any access to IE or IP, then at least one more instruction will be executed before any interrupt is vectored to.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at S5P2 of the previous machine cycle. Note than that if an interrupt flag is active but not being responded to for one of the above conditions, if the flag is not still active when the blocking condition is removed, the denied interrupt will not be serviced. In other words, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle is new.

The polling cycle/LCALL sequence is illustrated in Figure 40.

Note that if an interrupt of higher priority level goes active prior to S5P2 of the machine cycle labeled C3 in Figure 40, then in accordance with the above rules it will be vectored to during C5 and C6, without any instruction of the lower priority routine having been executed.

Thus the processor acknowledges an interrupt request by executing a hardware generated LCALL to the appropriate servicing routine. In some cases it also clears the flag that generated the interrupt, and in other cases it doesn't. It never clears the Serial Port or Timer 2 flags. This has to be done in the user's software. It clears an external interrupt flag (IE0 or IE1) only if it was transition-activated. The hardware-generated LCALL pushes the contents of the Program Counter on to the stack (but it does not save the PSW) and reloads the PC with an address that depends on the source of the interrupt being vectored to, as shown below:

Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI+TI	0023H

Execution proceeds from that location until the RETI instruction is encountered. The RETI instruction informs the processor that this interrupt routine is no longer in progress, then pops the top two bytes from the stack and reloads the Program Counter. Execution of the interrupted program continues from where it left off.

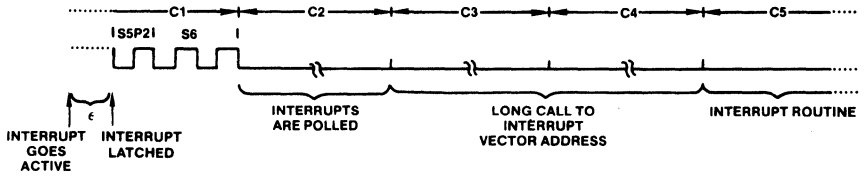
Note that a simple RET instruction would also have returned execution to the interrupted program, but it would have left the interrupt control system thinking an interrupt was still in progress.

EXTERNAL INTERRUPTS

The external sources can be programmed to be level-activated or transition-activated by setting or clearing bit IT1 or IT0 in Register TCON. If ITx = 0, external interrupt x is triggered by a detected low at the INTx pin. If ITx = 1, external interrupt x is edge triggered. In this mode if successive samples of the INTx pin show a high in one cycle and a low in the next cycle, interrupt request flag IEx in TCON is set. Flag bit IEx then requests the interrupt.

Since the external interrupt pins are sampled once each machine cycle, an input high or low should hold for at least 12 oscillator periods to ensure sampling. If the external interrupt is transition-activated, the external source has to hold the request pin high for at least one cycle, and then hold it low for at least one cycle. This is done to ensure that the transition is seen so that interrupt request flag IEx will be set. IEx will be automatically cleared by the CPU when the service routine is called.

If the external interrupt is level-activated, the external source has to hold the request active until the requested interrupt is actually generated. Then it has to deactivate the request before the interrupt service routine is completed, or else another interrupt will be generated.



This is the fastest possible response when C2 is the final cycle of an instruction other than RETI or an access to IE or IP.

Figure 40. Interrupt Response Timing Diagram

RESPONSE TIME

The $\overline{INT0}$ and $\overline{INT1}$ levels are inverted and latched into IE0 and IE1 at S5P2 of every machine cycle. The values are not actually polled by the circuitry until the next machine cycle. If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two cycles. Thus, a minimum of three complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine. Figure 40 shows interrupt response timings.

A longer response time would result if the request is blocked by one of the 3 previously listed conditions. If an interrupt of equal or higher priority level is already in progress, the additional wait time obviously depends on the nature of the other interrupt's service routine. If the instruction in progress is not in its final cycle, the additional wait time cannot be more the 3 cycles, since the longest instructions (MUL and DIV) are only 4 cycles long, and if the instruction in progress is RETI or an access to IE or IP, the additional wait time cannot be more than 5 cycles (a maximum of one more cycle to complete the instruction in progress, plus 4 cycles to complete the next instruction if the instruction is MUL or DIV).

Thus, in a single-interrupt system, the response time is always more than 3 cycles and less than 9 cycles.

SINGLE-STEP OPERATION

The 8051 interrupt structure allows single-step execution with very little software overhead. As previously noted, an interrupt request will not be responded to while an interrupt of equal priority level is still in progress, nor will it be responded to after RETI until at least one other instruction has been executed. Thus, once an interrupt routine has been entered, it cannot be re-entered until at least one instruction of the interrupted program is executed. One way to use this feature for single-step operation is to program one of the external interrupts (e.g. INT0) to be level-activated. The service routine for the interrupt will terminate with the following code:

```
JNB     P3.2,$   ;Wait Till INT0 Goes High
JB      P3.2,$   ;Wait Till INT0 Goes Low
RETI             ;Go Back and Execute One Instruction
```

Now if the $\overline{INT0}$ pin, which is also the P3.2 pin, is held normally low, the CPU will go right into the External Interrupt 0 routine and stay there until $\overline{INT0}$ is pulsed (from low to high to low). Then it will execute RETI, go back to the task program, execute one instruction, and immediately re-enter the External Interrupt 0 routine to await the next pulsing of P3.2. One step of the task program is executed each time P3.2 is pulsed.

RESET

The reset input is the RST pin, which is the input to a Schmitt Trigger. A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. The CPU responds by generating an internal reset, with the timing shown in Figure 41.

The external reset signal is asynchronous to the internal clock. The RST pin is sampled during State 5 Phase 2 of every machine cycle. The port pins will maintain their current activities for 19 oscillator periods after a logic 1 has been sampled at the RST pin; that is, for 19 to 31 oscillator periods after the external reset signal has been applied to the RST pin.

The internal reset algorithm writes 0s to all the SFRs except the port latches, the Stack Pointer, and SBUF. The port latches are initialized to FFH, the Stack Pointer to 07H, and SBUF is indeterminate. Table 11 lists the SFR reset values. The internal RAM is not affected by reset. On power up the RAM content is indeterminate.

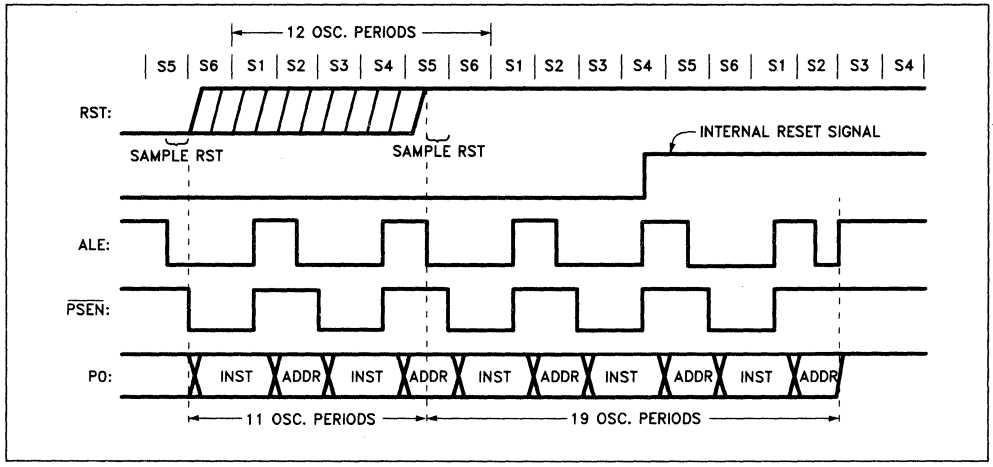


Figure 41. Reset Timing

Table 11. 8051 SFR Reset Values

Register	Reset Value
PC	0000H
ACC	00H
B	00H
PSW	00H
SP	07H
DPTR	0000H
P0–P3	FFH
IP	XXX00000B
IE	0XX00000B
TMOD	00H
TCON	00H
TH0	00H
TL0	00H
TH1	00H
TL1	00H
SCON	00H
SBUF	Indeterminate
PCON (HMOS)	0XXXXXXXB
PCON (CHMOS)	0XXX0000B

POWER-ON RESET

An automatic reset can be obtained when V_{CC} is turned on by connecting the RST pin to V_{CC} through a 10µf capacitor and to V_{SS} through an 8.2K resistor, providing the V_{CC} rise time does not exceed a millisecond and the oscillator start-up time does not exceed 10 milliseconds. This power-on reset circuit is shown in Figure 42. The CHMOS devices do not require the 8.2K pulldown resistor, although its presence does no harm.

When power is turned on, the circuit holds the RST pin high for an amount of time that depends on the value of the capacitor and the rate at which it charges. To ensure a good reset, the RST pin must be high long enough to allow the oscillator time to start-up (normally a few msec) plus two machine cycles.

Note that the port pins will be in a random state until the oscillator has started and the internal reset algorithm has written 1s to them.

With this circuit, reducing V_{CC} quickly to 0 causes the RST pin voltage to momentarily fall below 0V. However, this voltage is internally limited, and will not harm the device.

POWER-SAVING MODES OF OPERATION

For applications where power consumption is critical the CHMOS version provides power reduced modes of operation as a standard feature. The power down mode in HMOS devices is no longer a standard feature.

CHMOS POWER REDUCTION MODE

CHMOS versions have two power reducing modes, Idle and Power Down. The input through which backup power is supplied during these operations is V_{CC}. Figure 43 shows the internal circuitry which implements these features. In the Idle modes (IDL = 1), the oscillator continues to run and the Interrupt, Serial Port, 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. The address of this register is 87H. Figure 44 details its contents.

In the HMOS devices the PCON register only contains SMOD. The other four bits are implemented only in the CHMOS devices. User software should never write 1s to unimplemented bits, since they may be used in other 8051 Family products.

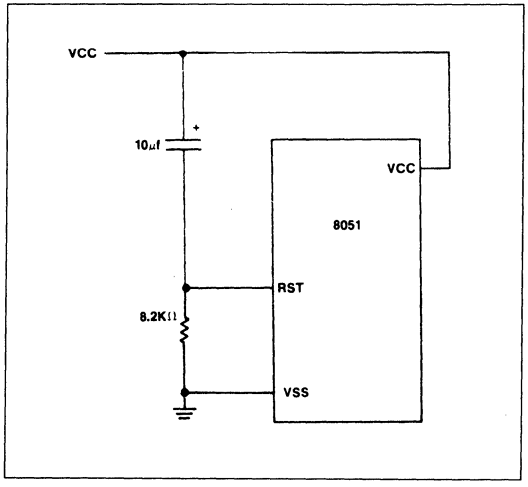


Figure 42. Power On Reset Circuit

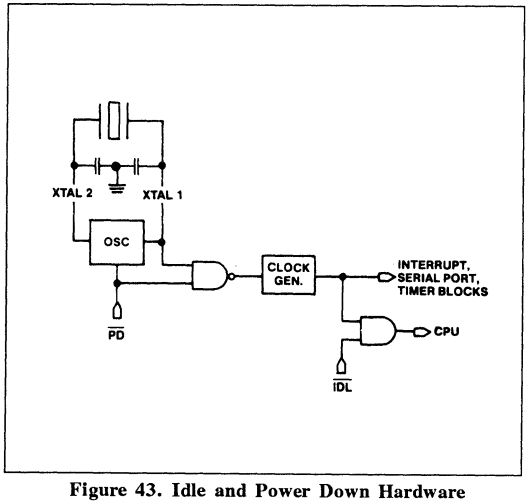


Figure 43. Idle and Power Down Hardware

IDLE MODE

An instruction that sets PCON.0 causes that to be the last instruction executed before going into the Idle mode, the internal clock signal is gated off to the CPU but not to the Interrupt, Timer and Serial Port functions. 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 $\overline{PSEN}$ hold at logic high levels.

(MSB)							(LSB)
SMOD	-	-	-	GF1	GF0	PD	IDL

Symbol	Position	Name and Function
SMOD	PCON.7	Double Baud rate bit. When set to a 1 and Timer 1 is used to generate baud rate, and the Serial Port is used in modes 1, 2, or 3.
—	PCON.6	(Reserved)
—	PCON.5	(Reserved)
—	PCON.4	(Reserved)
GF1	PCON.3	General-purpose flag bit.
GF0	PCON.2	General-purpose flag bit.
PD	PCON.1	Power Down bit. Setting this bit activates power down operation.
IDL	PCON.0	Idle mode bit. Setting this bit activates idle mode operation.

If 1s are written to PD and IDL at the same time, PD takes precedence. The reset value of PCON is (0XXX0000). In the HMOS devices the PCON register only contains SMOD. The other four bits are implemented only in the CHMOS devices. User software should never write 1s to unimplemented bits, since they may be used in future products.

Figure 44. Power Control (PCON) Register

There are two ways to terminate the Idle. 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 an 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. Since the clock oscillator is still running, the hardware reset needs to be held active for only two machine cycles (24 oscillator periods) to complete the 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. As shown in Figure 41, two or three machine cycles of program execution may take place before the internal reset algorithm takes control. On-chip hardware inhibits access to the internal RAM during this time, but access to the port pins is not inhibited. To eliminate the possibility of unexpected outputs at the port pins, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external Data RAM.

POWER DOWN MODE

An instruction that sets PCON.1 causes that to be the last instruction executed before going into the Power Down mode. In the Power Down mode, the on-chip oscillator is stopped. With the clock frozen, all functions

are stopped, the contents of the on-chip RAM and Special Function Registers are maintained. The port pins output the values held by their respective SFRs. The ALE and $\overline{PSEN}$ output are held low.

The only exit from Power Down is a hardware reset. Reset redefines all the SFRs, but does not change the on-chip RAM.

In the Power Down mode of operation, V_{CC} can be reduced to as low as 2V. Care must be taken, however, to ensure that V_{CC} is not reduced before the Power Down mode is invoked, and that V_{CC} is restored to its normal operating level, before the Power Down mode is terminated. The reset that terminates Power Down also frees the oscillator. The reset should not be activated before V_{CC} is restored to its normal operating level, and must be held active long enough to allow the oscillator to restart and stabilize (normally less than 10msec).

THE ON-CHIP OSCILLATORS

HMOS Versions

The on-chip oscillator circuitry for the HMOS (–I and –II) members of the 8051 family is a single stage linear inverter (Figure 45), intended for use as a crystal-controlled, positive reactance oscillator (Figure 46). In this application the crystal is operated in its fundamental response mode as an inductive reactance in parallel resonance with capacitance external to the crystal.

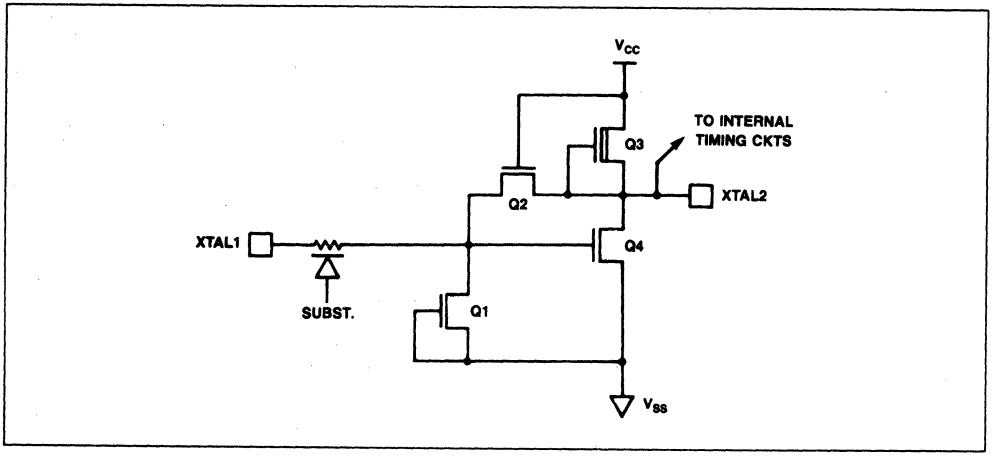


Figure 45. On-Chip Oscillator in the HMOS Version of the 8051 Family

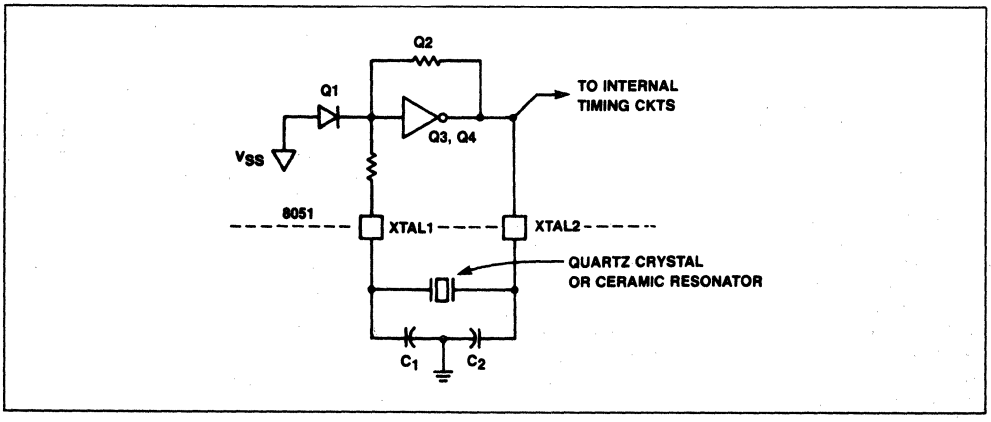


Figure 46. Using the HMOS On-Chip Oscillator

The crystal specifications and capacitance values (C1 and C2 in Figure 46) are not critical. 30pF can be used in these positions at any frequency with good quality crystals. A ceramic resonator can be used in place of the crystal in cost-sensitive applications. When a ceramic resonator is used, C1 and C2 are normally selected to be of somewhat higher values, typically, 47pF. The manufacturer of the ceramic resonator should be consulted for recommendation on the values of these capacitors.

To drive the HMOS parts with an external clock source, apply the external clock signal to XTAL2, and ground XTAL1, as shown in Figure 47. A pullup resistor may be used (to increase noise margin), but is optional if V_{OH} of the driving gate exceeds the V_{IH} minimum specification of XTAL.

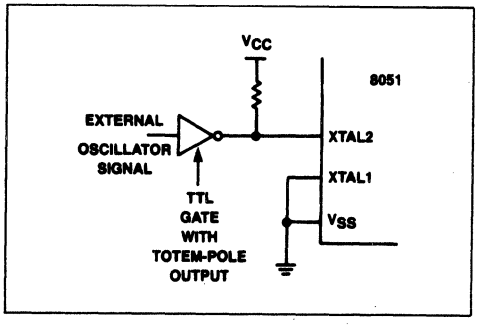


Figure 47. Driving the HMOS 8051 Family Parts with an External Clock

CHMOS VERSIONS

The on-chip oscillator circuitry for the 80C51BH, shown in Figure 48, consists of a single stage linear inverter intended for use as a crystal-controlled, positive reactance oscillator in the same manner as the HMOS parts. However, there are some important differences.

One difference is that the 80C51BH is able to turn off its oscillator under software control (by writing a 1 to the PD bit in PCON). Another difference is that, in the 80C51BH, the internal clocking circuitry is driven by the signal at XTAL1, whereas in the HMOS versions it is by the signal at XTAL2.

The feedback resistor R_f in Figure 48 consists of paralleled n-and p-channel FETs controlled by the PD bit, such that R_f is opened when PD = 1. The diodes D1 and D2, which act as clamps to V_{CC} and V_{SS}, are parasitic to the R_f FETs. The oscillator can be used with the same external components as the HMOS versions, as shown in Figure 49. Typically, C1 = C2 = 30pF when the feedback element is a quartz, and C1 = C2 = 47pF when a ceramic resonator is used.

To drive the CHMOS parts with an external clock source, apply the external clock signal to XTAL1, and leave XTAL2 float, as shown in Figure 50.

The reason for this change from the way the HMOS part is driven can be seen by comparing Figures 46 and 48. In the HMOS devices the internal timing circuits are driven by the signal at XTZL2. In the CHMOS devices the internal timing circuits are driven by the signal at XTAL1.

INTERNAL TIMING

Figures 51 through 54 show when the various strobe and port signals are clocked internally. The figures do not show rise and fall times of the signals, nor do they show propagation delays between the XTAL2 signal and events at other pins.

Rise and fall times are dependent on the external loading that each pin must drive. They are often taken to be something in the neighborhood of 10nsec, measured between 0.8V and 2.0V.

Propagation delays are different for different pins. For a given pin they vary with pin loading, temperature, V_{CC}, and manufacturing lot. If the XTAL2 waveform is taken as the timing reference, prop delays may vary from 25 to 125nsec.

The AC Timings section of the data sheets do not reference any timing to the XTAL2 waveform. Rather, they relate the critical edges of control and input signals to each other. The timings published in the data sheets include the effects of propagation delays under the specified test conditions.

8051 PIN DESCRIPTIONS

ALE/PROG: Address Latch Enable output pulse for latching the low byte of the address during accesses to external memory. ALE is emitted at a constant rate of 1/6 of the oscillator frequency, for external timing or clocking purposes, even when there are no accesses to external memory. (However, one ALE pulse is skipped during each access to external Data Memory.) This pin is also the program pulse (PROG during EPROM programming).

$\overline{\text{PSEN}}$: Program Store Enable is the read strobe to external Program Memory. When the device is executing out of external Program Memory, $\overline{\text{PSEN}}$ is activated twice each machine cycle (except that two $\overline{\text{PSEN}}$ activations are skipped during accesses to external Data Memory). $\overline{\text{PSEN}}$ is not activated when the device is executing out of internal Program Memory.

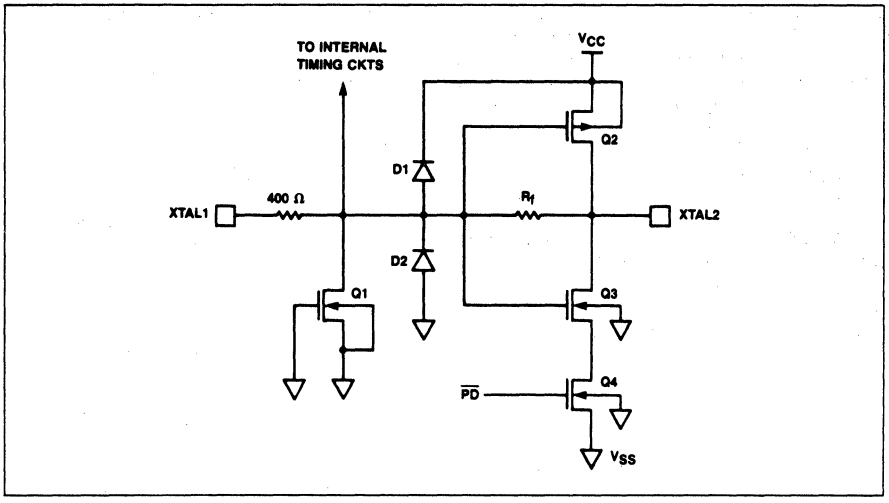


Figure 48. On-Chip Oscillator Circuitry in the CHMOS Versions of the 8051 Family

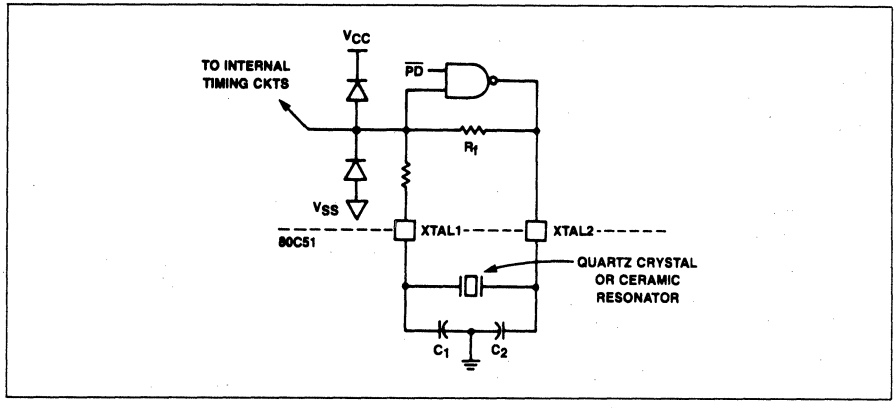


Figure 49. Using the CHMOS On-Chip Oscillator

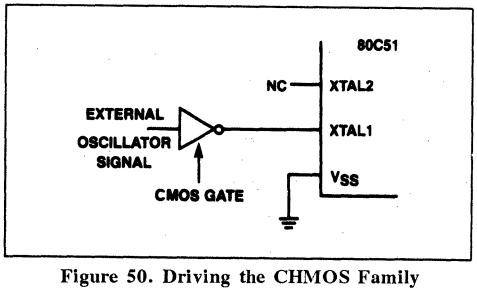


Figure 50. Driving the CHMOS Family Parts with an External Clock Source

$\overline{EA}$/VPP: When $\overline{EA}$ is held high the CPU executes out of internal Program Memory (unless the Program Counter exceeds 0FFFH in the 8051 or 80C51). Holding $\overline{EA}$ low forces the CPU to execute out of external memory regardless of the Program Counter value. In the 8031AH, $\overline{EA}$ must be externally wired low. In the EPROM devices, this pin also receives the programming supply voltage (VPP) during EPROM programming.

XTAL1: Input to the inverting oscillator amplifier.

XTAL2: Output from the inverting oscillator amplifier.

Port 0: Port 0 is an 8-bit open drain bidirectional port. As an open drain output port, it can sink eight LS TTL loads. Port 0 pins that have 1s written to them float, and in that state will function as high impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external memory. In this application it uses strong internal pullups when emitting 1s. Port 0 also emits code bytes during program verification. In that application, external pullups are required.

Port 1: Port 1 is an 8-bit bi-directional I/O port with internal pullups. Port 1 pins that have 1s written to them are pulled high by the internal pullups, and in that state can be used as inputs. As inputs, port 1 pins that are externally being pulled low will source current because of the internal pullups.

Port 2: Port 2 is an 8-bit bidirectional I/O port with internal pullups. Port 2 emits the high-order address byte during accesses to external memory that use 16-bit addresses. In this application, it uses the strong internal pullups when emitting 1s.

Port 3: Port 3 is an 8-bit bidirectional I/O port with internal pullups. It also serves the functions of various special features of the 8051 Family as follows:

Port Pin	Alternate Function
P3.0	RxD (serial input port)
P3.1	TxD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

V_{CC}: Supply voltage

V_{SS}: Circuit ground potential

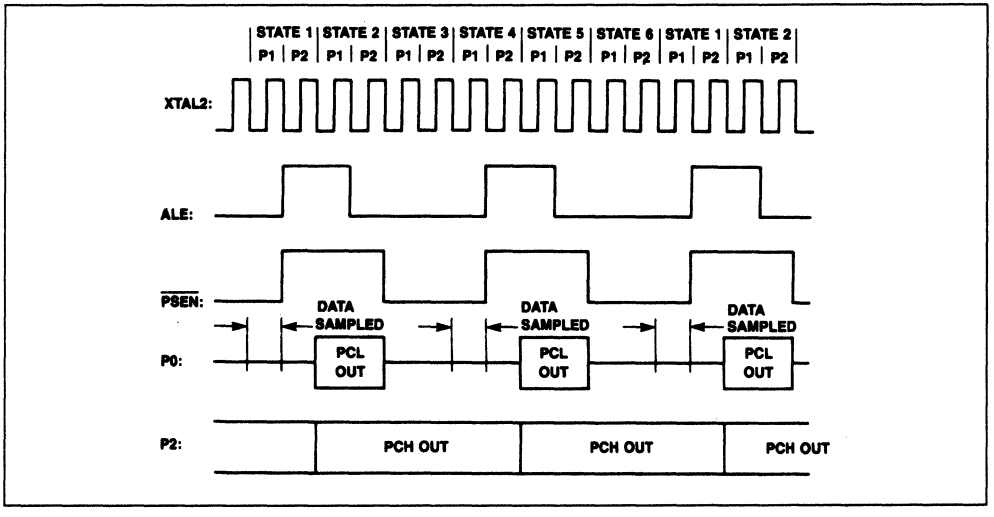


Figure 51. External Program Memory Fetches

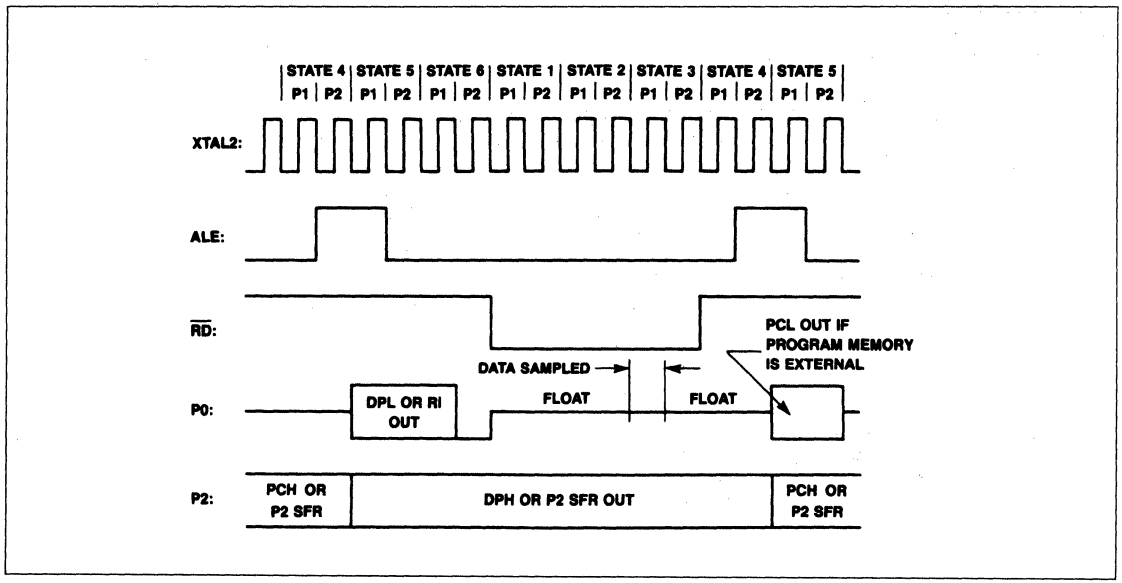


Figure 52. External Data Memory Read Cycle

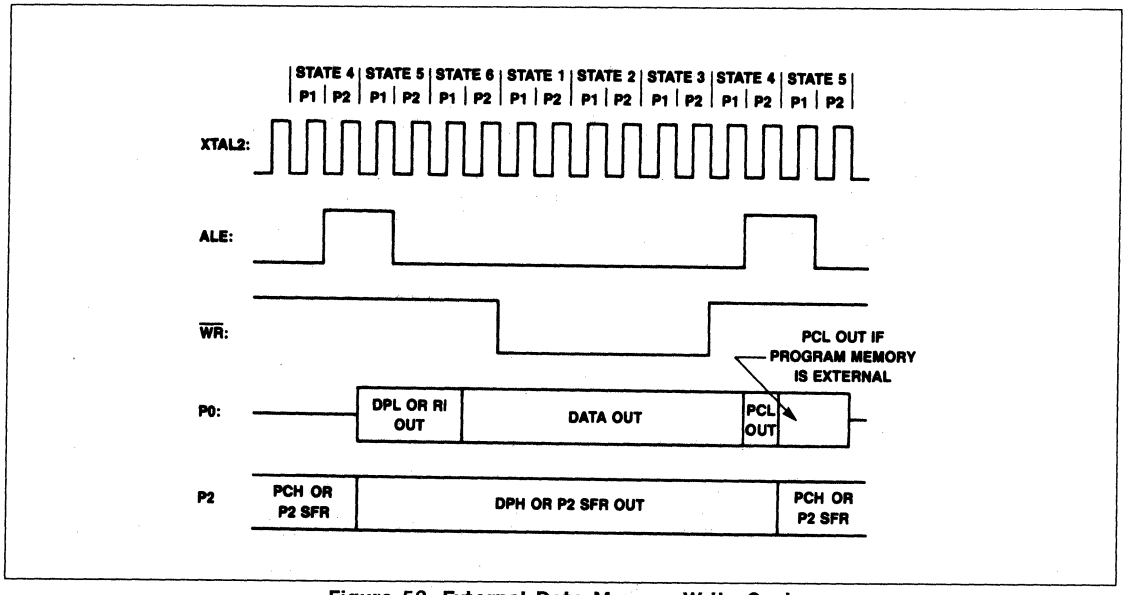


Figure 53. External Data Memory Write Cycle

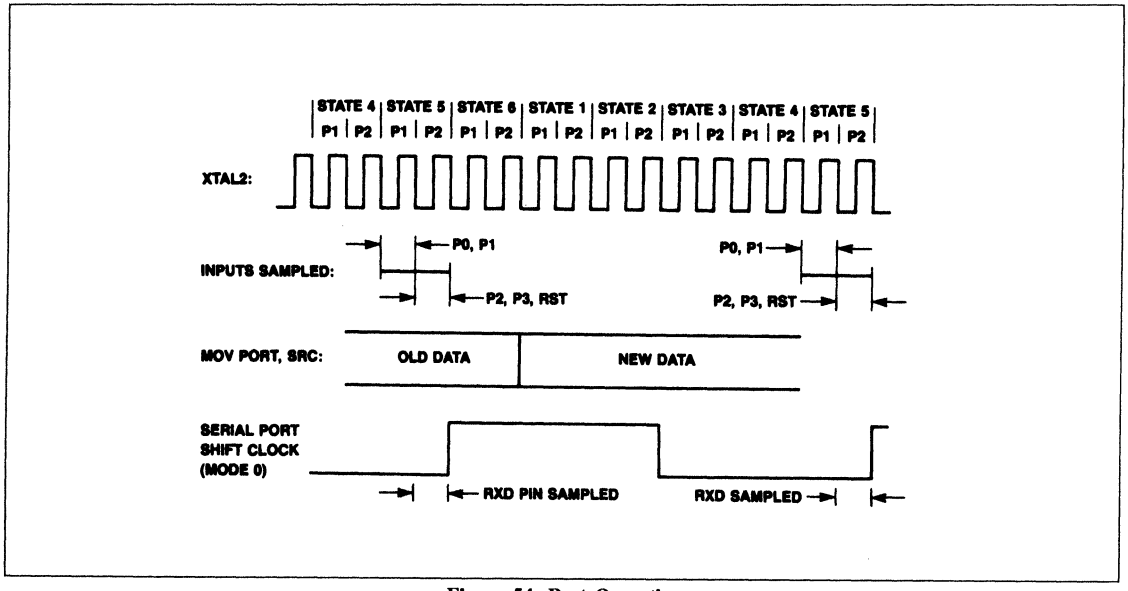


Figure 54. Port Operation

PROGRAMMER'S GUIDE AND INSTRUCTION SET

MEMORY ORGANIZATION

PROGRAM MEMORY

The 8051 has separate address spaces for program and data memory. The Program memory can be up to 64K bytes long. The lower 4K can reside on-chip. Figure 55 shows a map of the 8051 program memory.

The 8051 can address up to 64K bytes of data memory to the chip. The MOVX instruction is used to access the external data memory.

The 8051 has 128 bytes of on-chip RAM, plus a number of Special Function Registers (SFRs). The lower 128 bytes of RAM can be accessed either by direct addressing (MOV data addr) or by indirect addressing (MOV @Ri). Figure 56 shows the Data Memory organization.

DIRECT AND INDIRECT ADDRESS AREA

The 128 bytes of RAM which can be accessed by both direct and indirect addressing can be divided into three segments as listed below and shown in Figure 57.

1. Register Banks 0–3: Locations 0 through 1FH (32 bytes). The device after reset defaults to register bank 0. To use the other register banks, the user must select them in software. Each register bank contains eight 1-byte registers 0 through 7. Reset initializes the stack pointer to location 07H and it is incremented once to start from location 08H which is the first register (R0) of the second register bank. Thus in order to use more than one register bank, the SP should be initialized to a different location of the RAM where it is not used for data storage. (i.e. the higher part of the RAM).
2. Bit Addressable Area: 16 bytes have been assigned for this segment, 20H–2FH. Each one of the 128 bits of this segment can be directly addressed (0–7FH). The bits can be referred to in two ways both of which are acceptable by most assemblers. One way is to refer to their address (i.e. 0–7FH). The other way is with reference to bytes 20H to 2FH. Thus, bits 0–7 can also be referred to as bits 20.0–20.7, and bits 8–FH are the same as 21.0–21.7, and so on. Each of the 16 bytes in this segment can also be addressed as a byte.
3. Scratch Pad Area: 30H through 7FH are available to the user as data RAM. However, if the data pointer has been initialized to this area, enough bytes should be left aside to prevent SP data destruction.

Figure 56 shows the different segments of the on-chip RAM.

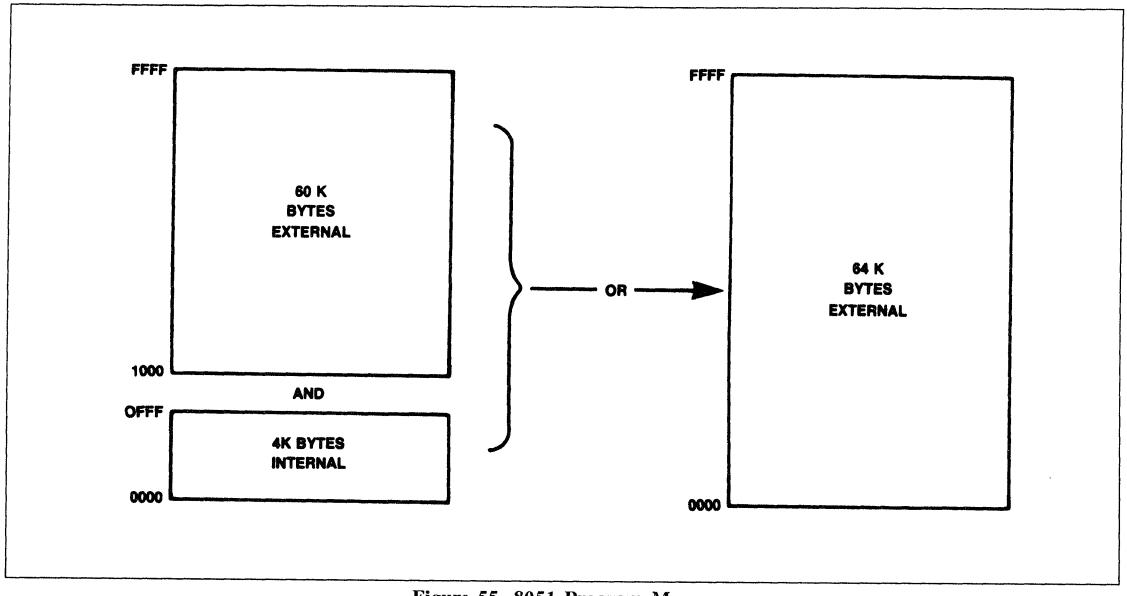


Figure 55. 8051 Program Memory

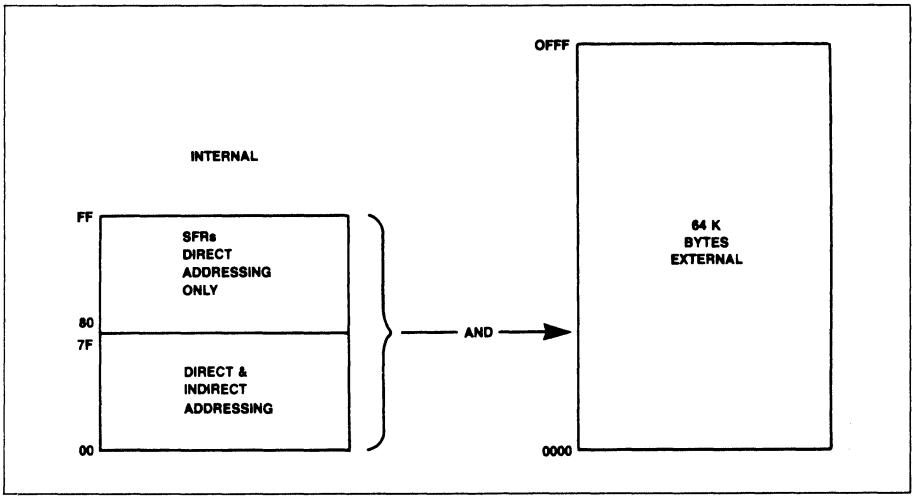


Figure 56. 8051 Data Memory

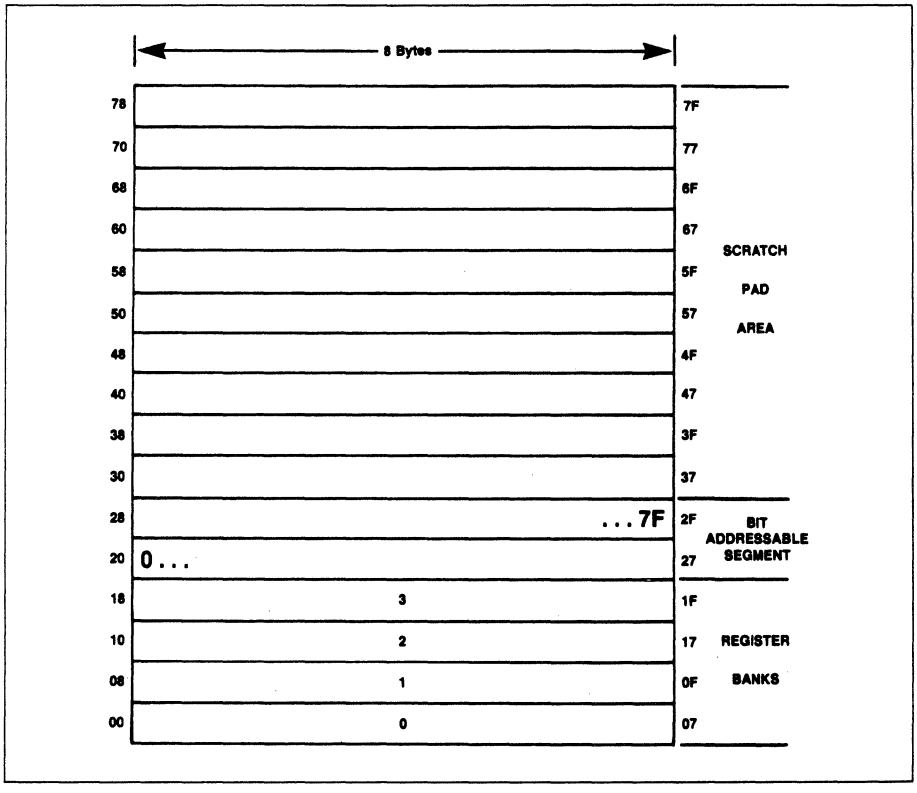


Figure 57. 128 Bytes of RAM Direct and Indirect Addressable

Table 12. 8051, 80C51 Special Function Registers

Symbol	Name	Address
*ACC	Accumulator	0E0H
*B	B Register	0F0H
*PSW	Program Status Word	0D0H
SP	Stack Pointer	81H
DPTR	Data Pointer 2 Bytes	
DPL	Low Byte	82H
DPH	High Byte	83H
*P0	Port 0	80H
*P1	Port 1	90H
*P2	Port 2	0A0H
*P3	Port 3	0B0H
*IP	Interrupt Priority Control	0B8H
*IE	Interrupt Enable Control	0A8H
TMOD	Timer/counter Mode Control	89H
*TCON	Timer/counter Control	88H
TH0	Timer/counter 0 High Byte	8CH
TL0	Timer/counter 0 Low Byte	8AH
TH1	Timer/counter 1 High Byte	8DH
TL1	Timer/counter 1 Low Byte	8BH
*SCON	Serial Control	98H
SBUF	Serial Data Buffer	99H
PCON	Power Control	87H

* = Bit addressable

Table 13. Contents of SFRs After a Reset

Register	Value in Binary
*ACC	00000000
*B	00000000
*PSW	00000000
SP	00000111
DPTR	
DPL	00000000
DPH	00000000
*P0	11111111
*P1	11111111
*P2	11111111
*P3	11111111
*IP	xxx00000
*IE	0xx00000
TMOD	00000000
*TCON	00000000
TH0	00000000
TL0	00000000
TH1	00000000
TL1	00000000
*SCON	00000000
SBUF	Indeterminate
PCON	HMOS 0xxxxxxx CHMOS 0xxx0000

X = Undefined
* = Bit addressable

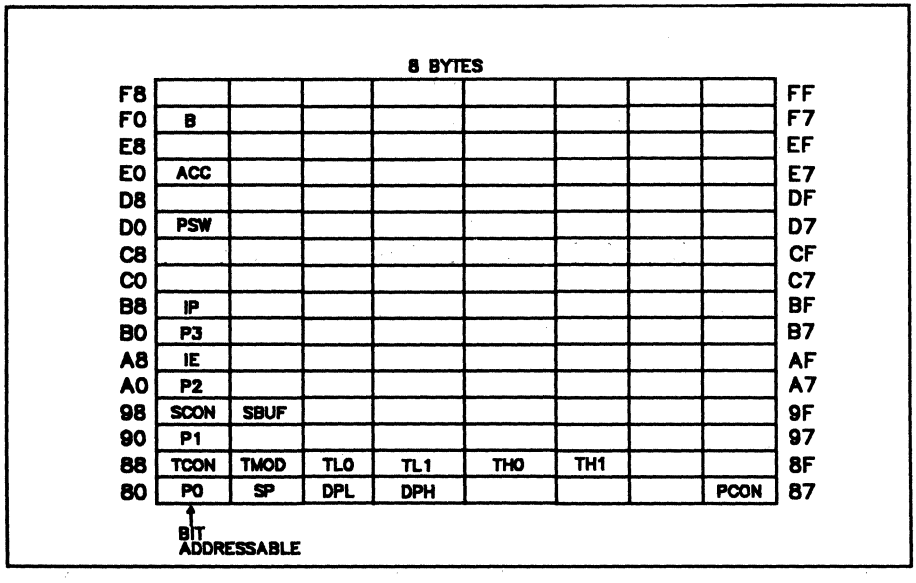


Figure 58. 8051 SFR Memory Map

Those SFRs that have their bits assigned for various functions are listed in this section. A brief description of each bit is provided for quick reference. For more detailed information refer to the Architecture Chapter of this book.

PSW: PROGRAM STATUS WORD. BIT ADDRESSABLE.

CY	AC	F0	RS1	RS0	OV	—	P

CY PSW.7 Carry Flag.

AC PSW.6 Auxiliary Carry Flag.

F0 PSW.5 Flag 0 available to the user for general purpose.

RS1 PSW.4 Register Bank selector bit 1 (SEE NOTE 1).

RS0 PSW.3 Register Bank selector bit 0 (SEE NOTE 1).

OV PSW.2 Overflow Flag.

— PSW.1 Usable as a general purpose flag.

P PSW.0 Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of '1' bits in the accumulator.

NOTE:
1. The value presented by RS0 and RS1 selects the corresponding register bank.

RS1	RS0	Register Bank	Address
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

PCON: POWER CONTROL REGISTER. NOT BIT ADDRESSABLE.

SMOD	—	—	—	GF1	GF0	PD	IDL

SMOD Double baud rate bit. If Timer 1 is used to generate baud rate and SMOD = 1, the baud rate is doubled when the Serial Port is used in modes 1, 2, or 3.

— Not implemented, reserved for future use.*

— Not implemented, reserved for future use.*

— Not implemented, reserved for future use.*

GF1 General purpose flag bit.

GF0 General purpose flag bit.

PD Power Down bit. Setting this bit activates Power Down operation in the 80C51BH. (Available only in CHMOS).

IDL Idle Mode bit. Setting this bit activates Idle Mode operation in the 80C51BH. (Available only in CHMOS).

If 1s are written to PD and IDL at the same time, PD takes precedence.

*User software should not write 1s to reserved bits. These bits may be used in future 8051 products to invoke new features.

INTERRUPTS:

In order to use any of the interrupts in the 8051 Family, the following three steps must be taken.

1. Set the EA (enable all) bit in the IE register to 1.
2. Set the corresponding individual interrupt enable bit in the IE register to 1.
3. Begin the interrupt service routine at the corresponding Vector Address of that interrupt. See Table below.

Interrupt Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI & TI	0023H

In addition, for external interrupts, pins $\overline{INT0}$ and $\overline{INT1}$ (P3.2 and P3.3) must be set to 1, and depending on whether the interrupt is to be level or transition activated, bits IT0 or IT1 in the TCON register may need to be set to 1.

ITx = 0 level activated

ITx = 1 transition activated

IE: INTERRUPT ENABLE REGISTER. BIT ADDRESSABLE.

If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled.

EA	—	—	ES	ET1	EX1	ET0	EX0

EA	IE.7	Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
—	IE.6	Not implemented, reserved for future use.*
—	IE.5	Not implemented, reserved for future use.*
ES	IE.4	Enable or disable the serial port interrupt.
ET1	IE.3	Enable or disable the Timer 1 overflow interrupt.
EX1	IE.2	Enable or disable External Interrupt 1.
ET0	IE.1	Enable or disable the Timer 0 overflow interrupt.
EX0	IE.0	Enable or disable External Interrupt 0.

*User software should not write 1s to reserved bits. These bits may be used in future 8051 products to invoke new features.

ASSIGNING HIGHER PRIORITY TO ONE OR MORE INTERRUPTS:

In order to assign higher priority to an interrupt the corresponding bit in the IP register must be set to 1.

Remember that while an interrupt service is in progress, it cannot be interrupted by a lower or same level interrupt.

PRIORITY WITHIN LEVEL:

Priority within level is only to resolve simultaneous requests of the same priority level.

From high to low, interrupt sources are listed below:

IE0
TF0
IE1
TF1
RI or TI

IP: INTERRUPT PRIORITY REGISTER. BIT ADDRESSABLE.

If the bit is 0, the corresponding interrupt has a lower priority and if the bit is 1 the corresponding interrupt has a higher priority.

—	—	—	PS	PT1	PX1	PT0	PX0

— IP. 7 Not implemented, reserved for future use.*
— IP. 6 Not implemented, reserved for future use.*
— IP. 5 Not implemented, reserved for future use.*
PS IP. 4 Defines the Serial Port interrupt priority level.
PT1 IP. 3 Defines the Timer 1 interrupt priority level.
PX1 IP. 2 Defines External Interrupt 1 priority level.
PT0 IP. 1 Defines the Timer 0 interrupt priority level.
PX0 IP. 0 Defines the External Interrupt 0 priority level.

*User software should not write 1s to reserved bits. These bits may be used in future 8051 products to invoke new features.

TCON: TIMER/COUNTER CONTROL REGISTER. BIT ADDRESSABLE.

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1 TCON. 7 Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows. Cleared by hardware as processor vectors to the interrupt service routine.

TR1 TCON. 6 Timer 1 run control bit. Set/cleared by software to turn Timer/Counter 1 ON/OFF.

TF0 TCON. 5 Timer 0 overflow flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as processor vectors to the service routine.

TR0 TCON. 4 Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF.

IE1 TCON. 3 External Interrupt 1 edge flag. Set by hardware when External Interrupt edge is detected. Cleared by hardware when interrupt is processed.

IT1 TCON. 2 Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.

IE0 TCON. 1 External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed.

IT0 TCON. 0 Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.

TMOD: TIMER/COUNTER MODE CONTROL REGISTER. NOT BIT ADDRESSABLE.

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
TIMER 1				TIMER 0			

GATE When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control).

C/$\overline{T}$ Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).

M1 Mode selector bit. (NOTE 1)

M0 Mode selector bit. (NOTE 1)

NOTE 1:

M1	M0	Operating Mode	
0	0	0	13-bit Timer (8048 compatible)
0	1	1	16-bit Timer/Counter
1	0	2	8-bit Auto-Reload Timer/Counter
1	1	3	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits, TH0 is an 8-bit Timer and is controlled by Timer 1 control bits.
1	1	3	(Timer 1) Timer/Counter 1 stopped.

TIMER SET-UP

Tables 14 through 17 give some values for TMOD which can be used to set up Timer 0 in different modes.

It is assumed that only one timer is being used at a time. If it is desired to run Timers 0 and 1 simultaneously, in any mode, the value in TMOD for Timer 0 must be ORed with the value shown for Timer 1 (Table 16 and 17).

For example, if it is desired to run Timer 0 in mode 1 GATE (external control), and Timer 1 in mode 2 COUNTER, then the value that must be loaded into TMOD is 69H (09H from Table 14 ORed with 60H from Table 17).

Moreover, it is assumed that the user, at this point, is not ready to turn the timers on and will do that at a different point in the program by setting bit TRx (in TCON) to 1.

TIMER/COUNTER 0

As a Timer:

Table 14

MODE	TIMER 0 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	00H	08H
1	16-bit Timer	01H	09H
2	8-bit Auto-Reload	02H	0AH
3	two 8-bit Timers	03H	0BH

As a Counter:

Table 15

MODE	COUNTER 0 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	04H	0CH
1	16-bit Timer	05H	0DH
2	8-bit Auto-Reload	06H	0EH
3	one 8-bit Counter	07H	0FH

NOTES:
1. The Timer is turned ON/OFF by setting/clearing bit TR0 in the software.
2. The Timer is turned ON/OFF by the 1 to 0 transition on $\overline{INT0}$ (P3.2) when TR0 = 1 (hardware control).

TIMER/COUNTER 1

As a Timer:

Table 16

MODE	TIMER 1 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	00H	80H
1	16-bit Timer	10H	90H
2	8-bit Auto-Reload	20H	A0H
3	does not run	30H	B0H

As a Counter:

Table 17

MODE	COUNTER 1 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	40H	C0H
1	16-bit Timer	50H	D0H
2	8-bit Auto-Reload	60H	E0H
3	not available	—	—

NOTES:
1. The Timer is turned ON/OFF by setting/clearing bit TR1 in the software.
2. The Timer is turned ON/OFF by the 1 to 0 transition on $\overline{INT1}$ (P3.3) when TR1 = 1 (hardware control).

SCON: (S0CON IN THE 83C652 AND 83C552) SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0 SCON. 7 Serial Port mode specifier. (NOTE 1).

SM1 SCON. 6 Serial Port mode specifier. (NOTE 1).

SM2 SCON. 5 Enables the multiprocessor communication feature in modes 2 & 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0. (See Table 18).

REN SCON. 4 Set/Cleared by software to Enable/Disable reception.

TB8 SCON. 3 The 9th bit that will be transmitted in modes 2 & 3. Set/Cleared by software.

RB8 SCON. 2 In modes 2 & 3, is 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 SCON. 1 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. Must be cleared by software.

RI SCON. 0 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 (except see SM2). Must be cleared by software.

NOTE 1:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	SHIFT REGISTER	Fosc./12
0	1	1	8-Bit UART	Variable
1	0	2	9-Bit UART	Fosc./64 OR Fosc./32
1	1	3	9-Bit UART	Variable

SERIAL PORT SET-UP:

Table 18

MODE	SCON	SM2 VARIATION
0 1 2 3	10H 50H 90H D0H	Single Processor Environment (SM2 = 0)
0 1 2 3	NA 70H B0H F0H	Multiprocessor Environment (SM2 = 1)

GENERATING BAUD RATES

Serial Port in Mode 0:

Mode 0 has a fixed baud rate which is 1/12 of the oscillator frequency. To run the serial port in this mode none of the Timer/Counters need to be set up. Only the SCON register needs to be defined.

$$\text{Baud Rate} = \frac{\text{Osc Freq}}{12}$$

Serial Port in Mode 1:

Mode 1 has a variable baud rate. The baud rate is generated by Timer 1.

USING TIMER/COUNTER 1 TO GENERATE BAUD RATES:

For this purpose, Timer 1 is used in mode 2 (Auto-Reload). Refer to Timer Setup section of this chapter.

$$\text{Baud Rate} = \frac{K \times \text{Oscillator Freq.}}{32 \times 12 \times [256 - (TH1)]}$$

If SMOD = 0, then K = 1.
If SMOD = 1, then K = 2. (SMOD is the PCON register).

Most of the time the user knows the baud rate and needs to know the reload value for TH1.
Therefore, the equation to calculate TH1 can be written as:

$$TH1 = 256 - \frac{K \times \text{Osc Freq.}}{384 \times \text{baud rate}}$$

TH1 must be an integer value. Rounding off TH1 to the nearest integer may not produce the desired baud rate. In this case, the user may have to choose another crystal frequency.

Since the PCON register is not bit addressable, one way to set the bit is logical ORing the PCON register. (ie, ORL PCON, #80H). The address of PCON is 87H.

SERIAL PORT IN MODE 2:

The baud rate is fixed in this mode and is $1/32$ or $1/64$ of the oscillator frequency depending on the value of the SMOD bit in the PCON register.

In this mode none of the Timers are used and the clock comes from the internal phase 2 clock.

SMOD = 1, Baud Rate = $1/32$ Osc Freq.

SMOD = 0, Baud Rate = $1/64$ Osc Freq.

To set the SMOD bit: ORL PCON, #80H. The address of PCON is 87H.

SERIAL PORT IN MODE 3:

The baud rate in mode 3 is variable and sets up exactly the same as in mode 1.

8051 FAMILY INSTRUCTION SET

Table 19. 8051 Instruction Set Summary

Interrupt Response Time: Refer to Hardware Description Chapter.

Instructions that Affect Flag Settings(1)

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	O		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	O	X		ANL C,/bit	X		
DIV	O	X		ORL C,bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

(1)Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) will also affect flag settings.

Note on instruction set and addressing modes:

Rn — Register R7–R0 of the currently selected Register Bank.

direct — 8-bit internal data location's address. This could be an Internal Data RAM location (0–127) or a SFR [i.e., I/O port, control register, status register, etc. (128–255)].

@Ri — 8-bit internal data RAM location (0–255) addressed indirectly through register R1 or R0.

#data — 8-bit constant included in instruction.

#data 16 — 16-bit constant included in instruction.

addr 16 — 16-bit destination address. Used by LCALL & LJMP. A branch can be anywhere within the 64K-byte Program Memory address space.

addr 11 — 11-bit destination address. Used by ACALL & AJMP. The branch will be within the same 2K-byte page of program memory as the first byte of the following instruction.

rel — Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is −128 to +127 bytes relative to first byte of the following instruction.

bit — Direct Addressed bit in Internal Data RAM or Special Function Register.

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS				
ADD	A,Rn	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@Ri	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,Rn	Add register to Accumulator with Carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with Carry	2	12
ADDC	A,@Ri	Add indirect RAM to Accumulator with Carry	1	12
ADDC	A,#data	Add immediate data to Acc with Carry	2	12
SUBB	A,Rn	Subtract Register from Acc with borrow	1	12
SUBB	A,direct	Subtract direct byte from Acc with borrow	2	12
SUBB	A,@Ri	Subtract indirect RAM from ACC with borrow	1	12
SUBB	A,#data	Subtract immediate data from Acc with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	Rn	Increment register	1	12
INC	direct	Increment direct byte	2	12
INC	@Ri	Increment indirect RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	Rn	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@Ri	Decrement indirect RAM	1	12

Table 19. 8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS (Continued)				
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A & B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12
LOGICAL OPERATIONS				
ANL	A,Rn	AND Register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@Ri	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,Rn	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@Ri	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,Rn	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12

Mnemonic		Description	Byte	Oscillator Period
LOGICAL OPERATIONS (Continued)				
RL	A	Rotate Accumulator Left	1	12
RLC	A	Rotate Accumulator Left through the Carry	1	12
RR	A	Rotate Accumulator Right	1	12
RRC	A	Rotate Accumulator Right through the Carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12
DATA TRANSFER				
MOV	A,Rn	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@Ri	Move indirect RAM to Accumulator	1	12
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	Rn,A	Move Accumulator to register	1	12
MOV	Rn,direct	Move direct byte to register	2	24
MOV	Rn,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,Rn	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@Ri	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@Ri,A	Move Accumulator to indirect RAM	1	12

Table 19. 8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
DATA TRANSFER (Continued)				
MOV	@Ri,direct	Move direct byte to indirect RAM	2	24
MOV	@Ri,#data	Move immediate data to indirect RAM	2	12
MOV	DPTR,#data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to Acc	1	24
MOVC	A,@A+PC	Move Code byte relative to PC to Acc	1	24
MOVX	A,@Ri	Move External RAM (8-bit addr) to Acc	1	24
MOVX	A,@DPTR	Move External RAM (16-bit addr) to Acc	1	24
MOVX	@Ri,A	Move Acc to External RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,Rn	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@Ri	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@Ri	Exchange low-order Digit indirect RAM with Acc	1	12
BOOLEAN VARIABLE MANIPULATION				
CLR	C	Clear Carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set Carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement Carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to CARRY	2	24
ANL	C,/bit	AND complement of direct bit to Carry	2	24
ORL	C,bit	OR direct bit to Carry	2	24
ORL	C,/bit	OR complement of direct bit to Carry	2	24
MOV	C,bit	Move direct bit to Carry	2	12
MOV	bit,C	Move Carry to direct bit	2	24
JC	rel	Jump if Carry is set	2	24
JNC	rel	Jump if Carry not set	2	24
JB	bit,rel	Jump if direct Bit is set	3	24
JNB	bit,rel	Jump if direct Bit is Not set	3	24
JBC	bit,rel	Jump if direct Bit is set & clear bit	3	24
PROGRAM BRANCHING				
ACALL	addr11	Absolute Subroutine Call	2	24
LCALL	addr16	Long Subroutine Call	3	24
RET		Return from Subroutine	1	24
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute Jump	2	24
LJMP	addr16	Long Jump	3	24
SJMP	rel	Short Jump (relative addr)	2	24

Table 19. 8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
PROGRAM BRANCHING (Continued)				
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is Zero	2	24
JNZ	rel	Jump if Accumulator is Not Zero	2	24
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3	24
CJNE	A,#data,rel	Compare immediate to Acc and Jump if Not Equal	3	24
CJNE	Rn,#data,rel	Compare immediate to register and Jump if Not Equal	3	24
CJNE	@Ri,#data,rel	Compare immediate to indirect and Jump if Not Equal	3	24
DJNZ	Rn,rel	Decrement register and Jump if Not Zero	2	24
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	24
NOP		No Operation	1	12

INSTRUCTION DEFINITIONS

ACALL addr11

Function: Absolute Call

Description: ACALL unconditionally calls a subroutine located at the indicated address. The instruction increments the PC twice to obtain the address of the following instruction, then pushes the 16-bit result onto the stack (low-order byte first) and increments the Stack Pointer twice. The destination address is obtained by successively concatenating the five high-order bits of the incremented PC, opcode bits 7-5, and the second byte of the instruction. The subroutine called must therefore start within the same 2K block of the program memory as the first byte of the instruction following ACALL. No flags are affected.

Example: Initially SP equals 07H. The label "SUBRTN" is at program memory location 0345 H. After executing the instruction,

ACALL SUBRTN

at location 0123H, SP will contain 09H, internal RAM locations 08H and 09H will contain 25H and 01H, respectively, and the PC will contain 0345H.

Bytes: 2

Cycles: 2

Encoding:

a10 a9 a8 1	0 0 0 1

a7 a6 a5 a4	a3 a2 a1 a0

Operation:
ACALL
$(PC) \leftarrow (PC) + 2$
$(SP) \leftarrow (SP) + 1$
$((SP)) \leftarrow (PC_{7\text{-}0})$
$(SP) \leftarrow (SP) + 1$
$((SP)) \leftarrow (PC_{15\text{-}8})$
$(PC_{10\text{-}0}) \leftarrow$ page address

ADD A,<src-byte>

Function: Add

Description: ADD adds the byte variable indicated to the Accumulator, leaving the result in the Accumulator. The carry and auxiliary-carry flags are set, respectively, if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates an overflow occured.

OV is set if there is a carry-out of bit 6 but not out of bit 7, or a carry-out of bit 7 but not bit 6; otherwise OV is cleared. When adding signed integers, OV indicates a negative number produced as the sum of two positive operands, or a positive sum from two negative operands.

Four source operand addressing modes are allowed: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B). The instruction,

ADD A,R0

will leave 6DH (01101101B) in the Accumulator with the AC flag cleared and both the carry flag and OV set to 1.

ADD A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 1 0	1 r r r

Operation: ADD
(A) ← (A) + (Rn)

ADD A,direct

Bytes: 2

Cycles: 1

Encoding:

0 0 1 0	0 1 0 1	direct address

Operation: ADD
(A) ← (A) + (direct)

ADD A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 1 0	0 1 1 i

Operation: ADD
$(A) \leftarrow (A) + ((R_i))$

ADD A,#data

Bytes: 2

Cycles: 1

Encoding:

0 0 1 0	0 1 0 0	immediate data

Operation: ADD
$(A) \leftarrow (A) + \#data$

ADDC A,<src-byte>

Function: Add with Carry

Description: ADDC simultaneously adds the byte variable indicated, the carry flag and the Accumulator contents, leaving the result in the Accumulator. The carry and auxiliary-carry flags are set, respectively, if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates an overflow occured.

OV is set if there is a carry-out of bit 6 but not out of bit 7, or a carry-out of bit 7 but not out of bit 6; otherwise OV is cleared. When adding signed integers, OV indicates a negative number produced as the sum of two positive operands or a positive sum from two negative operands.

Four source operand addressing modes are allowed: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B) with the carry flag set. The instruction,

ADDC A,R0

will leave 6EH (01101110B) in the Accumulator with AC cleared and both the Carry flag and OV set to 1.

ADDC A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 1 1	1 r r r

Operation: ADDC
$(A) \leftarrow (A) + (C) + (R_n)$

ADDC A,direct

Bytes: 2

Cycles: 1

Encoding:

0 0 1 1	0 1 0 1	direct address

Operation: ADDC
$(A) \leftarrow (A) + (C) + (direct)$

ADDC A,@RI

Bytes: 1

Cycles: 1

Encoding:

0 0 1 1	0 1 1 i

Operation: ADDC
$(A) \leftarrow (A) + (C) + ((R_i))$

ADDC A,#data

Bytes: 2

Cycles: 1

Encoding:

0 0 1 1	0 1 0 0	immediate data

Operation: ADDC
$(A) \leftarrow (A) + (C) + \#data$

AJMP addr11

Function: Absolute Jump

Description: AJMP transfers program execution to the indicated address, which is formed at run-time by concatenating the high-order five bits of the PC (*after* incrementing the PC twice), opcode bits 7-5, and the second byte of the instruction. The destination must therefore be within the same 2K block of program memory as the first byte of the instruction following AJMP.

Example: The label "JMPADR" is at program memory location 0123H. The instruction,

AJMP JMPADR

is at location 0345H and will load the PC with 0123H.

Bytes: 2

Cycles: 2

Encoding:

a10 a9 a8 0	0 0 0 1

a7 a6 a5 a4	a3 a2 a1 a0

Operation: AJMP
(PC) ← (PC) + 2
($PC_{10\text{-}0}$) ← page address

ANL <dest-byte>,<src-byte>

Function: Logical-AND for byte variables

Description: ANL performs the bitwise logical-AND operation between the variables indicated and stores the results in the destination variable. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (11000011B) and register 0 holds 55H (01010101B) then the instruction,

ANL A,R0

will leave 41H (01000001B) in the Accumulator.

When the destination is a directly addressed byte, this instruction will clear combinations of bits in any RAM location or hardware register. The mask byte determining the pattern of bits to be cleared would either be a constant contained in the instruction or a value computed in the Accumulator at run-time. The instruction,

ANL P1,#01110011B

will clear bits 7, 3, and 2 of output port 1.

ANL A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 1 0 1	1 r r r

Operation: ANL
(A) ← (A) ∧ (Rn)

ANL A,direct

Bytes: 2

Cycles: 1

Encoding:

0 1 0 1	0 1 0 1	direct address

Operation: ANL

(A) ← (A) ∧ (direct)

ANL A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 1 0 1	0 1 1 i

Operation: ANL

(A) ← (A) ∧ ((Ri))

ANL A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 0 1	0 1 0 0	immediate data

Operation: ANL
(A) ← (A) ∧ #data

ANL direct,A

Bytes: 2

Cycles: 1

Encoding:

0 1 0 1	0 0 1 0	direct address

Operation: ANL
(direct) ← (direct) ∧ (A)

ANL direct,#data

Bytes: 3

Cycles: 2

Encoding:

0 1 0 1	0 0 1 1	direct address	immediate data

Operation: ANL
(direct) ← (direct) ∧ #data

ANL C,<src-bit>

Function: Logical-AND for bit variables

Description: If the Boolean value of the source bit is a logical 0 then clear the carry flag; otherwise leave the carry flag in its current state. A slash ("/") preceding the operand in the assembly language indicates that the logical complement of the addressed bit is used as the source value, *but the source bit itself is not affected.* No other flags are affected.

Only direct addressing is allowed for the source operand.

Example: Set the carry flag if, and only if, P1.0 = 1, ACC. 7 = 1, and OV = 0:

```
MOV  C,P1.0     ;LOAD CARRY WITH INPUT PIN STATE
ANL  C,ACC.7    ;AND CARRY WITH ACCUM. BIT 7
ANL  C,/OV      ;AND WITH INVERSE OF OVERFLOW FLAG
```

ANL C,bit

Bytes: 2

Cycles: 2

Encoding:

1 0 0 0	0 0 1 0	bit address

Operation: ANL
(C) ← (C) ∧ (bit)

ANL C,/bit

Bytes: 2

Cycles: 2

Encoding:

1 0 1 1	0 0 0 0	bit address

Operation: ANL
(C) ← (C) ∧ ¬(bit)

CJNE <dest-byte>,<src-byte>, rel

Function: Compare and Jump if Not Equal.

Description: CJNE compares the magnitudes of the first two operands, and branches if their values are not equal. The branch destination is computed by adding the signed relative-displacement in the last instruction byte to the PC, after incrementing the PC to the start of the next instruction. The carry flag is set if the unsigned integer value of <dest-byte> is less than the unsigned integer value of <src-byte>; otherwise, the carry is cleared. Neither operand is affected.

The first two operands allow four addressing mode combinations: the Accumulator may be compared with any directly addressed byte or immediate data, and any indirect RAM location or working register can be compared with an immediate constant.

Example: The Accumulator contains 34H. Register 7 contains 56H. The first instruction in the sequence,

```
                CJNE   R7,#60H, NOT_EQ
;               ...    .....            ; R7 = 60H.
NOT_EQ:         JC     REQ_LOW          ; IF R7 < 60H.
;               ...    .....            ; R7 > 60H.
```

sets the carry flag and branches to the instruction at label NOT_EQ. By testing the carry flag, this instruction determines whether R7 is greater or less than 60H.

If the data being presented to Port 1 is also 34H, then the instruction,

```
WAIT:  CJNE  A,P1,WAIT
```

clears the carry flag and continues with the next instruction in sequence, since the Accumulator does equal the data read from P1. (If some other value was being input on P1, the program will loop at this point until the P1 data changes to 34H.)

CJNE A,direct,rel

Bytes: 3

Cycles: 2

Encoding:

1 0 1 1	0 1 0 1	direct address	rel. address

Operation:

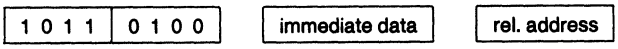
```
(PC) ← (PC) + 3
IF (A) <> (direct)
THEN
        (PC) ← (PC) + relative offset

IF (A) < (direct)
THEN
        (C) ← 1
ELSE
        (C) ← 0
```

CJNE A,#data,rel

Bytes: 3

Cycles: 2

Encoding:



Operation:

(PC) ← (PC) + 3
IF (A) <> *data*
THEN
(PC) ← (PC) + *relative offset*

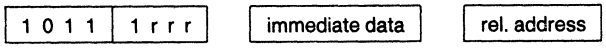
IF (A) < *data*
THEN
(C) ← 1
ELSE
(C) ← 0

CJNE Rn,#data,rel

Bytes: 3

Cycles: 2

Encoding:



Operation:

(PC) ← (PC) + 3
IF (Rn) <> *data*
THEN
(PC) ← (PC) + *relative offset*

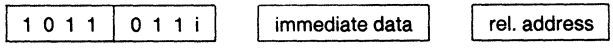
IF (Rn) < *data*
THEN
(C) ← 1
ELSE
(C) ← 0

CJNE @Ri,#data,rel

Bytes: 3

Cycles: 2

Encoding:



Operation:

(PC) ← (PC) + 3
IF ((Ri)) <> *data*
THEN
(PC) ← (PC) + *relative offset*

IF ((Ri)) < *data*
THEN
(C) ← 1
ELSE
(C) ← 0

CLR A

Function: Clear Accumulator

Description: The Accumulator is cleared (all bits reset to zero). No flags are affected.

Example: The Accumulator contains 5CH (01011100B). The instruction,

CLR A

will leave the Accumulator set to 00H (00000000B).

Bytes: 1

Cycles: 1

Encoding:

1 1 1 1	0 1 0 0

Operation:
CLR
(A) ← 0

CLR bit

Function: Clear bit

Description: The indicated bit is cleared (reset to zero). No other flags are affected. CLR can operate on the carry flag or any directly addressable bit.

Example: Port 1 has previously been written with 5DH (01011101B). The instruction,

CLR P1.2

will leave the port set to 59H (01011001B).

CLR C

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	0 0 1 1

Operation:
CLR
(C) ← 0

CLR bit

Bytes: 2

Cycles: 1

Encoding:

1 1 0 0	0 0 1 0	bit address

Operation:
CLR
(bit) ← 0

CPL A

Function: Complement Accumulator

Description: Each bit of the Accumulator is logically complemented (one's complement). Bits which previously contained a one are changed to a zero and vice-versa. No flags are affected.

Example: The Accumulator contains 5CH (01011100B). The instruction,

CPL A

will leave the Accumulator set to 0A3H (10100011B).

Bytes: 1

Cycles: 1

Encoding:

1 1 1 1	0 1 0 0

Operation:
CPL
(A) ← ¬(A)

CPL bit

Function: Complement bit

Description: The bit variable specified is complemented. A bit which had been a one is changed to zero and vice-versa. No other flags are affected. CLR can operate on the carry or any directly addressable bit.

Note: When this instruction is used to modify an output pin, the value used as the original data will be read from the output data latch, *not* the input pin.

Example: Port 1 has previously been written with 5BH (01011101B). The instruction sequence,

CPL P1.1

CPL P1.2

will leave the port set to 5BH (01011011B).

CPL C

Bytes: 1

Cycles: 1

Encoding:

1 0 1 1	0 0 1 1

Operation:
CPL
(C) ← ¬(C)

CPL bit

Bytes: 2

Cycles: 1

Encoding:

1 0 1 1	0 0 1 0	bit address

Operation: CPL
(bit) ← ¬ (bit)

DA A

Function: Decimal-adjust Accumulator for Addition

Description: DA A adjusts the eight-bit value in the Accumulator resulting from the earlier addition of two variables (each in packed-BCD format), producing two four-bit digits. Any ADD or ADDC instruction may have been used to perform the addition.

If Accumulator bits 3-0 are greater than nine (xxxx1010-xxxx1111), or if the AC flag is one, six is added to the Accumulator producing the proper BCD digit in the low-order nibble. This internal addition would set the carry flag if a carry-out of the low-order four-bit field propagated through all high-order bits, but it would not clear the carry flag otherwise.

If the carry flag is now set, or if the four high-order bits now exceed nine (1010xxxx-111xxxx), these high-order bits are incremented by six, producing the proper BCD digit in the high-order nibble. Again, this would set the carry flag if there was a carry-out of the high-order bits, but wouldn't clear the carry. The carry flag thus indicates if the sum of the original two BCD variables is greater than 100, allowing multiple precision decimal addition. OV is not affected.

All of this occurs during the one instruction cycle. Essentially, this instruction performs the decimal conversion by adding 00H, 06H, 60H, or 66H to the Accumulator, depending on initial Accumulator and PSW conditions.

Note: DA A *cannot* simply convert a hexadecimal number in the Accumulator to BCD notation, nor does DA A apply to decimal subtraction.

Example: The Accumulator holds the value 56H (01010110B) representing the packed BCD digits of the decimal number 56. Register 3 contains the value 67H (01100111B) representing the packed BCD digits of the decimal number 67. The carry flag is set. The instruction sequence.

```
ADDC  A,R3
DA    A
```

will first perform a standard twos-complement binary addition, resulting in the value 0BEH (10111110) in the Accumulator. The carry and auxiliary carry flags will be cleared.

The Decimal Adjust instruction will then alter the Accumulator to the value 24H (00100100B), indicating the packed BCD digits of the decimal number 24, the low-order two digits of the decimal sum of 56, 67, and the carry-in. The carry flag will be set by the Decimal Adjust instruction, indicating that a decimal overflow occurred. The true sum 56, 67, and 1 is 124.

BCD variables can be incremented or decremented by adding 01H or 99H. If the Accumulator initially holds 30H (representing the digits of 30 decimal), then the instruction sequence,

```
ADD   A,#99H

DA    A
```

will leave the carry set and 29H in the Accumulator, since 30 + 99 = 129. The low-order byte of the sum can be interpreted to mean 30 − 1 = 29.

Bytes: 1

Cycles: 1

Encoding:

1 1 0 1	0 1 0 0

Operation:

```
DA
-contents of Accumulator are BCD
IF    [[(A3-0) > 9] ∨ [(AC) = 1]]
      THEN(A3-0) ← (A3-0) + 6
           AND

IF    [[(A7-4) > 9] ∨ [(C) = 1]]
      THEN (A7-4) ← (A7-4) + 6
```

DEC byte

Function: Decrement

Description: The variable indicated is decremented by 1. An original value of 00H will underflow to 0FFH. No flags are affected. Four operand addressing modes are allowed: accumulator, register, direct, or register-indirect.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Register 0 contains 7FH (01111111B). Internal RAM locations 7EH and 7FH contain 00H and 40H, respectively. The instruction sequence,

```
DEC  @R0
DEC  R0
DEC  @R0
```

will leave register 0 set to 7EH and internal RAM locations 7EH and 7FH set to 0FFH and 3FH.

DEC A

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	0 1 0 0

Operation:
DEC
(A) ← (A) − 1

DEC Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	1 r r r

Operation:
DEC
(Rn) ← (Rn) − 1

DEC direct

Bytes: 2

Cycles: 1

Encoding:

0 0 0 1	0 1 0 1	direct address

Operation: DEC
(direct) ← (direct) − 1

DEC @Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	0 1 1 i

Operation: DEC
((Ri)) ← ((Ri)) − 1

DIV AB

Function: Divide

Description: DIV AB divides the unsigned eight-bit integer in the Accumulator by the unsigned eight-bit integer in register B. The Accumulator receives the integer part of the quotient; register B receives the integer remainder. The carry and OV flags will be cleared.

Exception: if B had originally contained 00H, the values returned in the Accumulator and B-register will be undefined and the overflow flag will be set. The carry flag is cleared in any case.

Example: The Accumulator contains 251 (0FBH or 11111011B) and B contains 18 (12H or 00010010B). The instruction,

DIV AB

will leave 13 in the Accumulator (0DH or 00001101B) and the value 17 (11H or 00010001B) in B, since 251 = (13 X 18) + 17. Carry and OV will both be cleared.

Bytes: 1

Cycles: 4

Encoding:

1 0 0 0	0 1 0 0

Operation: DIV
$(A)_{15\text{-}8}$ ← (A)/(B)
$(B)_{7\text{-}0}$

DJNZ <byte>,<rel-addr>

Function: Decrement and Jump if Not Zero

Description: DJNZ decrements the location indicated by 1, and branches to the address indicated by the second operand if the resulting value is not zero. An original value of 00H will underflow to 0FFH. No flags are affected. The branch destination would be computed by adding the signed relative-displacement value in the last instruction byte to the PC, after incrementing the PC to the first byte of the following instruction.

The location decremented may be a register or directly addressed byte.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Internal RAM locations 40H, 50H, and 60H contain the values 01H, 70H, and 15H, respectively. The instruction sequence,

```
DJNZ  40H,LABEL__1
DJNZ  50H,LABEL__2
DJNZ  60H,LABEL__3
```

will cause a jump to the instruction at label LABEL__2 with the values 00H, 6FH, and 15H in the three RAM locations. The first jump was *not* taken because the result was zero.

This instruction provides a simple way of executing a program loop a given number of times, or for adding a moderate time delay (from 2 to 512 machine cycles) with a single instruction. The instruction sequence,

```
        MOV     R2,#8
TOGGLE: CPL     P1.7
        DJNZ    R2,TOGGLE
```

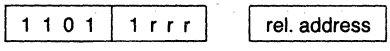
will toggle P1.7 eight times, causing four output pulses to appear at bit 7 of output Port 1. Each pulse will last three machine cycles; two for DJNZ and one to alter the pin.

DJNZ Rn,rel

Bytes: 2

Cycles: 2

Encoding:



Operation:

```
DJNZ
(PC) ← (PC) + 2
(Rn) ← (Rn) − 1
IF  (Rn) > 0 or (Rn) < 0
    THEN
        (PC) ← (PC) + rel
```

DJNZ direct,rel

Bytes: 3

Cycles: 2

Encoding:

1 1 0 1	0 1 0 1	direct address	rel. address

Operation:
DJNZ
(PC) ← (PC) + 2
(direct) ← (direct) − 1
IF (direct) > 0 or (direct) < 0
THEN
(PC) ← (PC) + rel

INC <byte>

Function: Increment

Description: INC increments the indicated variable by 1. An original value of 0FFH will overflow to 00H. No flags are affected. Three addressing modes are allowed: register, direct, or register-indirect.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Register 0 contains 7EH (011111110B). Internal RAM locations 7EH and 7FH contain 0FFH and 40H, respectively. The instruction sequence,

```
INC  @R0
INC  R0
INC  @R0
```

will leave register 0 set to 7FH and internal RAM locations 7EH and 7FH holding (respectively) 00H and 41H.

INC A

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 1 0 0

Operation:
INC
(A) ← (A) + 1

INC Rn

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	1 r r r

Operation: INC
(Rn) ← (Rn) + 1

INC direct

Bytes: 2

Cycles: 1

Encoding:

0 0 0 0	0 1 0 1	direct address

Operation: INC
(direct) ← (direct) + 1

INC @Ri

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 1 1 i

Operation: INC
((Ri)) ← ((Ri)) + 1

INC DPTR

Function: Increment Data Pointer

Description: Increment the 16-bit data pointer by 1. A 16-bit increment (modulo 2^{16}) is performed; an overflow of the low-order byte of the data pointer (DPL) from 0FFH to 00H will increment the high-order byte (DPH). No flags are affected.

This is the only 16-bit register which can be incremented.

Example: Registers DPH and DPL contain 12H and 0FEH, respectively. The instruction sequence,

```
INC  DPTR
INC  DPTR
INC  DPTR
```

will change DPH and DPL to 13H and 01H.

Bytes: 1

Cycles: 2

Encoding:

1 0 1 0	0 0 1 1

Operation: INC
(DPTR) ← (DPTR) + 1

JB bit,rel

Function: Jump if Bit set

Description: If the indicated bit is a one, jump to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. *The bit tested is not modified.* No flags are affected.

Example: The data present at input port 1 is 11001010B. The Accumulator holds 56 (01010110B). The instruction sequence,

```
JB   P1.2,LABEL1

JB   ACC.2,LABEL2
```

will cause program execution to branch to the instruction at label LABEL2.

Bytes: 3

Cycles: 2

Encoding:

0 0 1 0	0 0 0 0	bit address	rel. address

Operation:

```
JB
(PC) ← (PC) + 3
IF  (bit) = 1
    THEN
            (PC) ← (PC) + rel
```

JBC bit,rel

Function: Jump if Bit is set and Clear bit

Description: If the indicated bit is one, branch to the address indicated; otherwise proceed with the next instruction. *The bit will not be cleared if it is already a zero.* The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. No flags are affected.

Note: When this instruction is used to test an output pin, the value used as the original data will be read from the output data latch, *not* the input pin.

Example: The Accumulator holds 56H (01010110B). The instruction sequence,

```
JBC  ACC.3,LABEL1
JBC  ACC.2,LABEL2
```

will cause program execution to continue at the instruction identified by the label LABEL2, with the Accumulator modified to 52H (01010010B).

Bytes: 3

Cycles: 2

Encoding:

0 0 0 1	0 0 0 0	bit address	rel. address

Operation:

```
JBC
(PC) ← (PC) + 3
IF  (bit) = 1
    THEN
            (bit) ← 0
            (PC) ← (PC) + rel
```

JC rel

Function: Jump if Carry is set

Description: If the carry flag is set, branch to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. No flags are affected.

Example: The carry flag is cleared. The instruction sequence,

```
JC    LABEL1
CPL   C
JC    LABEL 2
```

will set the carry and cause program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 0 0	0 0 0 0	rel. address

Operation:

```
JC
(PC) ← (PC) + 2
IF  (C) = 1
    THEN
            (PC) ← (PC) + rel
```

JMP @A+DPTR

Function: Jump indirect

Description: Add the eight-bit unsigned contents of the Accumulator with the sixteen-bit data pointer, and load the resulting sum to the program counter. This will be the address for subsequent instruction fetches. Sixteen-bit addition is performed (modulo 2^{16}): a carry-out from the low-order eight bits propagates through the higher-order bits. Neither the Accumulator nor the Data Pointer is altered. No flags are affected.

Example: An even number from 0 to 6 is in the Accumulator. The following sequence of instructions will branch to one of four AJMP instructions in a jump table starting at JMP_TBL:

```
            MOV   DPTR,#JMP_TBL
            JMP   @A+DPTR
JMP_TBL:    AJMP  LABEL0
            AJMP  LABEL1
            AJMP  LABEL2
            AJMP  LABEL3
```

If the Accumulator equals 04H when starting this sequence, execution will jump to label LABEL2. Remember that AJMP is a two-byte instruction, so the jump instructions start at every other address.

Bytes: 1

Cycles: 2

Encoding:

0 1 1 1	0 0 1 1

Operation: JMP
(PC) ← (A) + (DPTR)

JNB bit,rel

Function: Jump if Bit Not set

Description: If the indicated bit is a zero, branch to the indicated address; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. *The bit tested is not modified.* No flags are affected.

Example: The data present at input port 1 is 11001010B. The Accumulator holds 56H (01010110B). The instruction sequence,

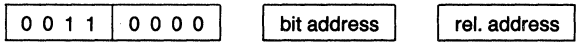
```
JNB  P1.3,LABEL1
JNB  ACC.3,LABEL2
```

will cause program execution to continue at the instruction at label LABEL2.

Bytes: 3

Cycles: 2

Encoding:



Operation:

```
JNB
(PC) ← (PC) + 3
IF  (bit) = 0
    THEN (PC) ← (PC) + rel.
```

JNC rel

Function: Jump if Carry not set

Description: If the carry flag is a zero, branch to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice to point to the next instruction. The carry flag is not modified.

Example: The carry flag is set. The instruction sequence,

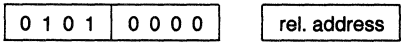
```
JNC  LABEL1
CPL  C
JNC  LABEL2
```

will clear the carry and cause program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:



Operation:

```
JNC
(PC) ← (PC) + 2
IF  (C) = 0
    THEN  (PC) ← (PC) + rel
```

JNZ rel

Function: Jump if Accumulator Not Zero

Description: If any bit of the Accumulator is a one, branch to the indicated address; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. The Accumulator is not modified. No flags are affected.

Example: The Accumulator originally holds 00H. The instruction sequence,

```
JNZ  LABEL1
INC  A
JNZ  LABEL2
```

will set the Accumulator to 01H and continue at label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 1 1	0 0 0 0	rel. address

Operation:

```
JNZ
(PC) ← (PC) + 2
IF   (A) ≠ 0
     THEN  (PC) ← (PC) + rel
```

JZ rel

Function: Jump if Accumulator Zero

Description: If all bits of the Accumulator are zero, branch to the address indicated; otherwise proceed with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. The Accumulator is not modified. No flags are affected.

Example: The Accumulator originally contains 01H. The instruction sequence,

```
JZ   LABEL1
DEC A
JZ   LABEL2
```

will change the Accumulator to 00H and cause program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0 1 1 0	0 0 0 0	rel. address

Operation:

```
JZ
(PC) ← (PC) + 2
IF   (A) = 0
     THEN  (PC) ← (PC) + rel
```

LCALL addr16 (Not implemented in 8xC751 and 8xC752)

Function: Long call

Description: LCALL calls a subroutine located at the indicated address. The instruction adds three to the program counter to generate the address of the next instruction and then pushes the 16-bit result onto the stack (low byte first), incrementing the Stack Pointer by two. The high-order and low-order bytes of the PC are then loaded, respectively, with the second and third bytes of the LCALL instruction. Program execution continues with the instruction at this address. The subroutine may therefore begin anywhere in the full 64K-byte program memory address space. No flags are affected.

Example: Initially the Stack Pointer equals 07H. The label "SUBRTN" is assigned to program memory location 1234H. After executing the instruction,

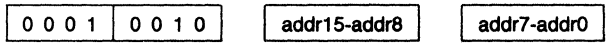
LCALL SUBRTN

at location 0123H, the Stack Pointer will contain 09H, internal RAM locations 08H and 09H will contain 26H and 01H, and the PC will contain 1235H.

Bytes: 3

Cycles: 2

Encoding:



Operation: LCALL
(PC) ← (PC) + 3
(SP) ← (SP) + 1
((SP)) ← (PC_{7-0})
(SP) ← (SP) + 1
((SP)) ← (PC_{15-8})
(PC) ← $addr_{15-0}$

LJMP addr16 (Implemented in 87C751 and 87C752, for use in in-circuit emulation).

Function: Long Jump

Description: LJMP causes an unconditional branch to the indicated address, by loading the high-order and low-order bytes of the PC (respectively) with the second and third instruction bytes. The destination may therefore be anywhere in the full 64K program memory address space. No flags are affected.

Example: The label "JMPADR" is assigned to the instruction at program memory location 1234H. The instruction,

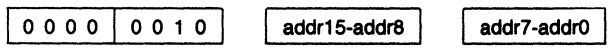
LJMP JMPADR

at location 0123H will load the program counter with 1234H.

Bytes: 3

Cycles: 2

Encoding:



Operation: LJMP
(PC) ← $addr_{15-0}$

MOV <dest-byte>,<src-byte>

Function: Move byte variable

Description: The byte variable indicated by the second operand is copied into the location specified by the first operand. The source byte is not affected. No other register or flag is affected.

This is by far the most flexible operation. Fifteen combinations of source and destination addressing modes are allowed.

Example: Internal RAM location 30H holds 40H. The value of RAM location 40H is 10H. The data present at input port 1 is 11001010B (0CAH).

```
MOV  R0,#30H  ;R0 <= 30H
MOV  A,@R0    ;A <= 40H
MOV  R1,A     ;R1 <= 40H
MOV  R,@R1    ;B <= 10H
MOV  @R1,P1   ;RAM (40H) <= 0CAH
MOV  P2,P1    ;P2 #0CAH
```

leaves the value 30H in register 0, 40H in both the Accumulator and register 1, 10H in register B, and 0CAH (11001010B) both in RAM location 40H and output on port 2.

MOV A,Rn

Bytes: 1

Cycles: 1

Encoding:

1 1 1 0	1 r r r

Operation:
MOV
(A) ← (Rn)

***MOV A,direct**

Bytes: 2

Cycles: 1

Encoding:

1 1 1 0	0 1 0 1

direct address

Operation:
MOV
(A) ← (direct)

***MOV A,ACC is not a valid instruction.**

MOV A,@Ri

Bytes: 1

Cycles: 1

Encoding:

1 1 1 0	0 1 1 i

Operation: MOV
(A) ← ((Ri))

MOV A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 1	0 1 0 0	immediate data

Operation: MOV
(A) ← #data

MOV Rn,A

Bytes: 1

Cycles: 1

Encoding:

1 1 1 1	1 r r r

Operation: MOV
(Rn) ← (A)

MOV Rn,direct

Bytes: 2

Cycles: 2

Encoding:

1 0 1 0	1 r r r	direct addr.

Operation: MOV
(Rn) ← (direct)

MOV Rn,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 1	1 r r r	immediate data

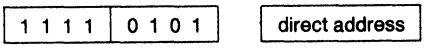
Operation: MOV
(Rn) ← #data

MOV direct,A

Bytes: 2

Cycles: 1

Encoding:



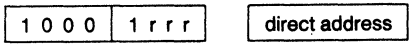
Operation: MOV
(direct) ← (A)

MOV direct,Rn

Bytes: 2

Cycles: 2

Encoding:



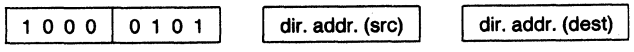
Operation: MOV
(direct) ← (Rn)

MOV direct,direct

Bytes: 3

Cycles: 2

Encoding:



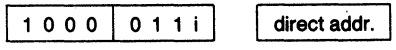
Operation: MOV
(direct) ← (direct)

MOV direct,@Ri

Bytes: 2

Cycles: 2

Encoding:



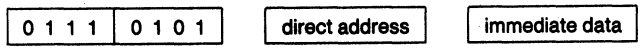
Operation: MOV
(direct) ← ((Ri))

MOV direct,#data

Bytes: 3

Cycles: 2

Encoding:



Operation: MOV
(direct) ← #data

MOV @Ri,A

Bytes: 1

Cycles: 1

Encoding:

1 1 1 1	0 1 1 i

Operation: MOV
((Ri)) ← (A)

MOV @Ri,direct

Bytes: 2

Cycles: 2

Encoding:

1 0 1 0	0 1 1 i	direct addr.

Operation: MOV
((Ri)) ← (direct)

MOV @Ri,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 1	0 1 1 i	immediate data

Operation: MOV
((RI)) ← #data

MOV <dest-bit>,<src-bit>

Function: Move bit data

Description: The Boolean variable indicated by the second operand is copied into the location specified by the first operand. One of the operands must be the carry flag; the other may be any directly addressable bit. No other register or flag is affected.

Example: The carry flag is originally set. The data present at input Port 3 is 11000101B. The data previously written to output Port 1 is 35H (00110101B).

```
MOV  P1.3,C
MOV  C,P3.3
MOV  P1.2,C
```

will leave the carry cleared and change Port 1 to 39H (00111001B).

MOV C,bit

Bytes: 2

Cycles: 1

Encoding:

1 0 1 0	0 0 1 0	bit address

Operation: MOV
(C) ← (bit)

MOV bit,C

Bytes: 2

Cycles: 2

Encoding:

1 0 0 1	0 0 1 0	bit address

Operation: MOV
(bit) ← (C)

MOV DPTR,#data16

Function: Load Data Pointer with a 16-bit constant

Description: The Data Pointer is loaded with the 16-bit constant indicated. The 16-bit constant is loaded into the second and third bytes of the instruction. The second byte (DPH) is the high-order byte, while the third byte (DPL) holds the low-order byte. No flags are affected.

This is the only instruction which moves 16 bits of data at once.

Example: The instruction,

MOV DPTR,#1234H

will load the value 1234H into the Data Pointer: DPH will hold 12H and DPL will hold 34H.

Bytes: 3

Cycles: 2

Encoding:

1 0 0 1	0 0 0 0	immed. data15-8	immed. data7-0

Operation: MOV
(DPTR) ← #$data_{15-0}$
DPH ☐ DPL ← #$data_{15-8}$ ☐ #$data_{7-0}$

MOVC A,@A+ <base-reg>

Function: Move Code byte

Description: The MOVC instructions load the Accumulator with a code byte, or constant from program memory. The address of the byte fetched is the sum of the original unsigned eight-bit Accumulator contents and the contents of a sixteen-bit base register, which may be either the Data Pointer or the PC. In the latter case, the PC is incremented to the address of the following instruction before being added with the Accumulator; otherwise the base register is not altered. Sixteen-bit addition is performed so a carry-out from the low-order eight bits may propagate through higher-order bits. No flags are affected.

Example: A value between 0 and 3 is in the Accumulator. The following instructions will translate the value in the Accumulator to one of four values defined by the DB (define byte) directive.

```
REL_PC: INC   A
        MOVC  A,@A+PC
        RET
        DB    66H
        DB    77H
        DB    88H
        DB    99H
```

If the subroutine is called with the Accumulator equal to 01H, it will return with 77H in the Accumulator. The INC A before the MOVC instruction is needed to "get around" the RET instruction above the table. If several bytes of code separated the MOVC from the table, the corresponding number would be added to the Accumulator instead.

MOVC A,@A+DPTR

Bytes: 1

Cycles: 2

Encoding:

1 0 0 1	0 0 1 1

Operation: MOVC
(A) ← ((A) + (DPTR))

MOVC A,@A + PC

Bytes: 1

Cycles: 2

Encoding:

1 0 0 0	0 0 1 1

Operation: MOVC
(PC) ← (PC) + 1
(A) ← ((A) + (PC))

MOVX <**dest-byte**>,<**src-byte**> (Not implemented in 8xC751 and 8xC752)

Function: Move External

Description: The MOVX instructions transfer data between the Accumulator and a byte of external data memory, hence the "X" appended to MOV. There are two types of instructions, differing in whether they provide an eight-bit or sixteen-bit indirect address to the external data RAM.

In the first type, the contents of R0 or R1 in the current register bank provide an eight-bit address multiplexed with data on P0. Eight bits are sufficient for external I/O expansion decoding or for a relatively small RAM array. For somewhat larger arrays, any output port pins can be used to output higher-order address bits. These pins would be controlled by an output instruction preceding the MOVX.

In the second type of MOVX instruction, the Data Pointer generates a sixteen-bit address. P2 outputs the high-order eight address bits (the contents of DPH) while P0 multiplexes the low-order eight bits (DPL) with data. The P2 Special Function Register retains its previous contents while the P2 output buffers are emitting the contents of DPH. This form is faster and more efficient when accessing very large data arrays (up to 64K bytes), since no additional instructions are needed to set up the output ports.

It is possible in some situations to mix the two MOVX types. A large RAM array with its high-order address lines driven by P2 can be addressed via the Data Pointer, or with code to output high-order address bits to P2 followed by a MOVX instruction using R0 or R1.

Example: An external 256 byte RAM using multiplexed address/data lines is connected to the 8051 Port 0. Port 3 provides control lines for the external RAM. Ports 1 and 2 are used for normal I/O. Registers 0 and 1 contain 12H and 34H. Location 34H of the external RAM holds the value 56H. The instruction sequence,

```
MOVX  A,@R1

MOVX  @R0,A
```

copies the value 56H into both the Accumulator and external RAM location 12H.

MOVX A,@Ri

Bytes: 1

Cycles: 2

Encoding:

1 1 1 0	0 0 1 i

Operation: MOVX
(A) ← ((Ri))

MOVX A,@DPTR

Bytes: 1

Cycles: 2

Encoding:

1 1 1 0	0 0 0 0

Operation: MOVX
(A) ← ((DPTR))

MOVX @Ri,A

Bytes: 1

Cycles: 2

Encoding:

1 1 1 1	0 0 1 i

Operation: MOVX
((Ri)) ← (A)

MOVX @DPTR,A

Bytes: 1

Cycles: 2

Encoding:

1 1 1 1	0 0 0 0

Operation: MOVX
(DPTR) ← (A)

NOP

Function: No Operation

Description: Execution continues at the following instruction. Other than the PC, no registers or flags are affected.

Example: It is desired to produce a low-going output pulse on bit 7 of Port 2 lasting exactly 5 cycles. A simple SETB/CLR sequence would generate a one-cycle pulse, so four additional cycles must be inserted. This may be done (assuming no interrupts are enabled) with the instruction sequence,

```
CLR   P2.7
NOP
NOP
NOP
NOP
SETB  P2.7
```

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 0 0 0

Operation: NOP
$(PC) \leftarrow (PC) + 1$

MUL AB

Function: Multiply

Description: MUL AB multiplies the unsigned eight-bit integers in the Accumulator and register B. The low-order byte of the sixteen-bit product is left in the Accumulator, and the high-order byte in B. If the product is greater than 255 (0FFH) the overflow flag is set; otehrwise it is cleared. The carry flag is always cleared.

Example: Originally the Accumulator holds the value 80 (50H). Register B holds the value 160 (0A0H). The instruction,

```
MUL   AB
```

will give the product 12,800 (3200H), so B is changed to 32H (00110010B) and the Accumulator is cleared. The overflow flag is set, carry is cleared.

Bytes: 1

Cycles: 4

Encoding:

1 0 1 0	0 1 0 0

Operation: MUL
$(A)_{7\text{-}0} \leftarrow (A) \times (B)$
$(B)_{15\text{-}8}$

ORL <dest-byte> <src-byte>

Function: Logical-OR for byte variables

Description: ORL performs the bitwise logical-OR operation between the indicated variables, storing the results in the destination byte. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (11000011B) and R0 holds 55H (01010101B) then the instruction,

ORL A,R0

will leave the Accumulator holding the value 0D7H (11010111B).

When the destination is a directly addressed byte, the instruction can set combinations of bits in any RAM location or hardware register. The pattern of bits to be set is determined by a mask byte, which may be either a constant data value in the instruction or a variable computed in the Accumulator at run-time. The instruction,

ORL P1,#00110010B

will set bits 5, 4, and 1 of output Port 1.

ORL A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 1 0 0	1 r r r

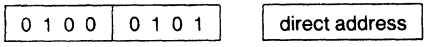
Operation: ORL
(A) ← (A) ∨ (Rn)

ORL A,direct

Bytes: 2

Cycles: 1

Encoding:



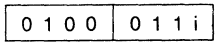
Operation: ORL
(A) ← (A) ∨ (direct)

ORL A,@Ri

Bytes: 1

Cycles: 1

Encoding:



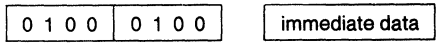
Operation: ORL
(A) ← (A) ∨ ((Ri))

ORL A,#data

Bytes: 2

Cycles: 1

Encoding:



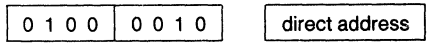
Operation: ORL
(A) ← (A) ∨ #data

ORL direct,A

Bytes: 2

Cycles: 1

Encoding:



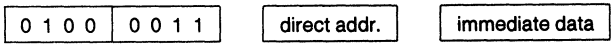
Operation: ORL
(direct) ← (direct) ∨ (A)

ORL direct,#data

Bytes: 3

Cycles: 2

Encoding:



Operation: ORL
(direct) ← (direct) ∨ #data

ORL C,<src-bit>

Function: Logical-OR for bit variables

Description: Set the carry flag if the Boolean value is a logical 1; leave the carry in its current state otherwise . A slash ("/") preceding the operand in the assembly language indicates that the logical complement of the addressed bit is used as the source value, but the source bit itself is not affected. No other flags are affected.

Example: Set the carry flag if and only if P1.0 = 1, ACC. 7 = 1, or OV = 0:

```
MOV  C,P1.0     ;LOAD CARRY WITH INPUT PIN P10
ORL  C,ACC.7    ;OR CARRY WITH THE ACC. BIT 7
ORL  C,/OV      ;OR CARRY WITH THE INVERSE OF OV.
```

ORL C,bit

Bytes: 2

Cycles: 2

Encoding:

0 1 1 1	0 0 1 0	bit address

Operation: ORL
(C) ← (C) ∨ (bit)

ORL C,/bit

Bytes: 2

Cycles: 2

Encoding:

1 0 1 0	0 0 0 0	bit address

Operation: ORL
(C) ← (C) ∨ ($\overline{\text{bit}}$)

POP direct

Function: Pop from stack.

Description: The contents of the internal RAM location addressed by the Stack Pointer is read, and the Stack Pointer is decremented by one. The value read is then transferred to the directly addressed byte indicated. No flags are affected.

Example: The Stack Pointer originally contains the value 32H, and internal RAM locations 30H through 32H contain the values 20H, 23H, and 01H, respectively. The instruction sequence,

```
POP   DPH
POP   DPL
```

will leave the Stack Pointer equal to the value 30H and the Data Pointer set to 0123H. At this point the instruction,

```
POP   SP
```

will leave the Stack Pointer set to 20H. Note that in this special case the Stack Pointer was decremented to 2FH before being loaded with the value popped (20H).

Bytes: 2

Cycles: 2

Encoding:

1 1 0 1	0 0 0 0	direct address

Operation:
POP
(direct) ← ((SP))
(SP) ← (SP) − 1

PUSH direct

Function: Push onto stack

Description: The Stack Pointer is incremented by one. The contents of the indicated variable is then copied into the internal RAM location addressed by the Stack Pointer. Otherwise no flags are affected.

Example: On entering an interrupt routine the Stack Pointer contains 09H. The Data Pointer holds the value 0123H. The instruction sequence,

```
PUSH   DPL
PUSH   DPH
```

will leave the Stack Pointer set to 0BH and store 23H and 01H in internal RAM locations 0AH and 0BH, respectively.

Bytes: 2

Cycles: 2

Encoding:

1 1 0 0	0 0 0 0	direct address

Operation:
PUSH
(SP) ← (SP) + 1
((SP)) ← (direct)

RET

Function: Return from subroutine

Description: RET pops the high- and low-order bytes of the PC successively from the stack, decrementing the Stack Pointer by two. Program execution continues at the resulting address, generally the instruction immediately following an ACALL or LCALL. No flags are affected.

Example: The Stack Pointer originally contains the value 0BH. Internal RAM locations 0AH and 0BH contain the values 23H and 01H, respectively. The instruction,

RET

will leave the Stack Pointer equal to the value 09H. Program execution will continue at location 0123H.

Bytes: 1

Cycles: 2

Encoding:

0 0 1 0	0 0 1 0

Operation: RET
$(PC_{15-8}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$
$(PC_{7-0}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$

RETI

Function: Return from interrupt

Description: RETI pops the high- and low-order bytes of the PC successively from the stack, and restores the interrupt logic to accept additional interrupts at the same priority level as the one just processed. The Stack Pointer is left decremented by two. No other registers are affected; the PSW is *not* automatically restored to its pre-interrupt status. Program execution continues at the resulting address, which is generally the instruction immediately after the point at which the interrupt request was detected. If a lower- or same-level interrupt had been pending when the RETI instruction is executed, that one instruction will be executed before the pending interrupt is processed.

Example: The Stack Pointer originally contains the value 0BH. An interrupt was detected during the instruction ending at location 0122H. Internal RAM locations 0AH and 0BH contain the values 23H and 01H, respectively. The instruction,

RETI

will leave the Stack Pointer equal to 09H and return program execution to location 0123H.

Bytes: 1

Cycles: 2

Encoding:

0 0 1 1	0 0 1 0

Operation: RETI
$(PC_{15-8}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$
$(PC_{7-0}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$

RL A

Function: Rotate Accumulator Left

Description: The eight bits in the Accumulator are rotated one bit to the left. Bit 7 is rotated into the bit 0 position. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

RL A

leaves the Accumulator holding the value 8BH (10001011B) with the carry unaffected.

Bytes: 1

Cycles: 1

Encoding:

0 0 1 0	0 0 1 1

Operation: RL
(An+1)←(An), n = 0 – 6
(A0)←(A7)

RLC A

Function: Rotate Accumulator Left through the Carry flag

Description: The eight bits in the Accumulator and the carry flag are together rotated one bit to the left. Bit 7 moves into the carry flag; the original state of the carry flag moves into the bit 0 position. No other flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B), and the carry is zero. The instruction,

RLC A

leaves the Accumulator holding the value 8BH (10001010B) with the carry set.

Bytes: 1

Cycles: 1

Encoding:

0 0 1 1	0 0 1 1

Operation: RLC
(An+1)←(An), n = 0 – 6
(A0)←(C)
(C)←(A7)

RR A

Function: Rotate Accumulator Right

Description: The eight bits in the Accumulator are rotated one bit to the right. Bit 0 is rotated into the bit 7 position. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

RR A

leaves the Accumulator holding the value 0E2H (11100010B) with the carry unaffected.

Bytes: 1

Cycles: 1

Encoding:

0 0 0 0	0 0 1 1

Operation: RR
(An) ← (An+1), n = 0 - 6
(A7) ← (A0)

RRC A

Function: Rotate Accumulator Right through Carry flag

Description: The eight bits in the Accumulator and the carry flag are together rotated one bit to the right. Bit 0 moves into the carry flag; the original value of the carry flag moves into the bit 7 position. No other flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B), the carry is zero. The instruction,

RRC A

leaves the Accumulator holding the value 62 (01100010B) with the carry set.

Bytes: 1

Cycles: 1

Encoding:

0 0 0 1	0 0 1 1

Operation: RRC
(An) ← (An+1), n = 0 - 6
(A7) ← (C)
(C) ← (A0)

SETB <bit>

Function: Set Bit

Description: SETB sets the indicated bit to one. SETB can operate on the carry flag or any directly addressable bit. No other flags are affected.

Example: The carry flag is cleared. Output Port 1 has been written with the value 34H (00110100B). The instructions,

SETB C

SETB P1.0

will leave the carry flag set to 1 and change the data output on Port 1 to 35H (00110101B).

SETB C

Bytes: 1

Cycles: 1

Encoding:

1 1 0 1	0 0 1 1

Operation: SETB
(C) ← 1

SETB bit

Bytes: 2

Cycles: 1

Encoding:

1 1 0 1	0 0 1 0	bit address

Operation: SETB
(bit) ← 1

SJMP rel

Function: Short Jump

Description: Program control branches unconditionally to the address indicated. The branch destination is computed by adding the signed displacement in the second instruction byte to the PC, after incrementing the PC twice. Therefore, the range of destinations allowed is from 128 bytes preceding this instruction to 127 bytes following it.

Example: The label "RELADR" is assigned to an instruction at program memory location 0123H. The instruction,

SJMP RELADR

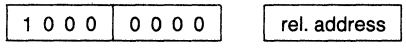
will assemble into location 0100H. After the instruction is executed, the PC will contain the value 0123H.

(*Note:* Under the above conditions the instruction following SJMP will be at 102H. Therefore, the displacement byte of the instruction will be the relative offset (0123H-0102H) = 21H. Put another way, an SJMP with a displacement of 0FEH would be a one-instruction infinite loop.)

Bytes: 2

Cycles: 2

Encoding:



Operation: SJMP
(PC) ← (PC) + 2
(PC) ← (PC) + rel

SUBB A,<src-byte>

Function: Subtract with borrow

Description: SUBB subtracts the indicated variable and the carry flag together from the Accumulator, leaving the result in the Accumulator. SUBB sets the carry (borrow) flag if a borrow is needed for bit 7, and clears C otherwise. (If C was set *before* executing a SUBB instruction, this indicates that a borrow was needed for the previous step in a multiple precision subtraction, so the carry is subtracted from the Accumulator along with the source operand.) AC is set if a borrow is needed for bit 3, and cleared otherwise. OV is set if a borrow is needed into bit 6, but not into bit 7, or into bit 7, but not bit 6.

When subtracting signed integers OV indicates a negative number produced when a negative value is subtracted from a positive value, or a positive result when a positive number is subtracted from a negative number.

The source operand allows four addressing modes: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C9H (11001001B), register 2 holds 54H (01010100B), and the carry flag is set. The instruction,

SUBB A,R2

will leave the value 74H (01110100B) in the accumulator, with the carry flag and AC cleared but OV set.

Notice that 0C9H minus 54H is 75H. The difference between this and the above result is due to the carry (borrow) flag being set before the operation. If the state of the carry is not known before starting a single or multiple-precision subtraction, it should be explicitly cleared by a CLR C instruction.

SUBB A,Rn

Bytes: 1

Cycles: 1

Encoding:

1 0 0 1	1 r r r

Operation: SUBB

$(A) \leftarrow (A) - (C) - (Rn)$

SUBB A,direct

Bytes: 2

Cycles: 1

Encoding:

1 0 0 1	0 1 0 1	direct address

Operation: SUBB
(A) ← (A) − (C) − (direct)

SUBB A,@RI

Bytes: 1

Cycles: 1

Encoding:

1 0 0 1	0 1 1 i

Operation: SUBB
(A) ← (A) − (C) − ((Ri))

SUBB A,#data

Bytes: 2

Cycles: 1

Encoding:

1 0 0 1	0 1 0 0	immediate data

Operation: SUBB
(A) ← (A) − (C) − #data

SWAP A

Function: Swap nibbles within the Accumulator

Description: SWAP A interchanges the low- and high-order nibbles (four-bit fields) of the Accumulator (bits 3-0 and bits 7-4). The operation can also be thought of as a four-bit rotate instruction. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

SWAP A

leaves the Accumulator holding the value 5CH (01011100B).

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	0 1 0 0

Operation: SWAP
$(A_{3-0}) \rightleftarrows (A_{7-4})$

XCH A,<byte>

Function: Exchange Accumulator with byte variable

Description: XCH loads the Accumulator with the contents of the indicated variable, at the same time writing the original Accumulator contents to the indicated variable. The source/destination operand can use register, direct, or register-indirect addressing.

Example: R0 contains the address 20H. The Accumulator holds the value 3FH (00111111B). Internal RAM location 20H holds the value 75H (01110101B). The instruction,

XCH A,@R0

will leave RAM location 20H holding the values 3FH (00111111B) and 75H (01110101B) in the accumulator.

XCH A,Rn

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	1 r r r

Operation: XCH
(A) $\rightleftarrows$ (Rn)

XCH A,direct

Bytes: 2

Cycles: 1

Encoding:

1 1 0 0	0 1 0 1	direct address

Operation: XCH
(A) $\rightleftarrows$ (direct)

XCH A,@Ri

Bytes: 1

Cycles: 1

Encoding:

1 1 0 0	0 1 1 i

Operation: XCH
(A) $\rightleftarrows$ ((Ri))

XCHD A,@Ri

Function: Exchange Digit

Description: XCHD exchanges the low-order nibble of the Accumulator (bits 3-0), generally representing a hexadecimal or BCD digit, with that of the internal RAM location indirectly addressed by the specified register. The high-order nibbles (bits 7-4) of each register are not affected. No flags are affected.

Example: R0 contains the address 20H. The Accumulator holds the value 36H (00110110B). Internal RAM location 20H holds the value 75H (01110101B). The instruction,

XCHD A,@R0

will leave RAM location 20H holding the value 76H (01110110B) and 35H (00110101B) in the Accumulator.

Bytes: 1

Cycles: 1

Encoding:

1 1 0 1	0 1 1 i

Operation: XCHD
$(A_{3\text{-}0}) \rightleftarrows ((Ri_{3\text{-}0}))$

XRL <dest-byte>,<src-byte>

Function: Logical Exclusive-OR for byte variables

Description: XRL performs the bitwise logical Exclusive-OR operation between the indicated variables, storing the results in the destination. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

(*Note:* When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.)

Example: If the Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B) then the instruction,

XRL A,R0

will leave the Accumulator holding the value 69H (01101001B).

When the destination is a directly addressed byte, this instruction can complement combinations of bits in any RAM location or hardware register. The pattern of bits to be complemented is then determined by a mask byte, either a constant contained in the instruction or a variable computed in the Accumulator at run-time. The instruction,

XRL P1,#00110001B

will complement bits 5, 4, and 0 of output Port 1.

XRL A,Rn

Bytes: 1

Cycles: 1

Encoding:

0 1 1 0	1 r r r

Operation: XRL
(A) ← (A) ∀ (Rn)

XRL A,direct

Bytes: 2

Cycles: 1

Encoding:

0 1 1 0	0 1 0 1	direct address

Operation: XRL
(A) ← (A) ∀ (direct)

XRL A,@Ri

Bytes: 1

Cycles: 1

Encoding:

0 1 1 0	0 1 1 i

Operation: XRL
(A) ← (A) ∀ ((Ri))

XRL A,#data

Bytes: 2

Cycles: 1

Encoding:

0 1 1 0	0 1 0 0	immediate data

Operation: XRL
(A) ← (A) ∀ #data

XRL direct,A

Bytes: 2

Cycles: 1

Encoding:

0 1 1 0	0 0 1 0	direct address

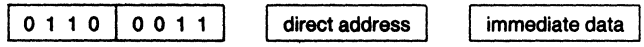
Operation: XRL
(direct) ← (direct) ∀ (A)

XRL direct,#data

Bytes: 3

Cycles: 2

Encoding:



Operation: XRL
(direct) ← (direct) ∀ #data

EPROM PRODUCTS

All 80C51 derivative products offered by Signetics are supported with an EPROM version. Currently available EPROM parts are the 87C51, 87C451, and the 87C751. EPROM versions of the 83C552, 83C652, and 83C752 are now in development.

All EPROM products are available in both windowed DIP and OTP package configurations. The windowed DIP package allows the EPROM to be erased under a strong UV light source making program development easier and faster. The OTP (One Time Programmable) version cannot be erased because there is no window through which the die could be exposed to UV light. While the EPROM can only be programmed once in the OTP package, the part costs less than in windowed DIP and therefore offers an advantage for those not desiring to use the masked ROM version of the part.

The EPROM products are fully supported on the industry standard EPROM programmers.

PROGRAMMING THE 87C51, 87C451 AND 87C552

The setup for programming the microcontroller is shown in Figure 59. Note that the part is running with a 4 to 6 MHz oscillator. The clock must be running because the device is executing internal address and program data transfers during the programming.

To program the SC87C51, SC87C451, or SC87C552, the address of the EPROM location to be programmed is applied to ports 1 and 2 as shown in Figure 59. The code byte to be programmed into this location is applied to port 0. RST, $\overline{PSEN}$ and the pins of ports 2 and 3 specified in Table 20 are held at the "Program Code Data" levels specified in the table. The ALE/$\overline{PROG}$ is then pulsed low 25 times to program the addressed location.

ENCRYPTION TABLE

The encryption table is a feature of the SC87C51, and its derivatives, that protects the code from being easily read by anyone other than the programmer. The encryption table is 16 bytes of code that are exclusive NORed with the program code data as it is read out. The first byte is XNORed with the first location read, the second with the second read, etc. through the sixteenth byte read. The seventeenth byte is XNORed with the first byte of the encryption table, the eighteenth with the second, etc. and on in sixteen byte groups.

After the Encryption table has been programmed the user has to know its contents in order to correctly decode the program code data. The encryption table itself cannot be read out.

The encryption table is programmed in the same manner as the program memory, but using the "Pgm Encryption Table" levels specified in Table 20. After the encryption table is programmed verification cycles will produce only encrypted information.

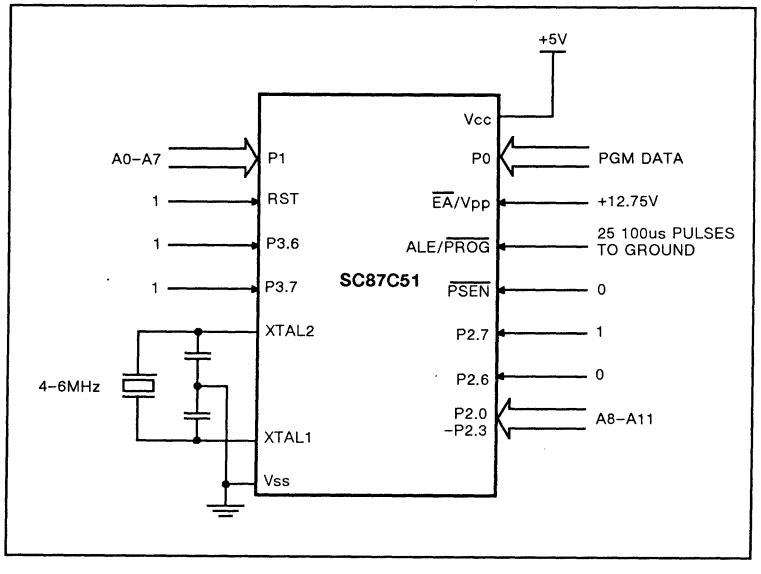


Figure 59. Programming Configuration

Table 20. EPROM Programming Modes

MODE	RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}$/V_{PP}	P2.7	P2.6	P3.7	P3.6
Read signature	1	0	1	1	0	0	0	0
Program code data	1	0	0*	V_{PP}	1	0	1	1
Verify code data	1	0	1	1	0	0	1	1
Pgm encryption table	1	0	0*	V_{PP}	1	0	1	0
Pgm lock bit 1	1	0	0*	V_{PP}	1	1	1	1
Pgm lock bit 2	1	0	0*	V_{PP}	1	1	0	0

NOTES:
1. "0" = valid low for that pin, "1" = valid high for that pin.
2. **V_{PP} = 12.75V ±0.25V.**
3. V_{CC} = 5V ±10% during programming and verification.

*ALE/$\overline{PROG}$ receives 25 programming pulses while V_{PP} is held at 12.75V. Each programming pulse is low for 100µs (±10µs) and high for a minimum of 10µs.

LOCK BIT

There are two lock bits on the SC87C51 that, when set, prevent the program data memory from being read out or programmed further. To program the lock bits repeat the programming sequence using the "Pgm Lock Bit" levels specified in Table 20.

After the first lock bit is programmed, further programming of the code memory or the encryption table is disabled. The other lock bit can of course still be programmed. With only lock bit one programmed, the memory can still be read out for program verification. After the second lock bit is programmed, it is no longer possible to read out (verify) the program memory.

PROGRAM VERIFICATION

If lock bit 2 has not been programmed the on-chip program memory can be read out for program verification. To verify the contents of the program memory, the address of the location to be read is applied to ports 1 and 2 as shown in Figure 60. The other pins are held at the "Verify Code Data" levels indicated in Table 20. The contents of the addressed location will appear on port 0. For this operation external pull-ups are required on port 0 as shown in figure 60. Note that if the encryption table has been programmed the data presented at port 0 will be the exclusive NOR of the program byte with a byte from the encryption table.

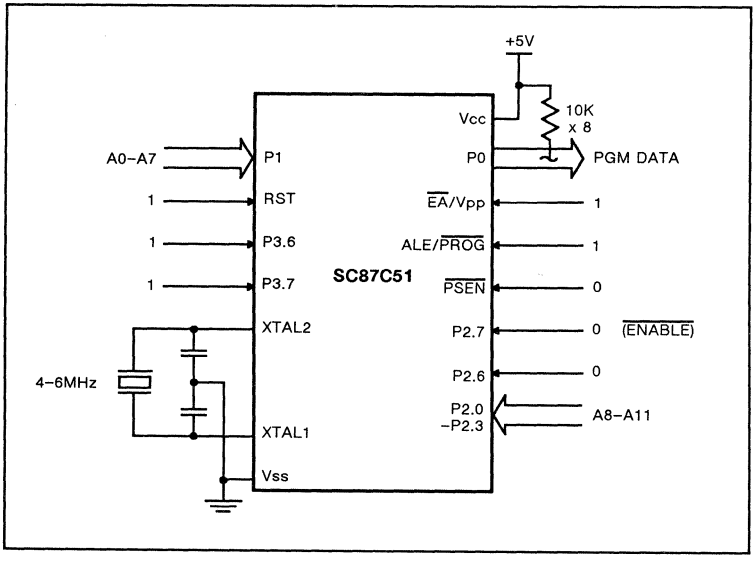


Figure 60. Program Verification

SIGNATURE BYTES

The SC87C51 contains two signature bytes that can be read and used by an EPROM programming system to identify the device. The signature bytes identify the device as an SC87C51 manufactured by Signetics .

The signature bytes are read by the same procedure as a normal verification of locations 030H and 031H, except that P3.6 and P3.7 need to be pulled to a logic low. The values are:

(030H) = 15H indicates the part made by Signetics
(031H) = 90H 87C451 94H 87C552
91H 87C751 95H 87C752
92H 87C51 96H 87C550
93H 87C652 97H 87C52

EPROM ERASURE

Erasure of the EPROM occurs when the chip is exposed to light with wavelengths shorter than 4000 angstroms. Sunlight and fluorescent lighting have wavelengths in this range, so exposure to these light sources over an extended period of time (about 1 week in sunlight, or 3 years in room level fluorescent lighting) could cause inadvertent erasure. It is recommended, for this reason, that an opaque label be placed over the window. If the part is subject to elevated temperatures or an environment where solvents are used, Kapton tape (Fluorglas part number 2345-5 or its equivalent) can be used.

The recommended erasure procedure is to expose the chip to ultraviolet light (at 2537 angstroms) to an integrated dose of at least 15W-sec/cm2. Exposing the EPROM to an ultraviolet lamp of 12,000μW/cm2 rating for 20 to 40 minutes, at a distance of 1 inch, is adequate.

PROGRAMMING THE 87C751 AND 87C752

The 87C751 and 87C752 are programmed using a Quickpulse programming algorithm that is similar to that used for the 87C51. It differs from the 87C51 in that a serial data stream is used to place the 87C751 in the programming mode.

Figure 61 shows a block diagram of the programming configuration for the 87C751. Port pin P0.2 is used for the programming voltage supply input (V_{PP} signal). Port pin P0.1 is used for the program (PGM) signal.

Port 3 accepts the address input for the EPROM location to be programmed. Both the high and low components of the eleven bit address are presented to the part through port 3. Multiplexing of the address components is performed using ASEL (P0.0).

Port 1 is used as a bidirectional data bus during programming and verify operations. During the programming mode, it accepts the byte to be programmed. In the verify mode, it returns the contents of the specified address location.

The X1 pin is the oscillator input and receives the master system clock. This clock should be between 1.2 and 6MHz.

The RESET pin is used to accept the serial data stream that places the 87C751 into various programming modes. This pattern consists of a 10-bit code with the LSB sent first. Each bit is synchronized to the clock input X1.

To program the 87C751 the part must be put into the programming mode by presenting the proper serial code (see Table 21) to the RESET pin. To do this RESET should be held high for at least two machine cycles. Port pins P0.1 and P0.2 will be at VOH as a result of this, but they must be driven high prior to sending the serial data stream on the RESET pin. The serial data bits can now be transmitted over the RESET pin placing the 87C751 into one of the programming modes. Following the transmission of the last data bit the reset pin should be held low.

Next the address information for the location to be programmed is placed on Port 3 and ASEL is used to perform the address multiplexing. ASEL should be driven high and then Port 3 driven with the high order address bits. ASEL is then driven low latching the high order bits internally. Port 3 can now be driven with the low 8 bits of the address, completing the addressing of the location to be programmed.

A high voltage V_{PP} level is now applied to the V_{PP} input. This sets Port 1 as an input port. The data to be programmed to the EPROM array should be placed on Port 1. A series of 25 programming pulses is now applied to the PGM pin (P0.1) to program the addressed EPROM location.

PROGRAM VERIFICATION

The EPROM array can be verified by placing the part in the programming mode as described above and forcing the V_{PP} pin to the V_{OH} level. Four machine cycles after addressing a location the contents of the addressed location will appear on Port 1.

87C751 AND 87C752 SIGNATURE BYTES

The signature bytes for the 87C751 and 87C752 are read differently and are in different locations than those on the 87C51. Due to its reduced pin count, the part has to be put into "Signature Byte Read Mode" by placing a 10-bit serial data stream on the Reset pin. The proper code and the conditions of P0.1 and P0.2, for this mode, are shown in Table 21.

Once the part has been placed into the Signature Byte Read Mode, the signature bytes can be read by the same procedure as a normal verification of locations 01EH and 01FH. The values are:

01EH = 15H indicates the part was made by Signetics
01FH = 91H – 87C751
01FH = 95H – 87C752

PROGRAMMING FEATURES

The 87C751 has all of the special programming features incorporated within its EPROM array that the 87C51 has. It has an encryption key table and two security bits (lock bits). These function exactly as they do in the 87C51. They are programmed or verified by sending the proper code over the RESET pin (see table 21) and then following the 87C751 programming procedure as described previously.

ERASURE CHARACTERISTICS

The erasure procedure is exactly the same as that described for the 87C51.

Table 21. Implementing Program/Verify Modes

Operation	Serial Code	P0.1 (PGM/)	P0.2 (V_{PP})
Program user EPROM	296H	~*	V_{PP}
Verify user EPROM	296H	V_{IH}	V_{IH}
Program key EPROM	292H	~*	V_{PP}
Verify key EPROM	292H	V_{IH}	V_{IH}
Program security bit 1	29AH	~*	V_{PP}
Program security bit 2	298H	~*	V_{PP}
Verify security bits	29AH	V_{IH}	V_{IH}
Read signature bytes	19CH	V_{IH}	V_{IH}

NOTE:
*Pulsed from V_{IH} to V_{IL} and returned to V_{IH}.

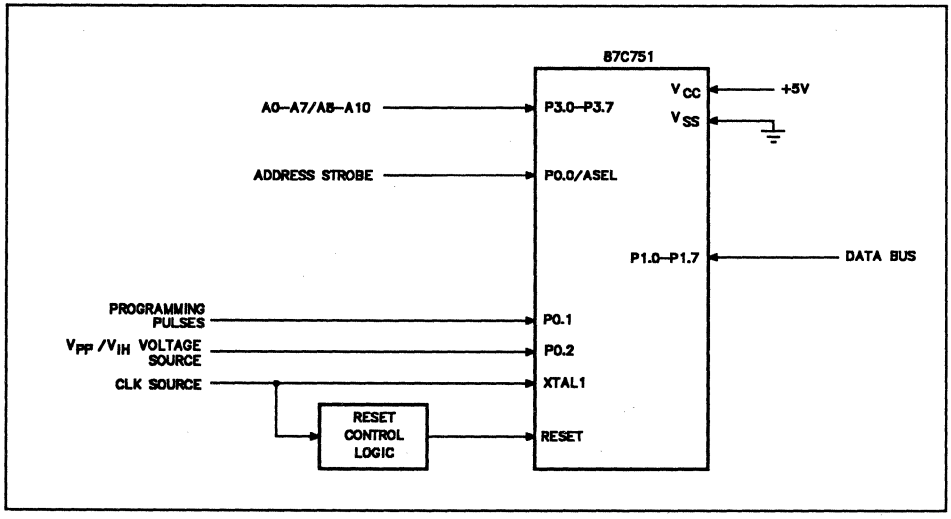


Figure 61. Programming Configuration

Signetics

SCN8031AH/SCN8051AH
Single-Chip 8-Bit Microcontroller

Product Specification

Microprocessor Division

DESCRIPTION

The Signetics SCN8031AH/SCN8051AH is a high-performance microcontroller fabricated using the Signetics highly reliable +5V, depletion-load, N-channel, silicon-gate, N500 MOS process technology. It provides the hardware features, architectural enhancements and instructions that are necessary to make it a powerful and cost-effective controller for applications requiring up to 64K bytes of program memory and/or up to 64K bytes of data storage.

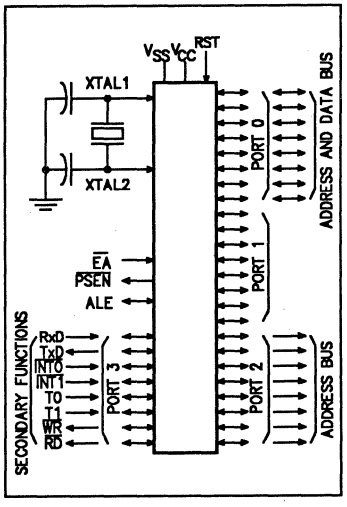
The SCN8051AH contains a 4K X 8 read-only program memory, a 128 X 8 read/write data memory, 32 I/O lines, two 16-bit timer/counters, a five-source two-priority-level nested interrupt structure, a serial I/O port for either multiprocessor communications, I/O expansion, or full-duplex UART, and on-chip oscillator and clock circuits. The SCN8031AH is identical, except that it lacks the program memory. For systems that require extra capability, the SCN8051AH can be expanded using standard TTL compatible memories and byte oriented peripheral controllers.

The SCN8051AH microcontroller, like its SCN8048 predecessor, is efficient both as a controller and as an arithmetic processor. It has extensive facilities for binary and BCD arithmetic and excels in bit-handling capabilities. Efficient use of program memory results from an instruction set consisting of 44% one-byte, 41% two-byte, and 15% three-byte instructions. With a 12MHz crystal, 58% of the instructions execute in 1μs, 40% in 2μs and multiply and divide require only 4μs.

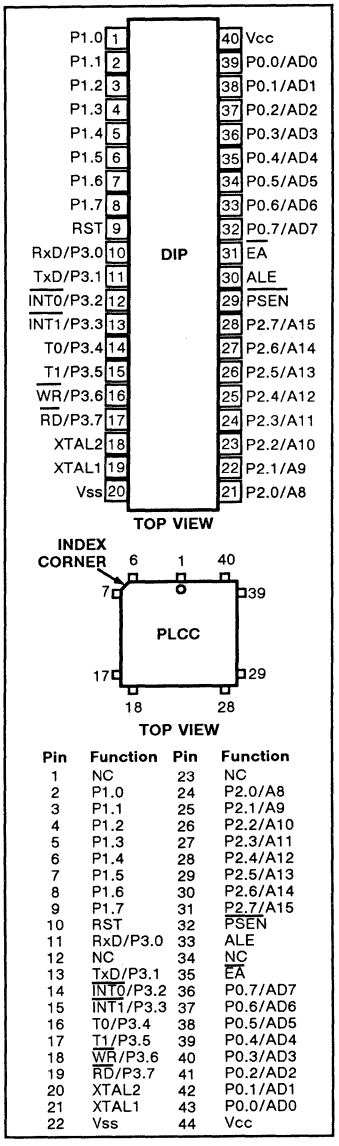
FEATURES

- **Reduced supply current**
- **4K X 8 ROM (SCN8051AH)**
- **128 X 8 RAM**
- **Four 8-bit ports, 32 I/O lines**
- **Two 16-bit timer/event counters**
- **High-performance full-duplex serial channel**
- **External memory expandable to 128K**
- **Boolean processor**
- **Industry standard 8051 architecture:**
  - **Non–paged jumps**
  - **Direct addressing**
  - **Four 8-register banks**
  - **Stack depth up to 128-bytes**
  - **Multiply, divide, subtract, compare**
- **Most instructions execute in 1μs**
- **4μs multiply and divide**

LOGIC SYMBOL



PIN CONFIGURATION



Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

ORDERING INFORMATION

SCN80 □□H□□□□□ (CPxxxx)

ROM/RAM (bytes)
31 = Ext/128
51 = 4K/128

Power Consumption
H = Reduced Power AH

Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

Pins
40 = 40-pin
44 = 44-pin

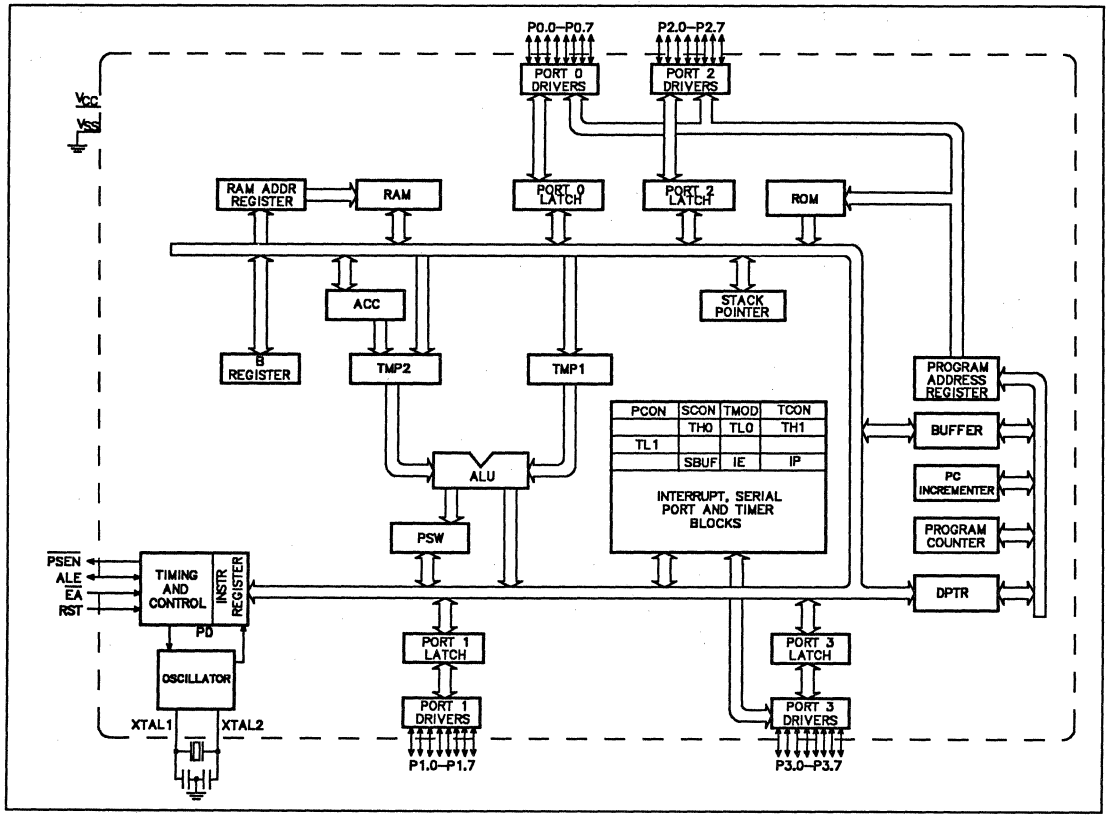
Package
A = Plastic LCC
I = Ceramic DIP
N = Plastic DIP

Speed
C = 3.5 to 12MHz
F = 3.5 to 15MHz

Operating Temperature Range
A = −40°C to +85°C
C = 0°C to +70°C

PART NUMBER SELECTION			
ROMless	**ROM**	**Temperature and Package**	**Frequency**
SCN8031HACN40	SCN8051HACN40	−40 to +85°C plastic DIP	3.5 to 12MHz
SCN8031HCCN40	SCN8051HCCN40	0 to +70°C plastic DIP	3.5 to 12MHz
SCN8031HCFN40	SCN8051HCFN40	0 to +70°C plastic DIP	3.5 to 15MHz
SCN8031HAFN40	SCN8051HAFN40	−40 to +85°C plastic DIP	3.5 to 15MHz
SCN8031HCCA44	SCN8051HCCA44	0 to +70°C plastic PLCC	3.5 to 12MHz
SCN8031HACA44	SCN8051HACA44	−40 to +85°C plastic PLCC	3.5 to 12MHz
SCN8031HCFA44	SCN8051HCFA44	0 to +70°C plastic PLCC	3.5 to 15MHz
SCN8031HAFA44	SCN8051HAFA44	−40 to +85°C plastic PLCC	3.5 to 15MHz

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. LCC	TYPE	NAME AND FUNCTION
V_{SS}	20	22	I	**Ground:** 0V reference.
V_{CC}	40	44	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–P0.7	39–32	43–36	I/O	**Port 0:** Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pullups when emitting 1s.
P1.0–P1.7	1–8	2–9	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. Port 1 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: I_{IL}).
P2.0–P2.7	21–28	24–31	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. Port 2 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pullups. (See DC Electrical Characteristics: I_{IL}). Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application, it uses strong internal pull-ups when emitting 1s. During accesses to exter- nal data memory that use 8-bit addresses (MOVX @Ri), port 2 emits the contents of the P2 special function register.
P3.0–P3.7	10–17	11, 13–19	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 3 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 3 is also used for the special features listed below:
	10	11	I	**RxD (P3.0):** Serial input port
	11	13	O	**TxD (P3.1):** Serial output port
	12	14	I	**INT0 (P3.2):** External interrupt
	13	15	I	**INT1 (P3.3):** External interrupt
	14	16	I	**T0 (P3.4):** Timer 0 external input
	15	17	I	**T1 (P3.5):** Timer 1 external input
	16	18	O	**WR (P3.6):** External data memory write strobe
	17	19	O	**RD (P3.7):** External data memory read strobe
RST	9	10	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal diffused resistor to V_{SS} permits a power-on reset using only an external capacitor to V_{CC}.
ALE	30	33	I/O	**Address Latch Enable:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory.
$\overline{PSEN}$	29	32	O	**Program Store Enable:** The read strobe to external program memory. When the device is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory.
$\overline{EA}$	31	35	I	**External Access Enable:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations 0000H and 0FFFH. If $\overline{EA}$ is held high, the device executes from internal program memory unless the program counter contains an address greater than 0FFFH.
XTAL1	19	21	I	**Crystal 1:** Input to the inverting oscillator amplifier.
XTAL2	18	20	O	**Crystal 2:** Output from the inverting oscillator amplifier and input to the internal clock generator circuits.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol, page 1.

To drive the device from an external clock source, XTAL2 should be driven while XTAL1 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

DESIGN CONSIDERATIONS

At power-on, the voltage on V_{CC} and RST should come up at the same time for a proper start-up.

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Storage temperature range	−65 to +150	°C
All voltages with respect to ground	−0.5 to +7.0	V
Power dissipation	1.0	W

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C , V_{CC} = 4.5 to 5.5V, V_{SS} = 0V[4, 5]

Symbol	Parameter	Test Conditions	Limits		Unit
			Min	Max	
V_{IL}	Input low voltage		−0.5	0.8	V
V_{IH}	Input high voltage, except RST and XTAL2		2	V_{CC}+0.5	V
V_{IH1}	Input high voltage to RST for reset, XTAL2	XTAL1 to V_{SS}	2.5	V_{CC}+0.5	V
V_{OL}	Output low voltage, ports 1, 2, 3[6]	I_{OL} = 1.6mA		0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$[6]	I_{OL} = 3.2mA		0.45	V
V_{OH}	Output high voltage, ports 1, 2, 3	I_{OH} = −80µA	2.4		V
V_{OH1}	Output high voltage port 0, ALE, $\overline{PSEN}$)[3]	I_{OH} = −400µA	2.4		V
I_{IL}	Logical 0 input current, ports 1, 2, 3	V_{IN} = 0.45V		−500	µA
I_{IH1}	Input high curent to RST for reset	V_{IN} < V_{CC} − 1.5V		500	µA
I_{LI}	Input leakage current, port 0, $\overline{EA}$	0.45 < V_{IN} < V_{CC}		±10	µA
I_{IL2}	Logical 0 input current for XTAL2	XTAL1 = V_{SS}, V_{IN} = 0.45V		−3.2	mA
I_{CC}	Power supply current:	All outputs disconnected and $\overline{EA}$ = V_{CC}		125	mA
C_{IO}	Pin capacitance			10	pF
T_A = −40°C to +85°C − Extended temperature range − SCN8051HAC only					
V_{IH}	Input high voltage, except RST and XTAL2		2.2	V_{CC}+0.5	V
V_{IH1}	Input high voltage to RST for reset, XTAL2	XTAL1 to V_{SS}	2.7		V
I_{CC}	Power supply current:	All outputs disconnected and $\overline{EA}$ = V_{CC}		175	mA

NOTES:

1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. For operating at elevated temperatures, the device must be derated based on +150°C maximum junction temperature.
3. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
4. Parameters are valid over operating temperature range unless otherwise specified.
5. All voltage measurements are referenced to ground. For testing, all input signals swing between 0.45V and 2.4V with a transition time of 20ns maximum. All time measurements are referenced at input voltages of 0.8V and 2.0V and at output voltages of 0.8V and 2.0V as appropriate.
6. V_{OL} is degraded when the device rapidly discharges external capacitance. This AC noise is most pronounced during emission of address data. When using external memory, locate the latch or buffer as close to the device as possible.

Datum	Emitting Ports	Degraded I/O Lines	V_{OL} (Peak Max)
Address	P2, P0	P1, P3	0.8V
Write Data	P0	P1, P3, ALE	0.8V

7. C_L = 100pF for port 0, ALE and $\overline{PSEN}$ outputs; C_L = 80pF for all other ports.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or −40°C to +85°C, V_{CC} = 5V ±20%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	1	Oscillator frequency: **Speed Versions** SCN8051 C SCN8051 F			 3.5 3.5	 12 15	 MHz MHz
t_{LHLL}	1	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	1	Address valid to ALE low	43		$t_{CLCL}-40$		ns
t_{LLAX}	1	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	1	ALE low to valid instruction in		233		$4t_{CLCL}-100$	ns
t_{LLPL}	1	ALE low to $\overline{PSEN}$ low	58		$t_{CLCL}-25$		ns
t_{PLPH}	1	$\overline{PSEN}$ pulse width	215		$3t_{CLCL}-35$		ns
t_{PLIV}	1	$\overline{PSEN}$ low to valid instruction in		125		$3t_{CLCL}-125$	ns
t_{PXIX}	1	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	1	Input instruction float after $\overline{PSEN}$		63		$t_{CLCL}-20$	ns
t_{AVIV}	1	Address to valid instruction in		302		$5t_{CLCL}-115$	ns
t_{PLAZ}	1	$\overline{PSEN}$ low to address float		20		20	ns
t_{PXAV}	1	$\overline{PSEN}$ to address valid	75		$t_{CLCL}-8$		ns
Data Memory							
t_{RLRH}	2, 3	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	2, 3	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	2, 3	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	2, 3	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	2, 3	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	2, 3	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	2, 3	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	2, 3	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	2, 3	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	2, 3	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{QVWH}	2, 3	Data vaid to $\overline{WR}$ high	433		$7t_{CLCL}-150$		ns
t_{WHQX}	2, 3	Data hold after $\overline{WR}$	33		$t_{CLCL}-8$		ns
t_{RLAZ}	2, 3	$\overline{RD}$ low to address float		20		20	ns
t_{WHLH}	2, 3	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
External Clock							
t_{CHCX}	5	High time	20		20		ns
t_{CLCX}	5	Low time	20		20		ns
t_{CLCH}	5	Rise time		20		20	ns
t_{CHCL}	5	Fall time		20		20	ns
Shift Register							
t_{XLXL}	4	Serial port clock cycle time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	4	Output data setup to clock rising edge	700		$10t_{CLCL}-133$		ns
t_{XHQX}	4	Output data hold after clock rising edge	50		$2t_{CLCL}-117$		ns
t_{XHDX}	4	Input data hold after clock rising edge	0		0		ns
t_{XHDV}	4	Clock rising edge to input data valid		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P – $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.

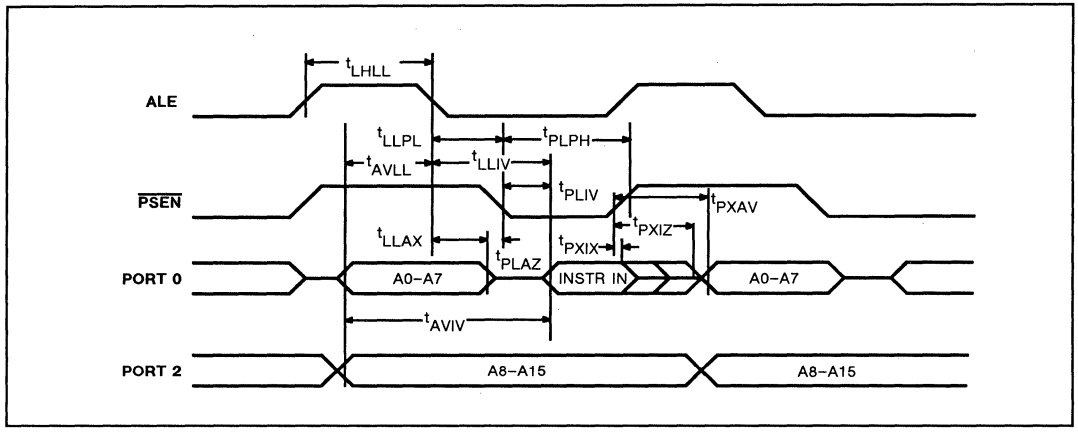


Figure 1. External Program Memory Read Cycle

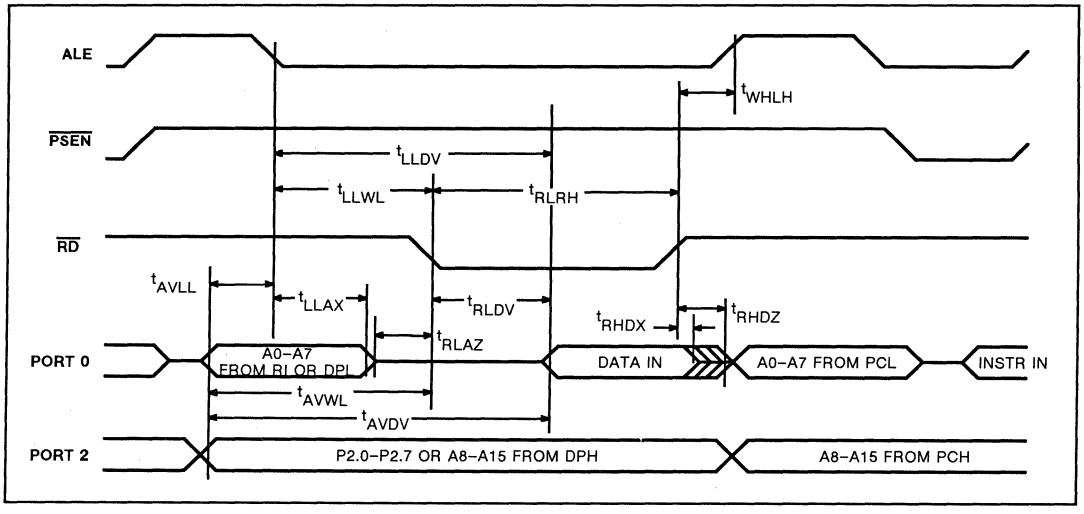


Figure 2. External Data Memory Read Cycle

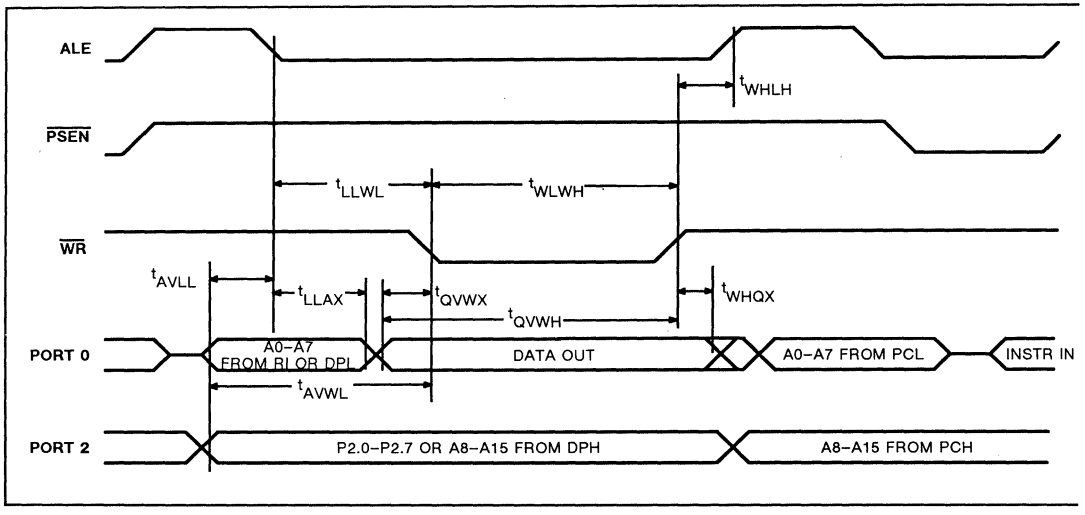


Figure 3. External Data Memory Write Cycle

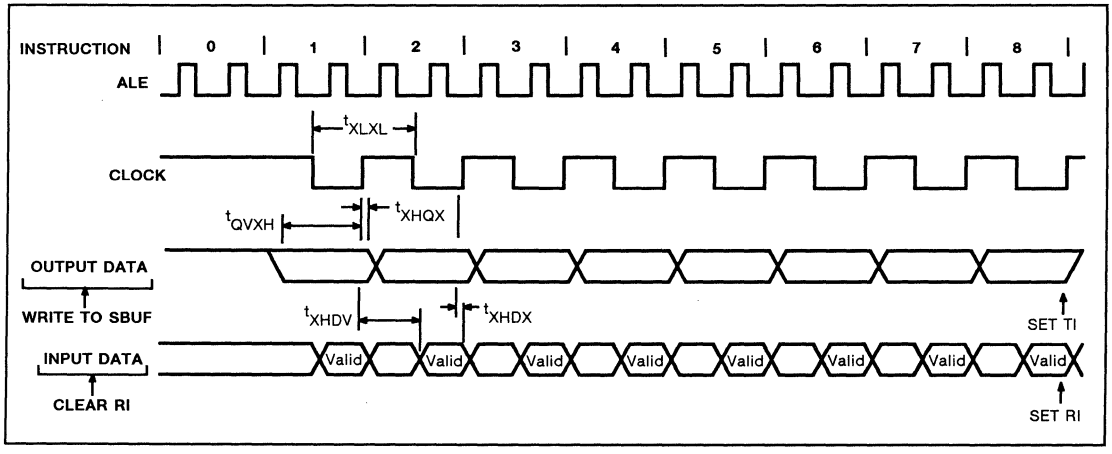


Figure 4. Shift Register Mode Timing

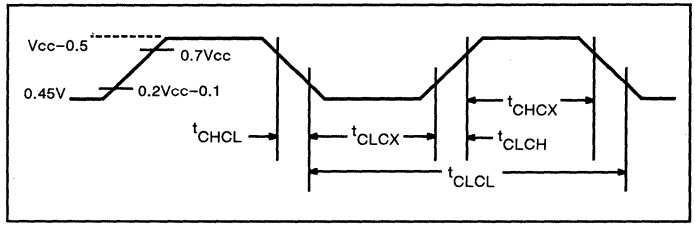


Figure 5. External Clock Drive

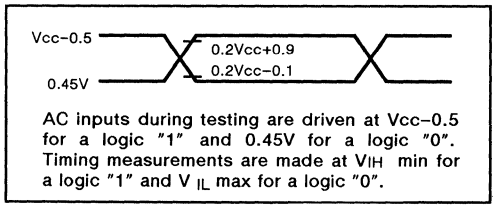


Figure 6. AC Testing Input/Output

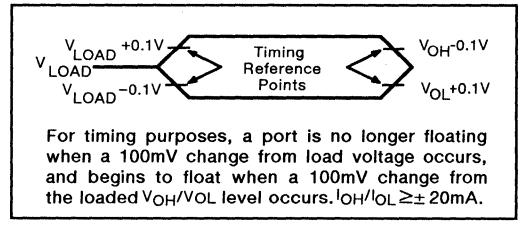


Figure 7. Float Waveform

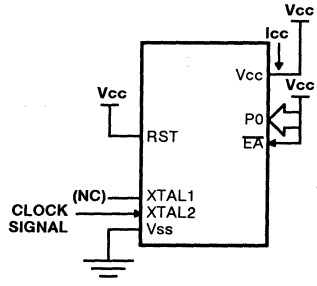


Figure 8. I_{CC} Test Condition, Active Mode
All other pins are disconnected

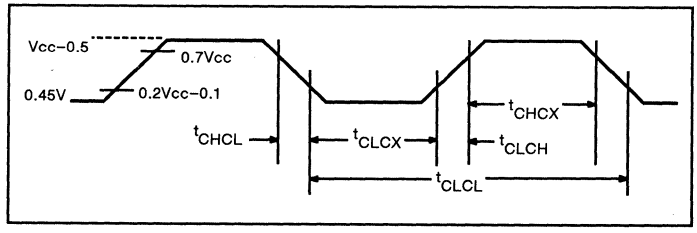


Figure 9. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

Signetics

SC80C31B/SC80C51B
CMOS Single-Chip 8-Bit Microcontroller

Product Specification

Microprocessor Division

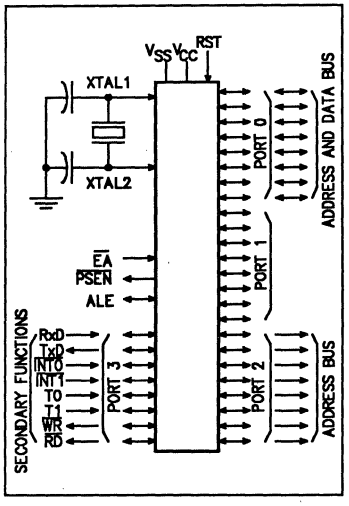
DESCRIPTION

The Signetics SC80C31B/SC80C51B is a high-performance microcontroller fabricated with Signetics high-density CMOS technology. The CMOS SC80C31B/SC80C51B is functionally compatible with the NMOS SCN8031/SCN8051 microcontrollers. The Signetics CMOS technology combines the high speed and density characteristics of HMOS with the low power attributes of CMOS. Signetics' epitaxial substrate minimizes latch-up sensitivity.

The SC80C31B/SC80C51B contains a 4K x 8 ROM, a 128 x 8 RAM, 32 I/O lines, two 16-bit counter/timers, a five-source, two priority level nested interrupt structure, a serial I/O port for either multi-processor communications, I/O expansion or full duplex UART, and on-chip oscillator and clock circuits.

In addition, the SC80C31B/SC80C51B has two software selectable modes of power reduction – idle mode and power-down mode. The idle mode freezes the CPU while allowing the RAM, timers, serial port, and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator, causing all other chip functions to be inoperative.

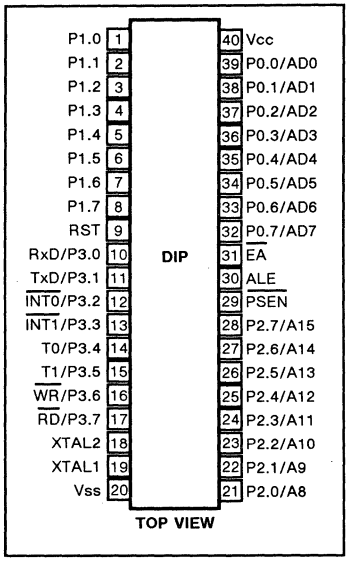
LOGIC SYMBOL



FEATURES

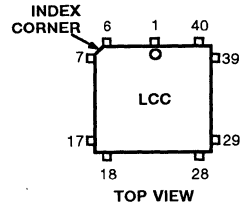
- **SCN8031/SCN8051/SC80C51 compatible**
  - **4K x 8 ROM**
  - **128 x 8 RAM**
  - **Two 16-bit counter/timers**
  - **Full duplex serial channel**
  - **Boolean processor**
- **Memory addressing capability**
  - **64K ROM and 64K RAM**
- **Power control modes:**
  - **Idle mode**
  - **Power-down mode**
- **CMOS and TTL compatible**
- **Three speed ranges at V_{CC} = 5V ±20%**
  - **3.5 to 12MHz**
  - **3.5 to 16MHz**
  - **0.5 to 12MHz**
- **Three package styles**

PIN CONFIGURATION



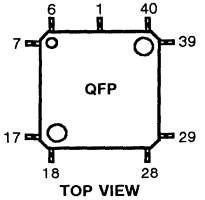
TOP VIEW

PIN CONFIGURATION (Cont)



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	Vss
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

CMOS Single-Chip 8-Bit Microcontroller — SC80C31B/SC80C51B

ORDERING INFORMATION

SC80C □1B□□□□□ (CPxxxx)

ROMless/ROM
3 = ROMless
5 = ROM

Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

Pins
40 = 40-pin
44 = 44-pin

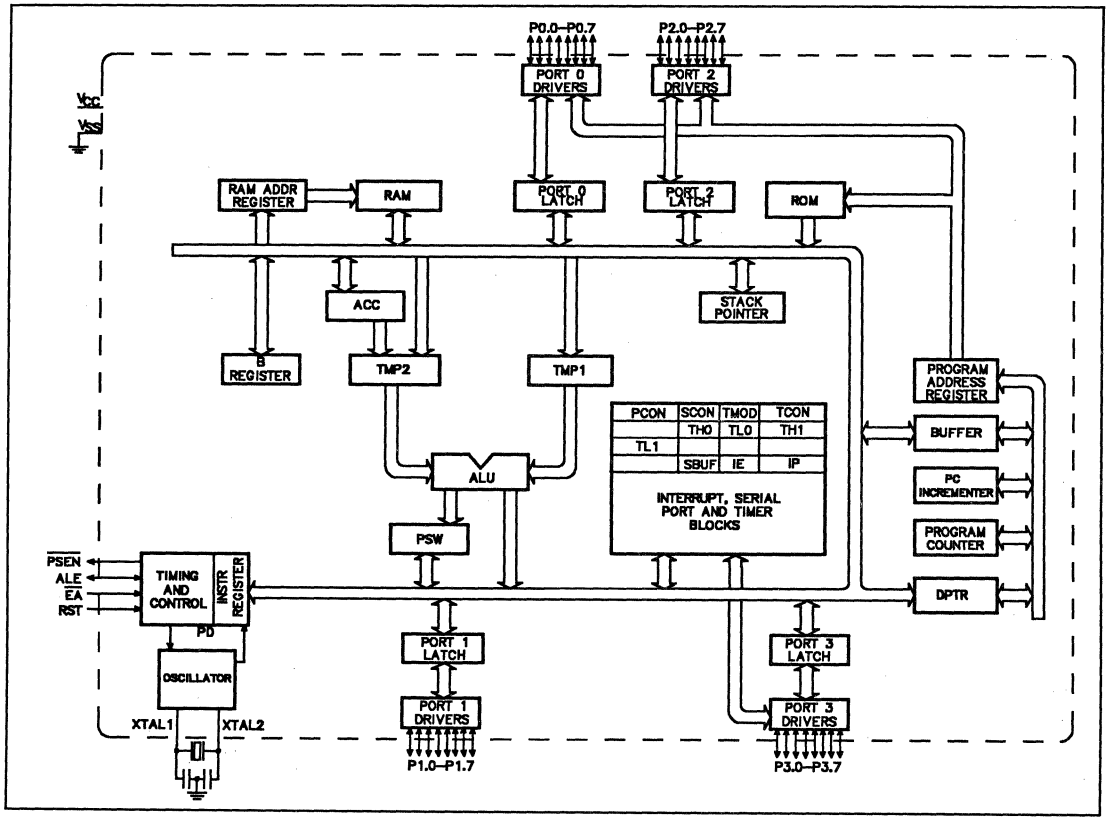
Package
A = Plastic LCC
N = Plastic DIP
B = Plastic QFP

Speed
B = 0.5 to 12MHz
C = 3.5 to 12MHz
G = 3.5 to 16MHz

Operating Temperature Range
A = −40°C to +85°C
C = 0°C to +70°C

PART NUMBER SELECTION			
ROMless	**ROM**	**Temperature and Package**	**Frequency**
SC80C31BCCN40	SC80C51BCCN40	0 to +70°C plastic DIP	3.5 to 12MHz
SC80C31BCGN40	SC80C51BCGN40	0 to +70°C plastic DIP	3.5 to 16MHz
SC80C31BCBN40	SC80C51BCBN40	0 to +70°C plastic DIP	0.5 to 12MHz
SC80C31BCCA44	SC80C51BCCA44	0 to +70°C plastic LCC	3.5 to 12MHz
SC80C31BCGA44	SC80C51BCGA44	0 to +70°C plastic LCC	3.5 to 16MHz
SC80C31BCBA44	SC80C51BCBA44	0 to +70°C plastic LCC	0.5 to 12MHz
SC80C31BACN40	SC80C51BACN40	−40 to +85°C plastic DIP	3.5 to 12MHz
SC80C31BAGN40	SC80C51BAGN40	−40 to +85°C plastic DIP	3.5 to 16MHz
SC80C31BACA44	SC80C51BACA44	−40 to +85°C plastic LCC	3.5 to 12MHz
SC80C31BAGA44	SC80C51BAGA44	−40 to +85°C plastic LCC	3.5 to 16MHz
SC80C31BCCB44	SC80C51BCCB44	0 to +70°C plastic QFP	3.5 to 12MHz
SC80C31BCGB44	SC80C51BCGB44	0 to +70°C plastic QFP	3.5 to 16MHz

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. LCC/QFP	TYPE	NAME AND FUNCTION
V_{SS}	20	22	I	**Ground:** 0V reference.
		23	I	**Ground:** 0V reference. (QFP only)
V_{CC}	40	44	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–P0.7	39–32	43–36	I/O	**Port 0:** Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pullups when emitting 1s. Port 0 also outputs the code bytes during program verification in the SC80C31B/SC80C51B. External pull-ups are required during program verification.
P1.0–P1.7	1–8	2–9	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pullups. Port 1 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pullups. (See DC Electrical Characteristics: I_{IL}). Port 1 also receives the low-order address byte during program memory verification.
P2.0–P2.7	21–28	24–31	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pullups. Port 2 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pullups. (See DC Electrical Characteristics: I_{IL}). Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application, it uses strong internal pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @Ri), port 2 emits the contents of the P2 special function register.
P3.0–P3.7	10–17	11, 13–19	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 3 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 3 also serves the special features of the SC80C51 family, as listed below:
	10	11	I	**RxD (P3.0):** Serial input port
	11	13	O	**TxD (P3.1):** Serial output port
	12	14	I	**INT0 (P3.2):** External interrupt
	13	15	I	**INT1 (P3.3):** External interrupt
	14	16	I	**T0 (P3.4):** Timer 0 external input
	15	17	I	**T1 (P3.5):** Timer 1 external input
	16	18	O	**WR (P3.6):** External data memory write strobe
	17	19	O	**RD (P3.7):** External data memory read strobe
RST	9	10	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal diffused resistor to V_{SS} permits a power-on reset using only an external capacitor to V_{CC}.
ALE	30	33	I/O	**Address Latch Enable:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory.
$\overline{PSEN}$	29	32	O	**Program Store Enable:** The read strobe to external program memory. When the SC80C31B/SC80C51B is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory.
$\overline{EA}$	31	35	I	**External Access Enable:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations 0000H and 0FFFH. If $\overline{EA}$ is held high, the device executes from internal program memory unless the program counter contains an address greater than 0FFFH.
XTAL1	19	21	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits.
XTAL2	18	20	O	**Crystal 2:** Output from the inverting oscillator amplifier.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol, page 1.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. To insure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

Table 1 shows the state of I/O ports during low current operating modes.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	Port 0	Port 1	Port 2	Port 3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power–down	Internal	0	0	Data	Data	Data	Data
Power–down	External	0	0	Float	Data	Data	Data

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Storage temperature range	−65 to +150	°C
Voltage on any other pin to V_{SS}	−0.5 to + 6.5	V
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.5	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V ±20%, V_{SS} = 0V

Symbol	Parameter	Test Conditions	Limits			Unit
			Min	Typical[1]	Max	
V_{IL}	Input low voltage, except $\overline{EA}$		−0.5		$0.2V_{CC}$−0.1	V
V_{IL1}	Input low voltage to $\overline{EA}$		0		$0.2V_{CC}$−0.3	V
V_{IH}	Input high voltage, except XTAL1, RST		$0.2V_{CC}$+.9		V_{CC}+0.5	V
V_{IH1}	Input high voltage, XTAL1, RST		$0.7V_{CC}$		V_{CC}+0.5	V
V_{OL}	Output low voltage, ports 1, 2, 3	I_{OL} = 1.6mA[2]			0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$	I_{OL} = 3.2mA[2]			0.45	V
V_{OH}	Output high voltage, ports 1, 2, 3	V_{CC} = 5V±10%, I_{OH} = −60µA	2.4			V
		I_{OH} = −25µA	$0.75V_{CC}$			V
		I_{OH} = −10µA	$0.9V_{CC}$			V
V_{OH1}	Output high voltage (port 0 in external bus mode, ALE, $\overline{PSEN}$)[3]	V_{CC} = 5V±10%, I_{OH} = −800µA	2.4			V
		I_{OH} = −300µA	$0.75V_{CC}$			V
		I_{OH} = −80µA	$0.9V_{CC}$			V
I_{IL}	Logical 0 input current, ports 1, 2, 3	V_{IN} = 0.45V			−50	µA
I_{TL}	Logical 1-to-0 transition current, ports 1, 2, 3	See note 4			−650	µA
I_{LI}	Input leakage current, port 0	V_{IN} = V_{IL} or V_{IH}			±10	µA
I_{CC}	Power supply current: Active mode @ 12MHz[5] Idle mode @ 12MHz[5] Power down mode	See note 6		11.5 1.3 3	25 4 50	mA mA µA
R_{RST}	Internal reset pulldown resistor		50		300	kΩ
C_{IO}	Pin capacitance				10	pF

NOTES:
1. Typical ratings are based on a limited number of samples taken from early manufacturing lots and are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the $0.9V_{CC}$ specification when the address bits are stabilizing.
4. Pins of ports 1, 2, and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. I_{CC}MAX at other frequencies is given by:
   Active mode: I_{CC}MAX = 0.94 X FREQ + 13.71
   Idle mode: I_{CC}MAX = 0.14 X FREQ + 2.31
   where FREQ is the external oscillator frequency in MHz. I_{CC}MAX is given in mA. See Figure 8.
6. See Figures 9 through 12 for I_{CC} test conditions.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V ±20%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	1	Oscillator frequency: **Speed Versions** SC80C31B/SC80C51B B SC80C31B/SC80C51B C SC80C31B/SC80C51B G			 0.5 3.5 3.5	 12 12 16	 MHz MHz MHz
t_{LHLL}	1	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	1	Address valid to ALE low	28		$t_{CLCL}-55$		ns
t_{LLAX}	1	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	1	ALE low to valid instruction in		234		$4t_{CLCL}-100$	ns
t_{LLPL}	1	ALE low to $\overline{PSEN}$ low	43		$t_{CLCL}-40$		ns
t_{PLPH}	1	$\overline{PSEN}$ pulse width	205		$3t_{CLCL}-45$		ns
t_{PLIV}	1	$\overline{PSEN}$ low to valid instruction in		145		$3t_{CLCL}-105$	ns
t_{PXIX}	1	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	1	Input instruction float after $\overline{PSEN}$		59		$t_{CLCL}-25$	ns
t_{AVIV}	1	Address to valid instruction in		312		$5t_{CLCL}-105$	ns
t_{PLAZ}	1	$\overline{PSEN}$ low to address float		10		10	ns
Data Memory							
t_{RLRH}	2, 3	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	2, 3	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	2, 3	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	2, 3	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	2, 3	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	2, 3	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	2, 3	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	2, 3	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	2, 3	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	2, 3	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{WHQX}	2, 3	Data hold after $\overline{WR}$	33		$t_{CLCL}-50$		ns
t_{RLAZ}	2, 3	$\overline{RD}$ low to address float		0		0	ns
t_{WHLH}	2, 3	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
External Clock							
t_{CHCX}	5	High time	20		20		ns
t_{CLCX}	5	Low time	20		20		ns
t_{CLCH}	5	Rise time		20		20	ns
t_{CHCL}	5	Fall time		20		20	ns
Shift Register							
t_{XLXL}	4	Serial port clock cycle time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	4	Output data setup to clock rising edge	700		$10t_{CLCL}-133$		ns
t_{XHQX}	4	Output data hold after clock rising edge	50		$2t_{CLCL}-117$		ns
t_{XHDX}	4	Input data hold after clock rising edge	0		0		ns
t_{XHDV}	4	Clock rising edge to input data valid		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P – $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.

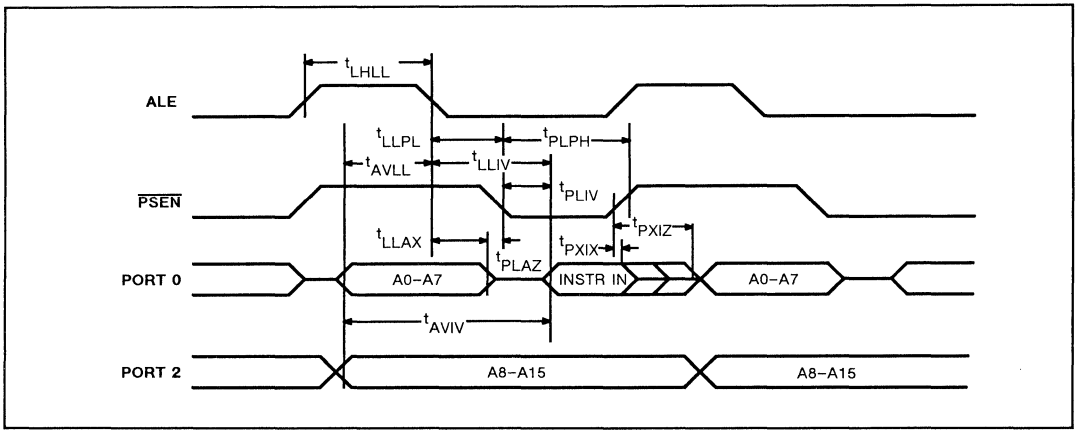


Figure 1. External Program Memory Read Cycle

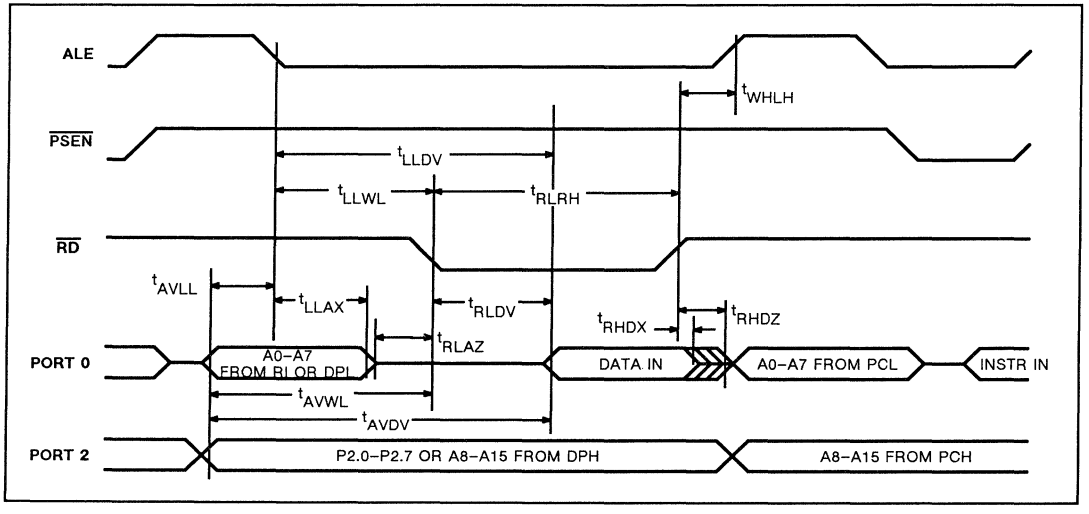


Figure 2. External Data Memory Read Cycle

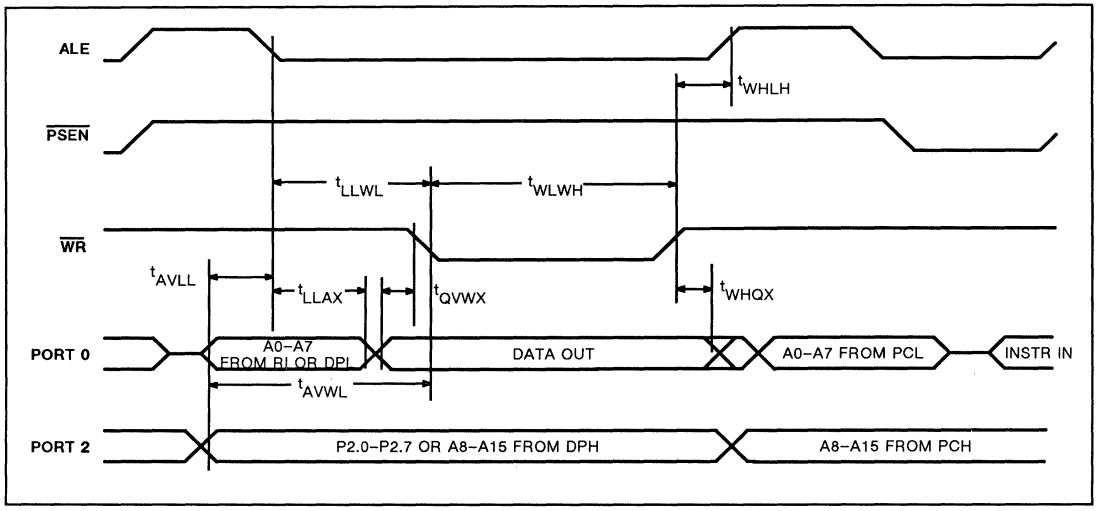


Figure 3. External Data Memory Write Cycle

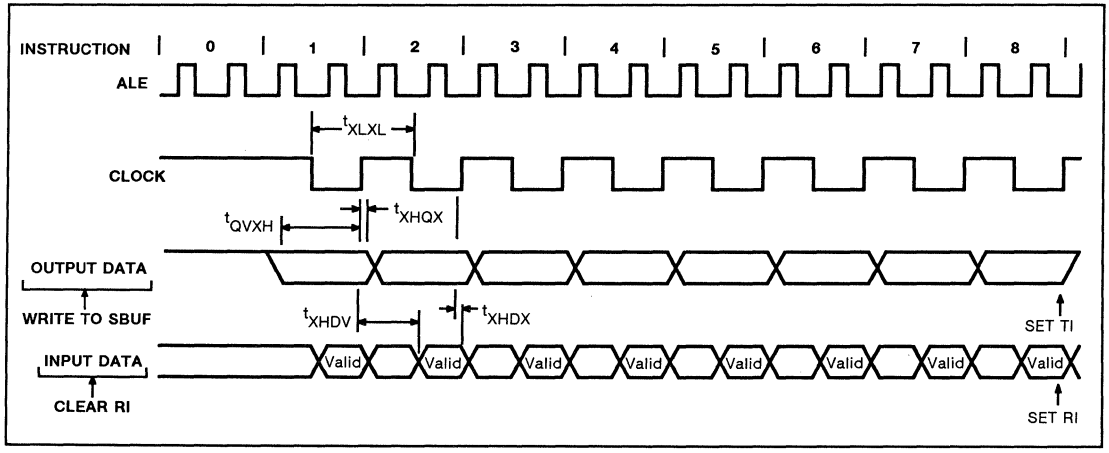


Figure 4. Shift Register Mode Timing

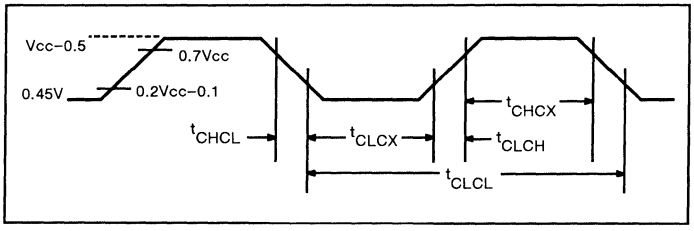
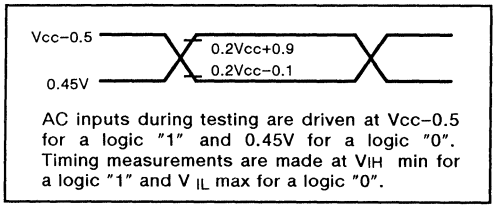
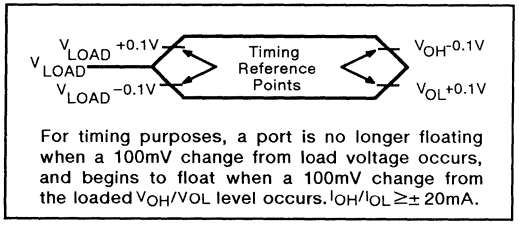


Figure 5. External Clock Drive



AC inputs during testing are driven at Vcc-0.5 for a logic "1" and 0.45V for a logic "0". Timing measurements are made at V_{IH} min for a logic "1" and V_{IL} max for a logic "0".

Figure 6. AC Testing Input/Output



For timing purposes, a port is no longer floating when a 100mV change from load voltage occurs, and begins to float when a 100mV change from the loaded V_{OH}/V_{OL} level occurs. $I_{OH}/I_{OL} \geq \pm 20$mA.

Figure 7. Float Waveform

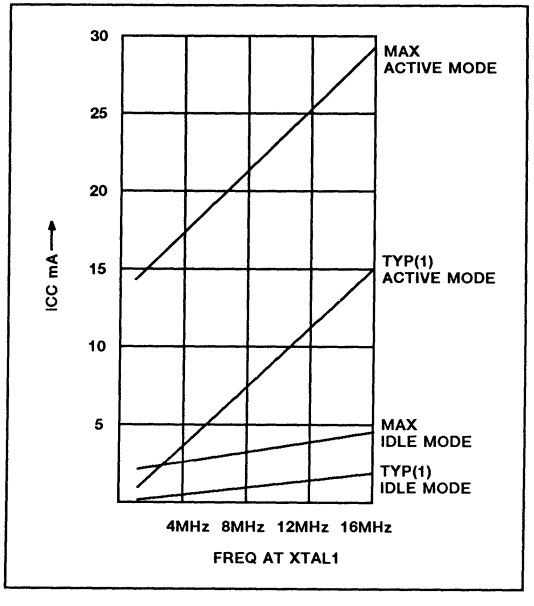


Figure 8. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test

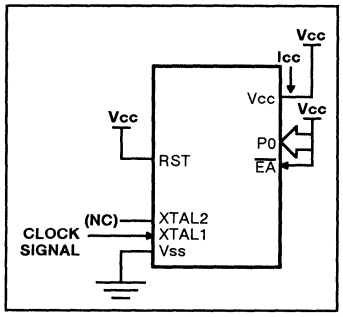


Figure 9. I_{CC} Test Condition, Active Mode
All other pins are disconnected

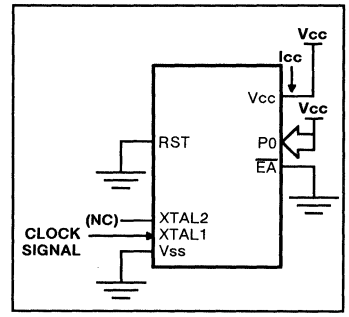


Figure 10. I_{CC} Test Condition, Idle Mode
All other pins are disconnected

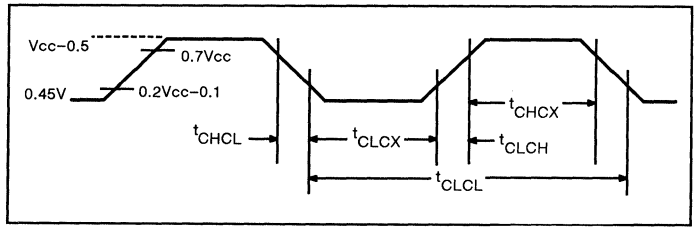


Figure 11. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

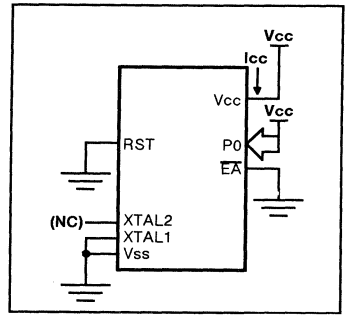


Figure 12. I_{CC} Test Conditions, Power Down Mode
All other pins are disconnected. V_{CC} = 2V to 5.5V

Signetics

SC87C51
CMOS Single-Chip 8-Bit EPROM Microcontroller

Product Specification

Microprocessor Division

DESCRIPTION

The Signetics SC87C51 is a high-performance microcontroller fabricated with Signetics high-density CMOS technology. The CMOS SC87C51 is functionally compatible with the NMOS SCN8031/SCN8051 and SC80C51 microcontrollers. The Signetics CMOS technology combines the high speed and density characteristics of HMOS with the low power attributes of CMOS. Signetics' epitaxial substrate minimizes latch-up sensitivity.

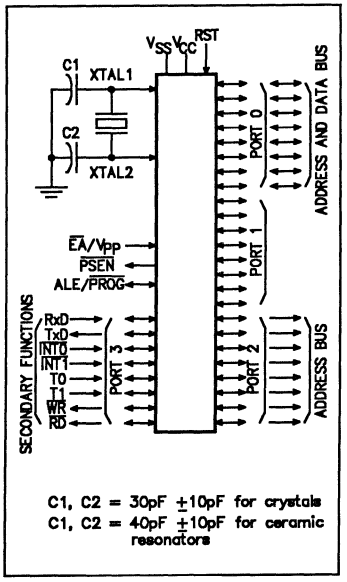
The SC87C51 contains a 4K x 8 EPROM, a 128 x 8 RAM, 32 I/O lines, two 16-bit counter/timers, a five-source, two priority level nested interrupt structure, a serial I/O port for either multiprocessor communications, I/O expansion or full duplex UART, and on-chip oscillator and clock circuits.

In addition, the SC87C51 has two software selectable modes of power reduction – idle mode and power-down mode. The idle mode freezes the CPU while allowing the RAM, timers, serial port, and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator, causing all other chip functions to be inoperative.

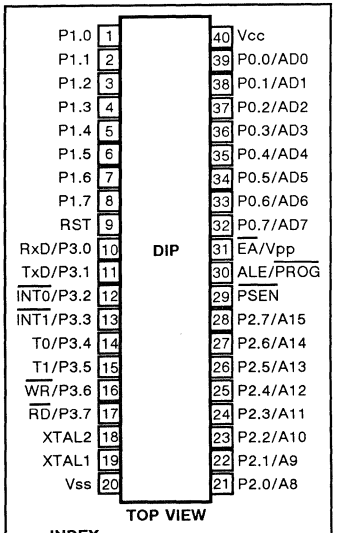
FEATURES

- **SCN8031/SCN8051/SC80C51 compatible**
  - **4K x 8 EPROM**
  - **128 x 8 RAM**
  - **Two 16-bit counter/timers**
  - **Full duplex serial channel**
  - **Boolean processor**
- **Memory addressing capability**
  - **64K ROM and 64K RAM**
- **Power control modes:**
  - **Idle mode**
  - **Power-down mode**
- **CMOS and TTL compatible**
- **Three speed ranges at Vcc = 5V ±10%**
  - **3.5 to 12MHz**
  - **3.5 to 16MHz**
  - **0.5 to 12MHz**
- **Four package styles**
- **Extended temperature ranges**
- **OTP package available**

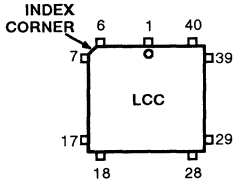
LOGIC SYMBOL



PIN CONFIGURATION



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE/PROG
12	NC	34	NC
13	TxD/P3.1	35	EA/Vpp
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

CMOS Single-Chip 8-Bit EPROM Microcontroller — SC87C51

ORDERING INFORMATION

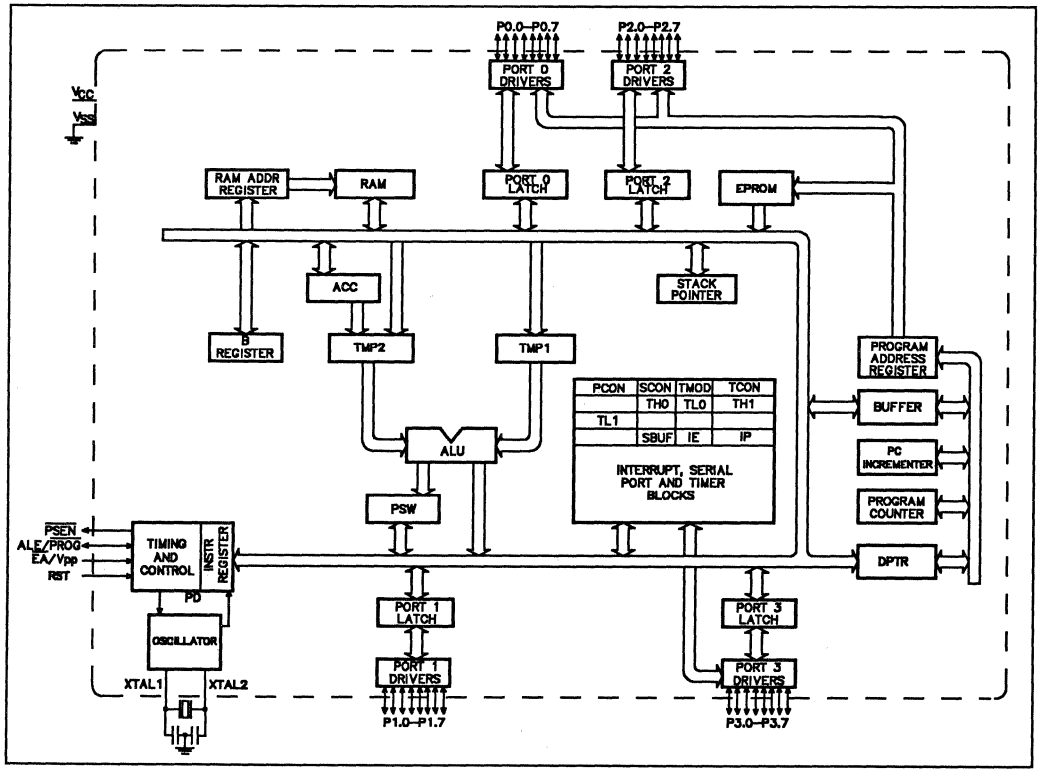
SC87C51 □□□□□

- OPERATING TEMPERATURE RANGE
  - A = −40 °C to +85 °C
  - C = 0 °C to +70 °C
- SPEED
  - B = 0.5 to 12MHz
  - C = 3.5 to 12MHz
  - G = 3.5 to 16MHz
- 40 = 40 pin DIP
- 44 = 44 pin LCC
- PACKAGE
  - A = Plastic LCC
  - F = Ceramic DIP
  - L = Ceramic LCC
  - N = Plastic DIP

PART NUMBER SELECTION		
Part Number	**Speed**	**Temperature and Package**
SC87C51CCF40	3.5 to 12MHz	0 to +70°C, ceramic DIP
SC87C51CGF40	3.5 to 16MHz	0 to +70°C, ceramic DIP
SC87C51CBF40	0.5 to 12MHz	0 to +70°C, ceramic DIP*
SC87C51CCL44	3.5 to 12MHz	0 to +70°C, ceramic LCC
SC87C51CGL44	3.5 to 16MHz	0 to +70°C, ceramic LCC
SC87C51CBL44	0.5 to 12MHz	0 to +70°C, ceramic LCC*
SC87C51CCN40	3.5 to 12MHz	0 to +70°C, plastic DIP
SC87C51CGN40	3.5 to 16MHz	0 to +70°C, plastic DIP
SC87C51CBN40	0.5 to 12MHz	0 to +70°C, plastic DIP*
SC87C51CCA44	3.5 to 12MHz	0 to +70°C, plastic LCC
SC87C51CGA44	3.5 to 16MHz	0 to +70°C, plastic LCC
SC87C51CBA44	0.5 to 12MHz	0 to +70°C, plastic LCC*
SC87C51ACN40	3.5 to 12MHz	−40 to +85°C, plastic DIP
SC87C51AGN40	3.5 to 16MHz	−40 to +85°C, plastic DIP
SC87C51ACA44	3.5 to 12MHz	−40 to +85°C, plastic LCC
SC87C51AGA44	3.5 to 16MHz	−40 to +85°C, plastic LCC
SC87C51ABN40	0.5 to 12MHz	−40 to +85°C, plastic DIP*
SC87C51ABA44	0.5 to 12MHz	−40 to +85°C, plastic LCC*
SC87C51ACF40	3.5 to 12MHz	−40 to +85°C, ceramic DIP
SC87C51ACL44	3.5 to 12MHz	−40 to +85°C, ceramic LCC
SC87C51ABL44	0.5 to 12MHz	−40 to +85°C, ceramic LCC*
SC87C51AGL44	3.5 to 16MHz	−40 to +85°C, ceramic LCC
SC87C51AGF40	3.5 to 16MHz	−40 to +85°C, ceramic DIP
SC87C51ABF40	0.5 to 12MHz	−40 to +85°C, ceramic DIP*

*Contact Factory

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. LCC	TYPE	NAME AND FUNCTION
V_{SS}	20	22	I	**Ground:** 0V reference.
V_{CC}	40	44	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–P0.7	39–32	43–36	I/O	**Port 0:** Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pullups when emitting 1s. Port 0 also outputs the code bytes during program verification in the SC87C51. External pull-ups are required during program verification.
P1.0–P1.7	1–8	2–9	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. Port 1 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 1 also receives the low-order address byte during program memory verification.
P2.0–P2.7	21–28	24–31	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. Port 2 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pullups. (See DC Electrical Characteristics: I_{IL}). Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application, it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @Ri), port 2 emits the contents of the P2 special function register.
P3.0–P3.7	10–17	11, 13–19	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 3 pins that have 1s written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 3 also serves the special features of the SC80C51 family, as listed below:
	10	11	I	**RxD (P3.0):** Serial input port
	11	13	O	**TxD (P3.1):** Serial output port
	12	14	I	**INT0 (P3.2):** External interrupt
	13	15	I	**INT1 (P3.3):** External interrupt
	14	16	I	**T0 (P3.4):** Timer 0 external input
	15	17	I	**T1 (P3.5):** Timer 1 external input
	16	18	O	**WR (P3.6):** External data memory write strobe
	17	19	O	**RD (P3.7):** External data memory read strobe
RST	9	10	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal diffused resistor to V_{SS} permits a power-on reset using only an external capacitor to V_{CC}.
ALE/$\overline{PROG}$	30	33	I/O	**Address Latch Enable/Program Pulse:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory. This pin is also the progam pulse input ($\overline{PROG}$) during EPROM programming.
$\overline{PSEN}$	29	32	O	**Program Store Enable:** The read strobe to external program memory. When the SC87C51 is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory.
$\overline{EA}$/V_{PP}	31	35	I	**External Access Enable/Programming Supply Voltage:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations 0000H through 0FFFH. If $\overline{EA}$ is held high, the device executes from internal program memory unless the program counter contains an address greater than 0FFFH. This pin also receives the 12.75V programming supply voltage (V_{PP}) during EPROM programming.
XTAL1	19	21	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits.
XTAL2	18	20	O	**Crystal 2:** Output from the inverting oscillator amplifier.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol, page 1.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

DESIGN CONSIDERATIONS

At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up.

When the idle mode is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write when idle is terminated by reset, the instruction following the one that invokes idle should not be one that writes to a port pin or to external memory.

Table 1 shows the state of I/O ports during low current operating modes.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	Port 0	Port 1	Port 2	Port 3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

Electrical Deviations from Commercial Specifications for Extended Temperature Range

D.C. and A.C. parameters not included here are the same as in the commercial temperature range table.

DC ELECTRICAL CHARACTERISTICS $T_A = -40°C$ to $+85°C$, $V_{CC} = 5V \pm 10\%$, $V_{SS} = 0V$

Symbol	Parameter	Test Conditions	Limits		Unit
			Min	Max	
V_{IL}	Input low volatage (except $\overline{EA}$)		-0.5	$0.2V_{CC}-0.15$	V
V_{IL1}	$\overline{EA}$		0	$0.2V_{CC}-0.35$	V
V_{IH}	Input high voltage (except XTAL1, RST)		$0.2V_{CC}+1$	$V_{CC}+0.5$	V
V_{IH1}	Input high voltage to XTAL1, RST		$0.7V_{CC}+0.1$	$V_{CC}+0.5$	V
I_{IL}	Logical 0 input current (port 1, 2, 3)	$V_{IN} = 0.45V$		-75	µA
I_{TL}	Logical 1 to 0 transition current (ports 1, 2, 3)	$V_{IN} = 2.0V$		-750	µA
I_{CC}	Power supply current Active mode Idle mode Power down mode	$V_{CC} = 4.5$–$5.5V$, Frequency range = 3.5 to 12MHz		 35 6 50	 mA mA µA

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Operating temperature under bias	0 to +70 or −40 to +85	°C
Storage temperature range	−65 to +150	°C
Voltage on $\overline{EA}$/V_{PP} pin to V_{SS}	0 to +13.0	V
Voltage on any other pin to V_{SS}	−0.5 to + 6.5	V
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.5	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or −40°C to +85°C, V_{CC} = 5V ±10%, V_{SS} = 0V

Symbol	Parameter	Test Conditions	Limits			Unit
			Min	Typical	Max	
V_{IL}	Input low voltage, except $\overline{EA}$[7]		−0.5		$0.2V_{CC}$−0.1	V
V_{IL1}	Input low voltage to $\overline{EA}$[7]		0		$0.2V_{CC}$−0.3	V
V_{IH}	Input high voltage, except XTAL1, RST[7]		$0.2V_{CC}$+.9		V_{CC}+0.5	V
V_{IH1}	Input high voltage, XTAL1, RST[7]		$0.7V_{CC}$		V_{CC}+0.5	V
V_{OL}	Output low voltage, ports 1, 2, 3	I_{OL} = 1.6mA[2]			0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$	I_{OL} = 3.2mA[2]			0.45	V
V_{OH}	Output high voltage, ports 1, 2, 3, ALE, $\overline{PSEN}$[3]	I_{OH} = −60µA I_{OH} = −25µA I_{OH} = −10µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
V_{OH1}	Output high voltage (port 0 in external bus mode)	I_{OH} = −800µA I_{OH} = −300µA I_{OH} = −80µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
I_{IL}	Logical 0 input current, ports 1, 2, 3[7]	V_{IN} = 0.45V			−50	µA
I_{TL}	Logical 1−to−0 transition current, ports 1, 2, 3[7]	See note 4			−650	µA
I_{LI}	Input leakage current, port 0	V_{IN} = V_{IL} or V_{IH}			±10	µA
I_{CC}	Power supply current:[7] Active mode @ 12MHz[5] Idle mode @ 12MHz[5] Power down mode	See note 6		11.5 1.3 3	25 4 50	 mA mA µA
R_{RST}	Internal reset pulldown resistor		50		300	kΩ
C_{IO}	Pin capacitance				10	pF

NOTES:
1. Typical ratings are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1−to−0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the $0.9V_{CC}$ specification when the address bits are stabilizing.
4. Pins of ports 1, 2, and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. I_{CC}MAX at other frequencies is given by:
   Active mode: I_{CC}MAX = 0.94 X FREQ + 13.71
   Idle mode: I_{CC}MAX = 0.14 X FREQ + 2.31
   where FREQ is the external oscillator frequency in MHz. I_{CC}MAX is given in mA. See Figure 8.
6. See Figures 9 through 12 for I_{CC} test conditions.
7. These values apply only to T_A = 0°C to +70°C. For T_A = −40°C to +85°C, see table on page 4.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or −40°C to +85°C, V_{CC} = 5V ±10%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	1	Oscillator frequency: **Speed Versions** SC87C51 B SC87C51 C SC87C51 G			 0.5 3.5 3.5	 12 12 16	 MHz MHz MHz
t_{LHLL}	1	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	1	Address valid to ALE low	28		$t_{CLCL}-55$		ns
t_{LLAX}	1	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	1	ALE low to valid instruction in		234		$4t_{CLCL}-100$	ns
t_{LLPL}	1	ALE low to $\overline{PSEN}$ low	43		$t_{CLCL}-40$		ns
t_{PLPH}	1	$\overline{PSEN}$ pulse width	205		$3t_{CLCL}-45$		ns
t_{PLIV}	1	$\overline{PSEN}$ low to valid instruction in		145		$3t_{CLCL}-105$	ns
t_{PXIX}	1	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	1	Input instruction float after $\overline{PSEN}$		59		$t_{CLCL}-25$	ns
t_{AVIV}	1	Address to valid instruction in		312		$5t_{CLCL}-105$	ns
t_{PLAZ}	1	$\overline{PSEN}$ low to address float		10		10	ns
Data Memory							
t_{RLRH}	2, 3	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	2, 3	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	2, 3	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	2, 3	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	2, 3	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	2, 3	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	2, 3	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	2, 3	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	2, 3	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	2, 3	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{WHQX}	2, 3	Data hold after $\overline{WR}$	33		$t_{CLCL}-50$		ns
t_{RLAZ}	2, 3	$\overline{RD}$ low to address float		0		0	ns
t_{WHLH}	2, 3	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
External Clock							
t_{CHCX}	5	High time	20		20		ns
t_{CLCX}	5	Low time	20		20		ns
t_{CLCH}	5	Rise time		20		20	ns
t_{CHCL}	5	Fall time		20		20	ns
Shift Register							
t_{XLXL}	4	Serial port clock cycle time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	4	Output data setup to clock rising edge	700		$10t_{CLCL}-133$		ns
t_{XHQX}	4	Output data hold after clock rising edge	50		$2t_{CLCL}-117$		ns
t_{XHDX}	4	Input data hold after clock rising edge	0		0		ns
t_{XHDV}	4	Clock rising edge to input data valid		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P – $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.

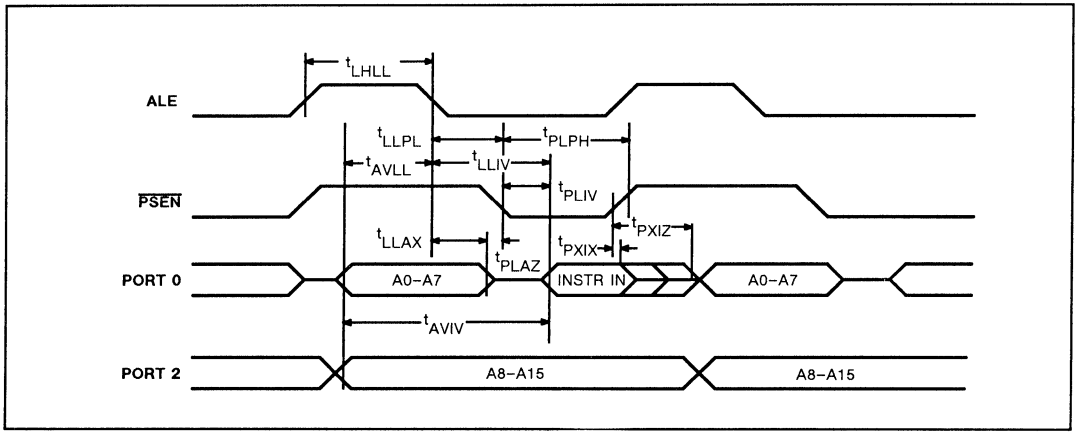


Figure 1. External Program Memory Read Cycle

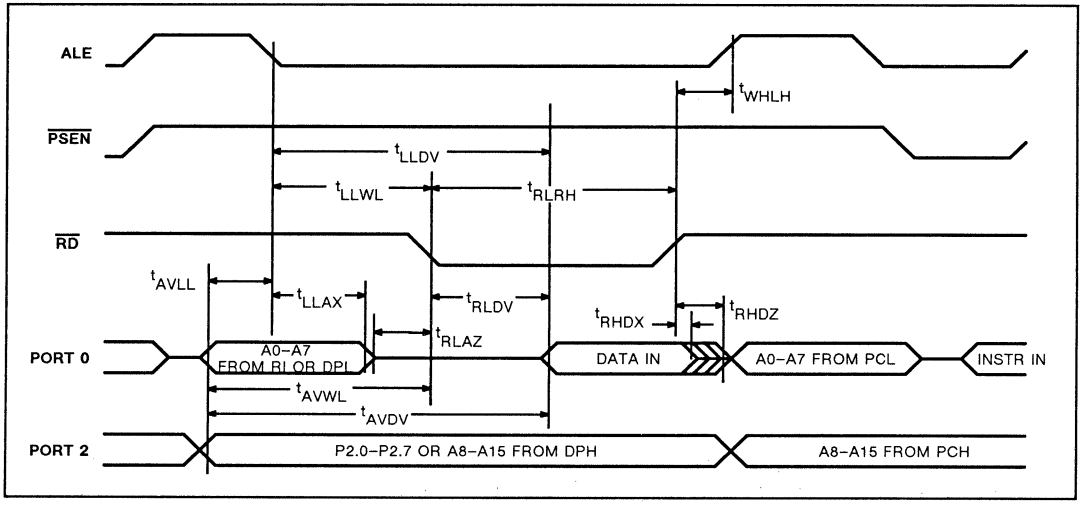


Figure 2. External Data Memory Read Cycle

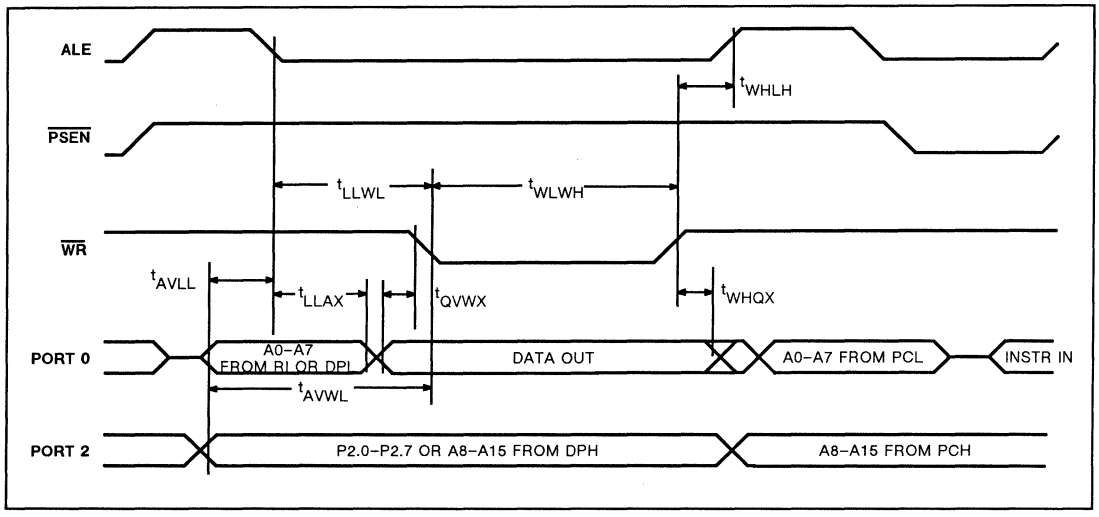


Figure 3. External Data Memory Write Cycle

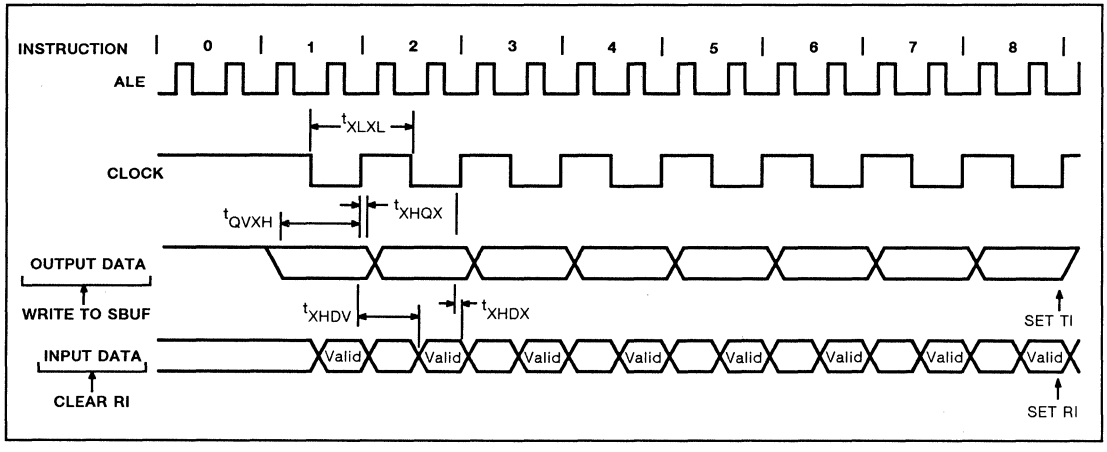


Figure 4. Shift Register Mode Timing

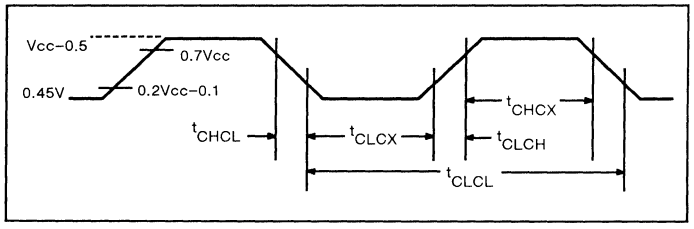


Figure 5. External Clock Drive

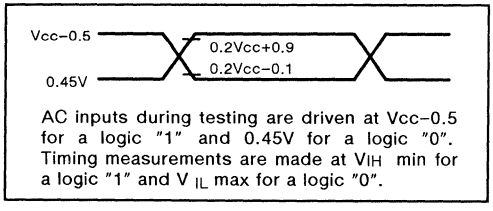


Figure 6. AC Testing Input/Output

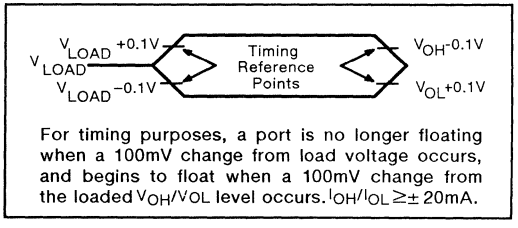


Figure 7. Float Waveform

EPROM CHARACTERISTICS

The SC87C51 is programmed by using a modified Quick-Pulse Programming™ algorithm. It differs from older methods in the value used for VPP (programming supply voltage) and in the width and number of the ALE/$\overline{PROG}$ pulses.

The SC87C51 contains two signature bytes that can be read and used by an EPROM programming system to identify the device. The signature bytes identify the device as an SC87C51 manufactured by Signetics Corporation.

Table 2 shows the logic levels for reading the signature byte, and for programming the program memory, the encryption table, and the lock bits. The circuit configuration and waveforms for quick-pulse programming are shown in Figures 13 and 14. Figure 15 shows the circuit configuration for normal program memory verification.

QUICK-PULSE PROGRAMMING

The setup for microcontroller quick-pulse programming is shown in Figure 13. Note that the SC87C51 is running with a 4 to 6MHz oscillator. The reason the oscillator needs to be running is that the device is executing internal address and program data transfers.

The address of the EPROM location to be programmed is applied to ports 1 and 2, as shown in Figure 13. The code byte to be programmed into that location is applied to port 0. RST, $\overline{PSEN}$ and pins of ports 2 and 3 specified in Table 2 are held at the "Program Code Data" levels indicated in Table 2. The ALE/$\overline{PROG}$ is pulsed low 25 times as shown in Figure 14.

To program the encryption table, repeat the 25 pulse programming sequence for addresses 0 through 1FH, using the "Pgm Encryption Table" levels. Do not forget that after the encryption table is programmed, verification cycles will produce only encrypted data.

To program the lock bits, repeat the 25 pulse programming sequence using the "Pgm Lock Bit" levels. After one lock bit is programmed, further programming of the code memory and encryption table is disabled. However, the other lock bit can still be programmed.

Note that the $\overline{EA}$/VPP pin must not be allowed to go above the maximum specified VPP level for any amount of time. Even a narrow glitch above that voltage can cause permanent damage to the device. The VPP source should be well regulated and free of glitches and overshoot.

Program Verification

If lock bit 2 has not been programmed, the on-chip program memory can be read out for program verification. The address of the program memory locations to be read is applied to ports 1 and 2 as shown in Figure 15. The other pins are held at the "Verify Code Data" levels indicated in Table 2. The contents of the address location will be emitted on port 0. External pull-ups are required on port 0 for this operation.

If the encryption table has been programmed, the data presented at port 0 will be the exclusive NOR of the gram byte with one of the encryption bytes. The user will have to know the encryption table contents in order to correctly decode the verification data. The encryption table itself cannot be read out.

Reading the Signature Bytes

The signature bytes are read by the same procedure as a normal verification of locations 030H and 031H, except that P3.6 and P3.7 need to be pulled to a logic low. The values are:

(030H) = 15H indicates manufactured by Signetics
(031H) = 92H indicates SC87C51

Program/Verify Algorithms

Any algorithm in agreement with the conditions listed in Table 2, and which satisfies the timing specifications, is suitable.

™Trademark phrase of Intel Corp.

Erasure Characteristics

Erasure of the EPROM begins to occur when the chip is exposed to light with wavelengths shorter than approximately 4,000 angstroms. Since sunlight and fluorescent lighting have wavelengths in this range, exposure to these light sources over an extended time (about 1 week in sunlight, or 3 years in room level fluorescent lighting) could cause inadvertent erasure. **For this and secondary effects, it is recommended that an opaque label be placed over the window.** For elevated temperature or solvent environments, use Kapton tape Fluorglas part number 2345-5 or equivalent.

The recommended erasure procedure is exposure to ultraviolet light (at 2537 angstroms) to an integrated dose of at least 15W-sec/cm^2. Exposing the EPROM to an ultraviolet lamp of 12,000μW/cm^2 rating for 20 to 39 minutes, at a distance of about 1 inch, should be sufficient.

Erasure leaves the array in an all 1s state.

Table 2. EPROM Programming Modes

MODE	RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}$/V_{PP}	P2.7	P2.6	P3.7	P3.6
Read signature	1	0	1	1	0	0	0	0
Program code data	1	0	0*	V_{PP}	1	0	1	1
Verify code data	1	0	1	1	0	0	1	1
Pgm encryption table	1	0	0*	V_{PP}	1	0	1	0
Pgm lock bit 1	1	0	0*	V_{PP}	1	1	1	1
Pgm lock bit 2	1	0	0*	V_{PP}	1	1	0	0

NOTES:

1. "0" = valid low for that pin, "1" = valid high for that pin.
2. **V_{PP} = 12.75V ±0.25V.**
3. V_{CC} = 5V ±10% during programming and verification.

*ALE/$\overline{PROG}$ receives 25 programming pulses while V_{PP} is held at 12.75V. Each programming pulse is low for 100μs (±10μs) and high for a minimum of 10μs.

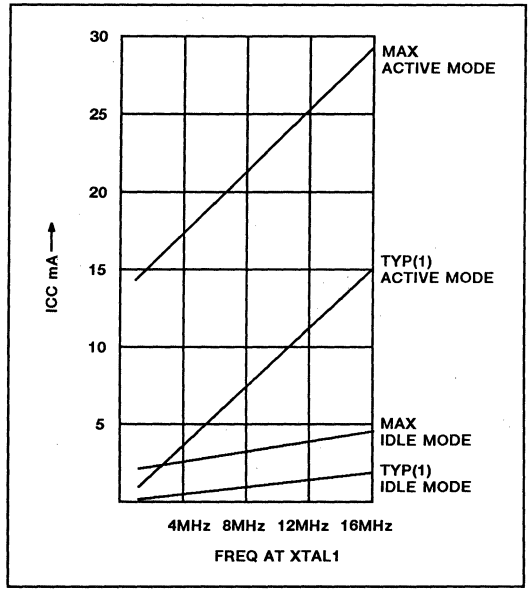


Figure 8. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test

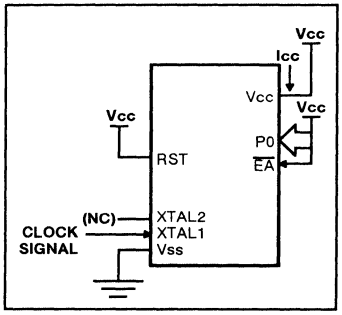


Figure 9. I_{CC} Test Condition, Active Mode
All other pins are disconnected

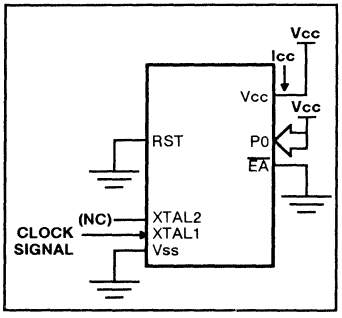


Figure 10. I_{CC} Test Condition, Idle Mode
All other pins are disconnected

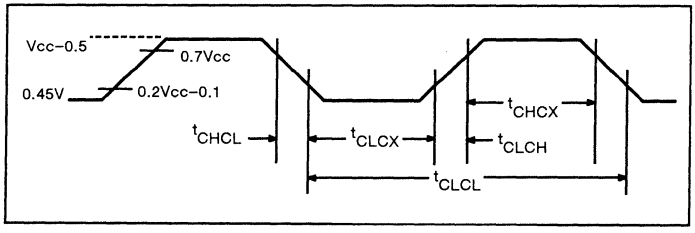


Figure 11. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

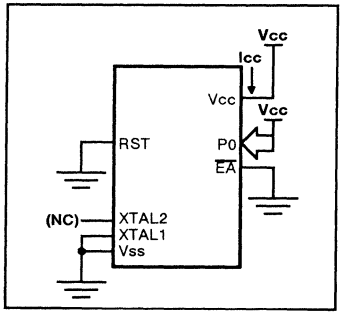


Figure 12. I_{CC} Test Conditions, Power Down Mode
All other pins are disconnected. V_{CC} = 2V to 5.5V

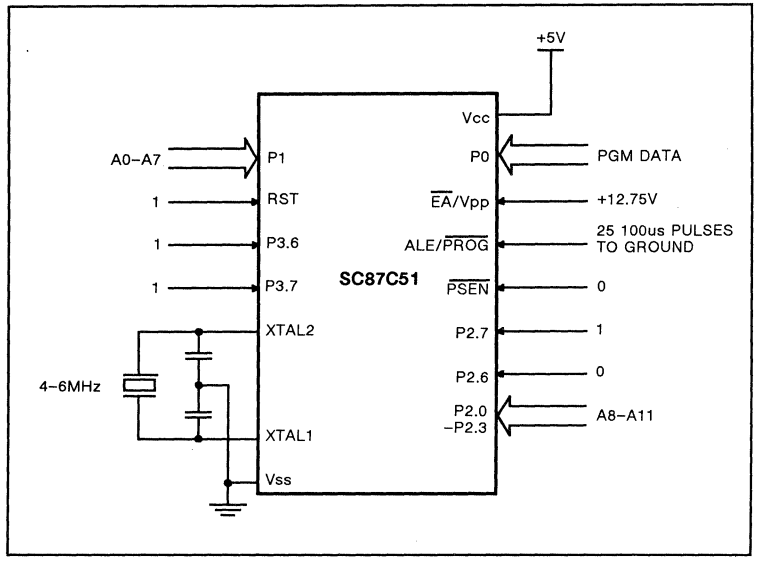


Figure 13. Programming Configuration

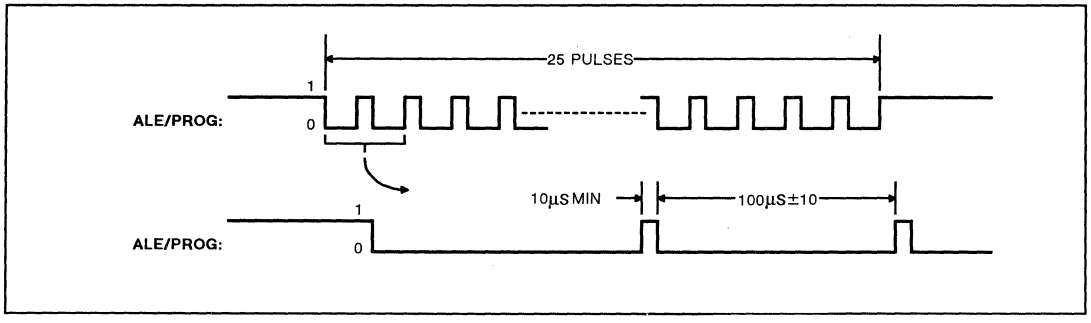


Figure 14. $\overline{\text{PROG}}$ Waveform

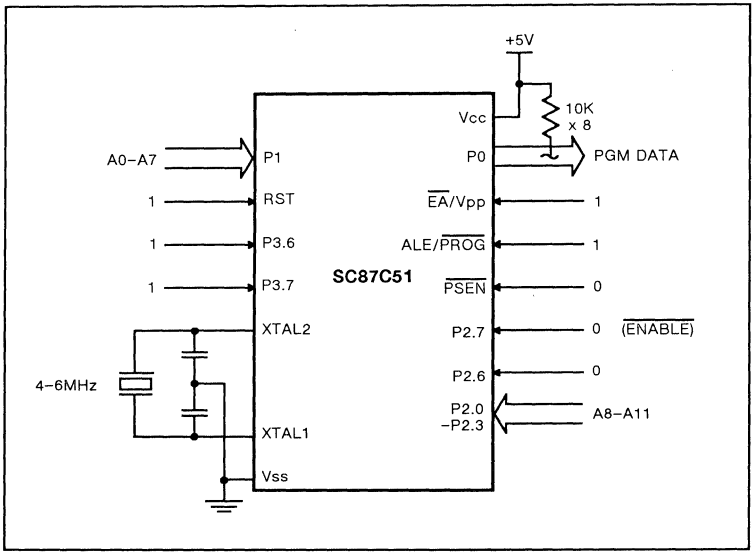


Figure 15. Program Verification

EPROM PROGRAMMING AND VERIFICATION CHARACTERISTICS T_A = 21°C to +27°C, V_{CC} = 5V±10%, V_{SS} = 0V (see Figure 16)

SYMBOL	PARAMETER	MIN	MAX	UNIT
V_{PP}	Programming supply voltage	12.5	13.0	V
I_{PP}	Programming supply current		50	mA
$1/t_{CLCL}$	Oscillator frequency	4	6	MHz
t_{AVGL}	Address setup to $\overline{PROG}$ low	$48t_{CLCL}$		
t_{GHAX}	Address hold after $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	Data setup to $\overline{PROG}$ low	$48t_{CLCL}$		
t_{GHDX}	Data hold after $\overline{PROG}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{ENABLE}$) high to Vpp	$48t_{CLCL}$		
t_{SHGL}	V_{PP} setup to $\overline{PROG}$ low	10		µs
t_{GHSL}	V_{PP} hold after $\overline{PROG}$	10		µs
t_{GLGH}	$\overline{PROG}$ width	90	110	µs
t_{AVQV}	Address to data valid		$48t_{CLCL}$	
t_{ELQV}	$\overline{ENABLE}$ low to data valid		$48t_{CLCL}$	
t_{EHQZ}	Data float after $\overline{ENABLE}$	0	$48t_{CLCL}$	
t_{GHGL}	$\overline{PROG}$ high to $\overline{PROG}$ low	10		µs

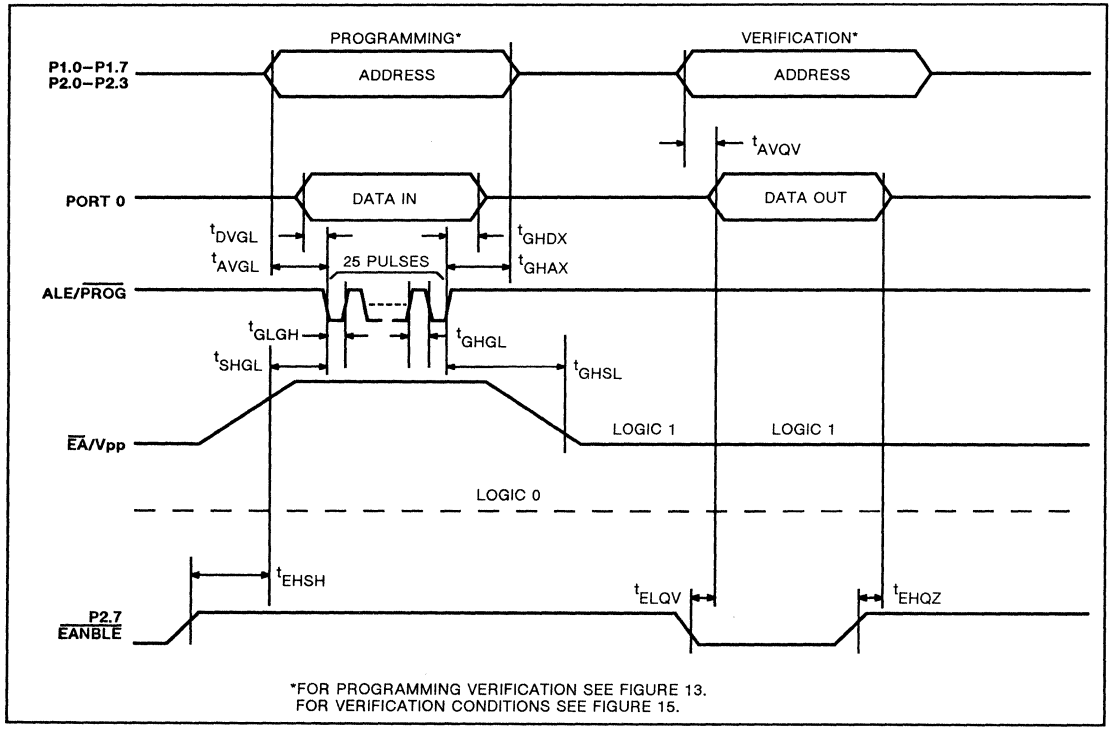


Figure 16. EPROM Programming and Verification

Signetics

Microprocessor Products

Section 2
8051 Family Derivatives

INDEX

INDEX (Continued)

Section 2
8051 FAMILY DERIVATIVES

8032/8052 OVERVIEW

The 8052/8032 are identical to the 8051/8031 respectively except for added features. The 8052/8032 has:

8K ROM (8052 only)
256 bytes RAM
Counter/timer 2

As a result, there are some additions to the interrupt structure, I/O pin alternate functions, and baud rate generation for the serial channel.

Since the similarity is so close, only the additional features of the 8052/8032 are described. Where necessary, some repetition of 8051 information will be made for the sake of clarity. The 8052 is pin for pin and fully code compatible with the 8051.

DIFFERENCES FROM THE 8051

PROGRAM MEMORY

The data and program memory are organized virtually identically to the 8051. The 8052 possesses 8K bytes of on-chip program memory. When $\overline{EA}$ is high, the 8052 fetches instructions from the internal ROM unless the address exceeds 1FFFH. Locations 2000H to FFFFH are fetched from external program memory. When $\overline{EA}$ is held low all instruction fetches are from external memory. The program memory space is shown in Figure 1.

The data memory organization is identical to the 8051 except that the 8052 has an additional 128 bytes of RAM overlapped with the special function register space. This additional RAM is addressed using indirect addressing only and is available as stack space. The 8052 data memory space is shown in Figure 2.

SPECIAL FUNCTION REGISTERS

The special function register space is the same as the 8051 except that the 8052 contains the additional special function registers T2CON, RCAP2L, RCAP2H, TL2 and TH2. Since the standard 8051 on-chip functions are identical in the 8052, the SFR locations, bit locations and operation are likewise identical. The only exceptions are in the interrupt mode and interrupt priority SFR's (see Table 1).

TIMER/COUNTERS

In addition to timer/counters 0 and 1 of the 8051, the 8052 contains timer/counter 2. Like timers 0 and 1, timer 2 can operate as either an event timer or as an event counter. This is selected by bit C/T2 in the special function register T2CON (see Figure 3). It has three operating modes:capture, auto–load, and baud rate generator, which are selected by bits in the T2CON as shown in Table 2.

In the Capture Mode there are two options which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then Timer 2 is a 16-bit timer or counter which upon over-

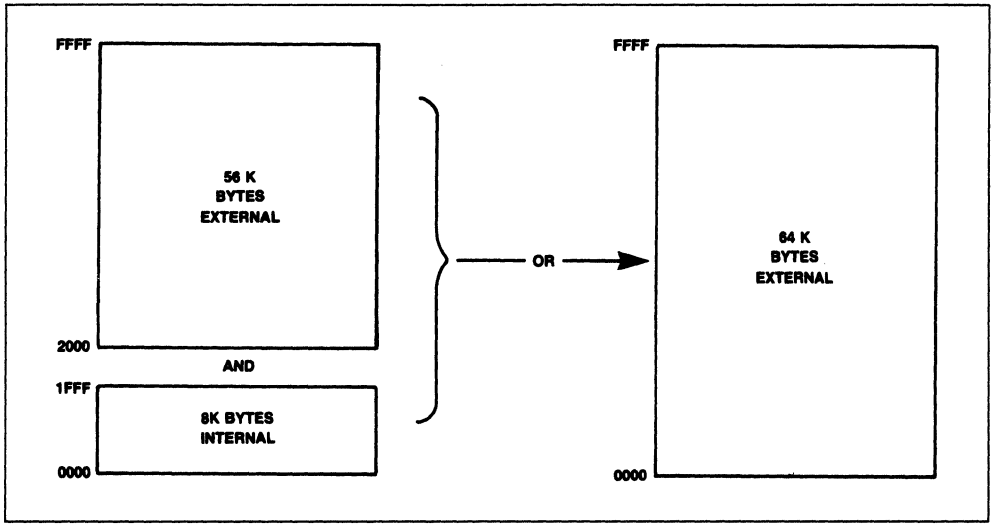


Figure 1. 8052 Program Memory

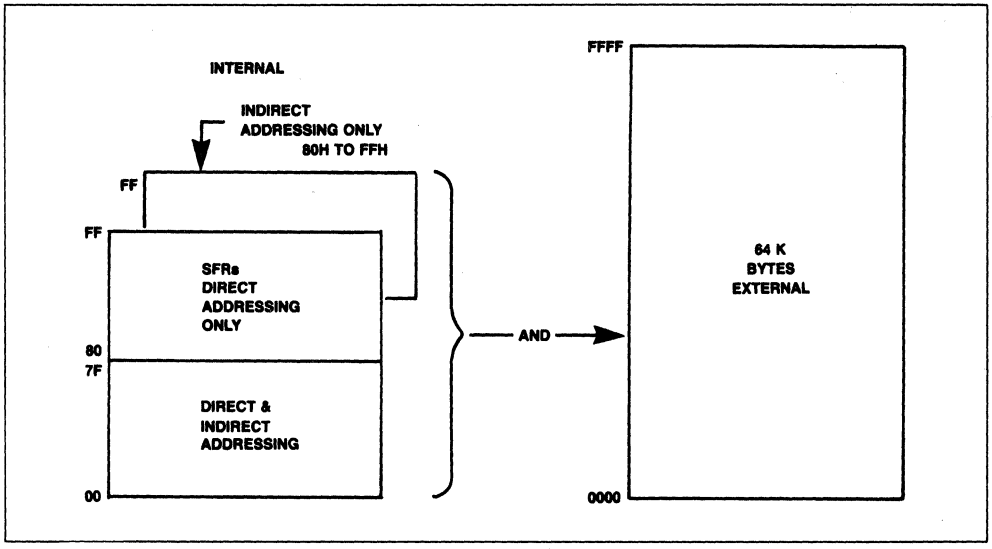


Figure 2. 8052 Data Memory

flowing sets bit TF2, the Timer 2 overflow bit, which can be used to generate an interrupt. If EXEN2 = 1, then 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, TL2 and TH2 to be captured into registers RCAP2L and RCAP2H, respectively. (RCAP2L and RCAP2H are new Special Function Registers in the 8052.) In addition, the transition at T2EX causes bit EXF2 in T2CON to be set, and EXF2 like TF2, can generate an interrupt. The Capture Mode is illustrated in Figure 4.

In the auto-reload mode there are again two options, which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then when Timer 2 rolls over it not only sets TF2 but also causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2L and RCAP2H, which are preset by software. If EXEN2 = 1, then Timer 2 still does the above, but with the added feature that a 1-to-0 transition at external input T2EX will also trigger the 16-bit reload and set EXF2. The auto–reload mode is illustrated in Figure 5.

The baud rate generation mode is selected by RCLK = 1 and/or TCLK = 1. It will be described in conjunction with the serial port.

SERIAL PORT

The serial port of the 8052/8032 is identical to that of the 8051 except that counter/timer 2 can be used to generate baud rates.

In the 8052, Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (see Figure 3). Note that the baud rate 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 6.

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.

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

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

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

$$\text{Modes 1,3 Baud Rate} = \frac{\text{Oscillator Frequency}}{32x\,[65536 - (\text{RCAP2H, RCA2L})]}$$

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

(MSB)							(LSB)
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$

Symbol	Position	Name and Significance
TF2	T2CON.7	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	T2CON.6	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.
RCLK	T2CON.5	Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	T2CON.4	Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	T2CON.3	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	T2CON.2	Start/stop control for Timer 2. A logic 1 starts the timer.
C/$\overline{T2}$	T2CON.1	Timer or counter select. (Timer 2) 0 = Internal timer (OSC/12) 1 = External event counter (falling edge triggered).
CP/$\overline{RL2}$	T2CON.0	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 3. Timer/Counter 2 (T2CON) Control Register

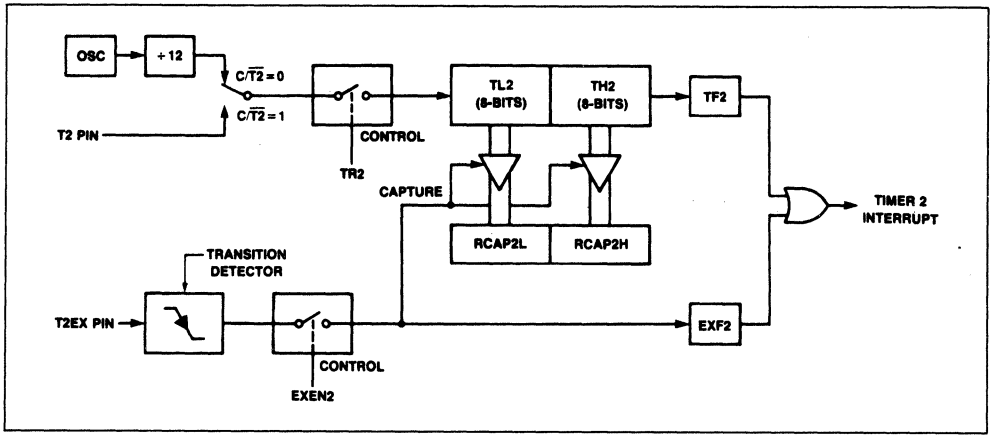


Figure 4. Timer 2 in Capture Mode

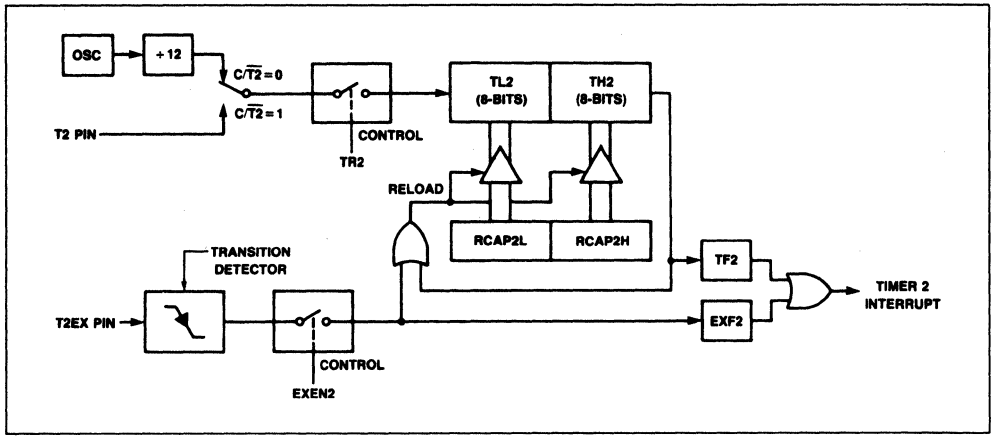


Figure 5. Timer 2 in Auto-Reload Mode

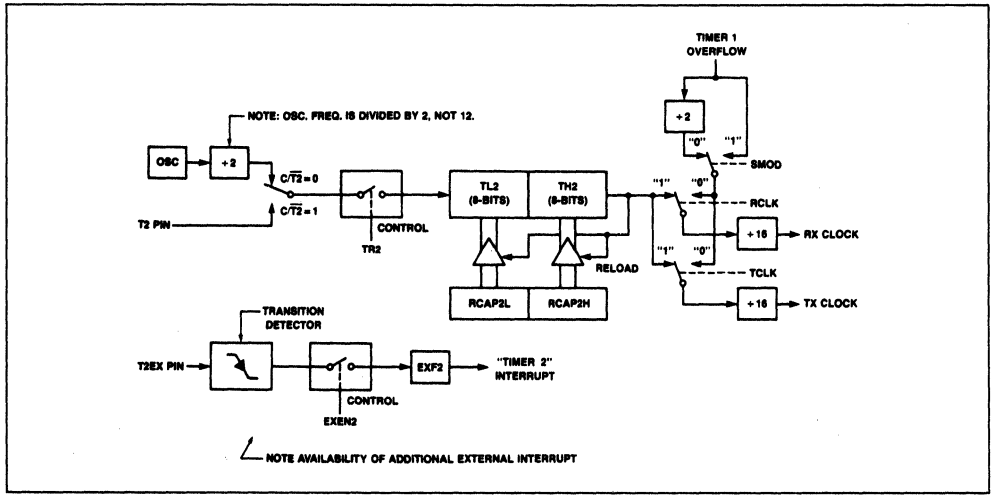


Figure 6. Timer 2 in Baud Rate Generator MOde

Table 1. Special Function Registers

Symbol	Description	Direct Address	Bit Address, Symbol or Alternative Port Function MSB							LSB	Reset Value
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
DPTR:	Data pointer (2 bytes)										
DPH	Data pointer high	83H									00H
DPL	Data pointer low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*	Interrupt enable	A8H	EA	–	ET2	ES	ET1	EX1	ET0	EX0	0x000000B
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt priority	B8H	–	–	PT2	PS	PT1	PX1	PT0	PX0	xx000000B
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	–	–	–	–	–	–	T2EX	T2	FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	A15	A14	A13	A12	A11	A10	A9	A8	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T0	T1	$\overline{INT1}$	$\overline{INT0}$	TxD	RxD	FFH
PCON	Power control	87H	SMOD	–	–	–	GF1	GF0	PD	IDL	0xxxxxxxB
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	–	P	00H
RCAP2H#	Capture high	CBH									00H
RCAP2L#	Capture low	CAH									00H
SBUF	Serial data buffer	99H									xxxxxxxxB
			9F	9E	9D	9C	9B	9A	99	98	
SCON*	Serial controller	98H	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	00H
SP	Stack pointer	81H									07H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00H
			CF	CE	CD	CC	CB	CA	C9	C8	
T2CON*#	Timer 2 control	C8H	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$	00H
TH0	Timer high 0	8CH									00H
TH1	Timer high 1	8DH									00H
TH2#	Timer high 2	CDH									00H
TL0	Timer low 0	8AH									00H
TL1	Timer low 1	8BH									00H
TL2#	Timer low 2	CCH									00H
TMOD	Timer mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H

* = Bit addressable

\# = SFRs are modified from or added to the 80C51 SFRs.

Table 2. Timer 2 Operating Modes

RCLK + TCLK	CP/$\overline{RL2}$	TR2	Mode
0	0	1	16–bit Auto–reload
0	1	1	16–bit Capture
1	X	1	Baud rate generator
X	X	0	(off)

Timer 2 as a baud rate generator is shown in Figure 6. This Figure is valid only if RCLK + 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 RCAP registers may be read, but should not be written to, because a write might overlap a reload and cause write and/or reload errors. Turn the Timer off (clear TR2) before accessing the Timer 2 or RCAP registers, in this case.

The serial port in Modes 1 and 3 with the timer 2 baud rate interface is shown in Figures 7 and 8.

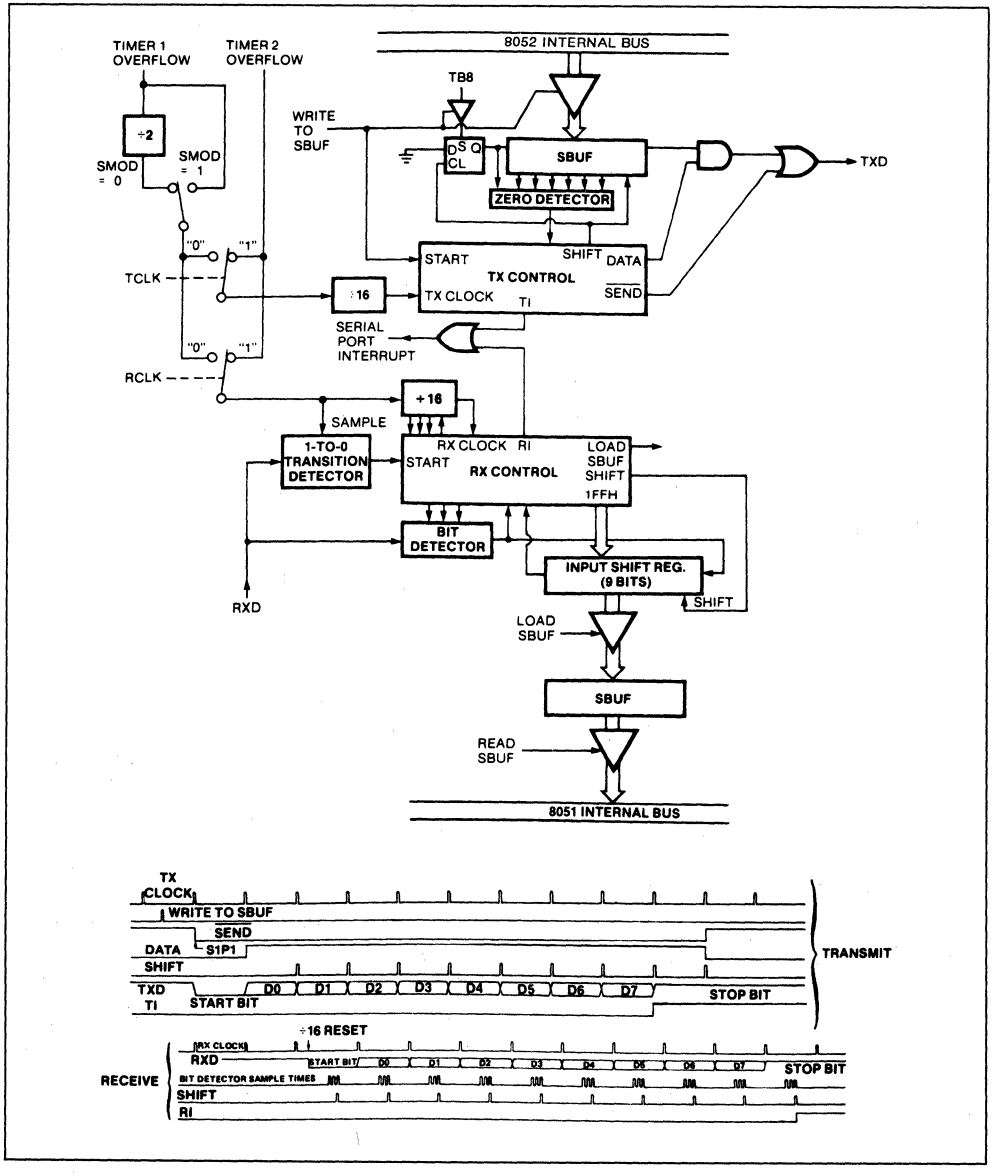


Figure 7. Serial Port Mode 1 in the 8052

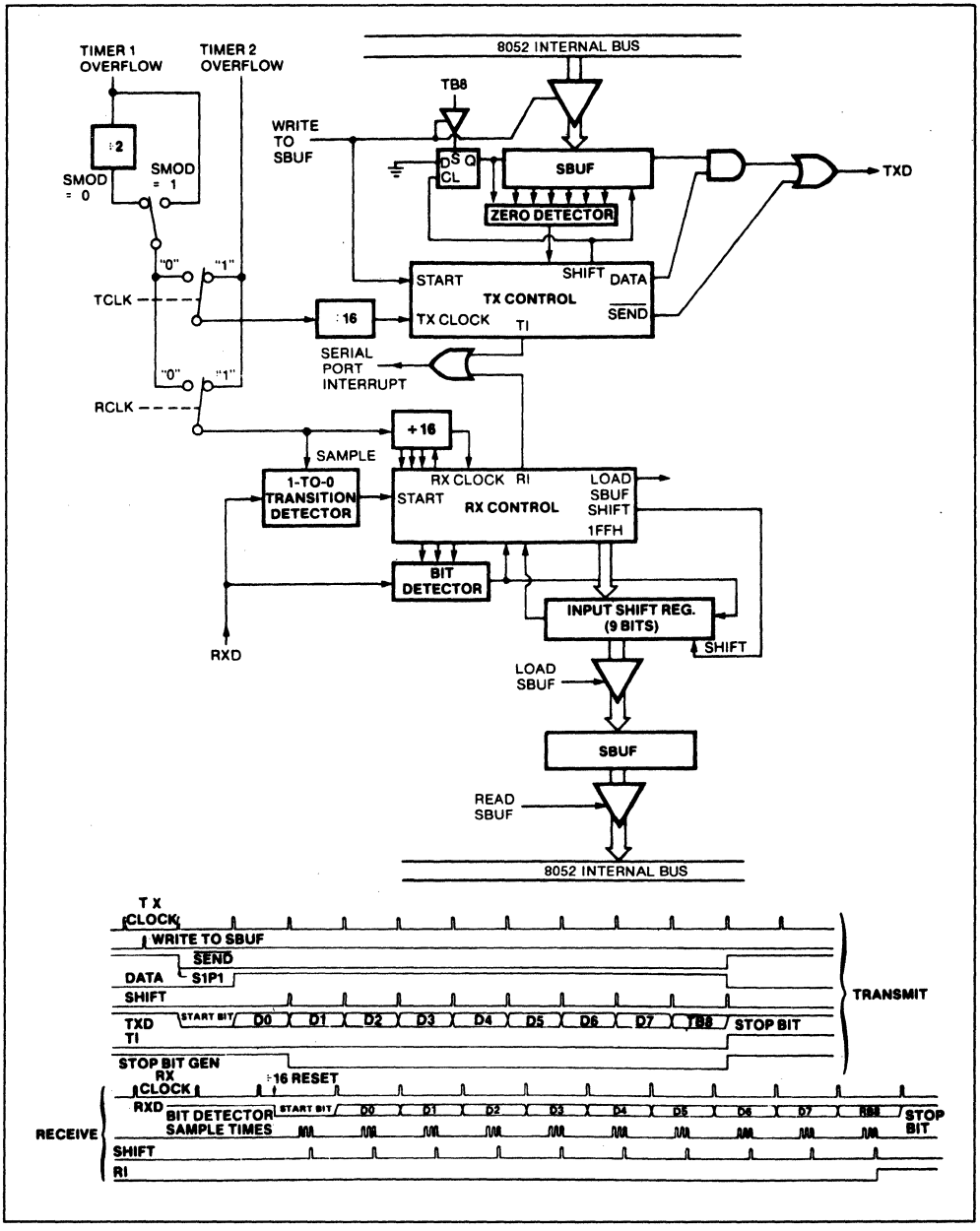


Figure 8. Serial Port Mode 3 in the 8052

TIMER/COUNTER 2 SET-UP

Except for the baud rate generator mode, the values given for T2CON do not include the setting of the TR2 bit. Therefore, bit TR2 must be set, separately, to turn the Timer on. See table 3 for set-up of timer 2 as a timer. See table 4 for set-up of timer 2 as a counter.

Table 3. Timer 2 as a Timer

Mode	T2CON	
	Internal Control (Note 1)	External Control (Note 2)
16–bit Auto–Reload	00H	08H
16–bit Capture	01H	09H
Baud rate generator receive and transmit same baud rate	34H	36H
Receive only	24H	26H
Transmit only	14H	16H

Table 4. Timer 2 as a Counter

Mode	TMOD	
	Internal Control (Note 1)	External Control (Note 2)
16–bit Auto–Reload	02H 03H	0AH 0BH

NOTES:
1. Capture/reload occurs only on timer/counter overflow.
2. Capture/reload occurs on timer/counter overflow and a 1 to 0 transition on T2EX (P1.1) pin except when timer 2 is used in the baud rate generator mode.

USING TIMER/COUNTER 2 TO GENERATE BAUD RATES

For this purpose, Timer 2 must be used in the baud rate generating mode. If Timer 2 is being clocked through pin T2 (P1.0) the baud rate is:

$$\text{Baud Rate} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

And if it is being clocked internally the baud rate is:

$$\text{Baud Rate} = \frac{\text{Osc Freq}}{32 \times [65536 - (\text{RCAP2H, RCAP2L})]}$$

To obtain the reload value for RCAP2H and RCAP2L the above equation can be rewritten as:

$$\text{RCAP2H, RCAP2L} = 65536 - \frac{\text{Osc Freq}}{32 \times \text{Baud Rate}}$$

INTERRUPTS

The 8052 has 6 interrupt sources as shown in Figure 9. All except TF2 and EXF2 are identical sources to those in the 8051.

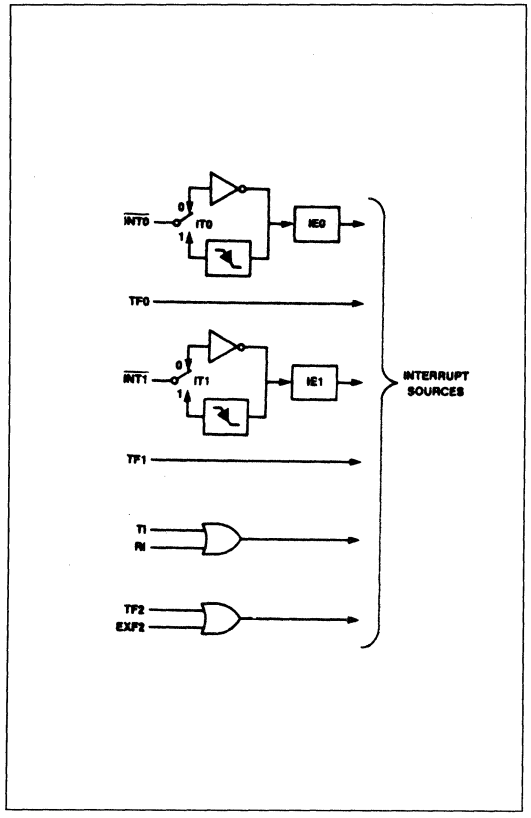


Figure 9. 8052/8032 Interrupt Sources

The Interrupt Enable Register and the Interrupt Priority Register are modified to include the additional 8052 interrupt sources. The operation of these registers is identical to the 8051. The registers are detailed in Figures 10, 11 and 12.

In the 8052, the Timer 2 Interrupt is generated by the logical OR of TF2 and EXF2. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and the bit will have to be cleared in software.

All of the bits that generate interrupts can be set or cleared by software, with the same result as though it had been set or cleared by hardware. That is, interrupts can be generated or pending interrupts can be canceled in software.

The interrupt vector addresses and the interrupt priority for requests in the same priority level are given in the following:

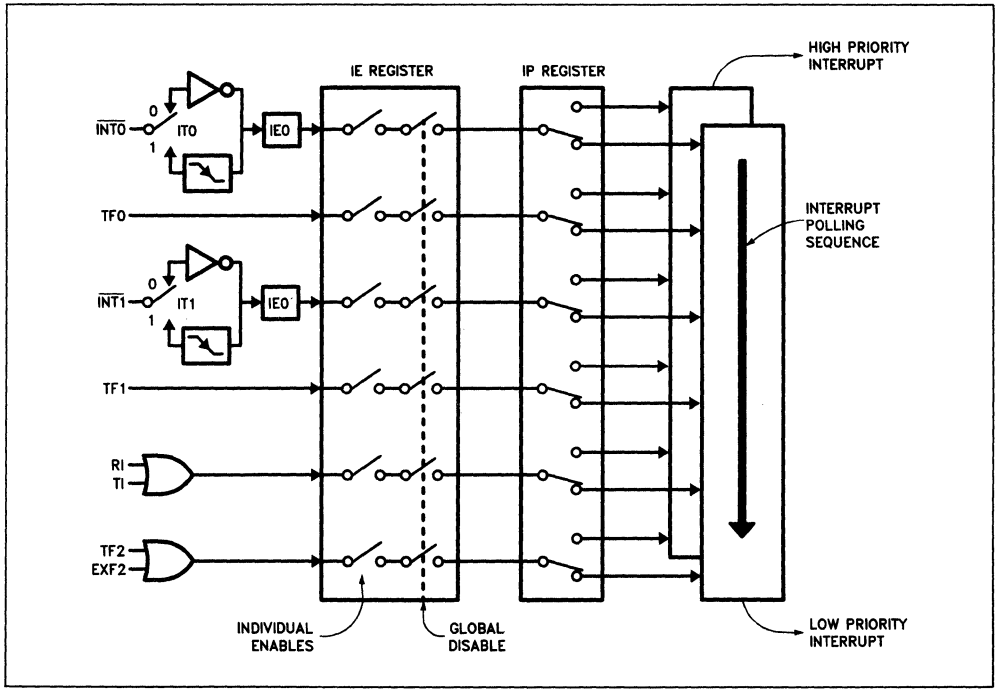


Figure 10. 8052 Interrupt Control System

Source	Vector Address	Priority Within Level
1. IE0	0003H	(highest)
2. TF0	000BH	
3. IE1	0013H	
4. TF1	001BH	
5. RI + TI	0023H	
6. TF2 + EXF2	002BH	(lowest)

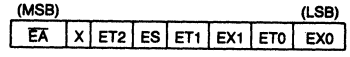
Note that they are identical to those in the 8051 except for the addition of the Timer 2 (TF1 and EXF2) interrupt at 002BH and at the lowest priority within a level.

PORT STRUCTURES

The port structures are identical in both parts, except that on the 8052/8032 ports P1.0 and P1.1 include the Timer 2 alternate functions as follows:

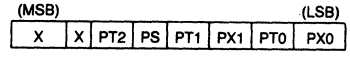
P1.0 T2 (Timer/counter 2 external input)
P1.1 T2EX (Timer/counter 2 capture/reload trigger)

As with the 8051, these alternate functions can only be activated if the corresponding bit latch in the port SFR contains a 1.



Symbol	Position	Function
$\overline{EA}$	IE.7	disables all interrupts. If $\overline{EA}$ = 0, no interrupt will be acknowledged. If $\overline{EA}$ = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
—	IE.6	reserved.
ET2	IE.5	enables or disables the Timer 2 Overflow or capture interrupt. If ET2 = 0, the Timer 2 interrupt is disabled.
ES	IE.4	enables or disables the Serial Port interrupt. If ES = 0, the Serial Port interrupt is disabled.
ET1	IE.3	enables or disables the Timer 1 Overflow interrupt. If ET1 = 0, the Timer 1 interrupt is disabled.
EX1	IE.2	enables or disables External Interrupt 1. If EX1 = 0, External Interrupt 1 is disabled.
ET0	IE.1	enables or disables the Timer 0 Overflow interrupt. If ET0 = 0, the Timer 0 interrupt is disabled.
EX0	IE.0	enables or disables External Interrupt 0. If EX0 = 0, External Interrupt 0 is disabled.

Figure 11. 8052 Interrupt Enable (IE) Register



Symbol	Position	Function
—	IP.7	reserved
—	IP.6	reserved
PT2	IP.5	defines the Timer 2 interrupt priority level. PT2 = 1 programs it to the higher priority level.
PS	IP.4	defines the Serial Port interrupt priority level. PS = 1 programs it to the higher priority level.
PT1	IP.3	defines the Timer 1 interrupt priority level. PT1 = 1 programs it to the higher priority level.
PX1	IP.2	defines the External Interrupt 1 priority level. PX1 = 1 programs it to the higher priority level.
PT0	IP.1	defines the Timer 0 interrupt priority level. PT0 = 1 programs it to the higher priority level.
PX0	IP.0	defines the External Interrupt 0 priority level. PX0 = 1 programs it to the higher priority level.

Figure 12. 8052 Interrupt Priority (IP) Register

Signetics

SCN8032AH/SCN8052AH
Single-Chip 8-Bit Microcontroller

Product Specification

Microprocessor Division

DESCRIPTION

The Signetics SCN8032AH/SCN8052AH is a high-performance microcontroller fabricated using the Signetics highly reliable +5V, depletion-load, N-channel, silicon-gate, N500 MOS process technology. It provides the hardware features, architectural enhancements and instructions that are necessary to make it a powerful and cost-effective controller for applications requiring up to 64K bytes of programming memory and up to 64K bytes of data storage.

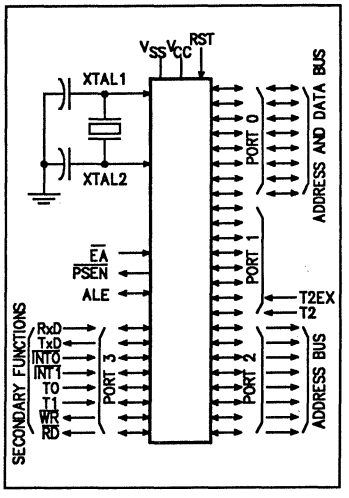
The SCN8032AH contains 256 bytes of read/write data memory, 32 I/O lines configured as four 8-bit ports, three 16-bit timer/counters, a six-source two-priority-level nested interrupt structure, a programmable serial I/O port and an on-chip oscillator and clock circuitry. The SCN8052AH has all of these features plus 8K bytes of non-volatile read-only program memory. Both microcontrollers have memory expansion capabilities of up to 64K bytes of data storage and 64K bytes of program memory that may be realized with standard TTL compatible memories.

Because of its extensive BCD/binary arithmetic and bit-handling facilities, the SCN8032AH/SCN8052AH microcontroller is efficient at both computational and control-oriented tasks. Efficient use of program memory is also achieved by using the familiar compact instruction set of the 8031/8051. Forty-four percent of the instructions are one-byte, 41% two-byte, and 15% three-byte instructions. With a 12MHz crystal, the majority of the instructions execute in just 1.0μs. The longest instructions, multiply and divide, require only 4μs at 12MHz.

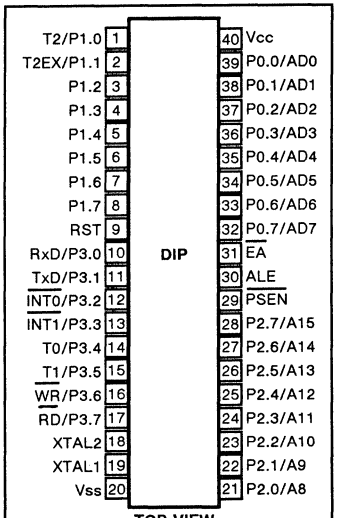
FEATURES

- **SCN8032AH – control-oriented CPU with RAM and I/O**
- **SCN8052AH – an SCN8032AH with factory mask-programmable ROM**
- **8K X 8 ROM (SCN8052AH only)**
- **256 X 8 RAM**
- **32 I/O lines (four 8-bit ports)**
- **Three 16-bit timer/counters**
- **Programmable full-duplex serial channel**
  - **Variable transmit/receive baud rate capability**
- **Timer 2 capture capability**
- **External memory addressing**
  - **64K ROM and 64K RAM**
- **Boolean processor**
- **128 user bit-addressable locations**
- **Upward compatible with SCN8031AH/SCN8051AH**

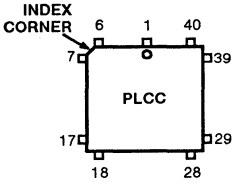
LOGIC SYMBOL



PIN CONFIGURATION



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	T2/P1.0	24	P2.0/A8
3	T2EX/P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

ORDERING INFORMATION

SCN80 □□H□□□□ (CPxxxx)

ROM/RAM (bytes)
32 = Ext/256
52 = 8K/256

Power Consumption
H = Reduced Power AH

Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

Pins
40 = 40-pin
44 = 44-pin

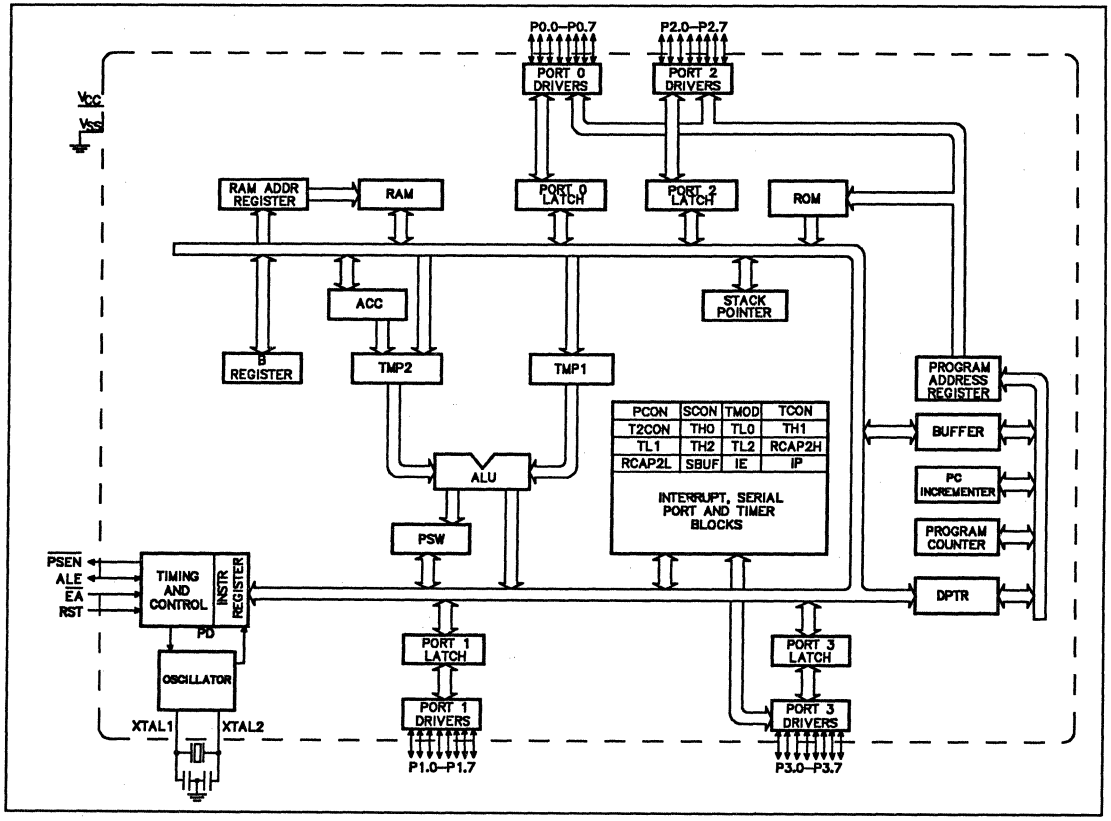
Package
A = Plastic LCC
I = Ceramic DIP
N = Plastic DIP

Speed
C = 3.5 to 12MHz
F = 3.5 to 15MHz

Operating Temperature Range
A = −40°C to +85°C
C = 0°C to +70°C

PART NUMBER SELECTION			
ROMless	**ROM**	**Temperature and Package**	**Frequency**
SCN8032HCCN40	SCN8052HCCN40	0 to +70°C, plastic DIP	3.5 to 12MHz
SCN8032HCCA44	SCN8052HCCA44	0 to +70°C, plastic LCC	3.5 to 12MHz
SCN8032HACN40	SCN8052HACN40	−40 to +85°C, plastic DIP	3.5 to 12MHz
SCN8032HACA44	SCN8052HACA44	−40 to +85°C, plastic LCC	3.5 to 12MHz
SCN8032HCFN40	SCN8052HCFN40	0 to +70°C, plastic DIP	3.5 to 15MHz
SCN8032HCFA44	SCN8052HCFA44	0 to +70°C, plastic PLCC	3.5 to 15MHz
SCN8032HAFN40	SCN8052HAFN40	−40 to +85°C, plastic DIP	3.5 to 15MHz
SCN8032HAFA44	SCN8052HAFA44	−40 to +85°C, plastic PLCC	3.5 to 15MHz

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. LCC	TYPE	NAME AND FUNCTION
V_{SS}	20	22	I	**Ground:** 0V reference.
V_{CC}	40	44	I	**Power Supply:** This is the power supply voltage for normal operation.
P0.0–P0.7	39–32	43–36	I/O	**Port 0:** Port 0 is an open–drain, bidirectional I/O port. Port 0 is also the multiplexed low–order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pullups when emitting 1s. Port 0 also outputs the code bytes during program verification.
P1.0–P1.7	1–8	2–9	I/O	**Port 1:** Port 1 is an 8–bit bidirectional I/O port with internal pull–ups. Pins P1.0 and P1.1 also correspond to the special functions T2, timer 2 counter trigger input, and T2EX, external input to timer 2. The output latch on these two special functions must be programmed to one for that function to operate. Port 1 also receives the low–order address byte during program verification.
	1	2	I	**T2 (P1.0):** Timer/counter 2 trigger input.
	2	3	I	**T2EX (P1.1):** Timer/counter 2 external count input.
P2.0–P2.7	21–28	24–31	I/O	**Port 2:** Port 2 is an 8–bit bidirectional I/O port with internal pull–ups. Port 2 emits the high–order address byte during fetches from external program memory and during accesses to external data memory that use 16–bit addresses (MOVX @DPTR). During accesses to external data memory that use 8–bit addresses (MOVX @Ri), port 2 emits the contents of the P2 special function register.
P3.0–P3.7	10–17	11, 13–19	I/O	**Port 3:** Port 3 is an 8–bit bidirectional I/O port with internal pullups. Port 3 is also used for the special features listed below:
	10	11	I	**RxD (P3.0):** Serial input port
	11	13	O	**TxD (P3.1):** Serial output port
	12	14	I	**$\overline{INT0}$ (P3.2):** External interrupt
	13	15	I	**$\overline{INT1}$ (P3.3):** External interrupt
	14	16	I	**T0 (P3.4):** Timer 0 external input
	15	17	I	**T1 (P3.5):** Timer 1 external input
	16	18	O	**$\overline{WR}$ (P3.6):** External data memory write strobe
	17	19	O	**$\overline{RD}$ (P3.7):** External data memory read strobe
RST	9	10	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. A small external pull–down resistor ($\cong$ 8.2KΩ) from RST to V_{SS} permits power–on reset when a capacitor ($\cong$ 10μf) is also connected from this pin to V_{CC}.
ALE	30	33	I/O	**Address Latch Enable:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory.
$\overline{PSEN}$	29	32	O	**Program Store Enable:** The read strobe to external program memory. When the device is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory.
$\overline{EA}$	31	35	I	**External Access Enable:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations 0000H and 1FFFH. If $\overline{EA}$ is held high, the device executes from internal program memory unless the program counter contains an address greater than 1FFFH.
XTAL1	19	21	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits.
XTAL2	18	20	O	**Crystal 2:** Output from the inverting oscillator amplifier.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol, page 1.

To drive the device from an external clock source, XTAL2 should be driven while XTAL1 should be grounded. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running.

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Storage temperature range	−65 to +150	°C
All voltages with respect to ground	−0.5 to +7.0	V
Power dissipation	2.0	W

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C , V_{CC} = 5V ±10%, V_{SS} = 0V[4, 5]

Symbol	Parameter	Test Conditions	Limits		Unit
			Min	Max	
V_{IL}	Input low voltage		−0.5	0.8	V
V_{IH}	Input high voltage, except RST and XTAL2		2	V_{CC}+0.5	V
V_{IH1}	Input high voltage to RST for reset, XTAL2	XTAL1 to V_{SS}	2.5	V_{CC}+0.5	V
V_{OL}	Output low voltage, ports 1, 2, 3[6]	I_{OL} = 1.6mA		0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$[6]	I_{OL} = 3.2mA		0.45	V
V_{OH}	Output high voltage, ports 1, 2, 3	I_{OH} = −80µA	2.4		V
V_{OH1}	Output high voltage port 0 in external bus mode, ALE, $\overline{PSEN}$)[3]	I_{OH} = −400µA	2.4		V
I_{IL}	Logical 0 input current, ports 1, 2, 3	V_{IN} = 0.45V		−800	µA
I_{IH1}	Input high curent to RST for reset	V_{IN} = V_{CC} − 1.5V		500	µA
I_{LI}	Input leakage current, port 0, $\overline{EA}$	0.45 < V_{IN} < V_{CC}		±10	µA
I_{IL2}	Logical 0 input current for XTAL2	XTAL1 = V_{SS}, V_{IN} = 0.45V		−3.2	mA
I_{CC}	Power supply current	All outputs disconnected and $\overline{EA}$ = V_{CC}		175	mA
C_{IO}	Pin capacitance	f_C = 1MHz, T_A = 25°C		10	pF
T_A = −40°C to +85°C − Extended temperature range − SCN8052HAC only					
V_{IH}	Input high voltage, except RST and XTAL2		2.2	V_{CC}+0.5	V
V_{IH1}	Input high voltage to RST for reset, XTAL2	XTAL1 to V_{SS}	2.7		V
I_{CC}	Power supply current	All outputs disconnected and $\overline{EA}$ = V_{CC}		175	mA

NOTES:

1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. For operating at elevated temperatures, the device must be derated based on +150°C maximum junction temperature.
3. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
4. Parameters are valid over operating temperature range unless otherwise specified.
5. All voltage measurements are referenced to ground. For testing, all input signals swing between 0.45V and 2.4V with a transition time of 20ns maximum. All time measurements are referenced at input voltages of 0.8V and 2.0V and at output voltages of 0.8V and 2.0V as appropriate.
6. V_{OL} is degraded when the device rapidly discharges external capacitance. This AC noise is most pronounced during emission of address data. When using external memory, locate the latch or buffer as close to the device as possible.

Datum	Emitting Ports	Degraded I/O Lines	V_{OL} (Peak Max)
Address	P2, P0	P1, P3	0.8V
Write Data	P0	P1, P3, ALE	0.8V

7. C_L = 100pF for port 0, ALE and $\overline{PSEN}$ outputs; C_L = 80pF for all other ports.

Single-Chip 8-Bit Microcontroller — SCN8032AH/SCN8052AH

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or −40°C to +85°C, V_{CC} = 5V ±10%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$		Oscillator frequency: **Speed Versions** SCN8052 C SCN8052 F			 3.5 3.5	 12 15	 MHz MHz
t_{LHLL}	1	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	1	Address valid to ALE low	43		$t_{CLCL}-40$		ns
t_{LLAX}	1	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	1	ALE low to valid instruction in		233		$4t_{CLCL}-100$	ns
t_{LLPL}	1	ALE low to $\overline{PSEN}$ low	58		$t_{CLCL}-25$		ns
t_{PLPH}	1	$\overline{PSEN}$ pulse width	215		$3t_{CLCL}-35$		ns
t_{PLIV}	1	$\overline{PSEN}$ low to valid instruction in		125		$3t_{CLCL}-125$	ns
t_{PXIX}	1	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	1	Input instruction float after $\overline{PSEN}$		63		$t_{CLCL}-20$	ns
t_{AVIV}	1	Address to valid instruction in		302		$5t_{CLCL}-115$	ns
t_{PLAZ}	1	$\overline{PSEN}$ low to address float		20		20	ns
t_{PXAV}	1	$\overline{PSEN}$ to address valid	75		$t_{CLCL}-8$		ns
Data Memory							
t_{RLRH}	2	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	3	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	2	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	2	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	2	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	2	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	2	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	2, 3	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	2, 3	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	3	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{QVWH}	3	Data vaid to $\overline{WR}$ high	433		$7t_{CLCL}-150$		ns
t_{WHQX}	3	Data hold after $\overline{WR}$	33		$t_{CLCL}-8$		ns
t_{RLAZ}	2	$\overline{RD}$ low to address float		20		20	ns
t_{WHLH}	2, 3	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
External Clock							
t_{CHCX}	5	High time			20		ns
t_{CLCX}	5	Low time			20		ns
t_{CLCH}	5	Rise time				20	ns
t_{CHCL}	5	Fall time				20	ns
Shift Register							
t_{XLXL}	4	Serial port clock cycle time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	4	Output data setup to clock rising edge	700		$10t_{CLCL}-133$		ns
t_{XHQX}	4	Output data hold after clock rising edge	50		$2t_{CLCL}-117$		ns
t_{XHDX}	4	Input data hold after clock rising edge	0		0		ns
t_{XHDV}	4	Clock rising edge to input data valid		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P – $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.

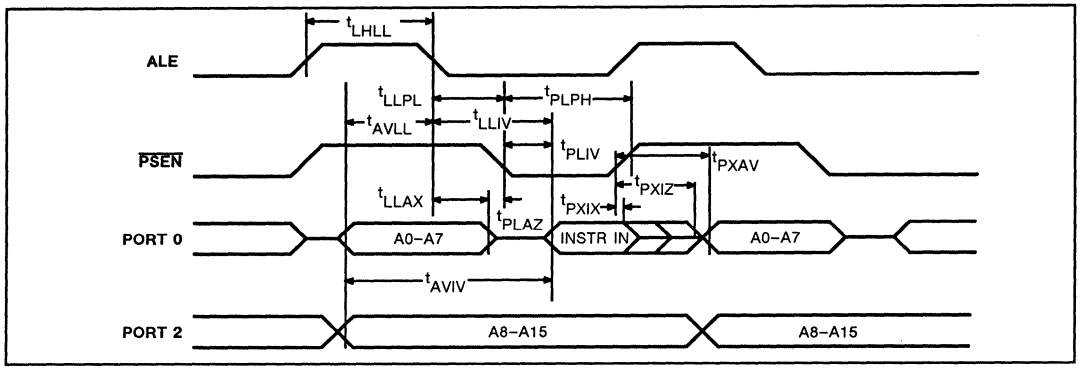


Figure 1. External Program Memory Read Cycle

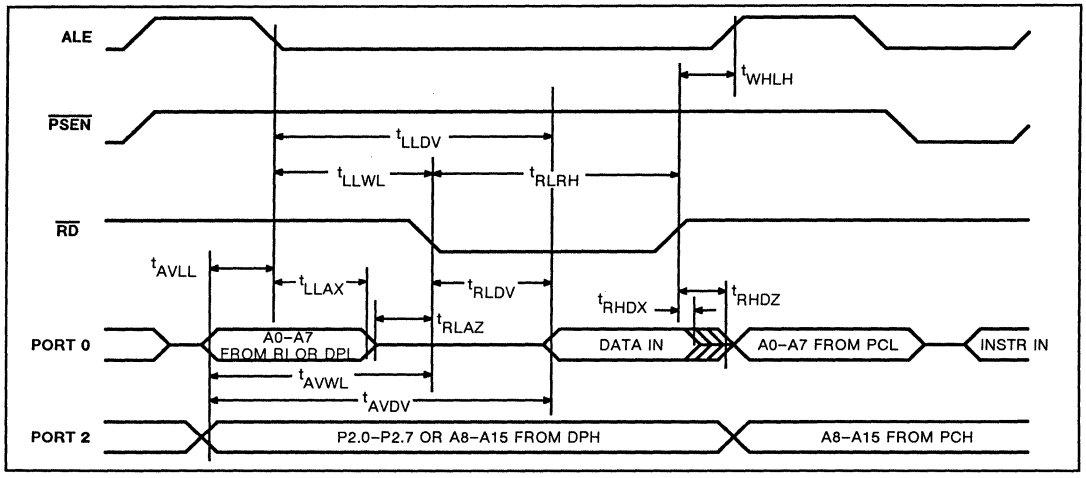


Figure 2. External Data Memory Read Cycle

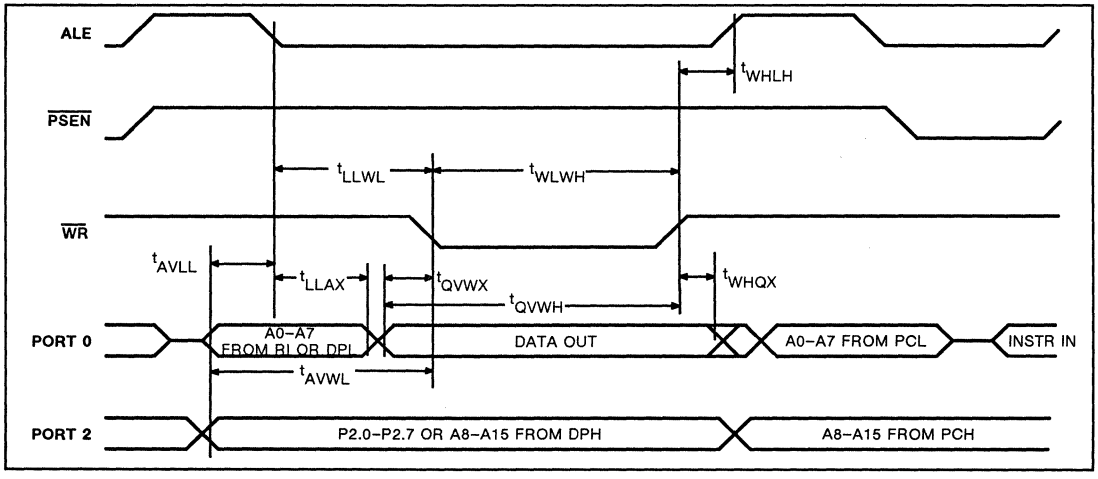


Figure 3. External Data Memory Write Cycle

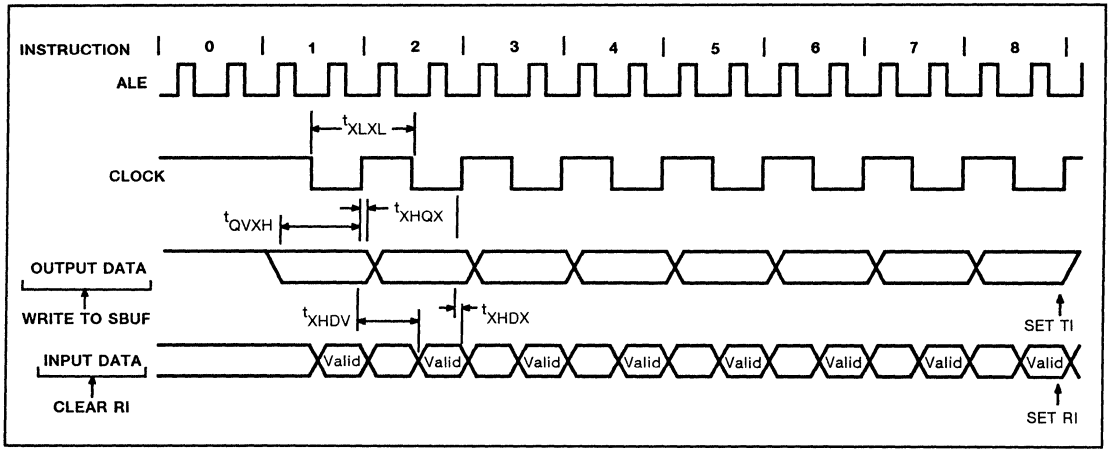


Figure 4. Shift Register Mode Timing

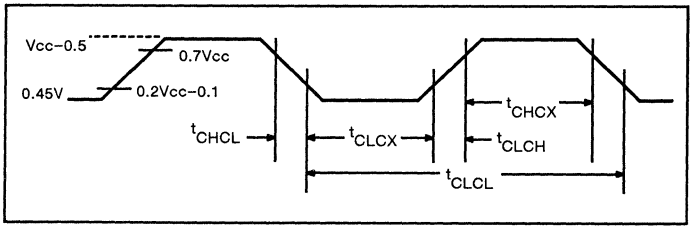


Figure 5. External Clock Drive

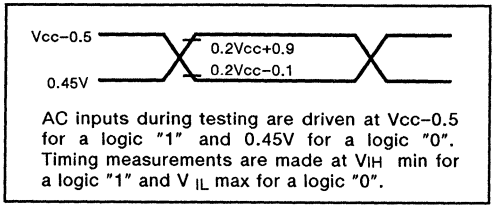


Figure 6. AC Testing Input/Output

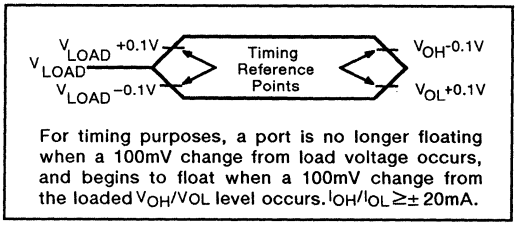


Figure 7. Float Waveform

8XC451 OVERVIEW

The SC80C451, the SC83C451, and the SC87C451 (hereafter referred to collectively as the 83C451) are I/O expanded versions of the 80C51. Three I/O ports have been added to the basic 80C51 architecture for a total of 7 on-chip I/O ports. The LCC version has a total of 68 pins. The DIP version has 64 pins. Port 6 has 4 control lines to facilitate high-speed asynchronous I/O functions.

The 83C451/87C451 includes a 4K X 8 ROM/EPROM, a 128 X 8 RAM, 56 (LCC) or 52 (DIP) I/O lines, two 16-bit timer/counters, a five source, two priority, level nested interrupt structure, a serial I/O port for either full duplex UART, I/O expansion, or multiprocessor communications, and an on-chip oscillator and clock circuits. The 80C451 includes all of the 83C451 features except the on-board 4K X 8 ROM.

The 83C451 has two software selectable modes of reduced activity for further power reduction: idle mode and power-down mode. Idle mode freezes the CPU while allowing the RAM, timers, serial port, and interrupt system to continue functioning. Power-down mode freezes the oscillator, causing all other chip functions to be inoperative while maintaining the RAM contents.

The 83C451 features include:

- 80C51 based architecture
- 68-pin LCC and 64-pin DIP packages
- Seven 8-bit I/O ports (LCC Version)
- Six 8-bit ports and one 4-bit port (DIP version)
- 4K X 8 ROM or EPROM
- 128 X 8 RAM
- Two 16-bit counter/timers
- Two external interrupts
- External memory addressing capability
  - 64K ROM and 64K RAM
- Low power consumption
  - Idle Mode
  - Power-down mode

DIFFERENCES FROM THE 8051

SPECIAL FUNCTION REGISTERS

The SFRs are identical to those of the standard 80C51 with the exception of four registers that have been added to allow control of the three additional I/O ports P4, P5, and P6. The additional registers are P4, P5, P6, and CSR. Registers P4, P5, and P6 function as port latches for ports 4, 5, and 6 respectively. These registers operate identically to those for ports 0 through 3 of the 80C51.

The Control Status Register (CSR) is used to control the mode of operation of port 6 and indicates the current status of port 6. All control status register bits can be read and written by the CPU except bits 0 and 1, which are read only. A Reset writes ones to bits 2–7 and zeros to bits 0 and 1. See Table 5 for the specific function of each bit in the Control Status register.

The SFR addresses for the 83C451 are identical to those in the 80C51 except for the additional registers P4, P5, P6, and CSR. Table 6 lists the SFR addresses, bit names and addresses (where applicable), and reset values for the 83C451. Table 7 is a detailed expansion of the special function registers.

I/O PORT STRUCTURE

The 8XC451 has a total of seven parallel I/O ports. The first four ports, P0 through P3, are identical in function to those present on the 80C51 family. The added ports 4 and 5 are identical in function to port 1, that is, they are standard quasi-bidirectional ports with no alternate functions and the standard output drive characteristics. Note that on the 68-pin LCC packages, port 4 is an 8-bit port, while on the 64-pin DIP packages, only the lower four bits of port 4 are available. Port 6 is a specialized 8-bit bidirectional I/O port with internal pullups. This special port can sink/source three LS TTL inputs and drive CMOS inputs without external pullups. The flexibility of this port facilitates high-speed parallel data communications. Port 6 operating modes are controlled by the port

Table 5. Control Status Register (CSR)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
BFLAG Mode Select		AFLAG Mode Select		Output Buffer Flag Clear Mode	Input Data Strobe Mode	Output Buffer Full Flag	Input Buffer Full Flag
0/0 = Logic 0 output* 0/1 = Logic 1 output* 1/0 = IBF output 1/1 = PE input (0 = Select) (1 = Disable I/O)		0/0 = Logic 0 output 0/1 = Logic 1 output 1/0 = OBF output* 1/1 = SEL input (0 = Data) (1 = Control/status)		0 = Negative edge of $\overline{ODS}$ 1 = Positive edge of $\overline{ODS}$	0 = Positive edge of $\overline{IDS}$ 1 = Low level of $\overline{IDS}$	0 = Output data buffer empty 1 = Output data buffer full	0 = Input data buffer empty 1 = Input data buffer full

NOTE:
*Output-always mode: MB1 = 0, MA1 = 1, and MA0 = 0. In this mode, port 6 is always enabled for output. $\overline{ODS}$ only clears the OBF flag.

Table 6. Special Function Register Addresses

REGISTER ADDRESS			BIT ADDRESS							
Name	Symbol	Address	MSB							LSB
Port 4	P4	C0	C7	C6	C5	C4	C3	C2	C1	C0
Port 5	P5	C8	CF	CE	CD	CC	CB	CA	C9	C8
Port 6 data	P6	D8	DF	DE	DD	DC	DB	DA	D9	D8
Port 6 control status	CSR	E8	EF	EE	ED	EC	EB	EA	E9	E8

6 Control Status Register (CSR). Port 6 and the CSR are addressed at the Special Function Addresses shown in Table 6. Port 6 can be used as a standard I/O port, or in strobed modes of operation in conjunction with the four port 6 control lines listed below:

ODS Output data strobe (active low)
IDS Input data strobe (active low)
BFLAG Bidirectional I/O pin. Can be programmed to output the Input Buffer Full flag (IBF), input an active low Port Enable (PE) signal, or output a high or low logic level.
AFLAG Bidirectional I/O pin. Can be programmed to output the Output Buffer Full (OBF) flag, input a register select signal (SEL), or output a high or low logic level.

Port 6 can be used in a number of different ways to facilitate data communication. It can be used as a processor bus interface, as a standard quasi-bidirectional I/O port, or as a parallel printer port (either polled or interrupt driven).

PROCESSOR BUS INTERFACE

Port 6 allows the use of an 83C451 as an element on a microprocessor type bus. The host processor could be a general purpose MPU or the data bus of a microcontroller like the 83C451 itself. Setting up the 83C451 as a processor bus interface allows single or multiple microcontrollers to be used on a bus as flexible peripheral processing elements. Applications can include: keyboard scanners, serial I/O controllers, servo controllers, etc.

On reset, Port 6 is programmed correctly (that is Special Function registers CSR and P6) for use as a bus interface. This prevents the interface from disrupting data on the bus of a host processor during power-up.

STANDARD QUASI-BIDIRECTIONAL I/O PORT

To use port 6 as a common I/O port, all of the control pins should be tied to ground. On hardware reset, bits 2–7 of the CSR are set to one. With the control pins grounded, the Port's operation and electrical characteristics will be identical to port 1 on the 80C51. No further software initialization is required.

PARALLEL PRINTER PORT

The 83C451 has the capacity to permit all of the intelligent features of a common printer to be handled by a single chip. The features of Port 6 allow a parallel port to be designed with only line driving and receiving chips required as additional hardware. The onboard UART allows RS232 interfacing with only level shifting chips added. The 8-bit parallel ports 0 to 6 are ample to drive onboard control functions, even when ports are used for external memory access, interrupts, and other functions. The RAM addressing ability of ports 0 to 2 can be used to address up to 64K bytes of a hardware buffer/spooler.

In addition, either end of a parallel interface can be implemented using port 6, and the interfaces can be interrupt driven or polled in either case. For more detailed information on port 6 usage, refer to the application notes contained in Section 3, entitled "80C451 Operation of Port 6" and "256K Centronics Printer Buffer Using the SC87C451 Microcontroller".

Table 7. 8X451 Special Function Registers

Symbol	Description	Direct Address	Bit Names and Addresses MSB							LSB	Reset Value
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
			EF	EE	ED	EC	EB	EA	E9	E8	
CSR*#	Port 6 command/status	E8H	MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF	FCH
DPTR:	Data pointer (2 bytes):										
DPH	High byte	83H									00H
DPL	Low byte	82H									00H
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt priority	B8H	–	–	–	PS	PT1	PX1	PT0	PX0	xxx00000B
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*	Interrupt enable	A8H	EA	–	–	ES	ET1	EX1	ET0	EX0	0xx00000B
P0*	Port 0	80H	87	B6	85	84	83	82	81	80	FFH
P1*	Port 1	90H	97	96	95	94	93	92	91	90	FFH
P2*	Port 2	A0H	A7	A6	A5	A4	A3	A2	A1	A0	FFH
P3*	Port 3	B0H	B7	B6	B5	B4	B3	B2	B1	B0	FFH
P4*#	Port 4	C0H	C7	C6	C5	C4	C3	C2	C1	C0	FFH
P5*#	Port 5	C8H	CF	CE	CD	CC	CB	CA	C9	C8	FFH
P6*#	Port 6	D8H	DF	DE	DD	DC	DB	DA	D9	D8	FFH
PCON	Power control	87H	SMOD	–	–	–	GF1	GF0	PD	IDL	0xxx0000B
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	–	P	00H
SBUF	Serial data buffer	99H									xxxxxxxxB
			9F	9E	9D	9C	9B	9A	99	98	
SCON*	Serial port control	98H	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	00H
SP	Stack pointer	81H									07H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer/counter control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00H
TMOD	Timer/counter mode	89H	GATE	C/T	M1	M0	GATE	C/T	M1	M0	00H
TH0	Timer 0 high byte	8CH									00H
TH1	Timer 1 high byte	8DH									00H
TL0	Timer 0 low byte	8AH									00H
TL1	Timer 1 low byte	8BH									00H

*SFRs are bit addressable.
#SFRs are modified from or added to the 80C51 SFRs.

Signetics

SC80C451/SC83C451
CMOS Single-Chip 8-Bit Microcontroller

Product Specification

Microprocessor Division

DESCRIPTION

The Signetics SC80C451/SC83C451 is an I/O expanded, single-chip microcontroller fabricated with Signetics high-density CMOS technology. Signetics epitaxial substrate minimizes latch-up sensitivity.

The SC80C451/SC83C451 is a functional extension of the SC80C51 microcontroller with three additional I/O ports and four I/O control lines. The LCC version has a total of 68 pins. Four control lines associated with port 6 facilitate high speed asynchronous I/O functions.

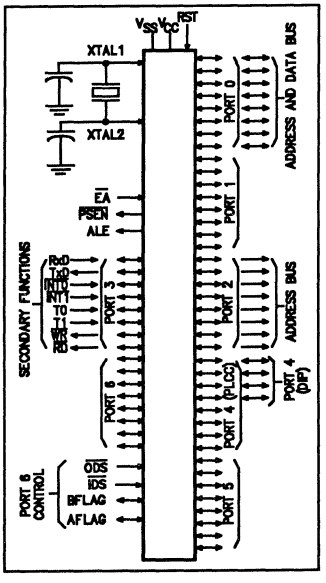
The SC83C451 includes a 4K X 8 ROM, a 128 X 8 RAM, 56 (LCC) or 52 (DIP) I/O lines, two 16-bit timer/counters, a five source, two priority level, nested interrupt structure, a serial I/O port for either a full duplex UART, I/O expansion, or multiprocessor communications, and an on-chip oscillator and clock circuits. The SC80C451 includes all the SC-83C451 features except the on-board 4K X 8 ROM.

The SC80C451/SC83C451 has two software selectable modes of reduced activity for further power reduction: idle mode and power-down mode. Idle mode freezes the CPU while allowing the RAM, timers, serial port, and interrupt system to continue functioning. Power-down mode freezes the oscillator, causing all other chip functions to be inoperative while maintaining the RAM contents.

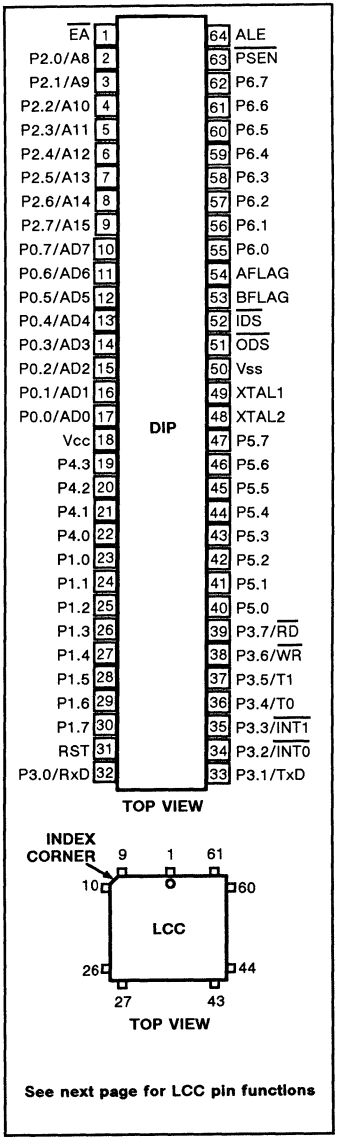
FEATURES

- **User programmable microcontroller**
- **SC80C51 based architecture**
- **68-pin LCC and 64-pin DIP packages:**
  - **Seven 8-bit I/O ports (LCC version)**
  - **Six 8-bit ports and one 4-bit port (DIP version)**
- **Port 6 features:**
  - **8 data pins**
  - **4 control pins**
  - **Direct MPU bus interface**
  - **Parallel printer interface**
- **On the microcontroller**
  - **4K X 8 ROM (SC83C451 only)**
  - **128 X 8 RAM**
  - **Two 16-bit counter/timers**
  - **Two external interrupts**
- **External memory addressing capability**
  - **64K ROM and 64K RAM**
- **Low power consumption:**
  - **Normal operation: less than 24mA at 5V, 12MHz**
  - **Idle mode**
  - **Power-down mode**

LOGIC SYMBOL



PIN CONFIGURATIONS



See next page for LCC pin functions

ORDERING INFORMATION

SC8□C451□□□□□ (CPxxxx)

ROMless/ROM
0 = ROMless
3 = ROM

Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

Pins
64 = 64-pin DIP
68 = 68-pin PLCC

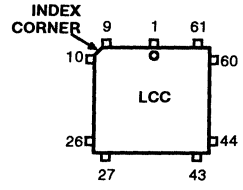
Package
A = Plastic PLCC
N = Plastic DIP

Speed
B = 0.5 to 12MHz
C = 3.5 to 12MHz
G = 3.5 to 16MHz

Operating Temperature Range
A = −40°C to +85°C
C = 0°C to +70°C

PART NUMBER SELECTION		
ROMless	**ROM**	**Temperature and Package**
SC80C451CCN64	SC83C451CCN64	0 to +70°C plastic DIP
SC80C451CGN64	SC83C451CGN64	0 to +70°C plastic DIP
SC80C451CBN64	SC83C451CBN64	0 to +70°C plastic DIP
SC80C451CCA68	SC83C451CCA68	0 to +70°C plastic LCC
SC80C451CGA68	SC83C451CGA68	0 to +70°C plastic LCC
SC80C451CBA68	SC83C451CBA68	0 to +70°C plastic LCC
SC80C451ACN64	SC83C451ACN64	−40 to +85°C plastic DIP
SC80C451AGN64	SC83C451AGN64	−40 to +85°C plastic DIP
SC80C451ACA68	SC83C451ACA68	−40 to +85°C plastic LCC
SC80C451AGA68	SC83C451AGA68	−40 to +85°C plastic LCC

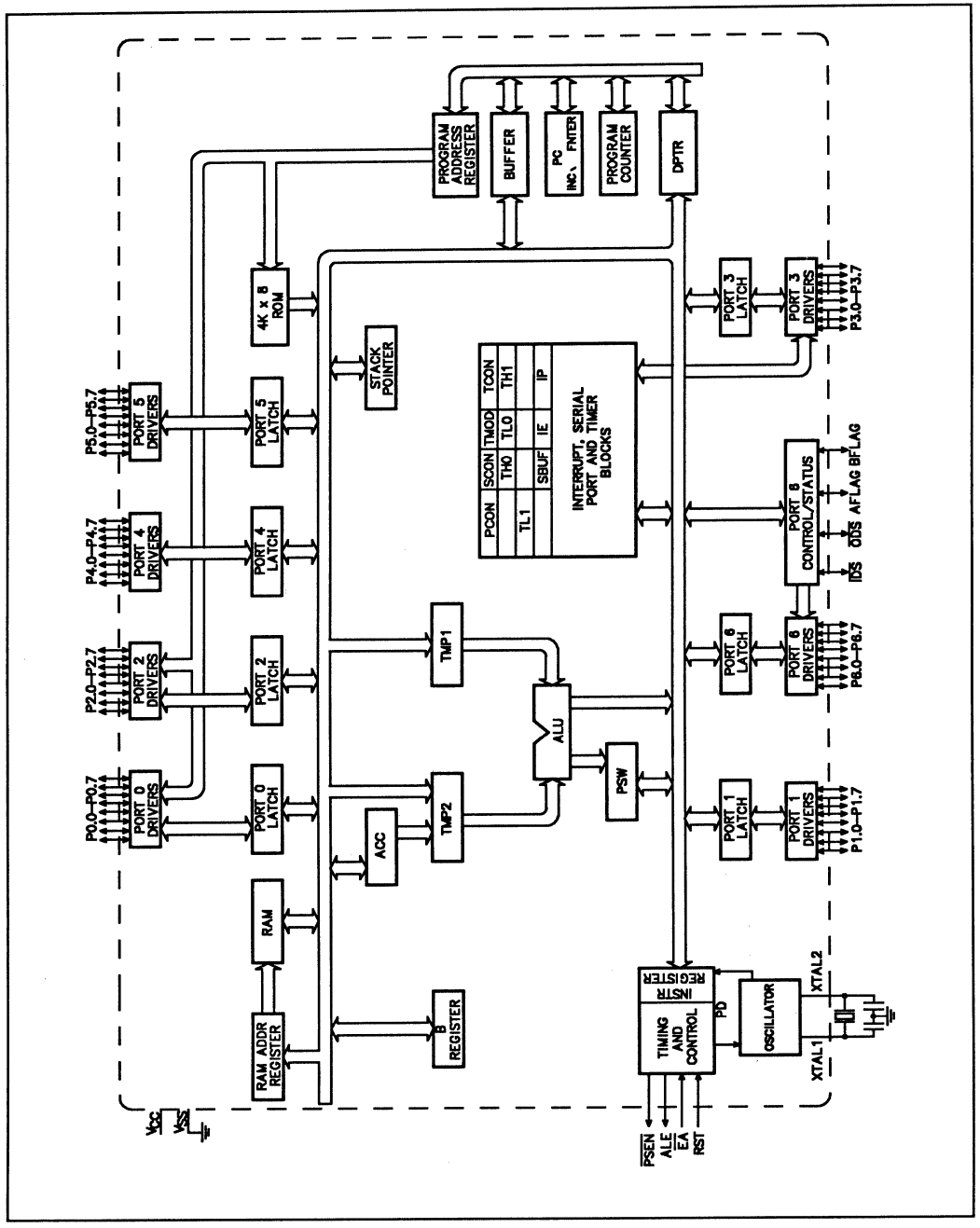
LCC PIN FUNCTIONS



TOP VIEW

Pin	Function	Pin	Function	Pin	Function
1	$\overline{EA}$	24	P4.2	47	P5.3
2	P2.0/A8	25	P4.1	48	P5.4
3	P2.1/A9	26	P4.0	49	P5.5
4	P2.2/A10	27	P1.0	50	P5.6
5	P2.3/A11	28	P1.1	51	P5.7
6	P2.4/A12	29	P1.2	52	XTAL2
7	P2.5/A13	30	P1.3	53	XTAL1
8	P2.6/A14	31	P1.4	54	Vss
9	P2.7/A15	32	P1.5	55	$\overline{ODS}$
10	P0.7/AD7	33	P1.6	56	$\overline{IDS}$
11	P0.6/AD6	34	P1.7	57	BFLAG
12	P0.5/AD5	35	RST	58	AFLAG
13	P0.4/AD4	36	P3.0/RxD	59	P6.0
14	P0.3/AD3	37	P3.1/$\overline{TxD}$	60	P6.1
15	P0.2/AD2	38	P3.2/$\overline{INT0}$	61	P6.2
16	P0.1/AD1	39	P3.3/$\overline{INT1}$	62	P6.3
17	P0.0/AD0	40	P3.4/T0	63	P6.4
18	Vcc	41	P3.5/$\overline{T1}$	64	P6.5
19	P4.7	42	P3.6/$\overline{WR}$	65	P6.6
20	P4.6	43	P3.7/$\overline{RD}$	66	P6.7
21	P4.5	44	P5.0	67	$\overline{PSEN}$
22	P4.4	45	P5.1	68	ALE
23	P4.3	46	P5.2		

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. PLCC	TYPE	NAME AND FUNCTION
V_{SS}	50	54	I	**Ground:** 0V reference.
V_{CC}	18	18	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–P0.7	17–10	17–10	I/O	**Port 0:** Port 0 is an 8-bit open-drain, bidirectional I/O port. Port 0 is also the multiplexed data and low-order address bus during accesses to external memory. External pullups are required during program verification. Port 0 can sink/source eight LS TTL inputs.
P1.0–P1.7	23–30	27–34	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pullups. Port 1 receives the low-order address bytes during program memory verification. Port 1 can sink/source three LS TTL inputs, and drive CMOS inputs without external pullups.
P2.0–P2.7	2–9	2–9	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pullups. Port 2 emits the high-order address bytes during access to external memory and receives the high-order address bits and control signals during program verification. Port 2 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups.
P3.0–P3.7	32–39	36–43	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 2 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups. Port 3 also serves the special functions listed below:
	32	36	I	**RxD (P3.0):** Serial input port
	33	37	O	**TxD (P3.1):** Serial output port
	34	38	I	**$\overline{\text{INT0}}$ (P3.2):** External interrupt 0
	35	39	I	**$\overline{\text{INT1}}$ (P3.3):** External interrupt 1
	36	40	I	**T0 (P3.4):** Timer 0 external input
	37	41	I	**T1 (P3.5):** Timer 1 external input
	38	42	O	**$\overline{\text{WR}}$ (P3.6):** External data memory write strobe
	39	43	O	**$\overline{\text{RD}}$ (P3.7):** External data memory read strobe
P4.0–P4.3	22–19		I/O	**Port 4:** Port 4 is a 4/8-bit (DIP/LCC) bidirectional I/O port with internal pullups. Port 4 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups.
P4.0–P4.7		26–19	I/O	
P5.0–P5.7	40–47	44–51	I/O	**Port 5:** Port 5 is an 8-bit bidirectional I/O port with internal pullups. Port 5 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups.
P6.0–P6.7	55–62	59–66	I/O	**Port 6:** Port 6 is a specialized 8-bit bidirectional I/O port with internal pullups. This special port can sink/source three LS TTL inputs and drive CMOS inputs without external pullups. Port 6 can be used in a strobed or non-strobed mode of operation. Port 6 works in conjunction with four control pins that serve the functions listed below: **Port 6 Control Lines:**
$\overline{\text{ODS}}$	51	55	I	**$\overline{\text{ODS}}$:** Output data strobe
$\overline{\text{IDS}}$	52	56	I	**$\overline{\text{IDS}}$:** Input data strobe
BFLAG	53	57	I/O	**BFLAG:** Bidirectional I/O pin with internal pullups
AFLAG	54	58	I/O	**AFLAG:** Bidirectional I/O pin with internal pullups
RST	31	35	I	**Reset:** A high on this pin, for two machine cycles while the oscillator is running, resets the device. An internal pull-down resistor permits a power-on reset using only a capacitor connected to V_{CC}.
ALE	64	68	O	**Address Latch Enable:** Output pulse for latching the low byte of the address during accesses to external memory. ALE is activated at a constant rate of 1/6 the oscillator frequency except during an external data memory access, at which time one ALE is skipped. ALE can sink/source eight LS TTL inputs and drive CMOS inputs without an external pullup.
$\overline{\text{PSEN}}$	63	67	O	**Program Store Enable:** The read strobe to external program memory. $\overline{\text{PSEN}}$ is activated twice each machine cycle during fetches from external program memory. However, when executing out of external program memory, two activations of $\overline{\text{PSEN}}$ are skipped during each access to external data memory. $\overline{\text{PSEN}}$ is not activated during fetches from internal program memory. $\overline{\text{PSEN}}$ can sink/source eight LS TTL inputs and drive CMOS inputs without an external pullup.
$\overline{\text{EA}}$	1	1	I	**Instruction Execution Control:** When $\overline{\text{EA}}$ is held high, the CPU executes out of internal program memory, unless the program counter exceeds 0FFFH. When $\overline{\text{EA}}$ is held low, the CPU executes out of external program memory. $\overline{\text{EA}}$ must never be allowed to float.
XTAL1	49	53	I	**Crystal 1:** Input to the inverting amplifier that forms the oscillator. This input receives the external oscillator when an external oscillator is used.
XTAL2	48	52	O	**Crystal 2:** An output of the inverting amplifier that forms the oscillator. This pin should be floated when an external oscillator is used.

PORTS 4 AND 5
Ports 4 and 5 are bidirectional I/O ports with internal pullups. Port 4 is an 8-bit port (LCC version) or a 4-bit port (DIP version). Port 4 and port 5 pins with ones written to them, are pulled high by the internal pullups, and in that state can be used as inputs. Port 4 and 5 are addressed at the special function register addresses shown in Table 1.

PORT 6
Port 6 is a special 8-bit bidirectional I/O port with internal pullups (see Figure 1). This port can be used as a standard I/O port, or in strobed modes of operation in conjunction with four special control lines: $\overline{ODS}$, $\overline{IDS}$, AFLAG, and BFLAG. Port 6 operating modes are controlled by the port 6 control status register (CSR). Port 6 and the CSR are addressed at the special function register addresses shown in Table 1. The following four control pins are used in conjunction with port 6:

ODS – Output data strobe input for port 6. $\overline{ODS}$ can be programmed to control the port 6 output drivers and the output buffer full flag (OBF), or to clear only the OBF flag bit in the CSR (output-always mode). $\overline{ODS}$ is active low for output driver control. The OBF flag can be programmed to be cleared on the negative or positive edge of $\overline{ODS}$.

IDS– Input data strobe for port 6. $\overline{IDS}$ is used to control the port 6 input latch and input buffer full flag (IBF) bit in the CSR. The input data latch can be programmed to be transparent when $\overline{IDS}$ is low and latched on the positive transition of $\overline{IDS}$, or to latch only on the positive transition of $\overline{IDS}$. Correspondingly, the IBF flag is set on the negative or positive transition of $\overline{IDS}$.

AFLAG – AFLAG is a bidirectional I/O pin which can be programmed to be an output set high or low under program control, or to output the state of the output buffer full flag. AFLAG can also be programmed to be an input which selects whether the contents of the output buffer, or the contents of the port 6 control status register will be output on port 6. This feature grants complete port 6 status to external devices.

BFLAG – BFLAG is a bidirectional I/O pin which can be programmed to be an output, set high or low under program control, or to output the state of the input buffer full flag. BFLAG can also be programmed to input an enable signal for port 6. When BFLAG is used as an enable input, port 6 output drivers are in the high-impedance state, and the input latch does not respond to the $\overline{IDS}$ strobe when BFLAG is high. Both features are enabled when BFLAG is low. This feature facilitates the use of the SC80C451/SC83C451 in bused multiprocessor systems.

Table 1. Special Function Register Addresses

REGISTER ADDRESS			BIT ADDRESS							
Name	**Symbol**	**Address**	**MSB**							**LSB**
Port 4	P4	C0	C7	C6	C5	C4	C3	C2	C1	C0
Port 5	P5	C8	CF	CE	CD	CC	CB	CA	C9	C8
Port 6 data	P6	D8	DF	DE	DD	DC	DB	DA	D9	D8
Port 6 control status	CSR	E8	EF	EE	ED	EC	EB	EA	E9	E8

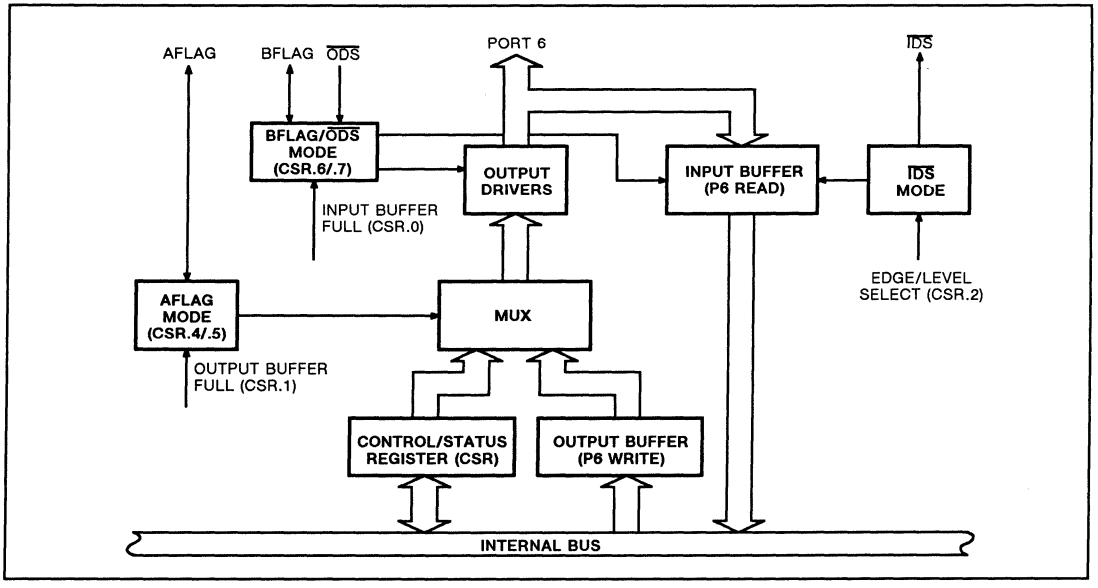


Figure 1. Port 6 Block Diagram

CONTROL STATUS REGISTER
The control status register (CSR) establishes the mode of operation for port 6 and indicates the current status of port 6 I/O registers. All control status register bits can be read and written by the CPU, except bits 0 and 1, which are read only. Reset writes ones to bits 2 through 7, and writes zeros to bits 0 and 1 (see Table 2).

CSR.0 Input Buffer Full Flag (IBF) (Read Only) – The IBF bit is set to a logic 1 when port 6 data is loaded into the input buffer under control of $\overline{IDS}$. This can occur on the negative or positive edge of $\overline{IDS}$, as determined by CSR.2. IBF is cleared when the CPU reads the input buffer register.

CSR.1 Output Buffer Full Flag (OBF) (Read Only) – The OBF flag is set to a logic 1 when the CPU writes to the port 6 output data buffer. OBF is cleared by the positive or negative edge of $\overline{ODS}$, as determined by CSR.3.

CSR.2 IDS Mode Select (IDSM) – When CSR.2 = 0, a low-to-high transition on the $\overline{IDS}$ pin sets the IBF flag. The port 6 input buffer is loaded on the $\overline{IDS}$ positive edge. When CSR.2 = 1, a high-to-low transition on the $\overline{IDS}$ pin sets the IBF flag. When port 6 input buffer is transparent when $\overline{IDS}$ is low, and latched when $\overline{IDS}$ is high.

CSR.3 Output Buffer Full Flag Clear Mode (OBFC) – When CSR.3 = 1, the positive edge of the $\overline{ODS}$ input clears the OBF flag. When CSR.3 = 0, the negative edge of the $\overline{ODS}$ input clears the OBF flag.

CSR.4, CSR.5 AFLAG Mode Select (MA0, MA1) – Bits 4 and 5 select the mode of operation for the AFLAG pin, as follows:

MA1	MA0	AFLAG Function
0	0	Logic 0 output
0	1	Logic 1 output
1	0	OBF flag output (CSR.1)
1	1	Select (SEL) input mode

The select (SEL) input mode is used to determine whether the port 6 data register or the control status register is output on port 6. When the select feature is enabled, the AFLAG input controls the source of port 6 output data. A logic 0 on AFLAG input selects the port 6 data register, and a logic 1 on AFLAG input selects the control status register.

CSR.6, CSR.7 BFLAG Mode Select (MB0, MB1) – Bits 6 and 7 select the mode operation as follows:

MB1	MB0	AFLAG Function
0	0	Logic 0 output
0	1	Logic 1 output
1	0	IBF flag output (CSR.0)
1	1	Port enable ($\overline{PE}$)

In the port enable mode, $\overline{IDS}$ and $\overline{ODS}$ inputs are disabled when BFLAG input is high. When the BFLAG input is low, the port is enabled for I/O.

SPECIAL FUNCTION REGISTER ADDRESSES
Special function register addresses for the SC80C451/SC83C451 are identical to those of the SC80C51, except for the additional registers listed in Table 1.

Table 2. Control Status Register (CSR)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
BFLAG Mode Select		AFLAG Mode Select		Output Buffer Flag Clear Mode	Input Data Strobe Mode	Output Buffer Full Flag	Input Buffer Full Flag
0/0 = Logic 0 output* 0/1 = Logic 1 output* 1/0 = $\overline{IBF}$ output 1/1 = $\overline{PE}$ input (0 = Select) (1 = Disable I/O)		0/0 = Logic 0 output 0/1 = Logic 1 output 1/0 = OBF output* 1/1 = SEL input (0 = Data) (1 = Control/status)		0 = Negative edge of $\overline{ODS}$ 1 = Positive edge of $\overline{ODS}$	0 = Positive edge of $\overline{IDS}$ 1 = Low level of $\overline{IDS}$	0 = Output data buffer empty 1 = Output data buffer full	0 = Input data buffer empty 1 = Input data buffer full

NOTE:
*Output-always mode: MB1 = 0, MA1 = 1, and MA0 = 0. In this mode, port 6 is always enabled for output. $\overline{ODS}$ only clears the OBF flag.

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Operating temperature under bias	0 to +70 −40 to +85	°C
Storage temperature range	−65 to +150	°C
Voltage on any other pin to V_{SS}	−0.5 to + 6.5	V
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.5	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V ±20%, V_{SS} = 0V

Symbol	Parameter	Test Conditions	Limits			Unit
			Min	Typical[1]	Max	
V_{IL}	Input low voltage, except $\overline{EA}$		−0.5		$0.2V_{CC}$−0.1	V
V_{IL1}	Input low voltage to $\overline{EA}$		0		$0.2V_{CC}$−0.3	V
V_{IH}	Input high voltage, except XTAL1, RST		$0.2V_{CC}$+.9		V_{CC}+0.5	V
V_{IH1}	Input high voltage, XTAL1, RST		$0.7V_{CC}$		V_{CC}+0.5	V
V_{OL}	Output low voltage, ports 1, 2, 3	I_{OL} = 1.6mA[2]			0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$	I_{OL} = 3.2mA[2]			0.45	V
V_{OH}	Output high voltage, ports 1, 2, 3	I_{OH} = −60µA I_{OH} = −25µA I_{OH} = −10µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
V_{OH1}	Output high voltage (port 0 in external bus mode, ALE, $\overline{PSEN}$)[3]	I_{OH} = −800µA I_{OH} = −300µA I_{OH} = −80µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
I_{IL}	Logical 0 input current, ports 1, 2, 3	V_{IN} = 0.45V			−50	µA
I_{TL}	Logical 1-to-0 transition current, ports 1, 2, 3	See note 4			−650	µA
I_{LI}	Input leakage current, port 0	V_{IN} = V_{IL} or V_{IH}			±10	µA
I_{CC}	Power supply current: Active mode @ 12MHz[5] Idle mode @ 12MHz[5] Power down mode	See note 6		11.5 1.3 3	25 4 50	 mA mA µA
R_{RST}	Internal reset pulldown resistor		50		300	kΩ
C_{IO}	Pin capacitance[7] − DIP package − PLCC package				15 10	pF pF

NOTES:
1. Typical ratings are based on a limited number of samples taken from early manufacturing lots and are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the $0.9V_{CC}$ specification when the address bits are stabilizing.
4. Pins of ports 1, 2, and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. I_{CC}MAX at other frequencies is given by:
   Active mode: I_{CC}MAX = 0.94 X FREQ + 13.71
   Idle mode: I_{CC}MAX = 0.14 X FREQ + 2.31
   where FREQ is the external oscillator frequency in MHz. I_{CC}MAX is given in mA. See Figure 13.
6. See Figures 14 through 17 for I_{CC} test conditions.
7. C_{IO} applies to Ports 1, 2, 3, 4, 5, 6, AFLAG, BFLAG, XTAL1, XTAL2.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V ±20%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	2	Oscillator frequency: **Speed Versions** SC80C451/SC83C451 B SC80C451/SC83C451 C SC80C451/SC83C451 G			 0.5 3.5 3.5	 12 12 16	 MHz MHz MHz
t_{LHLL}	2	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	2	Address valid to ALE low	28		$t_{CLCL}-55$		ns
t_{LLAX}	2	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	2	ALE low to valid instruction in		234		$4t_{CLCL}-100$	ns
t_{LLPL}	2	ALE low to $\overline{PSEN}$ low	43		$t_{CLCL}-40$		ns
t_{PLPH}	2	$\overline{PSEN}$ pulse width	205		$3t_{CLCL}-45$		ns
t_{PLIV}	2	$\overline{PSEN}$ low to valid instruction in		145		$3t_{CLCL}-105$	ns
t_{PXIX}	2	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	2	Input instruction float after $\overline{PSEN}$		59		$t_{CLCL}-25$	ns
t_{AVIV}	2	Address to valid instruction in		312		$5t_{CLCL}-105$	ns
t_{PLAZ}	2	$\overline{PSEN}$ low to address float		10		10	ns
Data Memory							
t_{RLRH}	3, 4	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	3, 4	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	3, 4	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	3, 4	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	3, 4	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	3, 4	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	3, 4	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	3, 4	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	3, 4	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	3, 4	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{WHQX}	3, 4	Data hold after $\overline{WR}$	33		$t_{CLCL}-50$		ns
t_{RLAZ}	3, 4	$\overline{RD}$ low to address float		0		0	ns
t_{WHLH}	3, 4	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
Shift Register							
t_{XLXL}	5	Serial port clock cycle time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	5	Output data setup to clock rising edge	700		$10t_{CLCL}-133$		ns
t_{XHQX}	5	Output data hold after clock rising edge	50		$2t_{CLCL}-117$		ns
t_{XHDX}	5	Input data hold after clock rising edge	0		0		ns
t_{XHDV}	5	Clock rising edge to input data valid		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.

AC ELECTRICAL CHARACTERISTICS (Continued)

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Port 6 Input (input rise and fall times = 5ns)							
t_{FLFH}	8	$\overline{PE}$ width	270		$3t_{CLCL}+20$		ns
t_{ILIH}	8	$\overline{IDS}$ width	270		$3t_{CLCL}+20$		ns
t_{DVIH}	8	Data setup to $\overline{IDS}$ high or $\overline{PE}$ high	0		0		ns
t_{IHDX}	8	Data hold after $\overline{IDS}$ high or $\overline{PE}$ high	30		30		ns
t_{IVFV}	9	$\overline{IDS}$ to BFLAG (IBF) delay		130		130	ns
Port 6 Output							
t_{OLOH}	6	$\overline{ODS}$ width	270		$3t_{CLCL}+20$		ns
t_{FVDV}	7	SEL to data out delay		85		85	ns
t_{OLDV}	6	$\overline{ODS}$ to data out delay		80		80	ns
t_{OHDZ}	6	$\overline{ODS}$ to data float delay		35		35	ns
t_{OVFV}	6	$\overline{ODS}$ to AFLAG (OBF) delay		100		100	ns
t_{FLDV}	6	$\overline{PE}$ to data out delay		120		120	ns
t_{OHFH}	7	$\overline{ODS}$ high to AFLAG (SEL) delay	100		100		ns
External Clock							
t_{CHCX}	10	High time	20		20		ns
t_{CLCX}	10	Low time	20		20		ns
t_{CLCH}	10	Rise time		20		20	ns
t_{CHCL}	10	Fall time		20		20	ns

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
F – $\overline{PE}$, SEL, or IBF
H – Logic level high
I – Instruction (program memory contents), or input data strobe
L – Logic level low, or ALE
O – Output data strobe
P– $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.L – Logic I

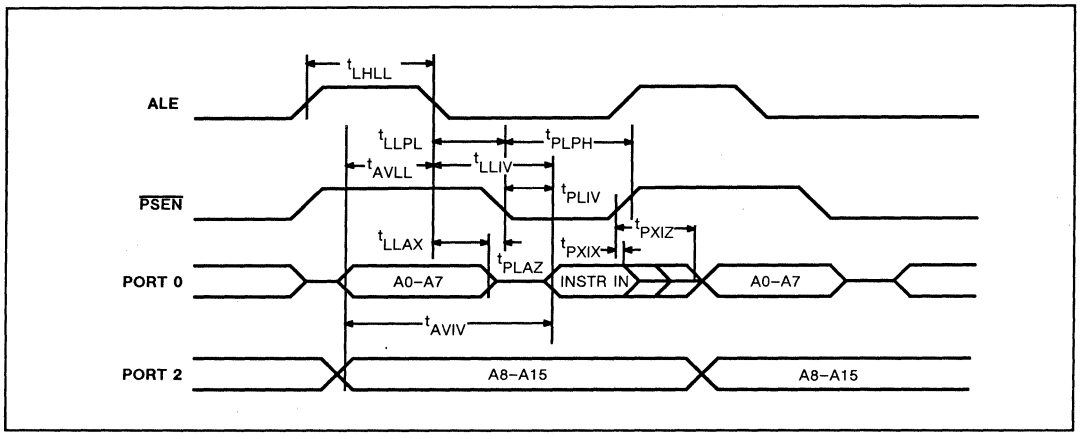


Figure 2. External Program Memory Read Cycle

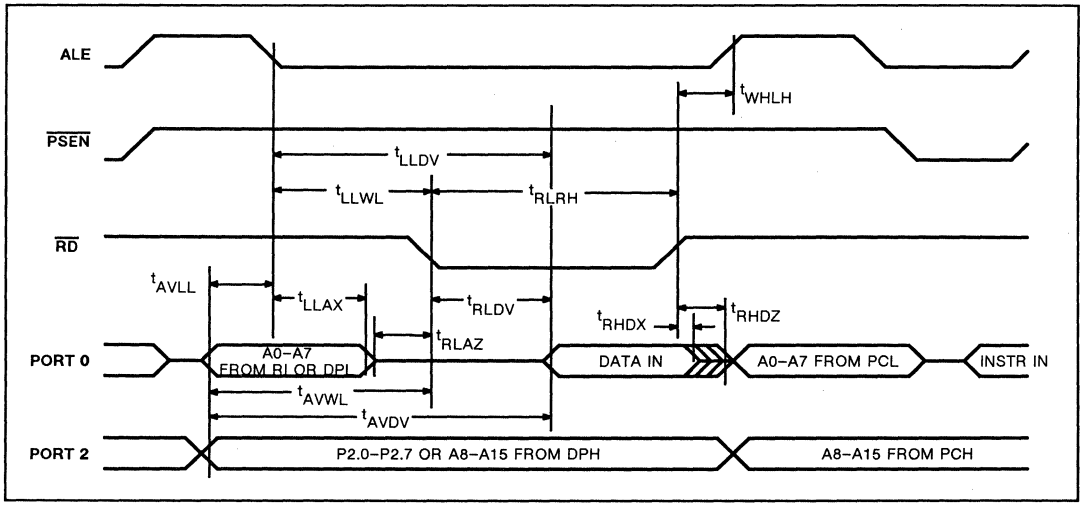


Figure 3. External Data Memory Read Cycle

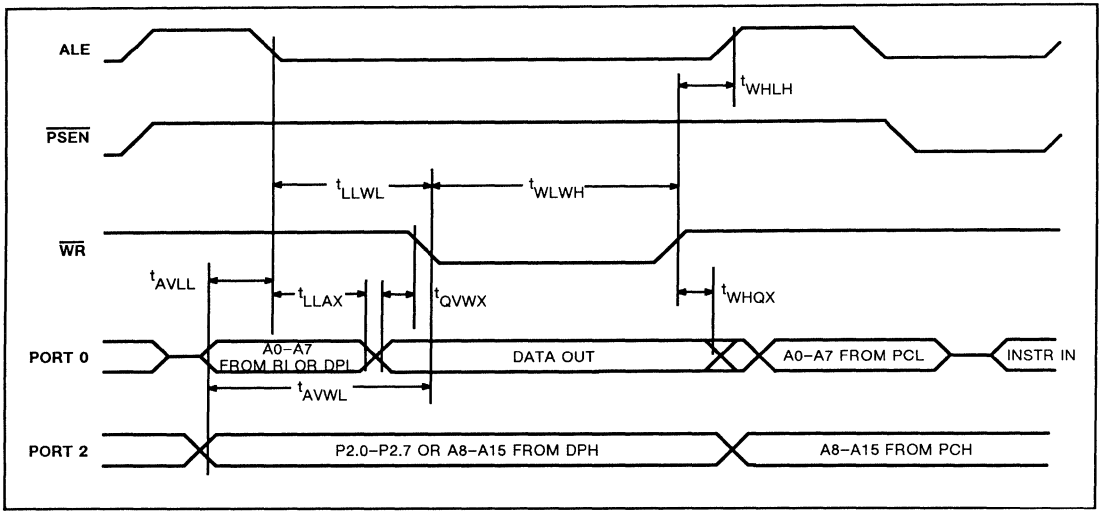


Figure 4. External Data Memory Write Cycle

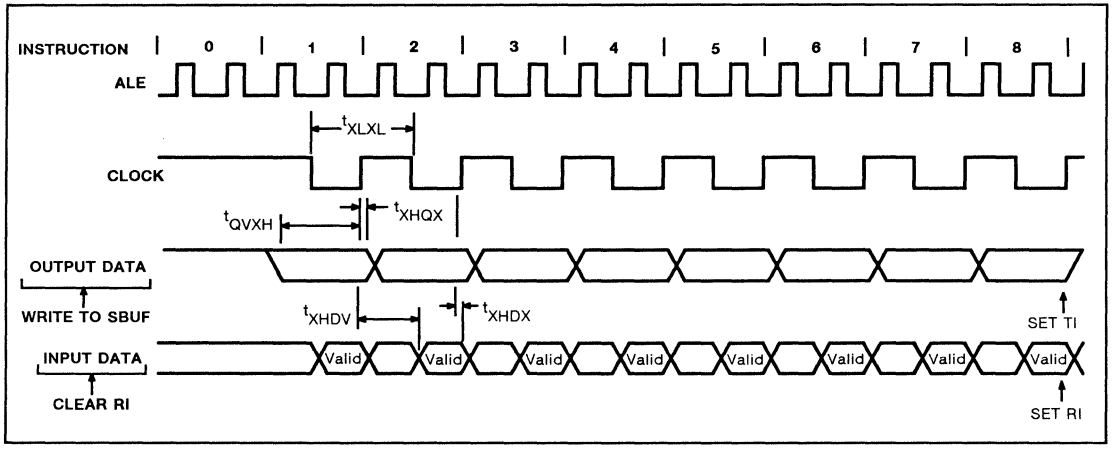


Figure 5. Shift Register Mode Timing

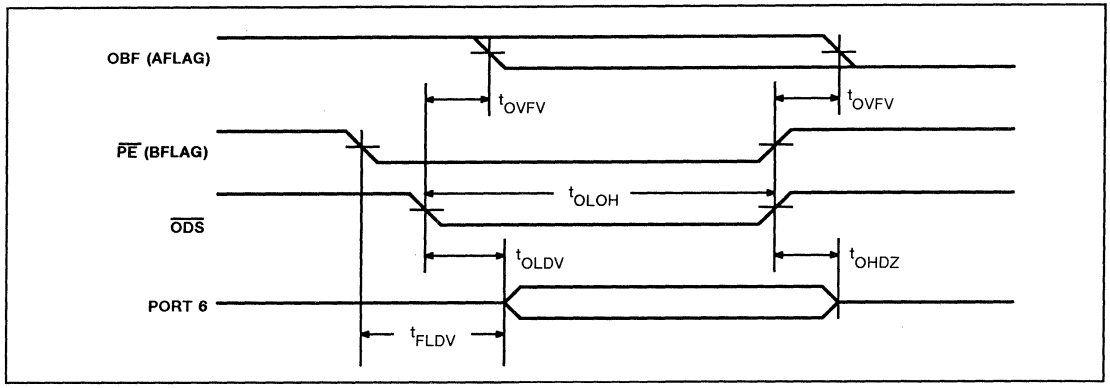


Figure 6. Port 6 Output

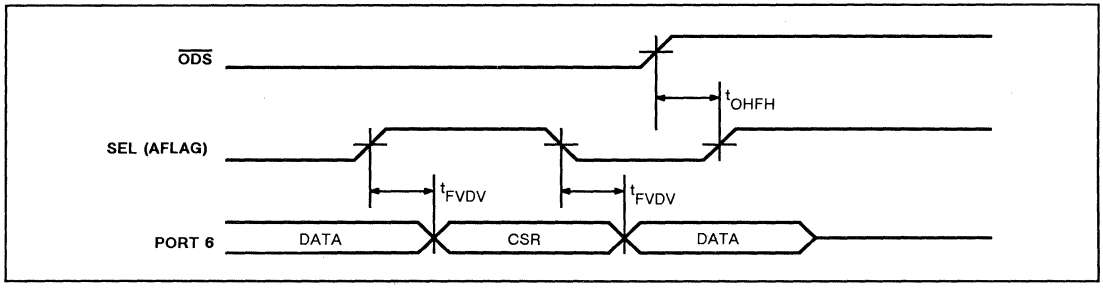


Figure 7. Port 6 Select Mode

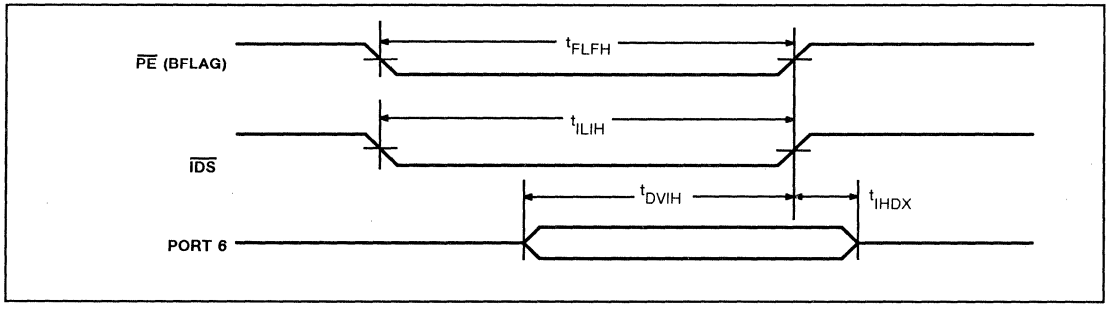


Figure 8. Port 6 Input

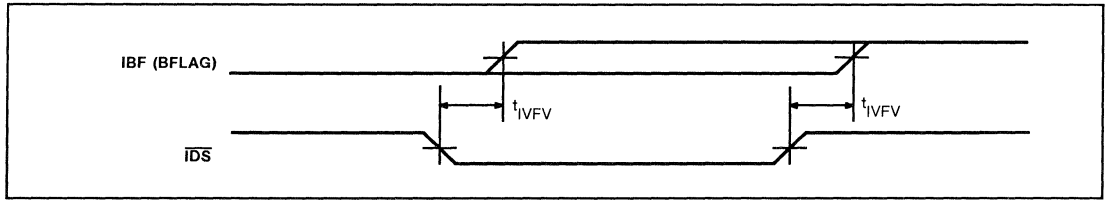


Figure 9. IBF Flag Output

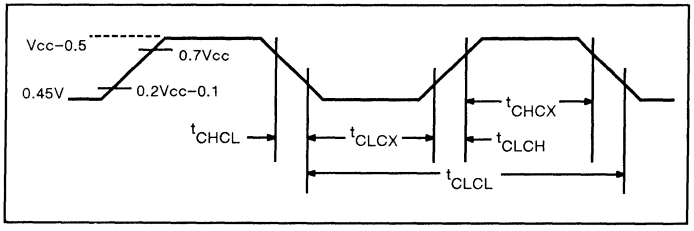
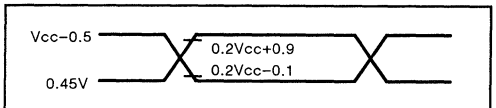
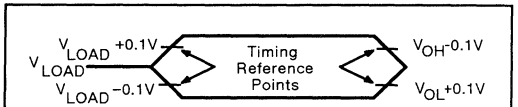


Figure 10. External Clock Drive



AC inputs during testing are driven at Vcc-0.5 for a logic "1" and 0.45V for a logic "0". Timing measurements are made at V_{IH} min for a logic "1" and V_{IL} max for a logic "0".

Figure 11. AC Testing Input/Output



For timing purposes, a port is no longer floating when a 100mV change from load voltage occurs, and begins to float when a 100mV change from the loaded V_{OH}/V_{OL} level occurs. $I_{OH}/I_{OL} \geq \pm 20mA$.

Figure 12. Float Waveform

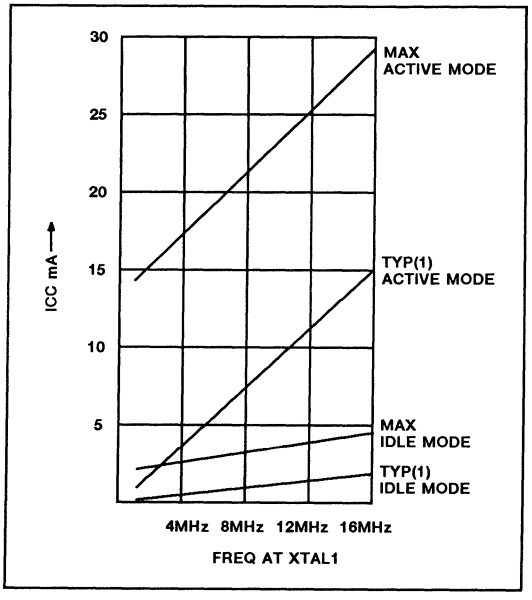


Figure 13. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test
TYP(1) – See DC Electrical Characteristics

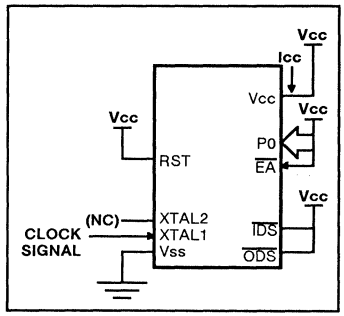


Figure 14. I_{CC} Test Condition, Active Mode
All other pins are disconnected

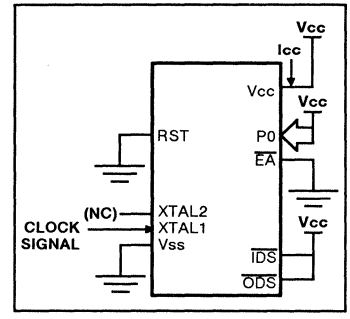


Figure 15. I_{CC} Test Condition, Idle Mode
All other pins are disconnected

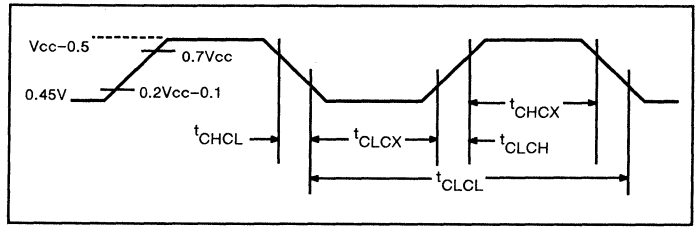


Figure 16. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

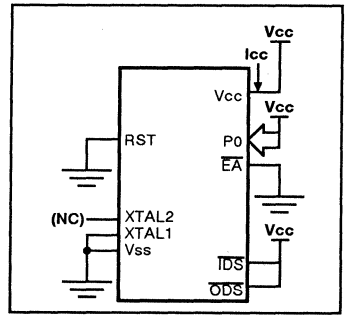


Figure 17. I_{CC} Test Conditions, Power Down Mode
All other pins are disconnected. V_{CC} = 2V to 5.5V

Signetics

SC87C451
CMOS Single-Chip 8-Bit EPROM Microcontroller

Product Specification

Microprocessor Division

DESCRIPTION

The Signetics SC87C451 is an I/O expanded, single-chip microcontroller fabricated with Signetics high-density CMOS technology. Signetics epitaxial substrate minimizes latch-up sensitivity.

The SC87C451 has 4K of EPROM on-chip as program memory and is otherwise identical to the SC83C451.

The SC87C451 is a functional extension of the SC87C51 microcontroller with three additional I/O ports and four I/O control lines. The LCC version has a total of 68 pins. Four control lines associated with port 6 facilitate high speed asynchronous I/O functions.

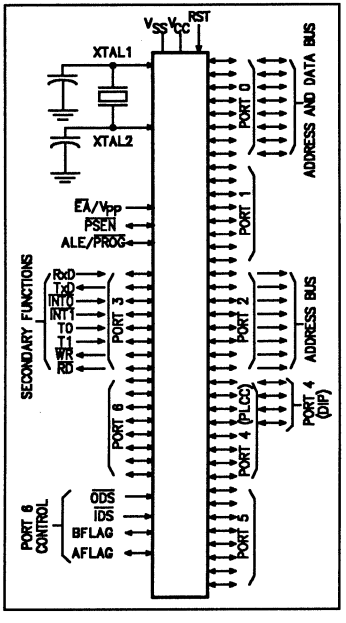
The SC87C451 includes a 4K X 8 EPROM, a 128 X 8 RAM, 58 (LCC) or 54 (DIP) I/O lines, two 16-bit timer/counters, a five source, two priority level, nested interrupt structure, a serial I/O port for either a full duplex UART, I/O expansion, or multiprocessor communications, and an on-chip oscillator and clock circuits.

The SC87C451 has two software selectable modes of reduced activity for further power reduction: idle mode and power-down mode. Idle mode freezes the CPU while allowing the RAM, timers, serial port, and interrupt system to continue functioning. Power-down mode freezes the oscillator, causing all other chip functions to be inoperative while maintaining the RAM contents.

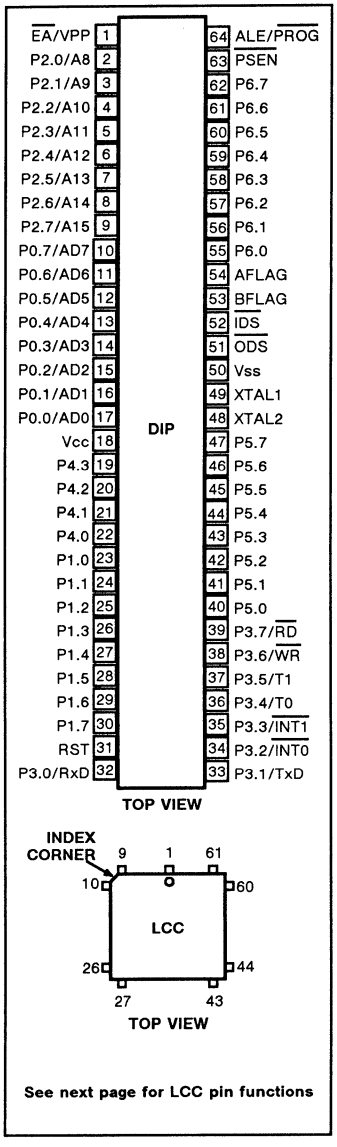
FEATURES

- **User programmable microcontroller**
- **SC80C51 based architecture**
- **68-pin LCC and 64-pin DIP packages:**
  - **Seven 8-bit I/O ports (LCC version)**
  - **Six 8-bit ports and one 4-bit port (DIP version)**
- **Port 6 features:**
  - **8 data pins**
  - **4 control pins**
  - **Direct MPU bus interface**
  - **Parallel printer interface**
- **On the microcontroller**
  - **4K X 8 EPROM**
  - **128 X 8 RAM**
  - **Two 16-bit counter/timers**
  - **Two external interrupts**
- **External memory addressing capability**
  - **64K ROM and 64K RAM**
- **Low power consumption:**
  - **Normal operation: less than 24mA at 5V, 12MHz**
  - **Idle mode**
  - **Power-down mode**

LOGIC SYMBOL



PIN CONFIGURATIONS

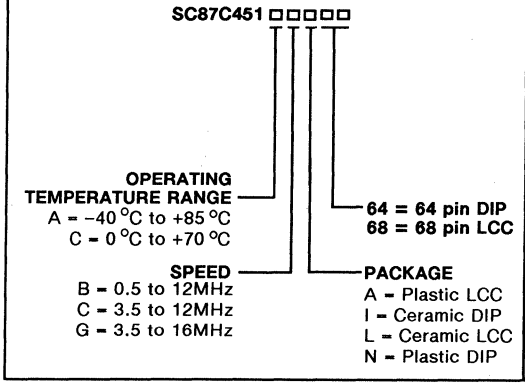


TOP VIEW

TOP VIEW

See next page for LCC pin functions

ORDERING INFORMATION

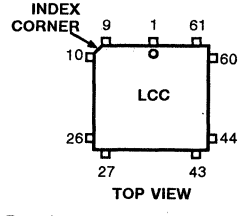


PART NUMBER SELECTION

Part No.	Speed	Temperature and Package
SC87C451CCI64	3.5 to 12MHz	0 to +70°C ceramic DIP
SC87C451CGI64	3.5 to 16MHz	0 to +70°C ceramic DIP*
SC87C451CBI64	0.5 to 12MHz	0 to +70°C ceramic DIP
SC87C451CCL68	3.5 to 12MHz	0 to +70°C ceramic LCC
SC87C451CGL68	3.5 to 16MHz	0 to +70°C ceramic LCC*
SC87C451CBL68	0.5 to 12MHz	0 to +70°C ceramic LCC
SC87C451CCN64	3.5 to 12MHz	0 to +70°C plastic DIP
SC87C451CGN64	3.5 to 16MHz	0 to +70°C plastic DIP*
SC87C451CBN64	0.5 to 12MHz	0 to +70°C plastic DIP
SC87C451CCA68	3.5 to 12MHz	0 to +70°C plastic LCC
SC87C451CGA68	3.5 to 16MHz	0 to +70°C plastic LCC*
SC87C451CBA68	0.5 to 12MHz	0 to +70°C plastic LCC

*PRELIMINARY SPECIFICATION

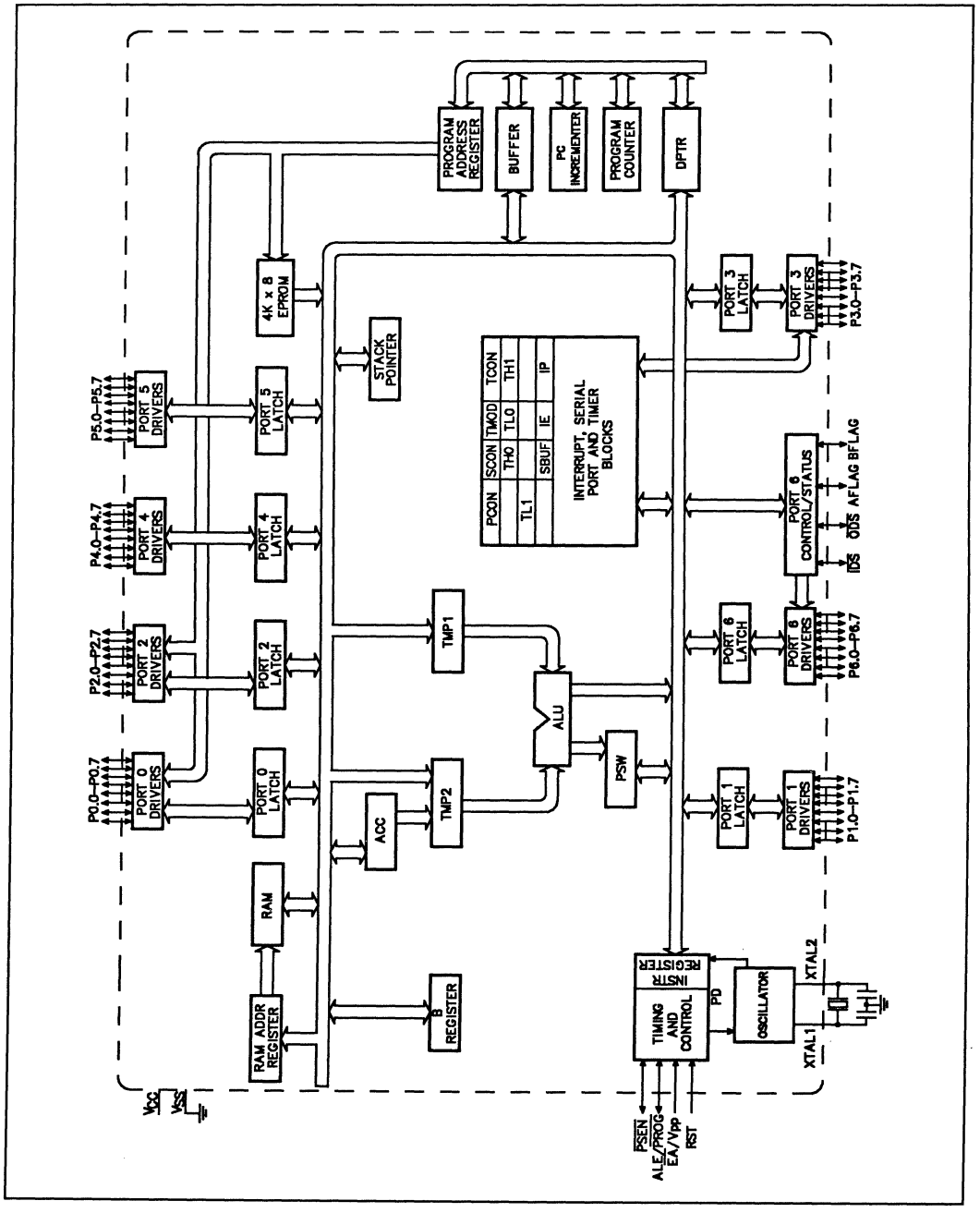
LCC PIN FUNCTIONS



TOP VIEW

Pin	Function	Pin	Function	Pin	Function
1	$\overline{EA}$/VPP	24	P4.2	47	P5.3
2	P2.0/A8	25	P4.1	48	P5.4
3	P2.1/A9	26	P4.0	49	P5.5
4	P2.2/A10	27	P1.0	50	P5.6
5	P2.3/A11	28	P1.1	51	P5.7
6	P2.4/A12	29	P1.2	52	XTAL2
7	P2.5/A13	30	P1.3	53	XTAL1
8	P2.6/A14	31	P1.4	54	Vss
9	P2.7/A15	32	P1.5	55	$\overline{ODS}$
10	P0.7/AD7	33	P1.6	56	$\overline{IDS}$
11	P0.6/AD6	34	P1.7	57	BFLAG
12	P0.5/AD5	35	RST	58	AFLAG
13	P0.4/AD4	36	P3.0/RxD	59	P6.0
14	P0.3/AD3	37	P3.1/$\overline{TxD}$	60	P6.1
15	P0.2/AD2	38	P3.2/$\overline{INT0}$	61	P6.2
16	P0.1/AD1	39	P3.3/$\overline{INT1}$	62	P6.3
17	P0.0/AD0	40	P3.4/T0	63	P6.4
18	Vcc	41	P3.5/$\overline{T1}$	64	P6.5
19	P4.7	42	P3.6/$\overline{WR}$	65	P6.6
20	P4.6	43	P3.7/$\overline{RD}$	66	P6.7
21	P4.5	44	P5.0	67	$\overline{PSEN}$
22	P4.4	45	P5.1	68	ALE/$\overline{PROG}$
23	P4.3	46	P5.2		

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. LCC	TYPE	NAME AND FUNCTION
V_{SS}	50	54	I	**Ground:** 0V reference.
V_{CC}	18	18	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–P0.7	17–10	17–10	I/O	**Port 0:** Port 0 is an 8-bit open-drain, bidirectional I/O port. Port 0 is also the multiplexed data and low-order address bus during accesses to external memory. External pullups are required during program verification. Port 0 can sink/source eight LS TTL inputs.
P1.0–P1.7	23–30	27–34	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pullups. Port 1 receives the low-order address bytes during program memory verification. Port 1 can sink/source three LS TTL inputs, and drive CMOS inputs without external pullups.
P2.0–P2.7	2–9	2–9	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pullups. Port 2 emits the high-order address bytes during access to external memory and receives the high-order address bits and control signals during program verification. Port 2 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups.
P3.0–P3.7	32–39	36–43	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 2 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups. Port 3 also serves the special functions listed below:
	32	36	I	**RxD (P3.0):** Serial input port
	33	37	O	**TxD (P3.1):** Serial output port
	34	38	I	**$\overline{INT0}$ (P3.2):** External interrupt 0
	35	39	I	**$\overline{INT1}$ (P3.3):** External interrupt 1
	36	40	I	**T0 (P3.4):** Timer 0 external input
	37	41	I	**T1 (P3.5):** Timer 1 external input
	38	42	O	**$\overline{WR}$ (P3.6):** External data memory write strobe
	39	43	O	**$\overline{RD}$ (P3.7):** External data memory read strobe
P4.0–P4.3	22–19		I/O	**Port 4:** Port 4 is a 4/8-bit (DIP/LCC) bidirectional I/O port with internal pullups. Port 4 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups.
P4.0–P4.7		26–19	I/O	
P5.0–P5.7	40–47	44–51	I/O	**Port 5:** Port 5 is an 8-bit bidirectional I/O port with internal pullups. Port 5 can sink/source three LS TTL inputs and drive CMOS inputs without external pullups.
P6.0–P6.7	55–62	59–66	I/O	**Port 6:** Port 6 is a specialized 8-bit bidirectional I/O port with internal pullups. This special port can sink/source three LS TTL inputs and drive CMOS inputs without external pullups. Port 6 can be used in a strobed or non-strobed mode of operation. Port 6 works in conjunction with four control pins that serve the functions listed below:
				Port 6 Control Lines:
	51	55		**$\overline{ODS}$:** Output data strobe
	52	56		**IDS:** Input data strobe
	53	57		**BFLAG:** Bidirectional I/O pin with internal pullups
	54	58		**AFLAG:** Bidirectional I/O pin with internal pullups
RST	31	35	I	**Reset:** A high on this pin, for two machine cycles while the oscillator is running, resets the device. An internal pull-down resistor permits a power-on reset using only a capacitor connected to V_{CC}.
ALE/$\overline{PROG}$	64	68	I/O	**Address Latch Enable/Program Pulse:** Output pulse for latching the low byte of the address during accesses to external memory. ALE is activated at a constant rate of 1/6 the oscillator frequency except during an external data memory access, at which time one ALE is skipped. ALE can sink/source eight LS TTL inputs and drive CMOS inputs without an external pullup. This pin is also the program pulse during EPROM programming.
$\overline{PSEN}$	63	67	I/O	**Program Store Enable:** The read strobe to external program memory. PSEN is activated twice each machine cycle during fetches from external program memory. However, when executing out of external program memory, two activations of PSEN are skipped during each access to external data memory. PSEN is not activated during fetches from internal program memory. PSEN can sink/source eight LS TTL inputs and drive CMOS inputs without an external pullup. This pin should be tied low during programming.
$\overline{EA}$/V_{PP}	1	1	I	**Instruction Execution Control/Programming Supply Voltage:** When $\overline{EA}$ is held high, the CPU executes out of internal program memory, unless the program counter exceeds 0FFFH. When $\overline{EA}$ is held low, the CPU executes out of external program memory. $\overline{EA}$ must never be allowed to float. This pin also receives the 12.75V programming supply voltage (V_{PP}) during EPROM programming.
XTAL1	49	53	I	**Crystal 1:** Input to the inverting amplifier that forms the oscillator. This input receives the external oscillator when an external oscillator is used.
XTAL2	48	52	O	**Crystal 2:** An output of the inverting amplifier that forms the oscillator. This pin should be floated when an external oscillator is used.

PORTS 4 AND 5
Ports 4 and 5 are bidirectional I/O ports with internal pullups. Port 4 is an 8-bit port (LCC version) or a 4-bit port (DIP version). Port 4 and port 5 pins with ones written to them, are pulled high by the internal pullups, and in that state can be used as inputs. Port 4 and 5 are addressed at the special function register addresses shown in Table 1.

PORT 6
Port 6 is a special 8-bit bidirectional I/O port with internal pullups (see Figure 1). This port can be used as a standard I/O port, or in strobed modes of operation in conjunction with four special control lines: $\overline{ODS}$, $\overline{IDS}$, AFLAG, and BFLAG. Port 6 operating modes are controlled by the port 6 control status register (CSR). Port 6 and the CSR are addressed at the special function register addresses shown in Table 1. The following four control pins are used in conjunction with port 6:

ODS – Output data strobe input for port 6. $\overline{ODS}$ can be programmed to control the port 6 output drivers and the output buffer full flag (OBF), or to clear only the OBF flag bit in the CSR (output-always mode). $\overline{ODS}$ is active low for output driver control. The OBF flag can be programmed to be cleared on the negative edge of $\overline{ODS}$.

IDS– Input data strobe for port 6. $\overline{IDS}$ is used to control the port 6 input latch and input buffer full flag (IBF) bit in the CSR. The input data latch can be programmed to be transparent when $\overline{IDS}$ is low and latched on the positive transition of $\overline{IDS}$, or to latch only on the positive transition of $\overline{IDS}$. Correspondingly, the IBF flag is set on the negative or positive transition of $\overline{IDS}$.

AFLAG – AFLAG is a bidirectional I/O pin which can be programmed to be an output set high or low under program control, or to output the state of the output buffer full flag. AFLAG can also be programmed to be an input which selects whether the contents of the output buffer, or the contents of the port 6 control status register will be output on port 6. This feature grants complete port 6 status to external devices.

BFLAG – BFLAG is a bidirectional I/O pin which can be programmed to be an output, set high or low under program control, or to output the state of the input buffer full flag. BFLAG can also be programmed to input an enable signal for port 6. When BFLAG is used as an enable input, port 6 output drivers are in the high-impedance state, and the input latch does not respond to the $\overline{IDS}$ strobe when BFLAG is high. Both features are enabled when BFLAG is low. This feature facilitates the use of the SC87C451 in bused multiprocessor systems.

Table 1. Special Function Register Addresses

REGISTER ADDRESS			BIT ADDRESS							
Name	Symbol	Address	MSB							LSB
Port 4	P4	C0	C7	C6	C5	C4	C3	C2	C1	C0
Port 5	P5	C8	CF	CE	CD	CC	CB	CA	C9	C8
Port 6 data	P6	D8	DF	DE	DD	DC	DB	DA	D9	D8
Port 6 control status	CSR	E8	EF	EE	ED	EC	EB	EA	E9	E8

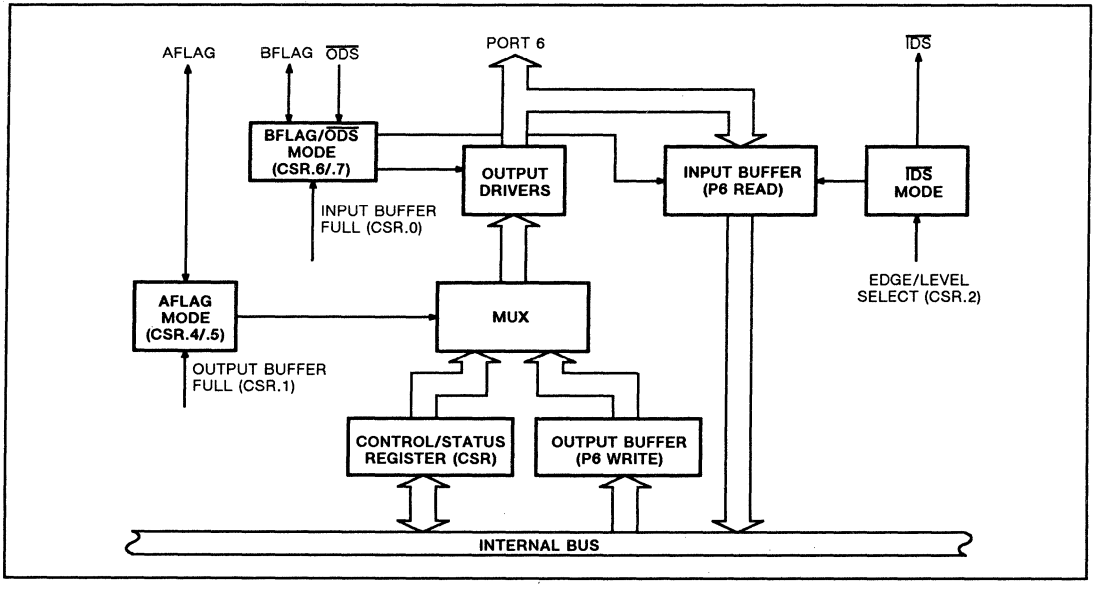


Figure 1. Port 6 Block Diagram

CONTROL STATUS REGISTER
The control status register (CSR) establishes the mode of operation for port 6 and indicates the current status of port 6 I/O registers. All control status register bits can be read and written by the CPU, except bits 0 and 1, which are read only. Reset writes ones to bits 2 through 7, and writes zeros to bits 0 and 1 (see Table 2).

CSR.0 Input Buffer Full Flag (IBF) (Read Only) – The IBF bit is set to a logic 1 when port 6 data is loaded into the input buffer under control of $\overline{IDS}$. This can occur on the negative or positive edge of $\overline{IDS}$, as determined by CSR.2. IBF is cleared when the CPU reads the input buffer register.

CSR.1 Output Buffer Full Flag (OBF) (Read Only) – The OBF flag is set to a logic 1 when the CPU writes to the port 6 output data buffer. OBF is cleared by the positive or negative edge of $\overline{ODS}$, as determined by CSR.3.

CSR.2 IDS Mode Select (IDSM) – When CSR.2 = 0, a low-to-high transition on the $\overline{IDS}$ pin sets the IBF flag. The port 6 input buffer is loaded on the $\overline{IDS}$ positive edge. When CSR.2 = 1, a high-to-low transition on the $\overline{IDS}$ pin sets the IBF flag. When port 6 input buffer is transparent when $\overline{IDS}$ is low, and latched when $\overline{IDS}$ is high.

CSR.3 Output Buffer Full Flag Clear Mode (OBFC) – When CSR.3 = 1, the positive edge of the $\overline{ODS}$ input clears the OBF flag. When CSR.3 = 0, the negative edge of the $\overline{ODS}$ input clears the OBF flag.

CSR.4, CSR.5 AFLAG Mode Select (MA0, MA1) – Bits 4 and 5 select the mode of operation for the AFLAG pin, as follows:

MA1	MA0	AFLAG Function
0	0	Logic 0 output
0	1	Logic 1 output
1	0	OBF flag output (CSR.1)
1	1	Select (SEL) input mode

The select (SEL) input mode is used to determine whether the port 6 data register or the control status register is output on port 6. When the select feature is enabled, the AFLAG input controls the source of port 6 output data. A logic 0 on AFLAG input selects the port 6 data register, and a logic 1 on AFLAG input selects the control status register.

CSR.6, CSR.7 BFLAG Mode Select (MB0, MB1) – Bits 6 and 7 select the mode operation as follows:

MA1	MA0	AFLAG Function
0	0	Logic 0 output
0	1	Logic 1 output
1	0	IBF flag output (CSR.0)
1	1	Port enable ($\overline{PE}$)

In the port enable mode, $\overline{IDS}$ and $\overline{ODS}$ inputs are disabled when BFLAG input is high. When the BFLAG input is low, the port is enabled for I/O.

SPECIAL FUNCTION REGISTER ADDRESSES
Special function register addresses for the SC87C451 are identical to those of the SC80C51, except for the additional registers listed in Table 1.

Table 2. Control Status Register (CSR)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
BFLAG Mode Select		AFLAG Mode Select		Output Buffer Flag Clear Mode	Input Data Strobe Mode	Output Buffer Full Flag	Input Buffer Full Flag
0/0 = Logic 0 output* 0/1 = Logic 1 output* 1/0 = IBF output 1/1 = PE input (0 = Select) (1 = Disable I/O)		0/0 = Logic 0 output 0/1 = Logic 1 output 1/0 = OBF output* 1/1 = SEL input (0 = Data) (1 = Control/status)		0 = Negative edge of $\overline{ODS}$ 1 = Positive edge of $\overline{ODS}$	0 = Positive edge of $\overline{IDS}$ 1 = Low level of $\overline{IDS}$	0 = Output data buffer empty 1 = Output data buffer full	0 = Input data buffer empty 1 = Input data buffer full

NOTE:
*Output-always mode: MB1 = 0, MA1 = 1, and MA0 = 0. In this mode, port 6 is always enabled for output. $\overline{ODS}$ only clears the OBF flag.

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Operating temperature under bias	0 to +70 −40 to +85	°C
Storage temperature range	−65 to +150	°C
Voltage on $\overline{EA}/V_{PP}$ pin to V_{SS}	0 to +13.0	V
Voltage on any other pin to V_{SS}	−0.5 to + 6.5	V
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.5	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V ±10%, V_{SS} = 0V

Symbol	Parameter	Test Conditions	Limits			Unit
			Min	Typical[1]	Max	
V_{IL}	Input low voltage, except $\overline{EA}$		−0.5		$0.2V_{CC}-0.1$	V
V_{IL1}	Input low voltage to $\overline{EA}$		0		$0.2V_{CC}-0.3$	V
V_{IH}	Input high voltage, except XTAL1, RST		$0.2V_{CC}+.9$		$V_{CC}+0.5$	V
V_{IH1}	Input high voltage, XTAL1, RST		$0.7V_{CC}$		$V_{CC}+0.5$	V
V_{OL}	Output low voltage, ports 1, 2, 3, 4, 5, 6	I_{OL} = 1.6mA[2]			0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$	I_{OL} = 3.2mA[2]			0.45	V
V_{OH}	Output high voltage, ports 1, 2, 3, 4, 5, 6	I_{OH} = −60µA I_{OH} = −25µA I_{OH} = −10µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
V_{OH1}	Output high voltage (port 0 in external bus mode, ALE, $\overline{PSEN}$)[3]	I_{OH} = −800µA I_{OH} = −300µA I_{OH} = −80µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
I_{IL}	Logical 0 input current, ports 1, 2, 3, 4, 5, 6	V_{IN} = 0.45V			−50	µA
I_{TL}	Logical 1-to-0 transition current, ports 1, 2, 3, 4, 5, 6	See note 4			−650	µA
I_{LI}	Input leakage current, port 0	V_{IN} = V_{IL} or V_{IH}			±10	µA
I_{CC}	Power supply current: Active mode @ 12MHz[5] Idle mode @ 12MHz[5] Power down mode	See note 6		11.5 1.3 3	25 4 50	 mA mA µA
R_{RST}	Internal reset pulldown resistor		50		300	kΩ
C_{IO}	Pin capacitance[7] − DIP package − PLCC package				15 10	pF pF

NOTES:
1. Typical ratings are based on a limited number of samples taken from early manufacturing lots and are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the $0.9V_{CC}$ specification when the address bits are stabilizing.
4. Pins of ports 1, 2, and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. I_{CC}MAX at other frequencies is given by:
   Active mode: I_{CC}MAX = 0.94 X FREQ + 13.71
   Idle mode: I_{CC}MAX = 0.14 X FREQ + 2.31
   where FREQ is the external oscillator frequency in MHz. I_{CC}MAX is given in mA. See Figure 13.
6. See Figures 14 through 17 for I_{CC} test conditions.
7. C_{IO} applies to Ports 1, 2, 3, 4, 5, 6, AFLAG, BFLAG, XTAL1, XTAL2.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V ±10%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	2	Oscillator frequency: **Speed Versions** SC87C451 B SC87C451 C SC87C451 G			 0.5 3.5 3.5	 12 12 16	 MHz MHz MHz
t_{LHLL}	2	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	2	Address valid to ALE low	28		$t_{CLCL}-55$		ns
t_{LLAX}	2	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	2	ALE low to valid instruction in		234		$4t_{CLCL}-100$	ns
t_{LLPL}	2	ALE low to $\overline{PSEN}$ low	43		$t_{CLCL}-40$		ns
t_{PLPH}	2	$\overline{PSEN}$ pulse width	205		$3t_{CLCL}-45$		ns
t_{PLIV}	2	$\overline{PSEN}$ low to valid instruction in		145		$3t_{CLCL}-105$	ns
t_{PXIX}	2	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	2	Input instruction float after $\overline{PSEN}$		59		$t_{CLCL}-25$	ns
t_{AVIV}	2	Address to valid instruction in		312		$5t_{CLCL}-105$	ns
t_{PLAZ}	2	$\overline{PSEN}$ low to address float		10		10	ns
Data Memory							
t_{RLRH}	3, 4	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	3, 4	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	3, 4	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	3, 4	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	3, 4	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	3, 4	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	3, 4	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	3, 4	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	3, 4	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	3, 4	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{WHQX}	3, 4	Data hold after $\overline{WR}$	33		$t_{CLCL}-50$		ns
t_{RLAZ}	3, 4	$\overline{RD}$ low to address float		0		0	ns
t_{WHLH}	3, 4	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
Shift Register							
t_{XLXL}	5	Serial port clock cycle time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	5	Output data setup to clock rising edge	700		$10t_{CLCL}-133$		ns
t_{XHQX}	5	Output data hold after clock rising edge	50		$2t_{CLCL}-117$		ns
t_{XHDX}	5	Input data hold after clock rising edge	0		0		ns
t_{XHDV}	5	Clock rising edge to input data valid		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and PSEN = 100pF, load capacitance for all other outputs = 80pF.

AC ELECTRICAL CHARACTERISTICS (Continued)

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Port 6 Input (input rise and fall times = 5ns)							
t_{FLFH}	8	$\overline{PE}$ width	270		$3t_{CLCL}+20$		ns
t_{ILIH}	8	$\overline{IDS}$ width	270		$3t_{CLCL}+20$		ns
t_{DVIH}	8	Data setup to $\overline{IDS}$ high or $\overline{PE}$ high	0		0		ns
t_{IHDX}	8	Data hold after $\overline{IDS}$ high or $\overline{PE}$ high	30		30		ns
t_{IVFV}	9	$\overline{IDS}$ to BFLAG (IBF) delay		130		130	ns
Port 6 Output							
t_{OLOH}	6	$\overline{ODS}$ width	270		$3t_{CLCL}+20$		ns
t_{FVDV}	7	SEL to data out delay		85		85	ns
t_{OLDV}	6	$\overline{ODS}$ to data out delay		80		80	ns
t_{OHDZ}	6	$\overline{ODS}$ to data float delay		35		35	ns
t_{OVFV}	6	$\overline{ODS}$ to AFLAG (OBF) delay		100		100	ns
t_{FLDV}	6	$\overline{PE}$ to data out delay		120		120	ns
t_{OHFH}	7	$\overline{ODS}$ high to AFLAG (SEL) delay	100		100		ns
External Clock							
t_{CHCX}	10	High time	20		20		ns
t_{CLCX}	10	Low time	20		20		ns
t_{CLCH}	10	Rise time		20		20	ns
t_{CHCL}	10	Fall time		20		20	ns

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
F – $\overline{PE}$, SEL, or IBF
H – Logic level high
I – Instruction (program memory contents), or input data strobe
L – Logic level low, or ALE
O – Output data strobe
P– $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.L – Logic I

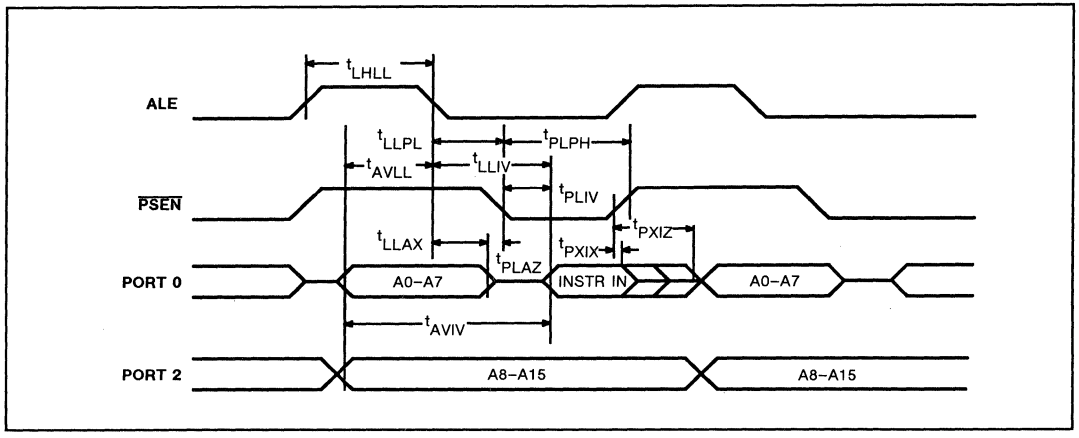


Figure 2. External Program Memory Read Cycle

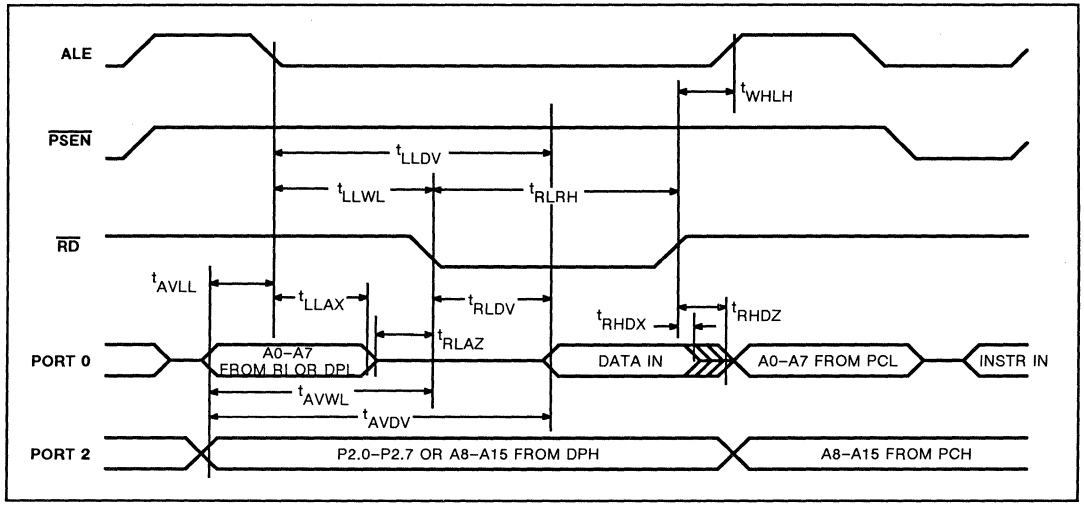


Figure 3. External Data Memory Read Cycle

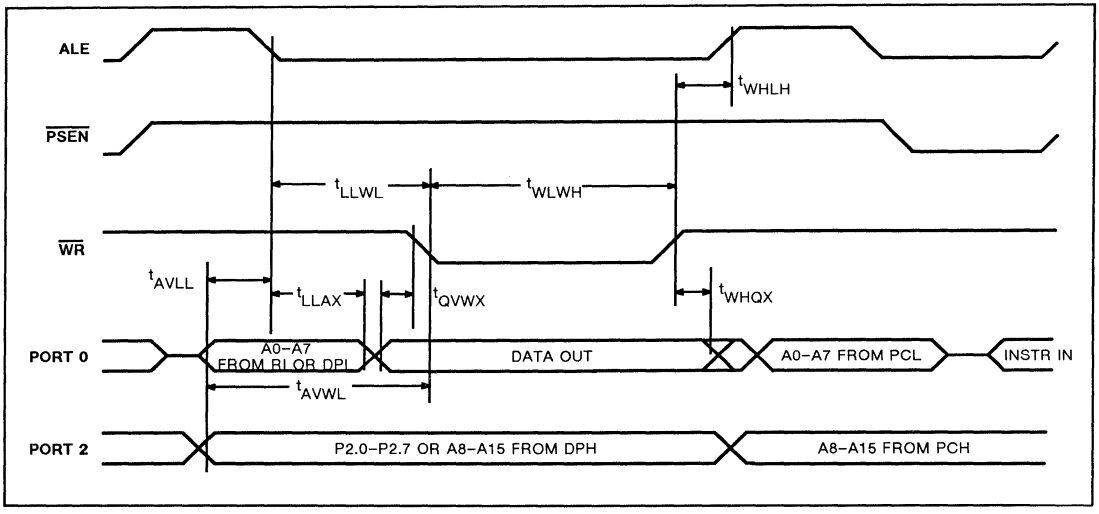


Figure 4. External Data Memory Write Cycle

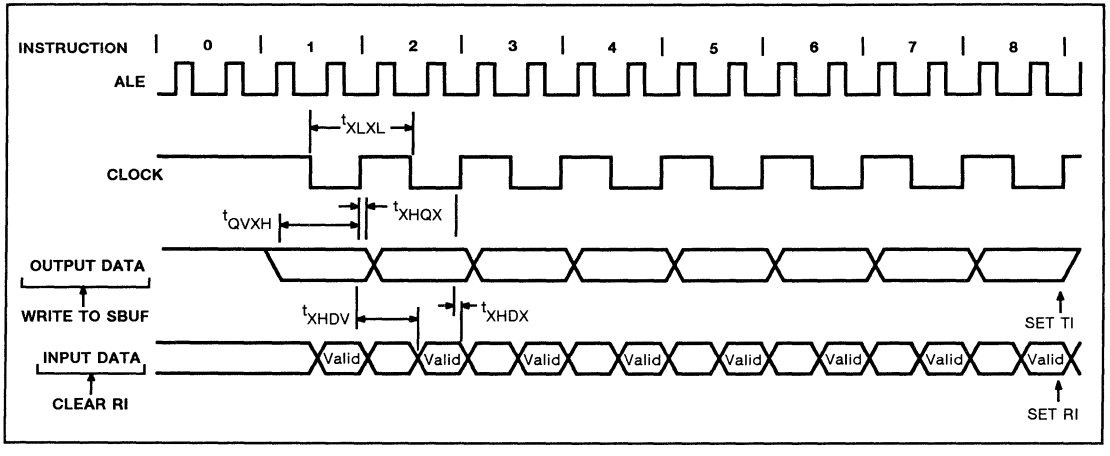


Figure 5. Shift Register Mode Timing

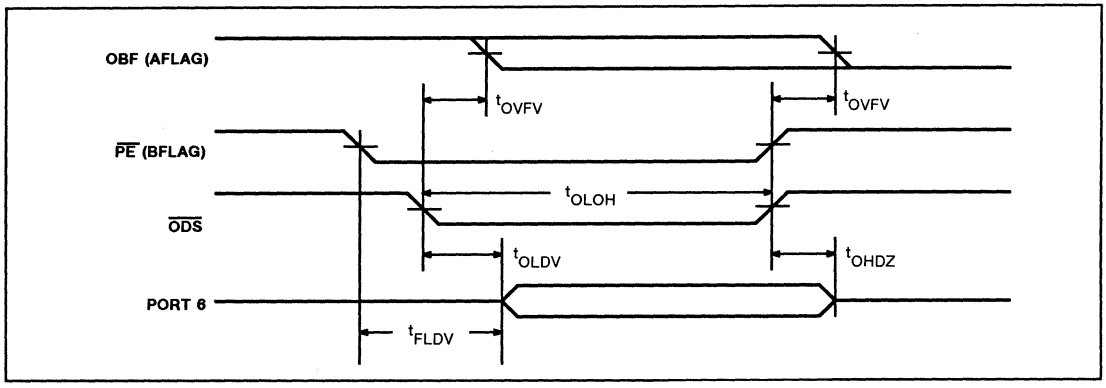


Figure 6. Port 6 Output

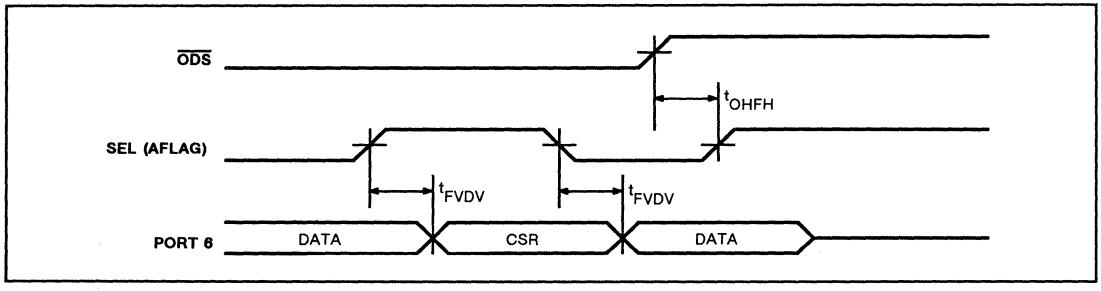


Figure 7. Port 6 Select Mode

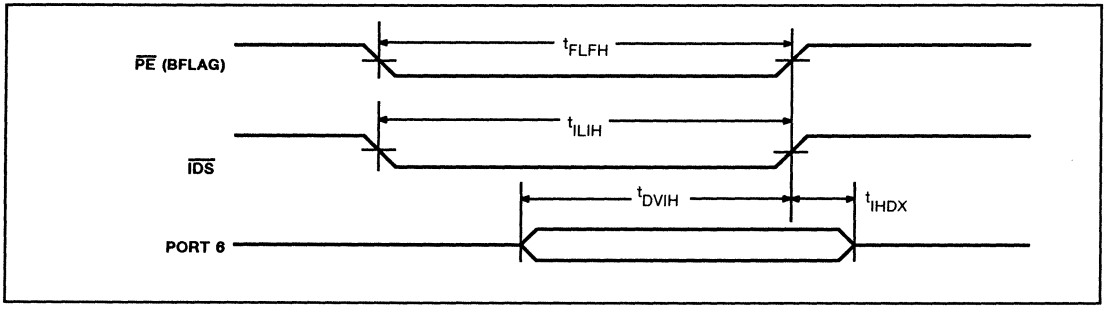


Figure 8. Port 6 Input

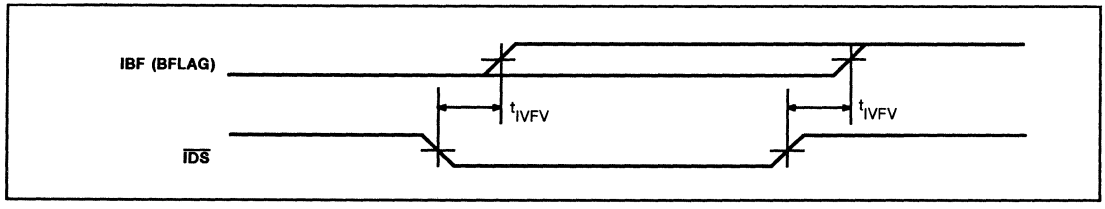


Figure 9. IBF Flag Output

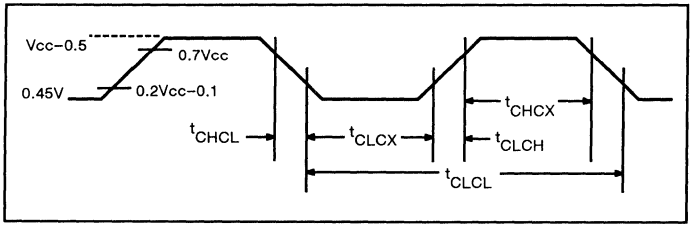


Figure 10. External Clock Drive

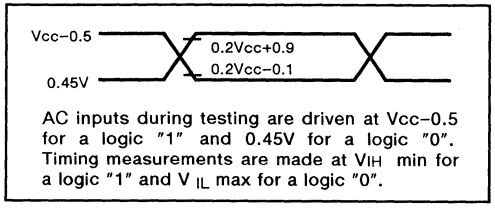


Figure 11. AC Testing Input/Output

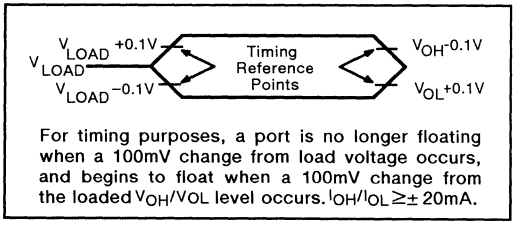


Figure 12. Float Waveform

EPROM CHARACTERISTICS

The SC87C451 is programmed by using a modified Quick-Pulse Programming™ algorithm. It differs from older methods in the value used for VPP (programming supply voltage) and in the width and number of the ALE/$\overline{PROG}$ pulses.

The SC87C451 contains two signature bytes that can be read and used by an EPROM programming system to identify the device. The signature bytes identify the device as an SC87C451 manufactured by Signetics Corporation.

Table 3 shows the logic levels for reading the signature byte, and for programming the program memory, the encryption table, and the lock bits. The circuit configuration and waveforms for quick-pulse programming are shown in Figures 18 and 19. Figure 20 shows the circuit configuration for normal program memory verification.

QUICK-PULSE PROGRAMMING

The setup for microcontroller quick-pulse programming is shown in Figure 18. Note that the SC87C451 is running with a 4 to 6MHz oscillator. The reason the oscillator needs to be running is that the device is executing internal address and program data transfers.

The address of the EPROM location to be programmed is applied to ports 1 and 2, as shown in Figure 18. The code byte to be programmed into that location is applied to port 0. RST, $\overline{PSEN}$ and pins of ports 2 and 3 specified in Table 2 are held at the "Program Code Data" levels indicated in Table 3. The ALE/$\overline{PROG}$ is pulsed low 25 times as shown in Figure 19.

To program the encryption table, repeat the 25 pulse programming sequence for addresses 0 through 1FH, using the "Pgm Encryption Table" levels. Do not forget that after the encryption table is programmed, verification cycles will produce only encrypted data.

To program the lock bits, repeat the 25 pulse programming sequence using the "Pgm Lock Bit" levels. After one lock bit is programmed, further programming of the code memory and encryption table is disabled. However, the other lock bit can still be programmed.

Note that the EA/VPP pin must not be allowed to go above the maximum specified VPP level for any amount of time. Even a narrow glitch above that voltage can cause permanent damage to the device. The VPP source should be well regulated and free of glitches and overshoot.

Program Verification

If lock bit 2 has not been programmed, the on-chip program memory can be read out for program verification. The address of the program memory locations to be read is applied to ports 1 and 2 as shown in Figure 20. The other pins are held at the "Verify Code Data" levels indicated in Table 3. The contents of the address location will be emitted on port 0. External pull-ups are required on port 0 for this operation.

If the encryption table has been programmed, the data presented at port 0 will be the exclusive NOR of the program byte with one of the encryption bytes. The user will have to know the encryption table contents in order to correctly decode the verification data. The encryption table itself cannot be read out.

Reading the Signature Bytes

The signature bytes are read by the same procedure as a normal verification of locations 030H and 031H, except that P3.6 and P3.7 need to be pulled to a logic low. The values are:

(030H) = 15H indicates manufactured by Signetics
(031H) = 90H indicates SC87C451

Program/Verify Algorithms

Any algorithm in agreement with the conditions listed in Table 3, and which satisfies the timing specifications, is suitable.

™Trademark phrase of Intel Corp.

Erasure Characteristics

Erasure of the EPROM begins to occur when the chip is exposed to light with wavelengths shorter than approximately 4,000 angstroms. Since sunlight and fluorescent lighting have wavelengths in this range, exposure to these light sources over an extended time (about 1 week in sunlight, or 3 years in room level fluorescent lighting) could cause inadvertent erasure. **For this and secondary effects, it is recommended that an opaque label be placed over the window.** For elevated temperature or solvent environment, use Kapton tape Fluorglas part number 2345–5 or equivalent.

The recommended erasure procedure is exposure to ultraviolet light (at 2537 angstroms) to an integrated dose of at least 15W–sec/cm^2. Exposing the EPROM to an ultraviolet lamp of 12,000μW/cm^2 rating for 20 to 39 minutes, at a distance of about 1 inch, should be sufficient.

Erasure leaves the array in an all 1s state.

Table 3. EPROM Programming Modes

MODE	RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}$/V_{PP}	P2.7	P2.6	P3.7	P3.6
Read signature	1	0	1	1	0	0	0	0
Program code data	1	0	0*	V_{PP}	1	0	1	1
Verify code data	1	0	1	1	0	0	1	1
Pgm encryption table	1	0	0*	V_{PP}	1	0	1	0
Pgm lock bit 1	1	0	0*	V_{PP}	1	1	1	1
Pgm lock bit 2	1	0	0*	V_{PP}	1	1	0	0

NOTES:

1. "0" = valid low for that pin, "1" = valid high for that pin.
2. **V_{PP} = 12.75V ±0.25V.**
3. V_{CC} = 5V ±10% during programming and verification.

*ALE/$\overline{PROG}$ receives 25 programming pulses while V_{PP} is held at 12.75V. Each programming pulse is low for 100μs (±10μs) and high for a minimum of 10μs.

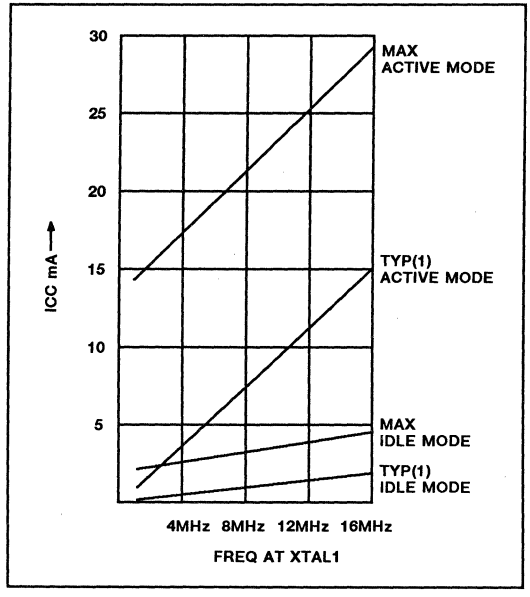


Figure 13. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test
TYP(1) – See DC Electrical Characteristics

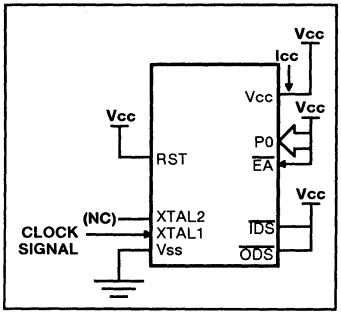


Figure 14. I_{CC} Test Condition, Active Mode
All other pins are disconnected

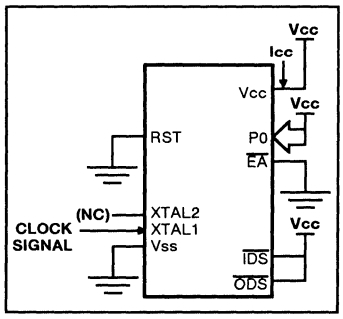


Figure 15. I_{CC} Test Condition, Idle Mode
All other pins are disconnected

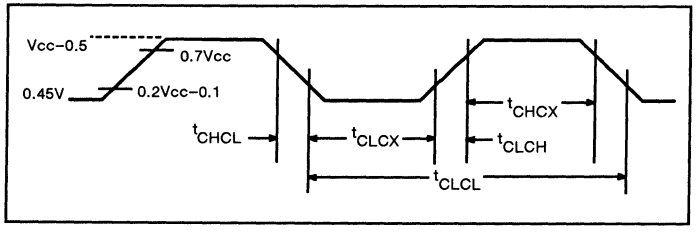


Figure 16. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

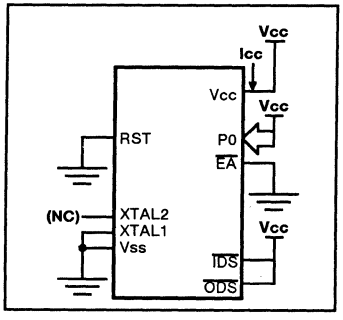


Figure 17. I_{CC} Test Conditions, Power Down Mode
All other pins are disconnected. V_{CC} = 2V to 5.5V

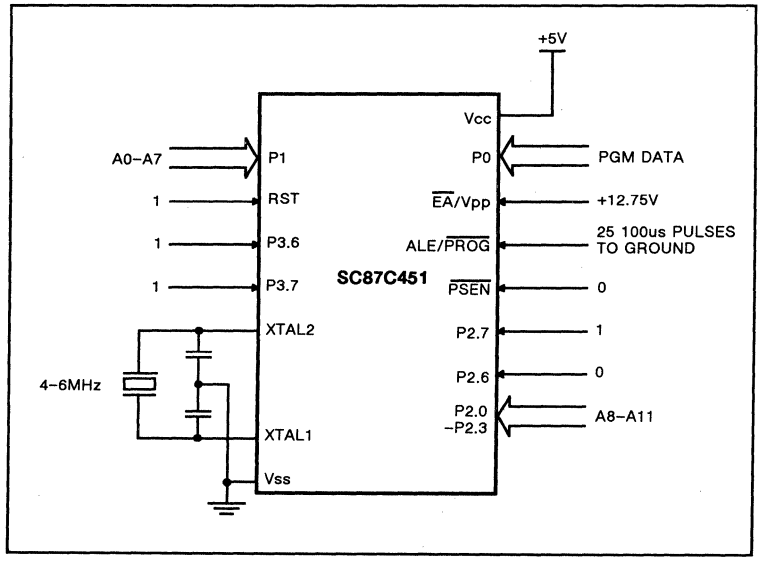


Figure 18. Programming Configuration

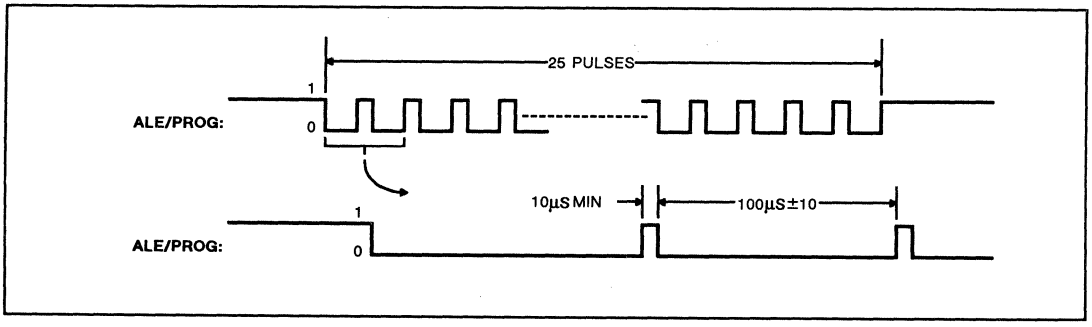


Figure 19. $\overline{\text{PROG}}$ Waveform

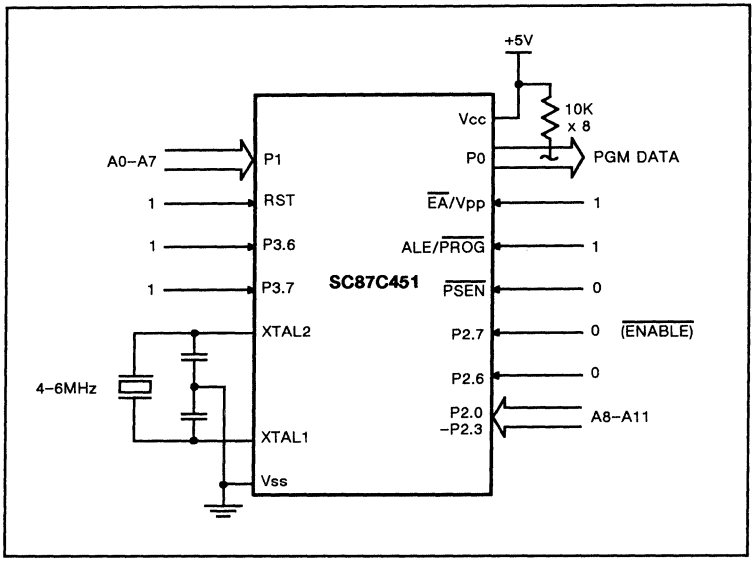


Figure 20. Programming Verification

EPROM PROGRAMMING AND VERIFICATION CHARACTERISTICS T_A = 21°C to +27°C, V_{CC} = 5V±10%, V_{SS} = 0V (see Figure 21)

SYMBOL	PARAMETER	MIN	MAX	UNIT
V_{PP}	Programming supply voltage	12.5	13.0	V
I_{PP}	Programming supply current		50	mA
$1/t_{CLCL}$	Oscillator frequency	4	6	MHz
t_{AVGL}	Address setup to $\overline{PROG}$ low	$48t_{CLCL}$		
t_{GHAX}	Address hold after $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	Data setup to $\overline{PROG}$ low	$48t_{CLCL}$		
t_{GHDX}	Data hold after $\overline{PROG}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{ENABLE}$) high to Vpp	$48t_{CLCL}$		
t_{SHGL}	V_{PP} setup to $\overline{PROG}$ low	10		µs
t_{GHSL}	V_{PP} hold after $\overline{PROG}$	10		µs
t_{GLGH}	$\overline{PROG}$ width	90	110	µs
t_{AVQV}	Address to data valid		$48t_{CLCL}$	
t_{ELQV}	$\overline{ENABLE}$ low to data valid		$48t_{CLCL}$	
t_{EHQZ}	Data float after $\overline{ENABLE}$	0	$48t_{CLCL}$	
t_{GHGL}	$\overline{PROG}$ high to $\overline{PROG}$ low	10		µs

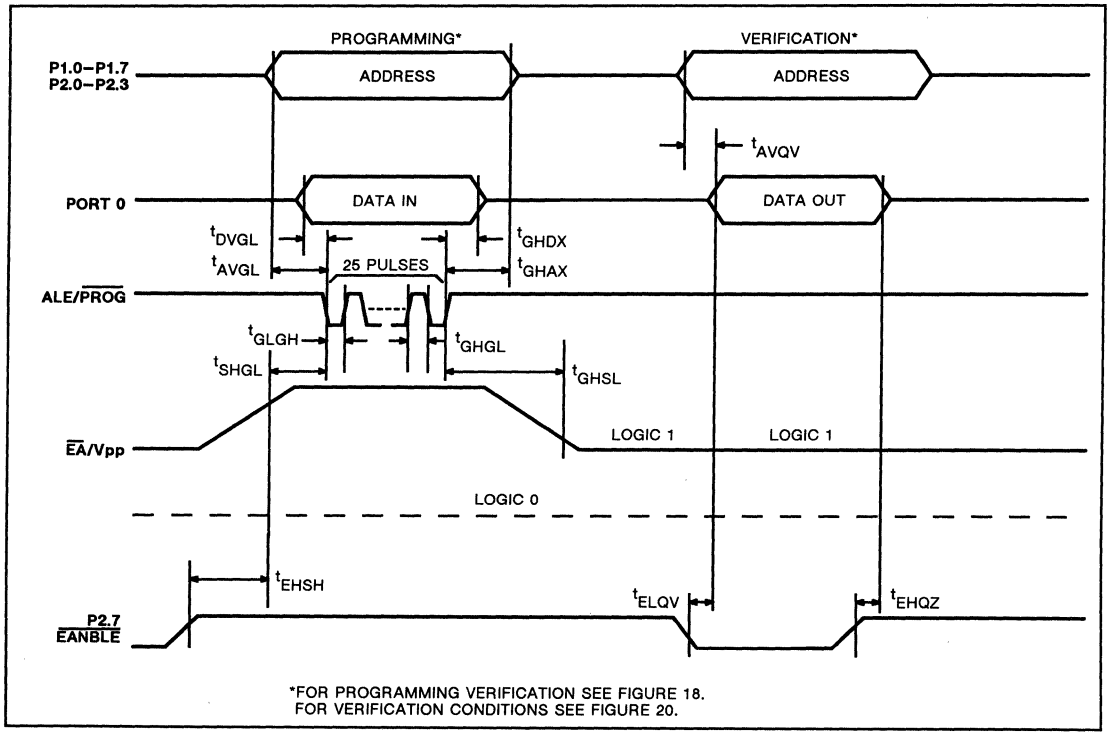


Figure 21. EPROM Programming and Verification

8XC552 OVERVIEW

The 83C552 is a stand-alone high-performance microcontroller designed for use in real-time applications such as instrumentation, industrial control, and automotive control applications such as engine management and transmission control. The device provides, in addition to the 80C51 standard functions, a number of dedicated hardware functions for these applications.

The 83C552 single-chip 8-bit microcontroller is manufactured in an advanced CMOS process and is a derivative of the 80C51 microcontroller family. The 83C552 uses the powerful instruction set of the 80C51. Additional special function registers are incorporated to control the on-chip peripherals. Three versions of the derivative exist although the generic term "83C552" is used to refer to family members:

83C552: 8K bytes mask-programmable ROM, 256 bytes RAM

87C552: 8K bytes EPROM, 256 bytes RAM

80C552: ROMless version of the 83C552

The 83C552 contains a non-volatile 8K x 8 read-only program memory, a volatile 256 x 8 read/write data memory, six 8-bit I/O ports, two 16-bit timer/event counters (identical to the timers of the 80C51), an additional 16-bit timer coupled to capture and compare latches, a fifteen-source, two-priority-level, nested interrupt structure, an 8-input ADC, a dual DAC pulse width modulated interface, two serial interfaces (UART and I2C bus), a 'watchdog' timer and on-chip oscillator and timing circuits. For systems that require extra capability, the 83C552 can be expanded using standard TTL compatible memories and logic.

The 83C552 has two software selectable modes of reduced activity for further power reduction – Idle and Power-down. The Idle mode freezes the CPU and resets Timer T2 and the ADC and PWM circuitry but allows the other timers, RAM, serial ports and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator causing all other chip functions to become inoperative.

DIFFERENCES FROM THE 8051

PROGRAM MEMORY

The 83C552 contains 8K-bytes of on-chip program memory which can be extended to 64K-bytes with external memories (see Figure 13). When the $\overline{EA}$ pin is held high, the 83C552 fetches instructions from internal ROM unless the address exceeds 1FFFH. Locations 2000H to FFFFH are fetched from external program memory. When the EA pin is held low, all instructions fetches are from external memory. ROM locations 0003H to 0073H are used by interrupt service routines.

DATA MEMORY

The internal data memory is divided into 3 sections: the lower 128 bytes of RAM, the upper 128 bytes of RAM and the 128-byte special function register areas. The lower 128 bytes of RAM are directly and indirectly addressable. While RAM locations 128 to 255 and the special function register area share the same address space, they are accessed through different addressing modes. RAM locations 128 to 255 are only indirectly addressable and the special function registers are only directly addressable. All other aspects of the internal RAM are identical to the 8051.

The stack may be located anywhere in the internal RAM by loading the eight bit stack pointer. Stack depth is 256 bytes maximum.

SPECIAL FUNCTION REGISTERS

The special function registers (directly addressable only) contain all of the 83C552 registers except the program counter and the four register banks. Most of the 56 special function registers are used to control the on-chip peripheral hardware. Other registers include arithmetic registers (ACC, B, PSW), stack pointer (SP) and data pointer registers (DHP, DPL). Sixteen of the SFR's contain 128 directly addressable bit locations. Table 8 lists the 83C552's special function registers.

The standard 80C51 SFR's are present and function identically in the 83C552 except where noted in the following sections.

TIMER T2

Timer T2 is a 16-bit timer consisting of two registers TMH2 (HIGH byte) and TML2 (LOW byte). The 16-bit timer/counter can be switched off or clocked via a prescaler from one of two sources: fosc/12 or an external signal. When Timer T2 is configured as a counter, the prescaler is clocked by an external signal on T2 (P1.4). A rising edge on T2 increments the prescaler and the maximum repetition rate is one count per machine cycle (1MHz with a 12MHz oscillator).

The maximum repetition rate for Timer T2 is twice the maximum repetition rate for Timer 0 and Timer 1. T2 (P1.4) is sampled at S2P1 and again at S5P1 (i.e., twice per machine cycle). A rising edge is detected when T2 is LOW during one sample and HIGH during the next sample. To ensure that a rising edge is detected, the input signal must be LOW for at least 1/2 cycle and then HIGH for at least 1/2 cycle. If a rising edge is detected before the end of S2P1, the timer will be incremented during the following cycle; otherwise it will be incremented one cycle later. The prescaler has a programmable division factor of 1, 2, 4 or 8 and is cleared if its division factor or input source is changed, or if the timer/counter is reset.

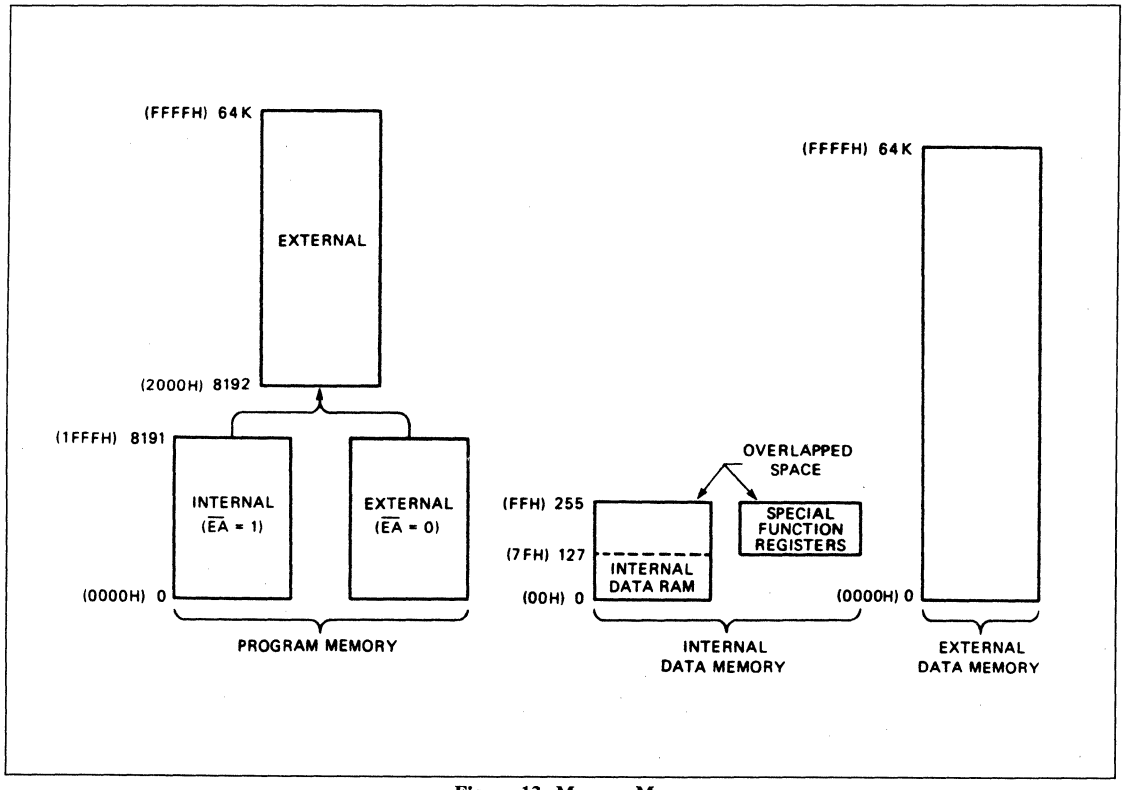


Figure 13. Memory Map

Timer T2 may be read "on the fly", but possesses no extra read latches and software precautions may have to be taken to avoid misinterpretation in the event of an overflow from least to most significant byte while Timer T2 is being read. Timer T2 is not loadable and is reset by the RST signal or by a rising edge on the input signal RT2, if enabled. RT2 is enabled by setting bit T2ER (TM2CON.5).

When the least significant byte of the timer overflows or when a 16-bit overflow occurs, an interrupt request may be generated. Either or both of these overflows can be programmed to request an interrupt. In both cases, the interrupt vector will be the same. When the lower byte (TML2) overflows, flag T2B0 (TM2CON) is set and flag T20V (TM2IR) is set when TMH2 overflows. These flags are set one cycle after an overflow occurs. Note that when T20V is set, T2B0 will also be set. To enable the byte overflow interrupt, bits ET2 (IEN1.7, enable overflow interrupt, see Figure 14) and T2IS0 (TM2CON.6, byte overflow interrupt select) must be set. Bit T2B0 (TM2CON.4) is the Timer T2 byte overflow flag.

	7	6	5	4	3	2	1	0
IEN1 (E8H)	ET2	ECM2	ECM1	ECM0	ECT3	ECT2	ECT1	ECT0
	MSB							LSB

BIT	Symbol	Function
IEN1.7	ET2	Enable Timer T2 overflow interrupt(s)
IEN1.6	ECM2	Enable T2 Comparator 2 interrupt
IEN1.5	ECM1	Enable T2 Comparator 1 interrupt
IEN1.4	ECM0	Enable T2 Comparator 0 interrupt
IEN1.3	ECT3	Enable T2 Capture register 3 interrupt
IEN1.2	ECT2	Enable T2 Capture register 2 interrupt
IEN1.1	ECT1	Enable T2 Capture register 1 interrupt
IEN1.0	ECT0	Enable T2 Capture register 0 interrupt

Figure 14. Timer T2 Interrupt Enable Register (IEN1)

Table 8. 8XC552 Special Function Registers

Symbol	Description	Direct Address	Bit Address, Symbol or Alternative Port Function MSB LSB								Reset Value
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
ADCH#	A/D converter high	C6H									xxxxxxxxB
ADCON#	Adc control	C5H	ADC.1	ADC.0	ADEX	ADCI	ADCS	AADR2	AADR1	AADR0	xx000000B
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
CTCON#	Capture control	EBH	CTN3	CTP3	CTN2	CTP2	CTN1	CTP	CTN0	CTP0	00H
CTH3#	Capture high 3	CFH									xxxxxxxxB
CTH2#	Capture high 2	CEH									xxxxxxxxB
CTH1#	Capture high 1	CDH									xxxxxxxxB
CTH0#	Capture high 0	CCH									xxxxxxxxB
CMH2#	Compare high 2	CBH									00H
CMH1#	Compare high 1	CAH									00H
CMH0#	Compare high 0	C9H									00H
CTL3#	Capture low 3	AFH									xxxxxxxxB
CTL2#	Capture low 2	AEH									xxxxxxxxB
CTL1#	Capture low 1	ADH									xxxxxxxxB
CTL0#	Capture low 0	ACH									xxxxxxxxB
CML2#	Compare low 2	ABH									00H
CML1#	Compare low 1	AAH									00H
CML0#	Compare low 0	A9H									00H
DPTR:	Data pointer (2 bytes)										
DPH	Data pointer high	83H									00H
DPL	Data pointer low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IEN0*	Interrupt enable 0	A8H	EA	EAD	ES1	ES0	ET1	EX1	ET0	ET1	00H
			EF	EE	ED	EC	EB	EA	E9	E8	
IEN1*#	Interrupt enable 1	E8H	ET2	ECM2	ECM1	ECM0	ECT3	ECT2	ECT1	ECT0	00H
			BF	BE	BD	BC	BB	BA	B9	B8	
IP0*	Interrupt priority 0	B8H	–	PAD	PS1	PS0	PT1	PX1	PT0	PX0	x0000000B
			FF	FE	FD	FC	FB	FA	F9	F8	
IP1*#	Interrupt priority 1	F8H	PT2	PCM2	PCM1	PCM0	PCT3	PCT2	PCT1	PCT0	00H
P5#	Port 5	C4H	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	xxxxxxxxB
			C7	C6	C5	C4	C3	C2	C1	C0	
P4*#	Port 4	C0H	CMT1	CMT0	CMSR5	CMSR4	CMSR3	CMSR2	CMSR1	CMSR0	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T1	T0	$\overline{INT1}$	$\overline{INT0}$	TXD	RXD	FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	A15	A14	A13	A12	A11	A10	A9	A8	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	SDA	SCL	RT2	T2	CT3I	CT2I	CT1I	CT0I	FFH
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
PCON	Power control	87H	SMOD	–	–	WLE	GF1	GF0	PD	IDL	00xx00000B
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	00H
PWMP#	PWM prescaler	FEH									00H
PWM1#	PWM register 1	FDH									00H
PWM0#	PWM register 0	FCH									00H
RTE#	Reset/toggle enable	EFH	TP47	TP46	RP45	RP44	RP43	RP42	RP41	RP40	00H
SP	Stack pointer	81H									07H
S0BUF	Serial 0 data buffer	99H									xxxxxxxxB

* = SFRs are bit addressable.
# = SFRs are modified from or added to the 80C51 SFRs.

Table 8. 8XC552 Special Function Registers (Continued)

Symbol	Description	Direct Address	Bit Address, Symbol or Alternative Port Function MSB LSB								Reset Value
			9F	9E	9D	9C	9B	9A	99	98	
S0CON*	Serial 0 control	98H	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	00H
S1ADR#	Serial 1 address	DBH	SLAVE ADDRESS							GC	00H
SIDAT#	Serial 1 data	DAH									00H
S1STA#	Serial 1 status	D9H	SC4	SC3	SC2	SC1	SC0	0	0	0	F8H
			DF	DE	DD	DC	DB	DA	D9	D8	
SICON#*	Serial 1 control	D8H	–	ENS1	STA	STO	SI	AA	CR1	CR0	x0000000B
STE#	Set enable	EEH	TG47	TG46	SP45	SP44	SP43	SP42	SP41	SP40	C0H
TH1	Timer high 1	8DH									00H
TH0	Timer high 0	8CH									00H
TL1	Timer low 1	8BH									00H
TL0	Timer low 0	8AH									00H
TMH2#	Timer high 2	EDH									00H
TML2#	Timer low 2	ECH									00H
TMOD	Timer mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00H
TM2CON#	Timer 2 control	EAH	T2IS1	TSIS0	T2ER	T2B0	T2P1	T2P0	T2MS1	T2MS0	00H
			CF	CE	CD	CC	CB	CA	C9	C8	
TM2IR#	Timer 2 int flag reg	C8H	T20V	CMI2	CMI1	CMI0	CTI3	CTI2	CTI1	CTI0	00H
T3#	Timer 3	FFH									00H

* = SFRs are bit addressable.
\# = SFRs are modified from or added to the 80C51 SFRs.

To enable the 16-bit overflow interrupt, bits ET2 (IE1.7, enable overflow interrupt) and T2IS1 (TM2CON.7, 16-bit overflow interrupt select) must be set. Bit T2OV (TM2IR.7) is the Timer T2 16-bit overflow flag. All interrupt flags must be reset by software. To enable both byte and 16-bit overflow, T2ISO and T2IS1 must be set and two interrupt service routines are required. A test on the overflow flags indicates which routine must be executed. For each routine, only the corresponding overflow flag must be cleared.

Timer T2 may be reset by a rising edge on RT2 (P1.5) if the Timer T2 external reset enable bit (T2ER) in T2CON is set. This reset also clears the prescaler. In the Idle mode, the timer/counter and prescaler are reset and halted. Timer T2 is controlled by the TM2CON special function register (see Figure 15).

Timer T2 Extension

When a 12MHz oscillator is used, a 16-bit overflow on Timer T2 occurs every 65.5, 131, 262 or 524ms depending on the prescaler division ratio i.e. the maximum cycle time is approximately 0.5 seconds. In applications where cycle times are greater than 0.5 seconds, it is necessary to extend Timer T2. This is achieved by selecting fosc/12 as the clock source (set T2MS0, reset T2MS1), setting the prescaler division ratio to 1/8 (set T2P0, set T2P1), disabling the byte overflow interrupt (reset T2IS0) and enabling the 16-bit overflow interrupt (set T2IS1). The following software routine is written for a three-byte extension which gives a maximum cycle time of approximately 2400 hours.

```
OVINT: PUSH   ACC      ;save accumulator
       PUSH   PSW      ;save status
       INC    TIMEX1   ;increment first
                       ;byte (low order)
                       ;of extended timer
       MOV    A,TIMEX1
       JNZ    INTEX    ;jump to INTEX if
                       ;there is no
                       ;overflow
       INC    TIMEX2   ;increment second
                       ;byte
       MOV    A,TIMEX2
       JNZ    INTEX    ;jump to INTEX if
                       ;there is no
                       ;overflow
       INC    TIMEX3   ;increment third
                       ;byte (high order)
INTEX: CLR    T2OV     ;reset interrupt
                       ;flag
       POP    PSW      ;restore status
       POP    ACC      ;restore accumulator
       RETI            ;return from
                       ;interrupt
```

Timer T2, Capture and Compare Logic

Timer T2 is connected to four 16-bit capture registers and three 16-bit compare registers. A capture register may be used to capture the contents of Timer T2 when a transition occurs on its corresponding input pin. A compare register may be used to set, reset or toggle Port 4 output pins at certain pre-programmable time intervals.

	7	6	5	4	3	2	1	0
TM2CON (EAH)	T2IS1	T2IS0	T2ER	T2BO	T2P1	T2P0	T2MS1	T2MS0
	MSB							LSB

Bit	Symbol	Function
TM2CON.7	T2IS1	Timer T2 16-bit overflow interrupt select
TM2CON.6	T2IS0	Timer T2 byte overflow interrupt select
TM2CON.5	T2ER	Timer T2 external reset enable. When this bit is set, Timer T2 may be reset by a rising edge on RT2 (P1.5).
TM2CON.4	T2BO	Timer T2 byte overflow interrupt flag
TM2CON.3	T2P1	Timer T2 prescaler select
TM2CON.2	T2P0	

T2P1	T2P0	Timer T2 clock
0	0	clock source
0	1	clock source/2
1	0	clock source/4
1	1	clock source/8

Bit	Symbol	Function
TM2CON.1	T2MS1	Timer T2 mode select
TM2CON.0	T2MS0	

T2MS1	T2MS0	Mode selected
0	0	Timer T2 halted (off)
0	1	T2 clock source = $f_{osc}/12$
1	0	Test mode; do not use
1	1	T2 clock source = pin T2

Figure 15. T2 Control Register (TM2CON)

The combination of Timer T2 and the capture and compare logic is very powerful in applications involving rotating machinery, automotive injection systems, etc. Timer T2 and the capture and compare logic are shown in Figure 16.

Capture Logic

The four 16-bit capture registers that Timer T2 is connected to are: CT0, CT1, CT2 and CT3. These registers are loaded with the contents of Timer T2 and an interrupt is requested upon receipt of the input signals CT0I, CT1I, CT2I or CT3I. These input signals are shared with Port 1. The four interrupt flags are in the Timer T2 interrupt register (TM2IR special function register). If the capture facility is not required, these inputs can be regarded as additional external interrupt inputs.

Using the capture control register CTCON (see Figure 17), these inputs may capture on a rising edge, a falling edge or on either a rising or falling edge. The inputs are sampled during S1P1 of each cycle. When a selected edge is detected, the contents of Timer T2 are captured at the end of the cycle.

Measuring Time Intervals using Capture Registers

When a recurring external event is represented in the form of rising or falling edges on one of the four capture pins, the time between two events can be measured using Timer T2 and a capture register. When an event occurs, the contents of Timer T2 are copied into the relevant capture register and an interrupt request is generated. The interrupt service routine may then compute the interval time if it knows the previous contents of Timer T2 when the last event occurred. With a 12MHz oscillator, Timer T2 can be programmed to overflow every 524ms. When event interval times are shorter than this, computing the interval time is simple and the interrupt service routine is short. For longer interval times, the Timer T2 extension routine may be used.

Compare Logic

Each time Timer T2 is incremented, the contents of the three 16-bit compare registers CM0, CM1 and CM2 are compared with the new counter value of Timer T2. When a match is found, the corresponding interrupt flag in TM2IR is set at the end of the following cycle. When a match with CM0 occurs, the controller sets bits 0–5 of Port 4 if the corresponding bits of the set enable register STE are at logic 1. When a match with CM1 occurs, the controller resets bits 0–5 of Port 4 if the corresponding bits of the reset/toggle enable register RTE are at logic 1 (see Figure 18 for RTE register function). If RTE is '0', then P4.n is not affect by a match between CM1 or CM2 and Timer 2. When a match with CM2 occurs, the controller 'toggles' bits 6 and 7 of Port 4 if the corresponding bits of the RTE are at logic 1. The port latches of bits 6 and 7 are not toggled. Two additional flip-flops store the last operation and it is these flip-flops that are toggled. Thus, if the current operation is 'set', the next operation will be 'reset' even if the port latch is reset by software before the 'reset' operation occurs. The first 'toggle' after a chip RESET will set the port latch. The contents of these two flip flops can be read at STE.6 and STE.7 (corresponding to P4.6 and P4.7 respectively). Bits STE.6 and STE.7 are read only (see Figure 19 for STE register function). A logic 1 indicates that the next toggle will set the port latch; a logic 0 indicates that the next toggle will reset the port latch. CM0, CM1 and CM2 are reset by the RST signal.

The modified port latch information appears at the port pin during S5P1 of the cycle following the cycle in which a match occurred. If the port is modified by software, the outputs change during S1P1 of the following cycle. Each port 4 bit can be set or reset by software at any time. A hardware modification resulting from a comparator match takes precedence over a software modification in the same cycle. When the comparator results require a 'set' and a 'reset' at the same time, the port latch will be reset.

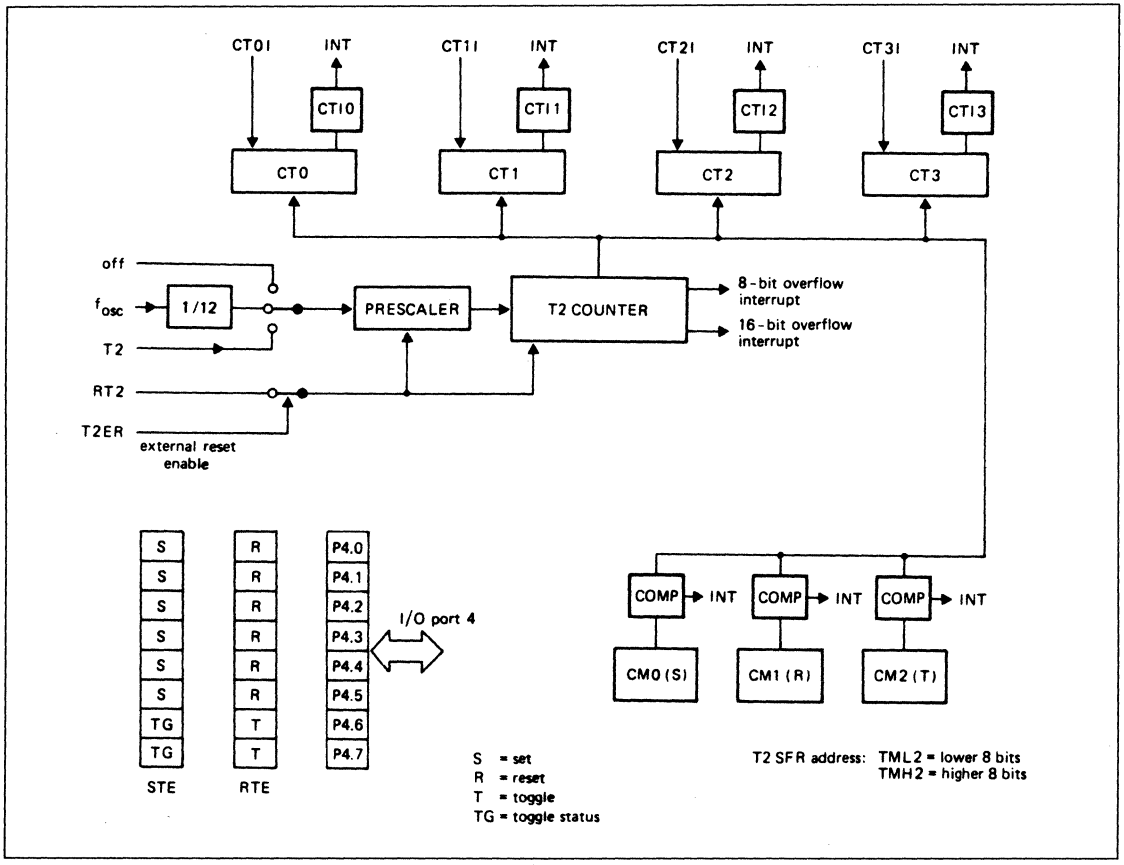


Figure 16. Block Diagram of Timer 2

CTCON (EBH)	7	6	5	4	3	2	1	0
	CTN3	CTP3	CTN2	CTP2	CTN1	CTP1	CTN0	CTP0
	MSB							LSB

Bit	Symbol	Capture/Interrupt on:
CTCON.7	CTN3	Capture Register 3 triggered by a falling edge on CT3I
CTCON.6	CTP3	Capture Register 3 triggered by a rising edge on CT3I
CTCON.5	CTN2	Capture Register 2 triggered by a falling edge on CT2I
CTCON.4	CTP2	Capture Register 2 triggered by a rising edge on CT2I
CTCON.3	CTN1	Capture Register 1 triggered by a falling edge on CT1I
CTCON.2	CTP1	Capture Register 1 triggered by a rising edge on CT1I
CTCON.1	CTN0	Capture Register 0 triggered by a falling edge on CT0I
CTCON.0	CTP0	Capture Register 0 triggered by a rising edge on CT0I

Figure 17. Capture Control Register (CTCON)

RTE (EFH)	7	6	5	4	3	2	1	0
	TP47	TP46	RP45	RP44	RP43	RP42	RP41	RP40
	MSB							LSB

BIT	Symbol	Function
RTE.7	TP47	if '1' then P4.7 toggles on a match between CM2 and Timer T2
RTE.6	TP46	if '1' then P4.6 toggles on a match between CM2 and Timer T2
RTE.5	RP45	if '1' then P4.5 is reset on a match between CM1 and Timer T2
RTE.4	RP44	if '1' then P4.4 is reset on a match between CM1 and Timer T2
RTE.3	RP43	if '1' then P4.3 is reset on a match between CM1 and Timer T2
RTE.2	RP42	if '1' then P4.2 is reset on a match between CM1 and Timer T2
RTE.1	RP41	if '1' then P4.1 is reset on a match between CM1 and Timer T2
RTE.0	RP40	if '1' then P4.0 is reset on a match between CM1 and Timer T2

Figure 18. Reset/Toggle Enable Register (RTE)

	7	6	5	4	3	2	1	0
STE (EEH)	TG47	TG46	SP45	SP44	SP43	SP42	SP41	SP40
	MSB							LSB

Bit	Symbol	Function
STE.7	TG47	Toggle flip-flops
STE.6	TG46	Toggle flip-flops
STE.5	SP45	If '1' then P4.5 is set on a match between CM0 and timer T2
STE.4	SP44	If '1' then P4.4 is set on a match between CM0 and timer T2
STE.3	SP43	If '1' then P4.3 is set on a match between CM0 and timer T2
STE.2	SP42	If '1' then P4.2 is set on a match between CM0 and timer T2
STE.1	SP41	If '1' then P4.1 is set on a match between CM0 and timer T2
STE.0	SP40	If '1' then P4.0 is set on a match between CM0 and timer T2

Figure 19. Set Enable Register (STE)

Timer T2 Interrupt Flag Register TM2IR

Eight of the nine Timer T2 interrupt flags are located in special function register TM2IR (see Figure 20). The ninth flag is TM2CON.4.

The CT0I and CT1I flags are set during S4 of the cycle in which the contents of Timer T2 are captured. CT0I is scanned by the interrupt logic during S2 and CT1I is scanned during S3. CT2I and CT3I are set during S6 and are scanned during S4 and S5. The associated interrupt requests are recognized during the following cycle. If these flags are polled, a transition at CT0I or CT1I will be recognized one cycle before a transition on CT2I or CT3I since registers are read during S5. The CMI0, CMI1 and CMI2 flags are set during S6 of the cycle following a match. CMI0 is scanned by the interrupt logic during S2; CMI1 and CMI2 are scanned during S3 and S4. A match will be recognized by the interrupt logic (or by polling the flags) two cycles after the match takes place.

The 16-bit overflow flag (T2OV) and the byte overflow flag (T2BO) are set during S6 of the cycle in which the overflow occurs. These flags are recognized by the interrupt logic during the next cycle.

Special function register IP1 (Figure 20) is used to determine the Timer T2 interrupt priority. Setting a bit high gives that function a high priority, and setting a bit low gives the function a low priority. The functions controlled by the various bits of the IP1 register are shown in Figure 20.

TIMER T3, THE WATCHDOG TIMER

In addition to Timer T2 and the standard timers, a watchdog timer is also incorporated on the 83C552. The purpose of a watchdog timer is to reset the microcontroller if it enters erroneous processor states (possibly caused by electrical noise or RFI) within a reasonable period of time. An analogy is the "dead man's handle" in railway locomotives. When enabled, the watchdog circuitry will generate a system reset if the user program fails to reload the watchdog timer within a specified length of time known as the 'watchdog interval'.

	7	6	5	4	3	2	1	0
TM2IR (C8H)	T2OV	CMI2	CMI1	CMI0	CTI3	CTI2	CTI1	CTI0
	MSB							LSB

Bit	Symbol	Function
TM2IR.7	T2OV	Timer T2 16-bit overflow interrupt flag
TM2IR.6	CMI2	CM2 interrupt flag
TM2IR.5	CMI1	CM1 interrupt flag
TM2IR.4	CMI0	CM0 interrupt flag
TM2IR.3	CTI3	CT3 interrupt flag
TM2IR.2	CTI2	CT2 interrupt flag
TM2IR.1	CTI1	CT1 interrupt flag
TM2IR.0	CTI0	CT0 interrupt flag

Interrupt Flag Register (TM2IR)

	7	6	5	4	3	2	1	0
IP1 (F8H)	PT2	PCM2	PCM1	PCM0	PCT3	PCT2	PCT1	PCT0
	MSB							LSB

Bit	Symbol	Function
IP1.7	PT2	Timer T2 overlfow interrupt(s) priority level
IP1.6	PCM2	Timer T2 comparator 2 interrupt priority level
IP1.5	PCM1	Timer T2 comparator 1 interrupt priority level
IP1.4	PCM0	Timer T2 comparator 0 interrupt priority level
IP1.3	PCT3	Timer T2 capture register 3 interrupt priority level
IP1.2	PCT2	Timer T2 capture register 2 interrupt priority level
IP1.1	PCT1	Timer T2 capture register 1 interrupt priority level
IP1.0	PCT0	Timer T2 capture register 0 interrupt priority level

Timer 2 Interrupt Priority Register (IP1)

Figure 20. Interrupt Flag Register (TM2IR) and Timer T2 Interrupt Priority Register (IP1)

Watchdog Circuit Description

The watchdog timer (Timer T3) consists of an 8-bit timer with an 11-bit prescaler as shown in Figure 21. The prescaler is fed with a signal whose frequency is 1/12 the oscillator frequency (1MHz with a 12 MHz oscillator). The 8-bit timer is incremented every 't' seconds where:

$t = 12 \times 2048 \times 1/fosc$ (= 2ms at fosc = 12MHz)

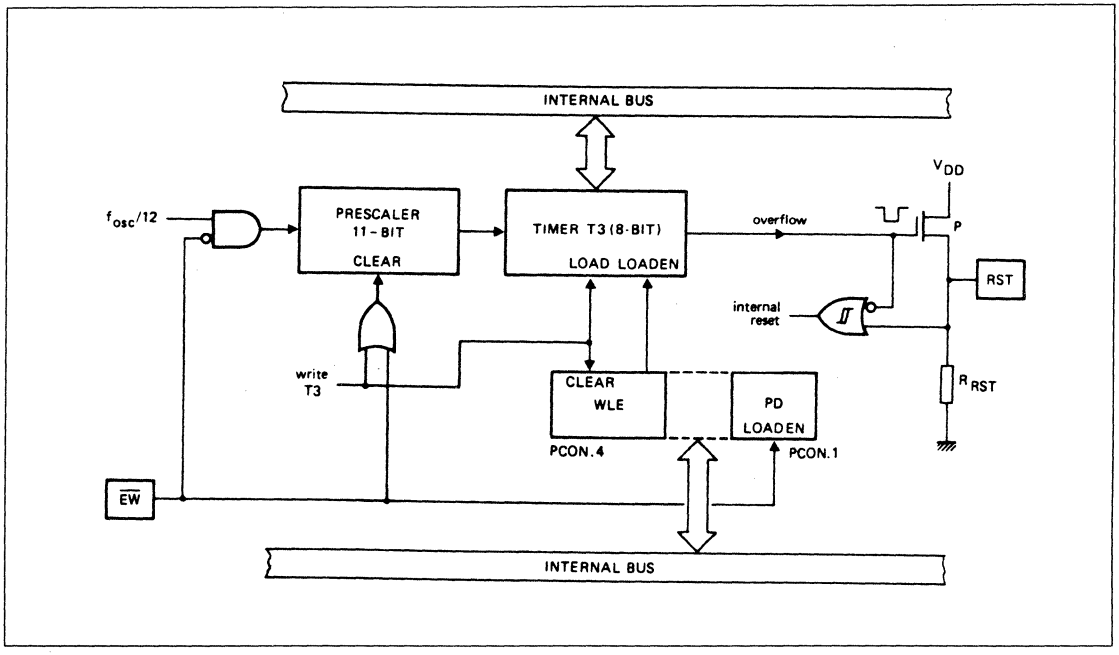


Figure 21. Watchdog Timer

If the 8-bit timer overflows, a short internal reset pulse is generated which will reset the 83C552. A short output reset pulse is also generated at the RST pin. This short output pulse (3 machine cycles) may be destroyed if the RST pin is connected to a capacitor. This would not, however, affect the internal reset operation.

Watchdog operation is activated when external pin $\overline{EW}$ is tied low. When $\overline{EW}$ is tied low, it is impossible to disable watchdog operation by software.

How to Operate the Watchdog Timer

The watchdog timer has to be reloaded within periods that are shorter than the programmed watchdog interval; otherwise the watchdog timer will overflow and a system reset will be generated. The user program must therefore continually execute sections of code which reload the watchdog timer. The period of time elapsed between execution of these sections of code must never exceed the watchdog interval. When using a 12MHz oscillator, the watchdog interval is programmable between 2ms and 510ms.

In order to prepare software for watchdog operation, a programmer should first determine how long his system can sustain an erroneous processor state. The result will be the maximum watchdog interval. As the maximum watchdog interval becomes shorter, it becomes more difficult for the programmer to ensure that the user program always reloads the watchdog timer within the watchdog interval and thus it becomes more difficult to implement watchdog operation.

The programmer must now partition the software in such a way that reloading of the watchdog is carried out in accordance with the above requirements. The programmer must determine the execution times of all software modules. The effect of possible conditional branches, subroutines, external and internal interrupts must all be taken into account. Since it may be very difficult to evaluate the execution times of some sections of code, the programmer should use worst case estimations. In any event, the programmer must make sure that the watchdog is not activated during normal operation.

The watchdog timer is reloaded in two stages in order to prevent erroneous software from reloading the watchdog. First, PCON.4 (WLE) must be set. Then T3 may be loaded. When T3 is loaded, PCON.4 (WLE) is automatically reset. T3 can not be loaded if PCON.4 (WLE) is reset. Reload code may be put in a subroutine as it is called frequently. Since Timer T3 is an up-counter, a reload value of 00H gives the maximum watchdog interval (510ms with a 12MHz oscillator) and a reload value of 0FFH gives the minimum watchdog interval (2ms with a 12MHz oscillator).

In the Idle mode, the watchdog circuitry remains active. When watchdog operation is implemented, the Power-down mode cannot be used since both states are contradictory. Thus, when watchdog operation is enabled by ty-

ing external pin $\overline{EW}$ low, it is impossible to enter the Power-down mode and an attempt to set the Power-down bit (PCON.1) will have no effect. PCON.1 will remain at logic 0.

During the early stages of software development/debugging, the watchdog may be disabled by tying the $\overline{EW}$ pin high. At a later stage, $\overline{EW}$ may be tied low to complete the debugging process.

Watchdog Software Example

The following example shows how watchdog operation might be handled in a user program.

```
;at the program start:

T3          EQU 0FFH ;address of
                     ;watchdog timer T3
PCON        EQU 087H ;address of PCON SFR
WATCH-INTV  EQU 156  ;watchdog interval
                     ;(e.g.2x100ms)
;to be inserted at each watchdog reload
;location within the user program:

            LCALL WATCHDOG

;watchdog service routine:

WATCHDOG: ORL  PCON,#10H    ;set condition
                            ;flag (PCON.4)
          MOV T3,WATCH-INV  ;load T3 with
                            ;watchdog
                            ;interval
          RET
```

If it is possible for this subroutine to be called in an erroneous state, then the condition flag WLE should be set at different parts of the main program.

SERIAL I/O

The 83C552 is equipped with two independent serial ports: SIO0 and SIO1. SIO0 is a full duplex UART port and is identical to the 80C51 serial port. SIO1 accommodates the 12C bus.

SIO0

SIO0 is a full duplex serial I/O port identical to that on the 80C51. Its operation is identical including the use of timer 1 as a baud rate generator.

SIO1, I²C Serial I/O

The I²C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the bus are:

- Bidirectional data transfer between masters and slaves
- Multimaster bus (no central master)
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
- The I²C bus may be used for test and diagnostic purposes

The output latches of P1.6 and P1.7 must be set to logic 1 in order to enable SIO1.

The 83C552 on-chip I²C logic provides a serial interface that meets the I²C bus specification and supports all transfer modes (other than the low speed mode) from and to the I²C bus. The SIO1 logic handles bytes transfer autonomously. It also keeps track of serial transfers, and a status register S1STA) reflects the status of SIO1 and the I²C bus.

The CPU interfaces to the I²C logic via the following four special function registers: S1CON (SIO1 control register), S1STA (SIO1 status register), S1DAT (SIO1 data register) and S1ADR (SIO1 slave address register). The SIO1 logic interfaces to the external I2C bus via two Port 1 pins: P1.6/SCL (serial clock line) and P1.7/ SDA (serial data line).

A typical I²C bus configuration is shown in Figure 22 and Figure 23 shows how a data transfer is accomplished on the bus. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I²C bus:

1. Data transfer from a master transmitter to a slave receiver. The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte.
2. Data transfer from a slave transmitter to a master receiver. The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. Next follows a number of data bytes transmitted by the slave to the master. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a 'not acknowledge' is returned.

The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the I²C bus will not be released.

Modes of Operation

The on-chip SIO1 logic may operate in the following four modes:

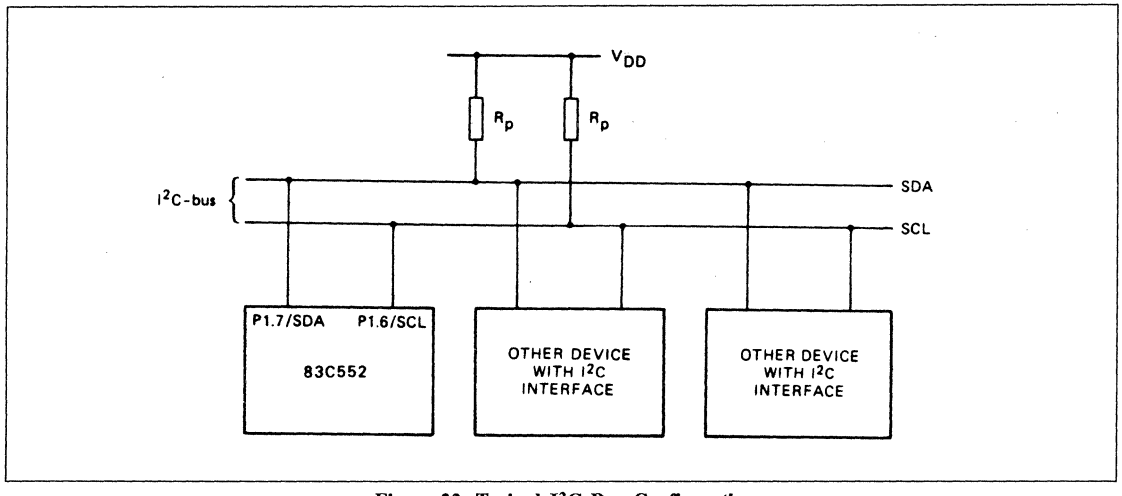


Figure 22. Typical I²C Bus Configuration

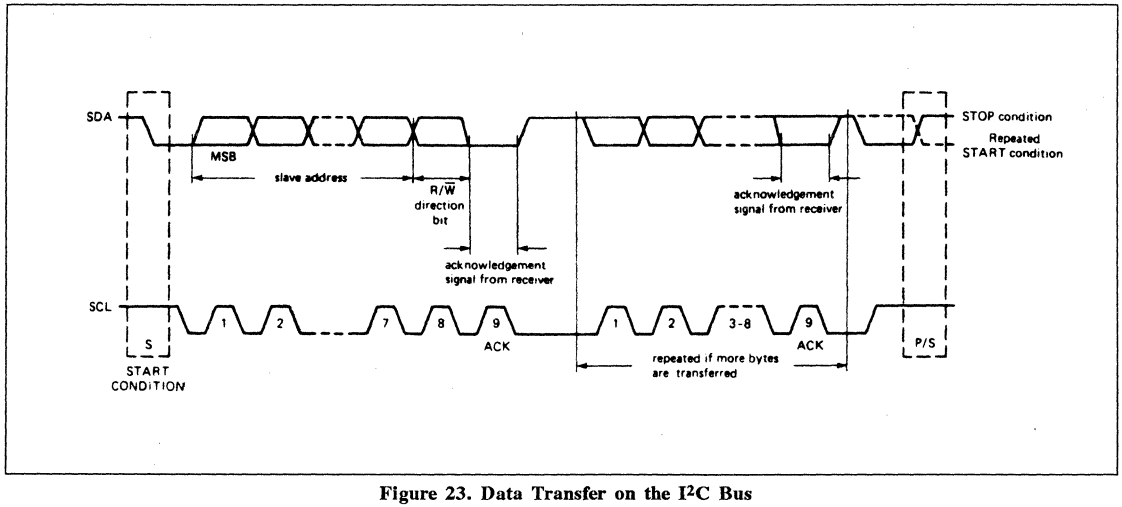


Figure 23. Data Transfer on the I²C Bus

1. Master Transmitter Mode:

Serial data output through P1.7/SDA while P1.6/SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the data direction bit. In this case the data direction bit ($R/\overline{W}$) will be logic 0 and we say that a 'W' is transmitted. Thus the first byte transmitted is SLA+W. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

2. Master Receiver Mode:

The first byte transmitted contains the slave address of the transmitting device (7 bits) and the data direction bit. In this case the data direction bit ($R/\overline{W}$) will be logic 1 and we say that an 'R' is transmitted. Thus the first byte transmitted is SLA+R. Serial data is received via P1.7/SDA while P1.6/SCL outputs the serial clock. Serial data is received eight bits at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are output to indicate the beginning and end of a serial transfer.

3. Slave Receiver Mode:

Serial data and the serial clock are received through P1.7/SDA and P1.6/SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

4. Slave Transmitter Mode:

The first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via P1.7/SDA while the serial clock is input through P1.6/SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer.

In a given application, SIO1 may operate as a master and as a slave. In the slave mode, the SIO1 hardware looks for its own slave address and the general call address. If one of these addresses is detected, an interrupt is requested. When the microcontroller wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave action is not interrupted. If bus arbitration is lost in the master mode, SIO1 switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

SIO1 Implementation and Operation

Figure 24 shows how the on-chip I2C bus interface is implemented and the following text describes the individual blocks.

Input Filters and Output Stages

The input filters have I2C compatible input levels. If the input voltage is less than 1.5V, the input logic level is interpreted as 0; if the input voltage is greater than 3.0 V, the input logic level is interpreted as 1. Input signals are synchronized with the internal clock (fosc/4) and spikes shorter than three oscillator periods are filtered out.

The output stages consist of open drain transistors that can sink 3mA at $V_{OUT} < 0.4V$. These open drain outputs do not have clamping diodes to V_{DD}. Thus, if the device is connected to the I2C bus and V_{DD} is switched off, the I2C bus is not affected.

Address Register, S1ADR

This 8-bit special function register may be loaded with the 7-bit slave address (7 most significant bits) to which SIO1 will respond when programmed as a slave transmitter or receiver. The LSB (GC) is used to enable general call address (00H) recognition.

Comparator

The comparator compares the received 7-bit slave address with its own slave address (7 most significant bits in S1ADR). It also compares the first received 8-bit byte with the general call address (00H). If an equality is found, the appropriate status bits are set and an interrupt is requested.

Shift Register, S1DAT

This 8-bit special function register contains a byte of serial data to be transmitted or a byte which has just been received. Data in S1DAT is always shifted from right to left; the first bit to be transmitted is the MSB (bit 7) and, after a byte has been received, the first bit of received data is located at the MSB of S1DAT. While data is being shifted out, data on the bus is simultaneously being shifted in; S1DAT always contains the last byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in S1DAT.

Arbitration and Synchronization Logic

In the master transmitter mode, the arbitration logic checks that every transmitted logic 1 actually appears as a logic 1 on the I2C bus. If another device on the bus overrules a logic 1 and pulls the SDA line low, arbitration is lost and SIO1 immediately changes from master transmitter to slave receiver. SIO1 will continue to output clock pulses (on SCL) until transmission of the current serial byte is complete.

Arbitration may also be lost in the master receiver mode. Loss of arbitration in this mode can only occur while SIO1 is returning a 'not acknowledge' (logic 1) to the bus. Arbitration is lost when another device on the bus pulls this signal LOW. Since this can occur only at the end of a serial byte, SIO1 generates no further clock pulses. Figure 25 shows the arbitration procedure.

The synchronization logic will synchronize the serial clock generator with the clock pulses on the SCL line from another device. If two or more master devices generate clock pulses, the 'mark' duration is determined by the device that generates the shortest 'marks' and the 'space' duration is determined by the device that generates the longest 'spaces'. Figure 26 shows the synchronization procedure.

A slave may stretch the space duration to slow down the bus master. The space duration may also be stretched for handshaking purposes. This can be done after each bit or after a complete byte transfer. SIO1 will stretch the SCL space duration after a byte has been transmitted or received and the acknowledge bit has been transferred. The serial interrupt flag (SI) is set and the stretching continues until the serial interrupt flag is cleared.

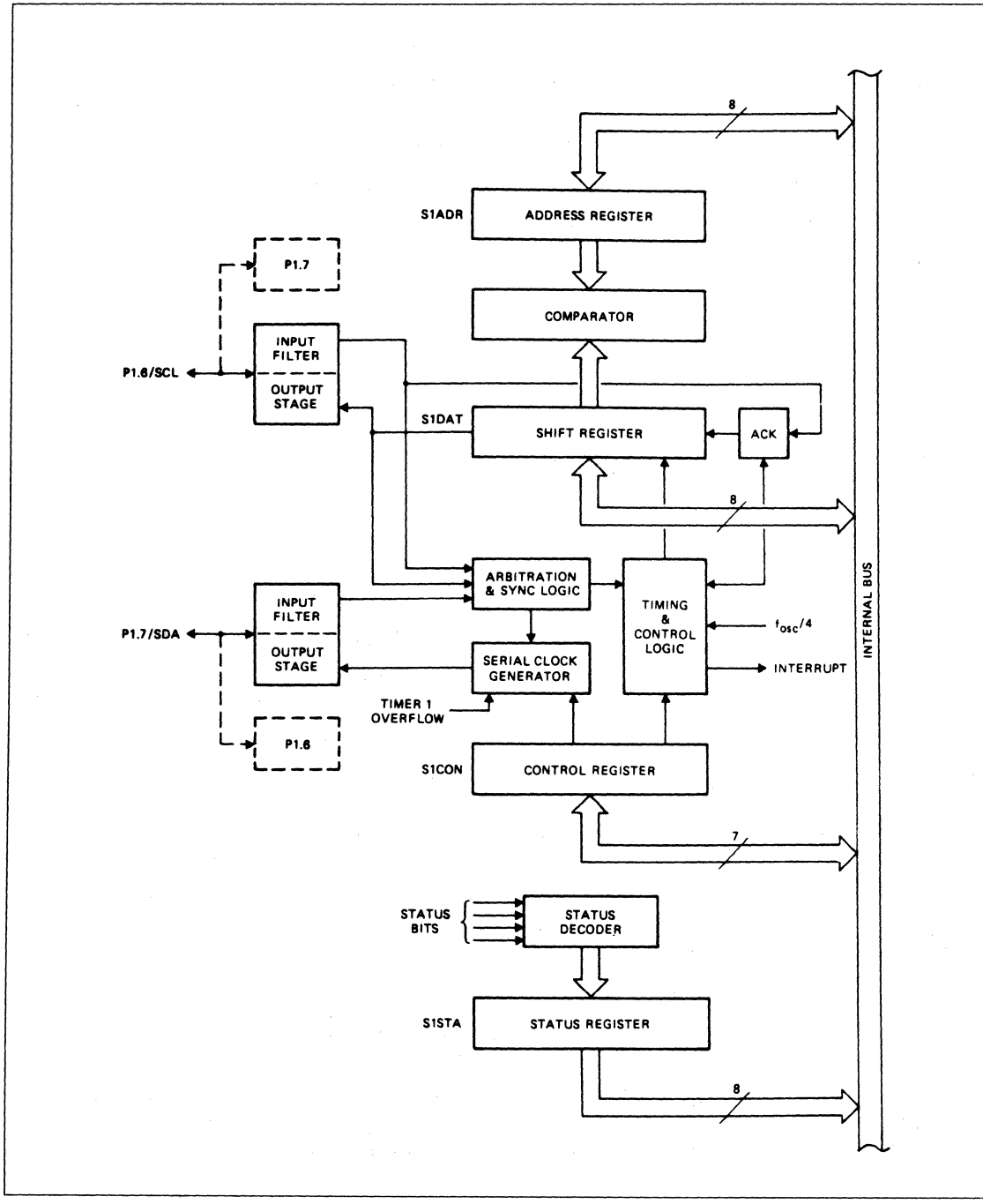
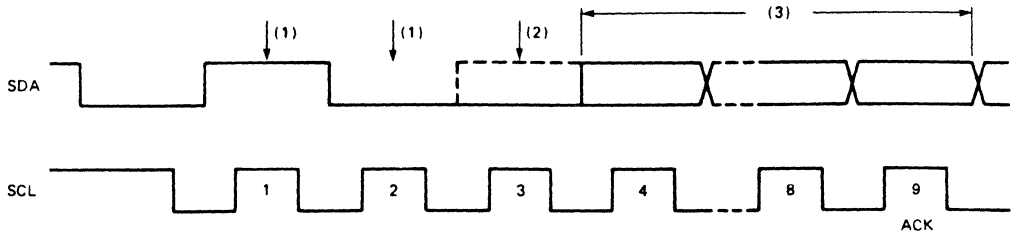
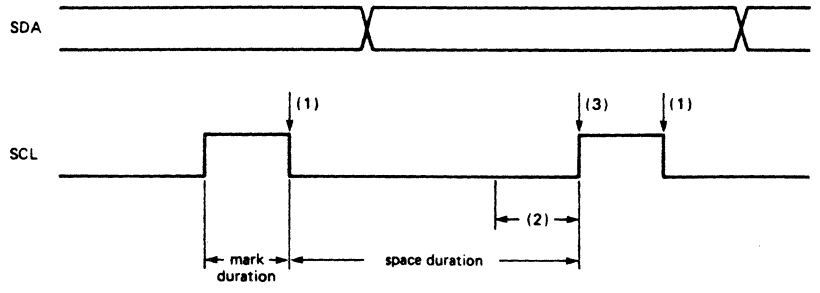


Figure 24. Block Diagram of the I²C Bus Serial Interface



1. Another device transmits identical serial data
2. Another device overrules a logic 1 (dotted line) transmitted by SIO1 (master) by pulling the SDA line LOW. Arbitration is lost and SIO1 enters the slave receiver mode
3. SIO1 is in the slave receiver mode but still generates clock pulses until the current byte has been transmitted. SIO1 will not generate clock pulses for the next byte. Data on SDA originates from the new master once it has won arbitration.

Figure 25. Arbitration Procedure



1. Another sevice pulls the the SCL line low before the SIO1 'mark' duration is complete. The serial clock generator is immediately reset and commences with the 'space' duration by pulling SCL LOW.
2. Another device stills pulls the SCL line LOW after SIO1 releases SCL. The serial clock generator is forced into the wait state until the SCL line is released.
3. The SCL line is released and the serial clock generator commences with the mark duration.

Figure 26. Serial Clock Synchronization

Serial Clock Generator

This programmable clock pulse generator provides the SCL clock pulses when SIO1 is in the master transmitter or master receiver mode. It is switched off when SIO1 is in a slave mode. The programmable output clock frequencies are: fosc/120, fosc/9600 and the Timer 1 overflow rate divided by eight. The output clock pulses have a 50% duty cycle unless the clock generator is synchronized with other SCL clock sources as described above.

Timing and Control

The timing and control logic generates the timing and control signals for serial byte handling. This logic block provides the shift pulses for S1DAT, enables the comparator, generates and detects start and stop conditions, receives and transmits acknowledge bits, controls the master and slave modes, contains interrupt request logic and monitors the I2C bus status.

Control Register, S1CON

This 7-bit special function register is used by the microcontroller to control the following SIO1 functions: start and restart of a serial transfer, termination of a serial transfer, bit rate, address recognition and acknowledgment.

Status Decoder and Status Register

The status decoder takes all of the internal status bits and compresses them into a 5-bit code. This code is unique for each I2C bus status. The 5-bit code may be used to generate vector addresses for fast processing of the various service routines. Each service routine processes a particular bus status. There are 26 possible bus states if all four modes of SIO1 are used. The 5-bit status code is latched into the five most significant bits of the status register when the serial interrupt flag is set (by hardware) and remains stable until the interrupt flag is cleared by software. The three least significant bits of the status register are always zero. If the status code is used as a vector to service routines, then the routines are displaced by eight address locations. Eight bytes of code is sufficient for most of the service routines (see the software example in this section).

More Information on SIO1 Operating Modes

The four operating modes are:

- Master Transmitter
- Master Receiver
- Slave Receiver
- Slave Transmitter

Data transfers in each mode of operation are shown in Figures 27–30. These figures contain the following abbreviations:

Abbreviation	Explanation
S	Start condition
SLA	7-bit slave address
R	Read bit (high level at SDA)
W	Write bit (low level at SDA)
A	Acknowledge bit (low level at SDA)
$\overline{A}$	Not acknowledge bit (high level at SDA)
DATA	8-bit data byte
P	Stop condition

In Figure 27–30, circles are used to indicate when the serial interrupt flag is set. The numbers in the circles show the status code held in the S1STA register. At these points, a service routine must be executed to continue or complete the serial transfer. These service routines are not critical since the serial transfer is suspended until the serial interrupt flag is cleared by software.

When a serial interrupt routine is entered, the status code in S1STA is used to branch to the appropriate service routine. For each status code, the required software action and details of the following serial transfer are given in Tables 9–12.

Master Transmitter Mode

In the master transmitter mode, a number of data bytes are transmitted to a slave receiver (see Figure 27). Before the master transmitter mode can be entered, S1CON must be initialized as follows:

	7	6	5	4	3	2	1	0
S1CON (D8H)	–	ENS1	STA	STO	SI	AA	CR1	CR0
	X	1	0	0	0	X	–bit rate–	

CR0 and CR1 define the serial bit rate. ENS1 must be set to logic 1 to enable SIO1. If the AA bit is reset, SIO1 will not acknowledge its own slave address or the general call address in the event of another device becoming master of the bus. In other words, if AA is reset, SIO0 can not enter a slave mode. STA, STO and SI must be reset.

The master transmitter mode may now be entered by setting the STA bit using the SETB instruction. The SIO1 logic will now test the I2C bus and generate a start condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI) is set and the status code in the status register (S1STA) will be 08H. This status code must be used to vector to an interrupt service routine that loads S1DAT with the slave address and the data direction bit (SLA+W). The SI bit in S1CON must then be reset before the serial transfer can continue.

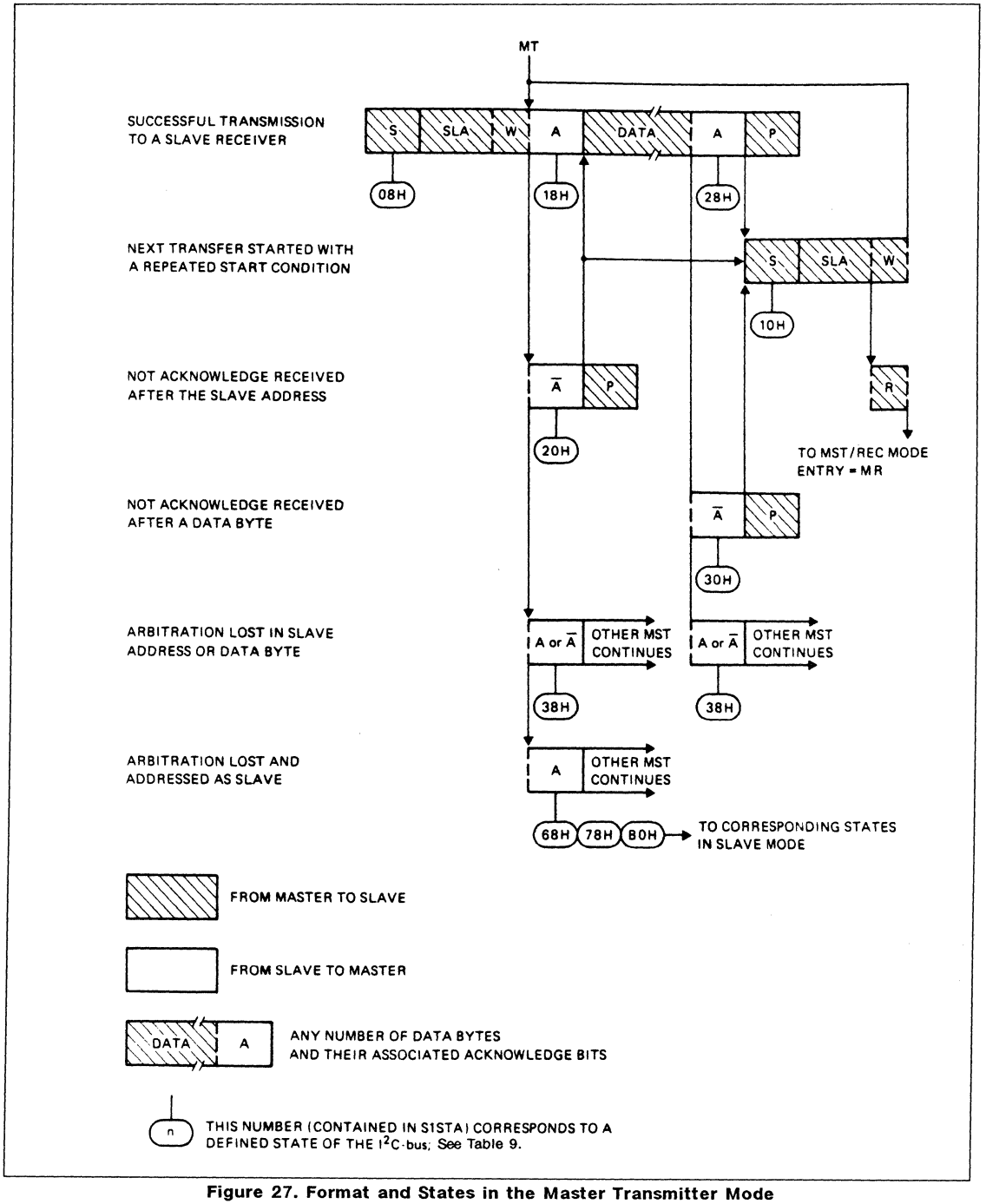


Figure 27. Format and States in the Master Transmitter Mode

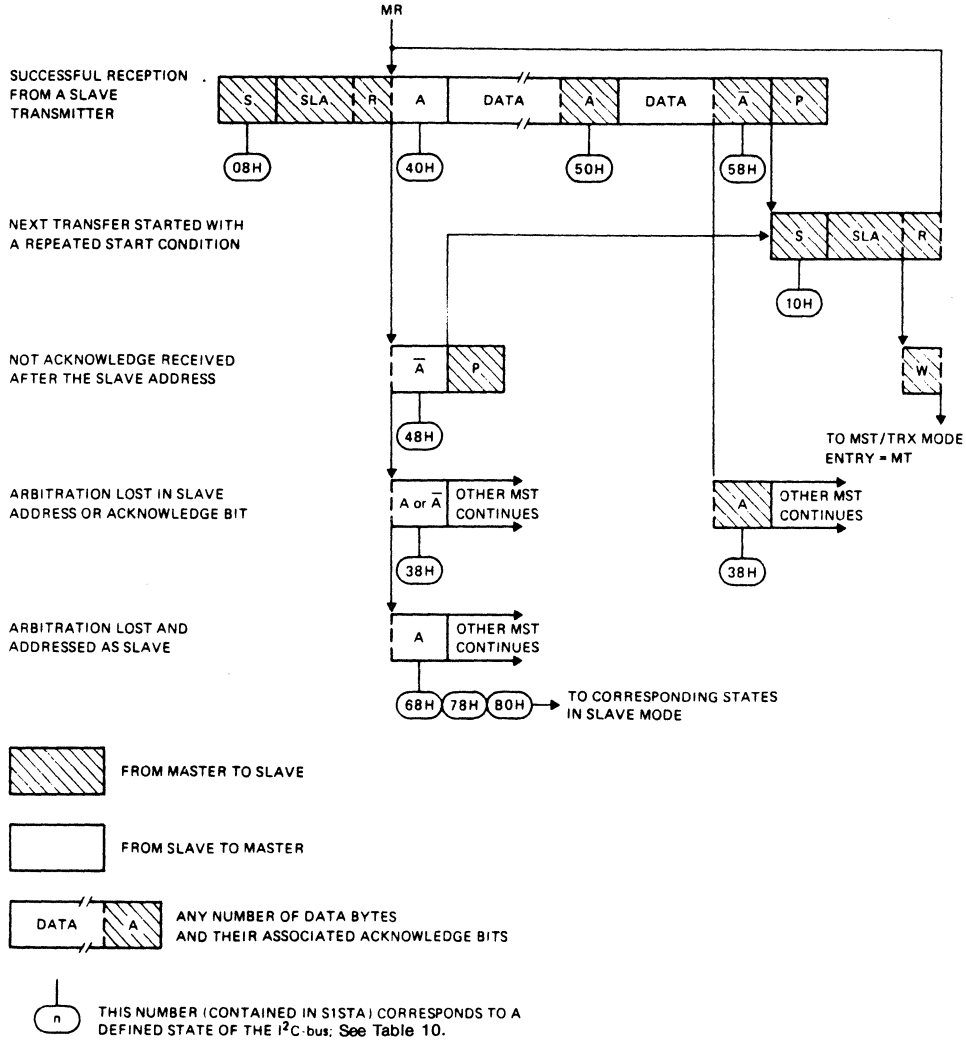


Figure 28. Format and States in the Master Receiver Mode

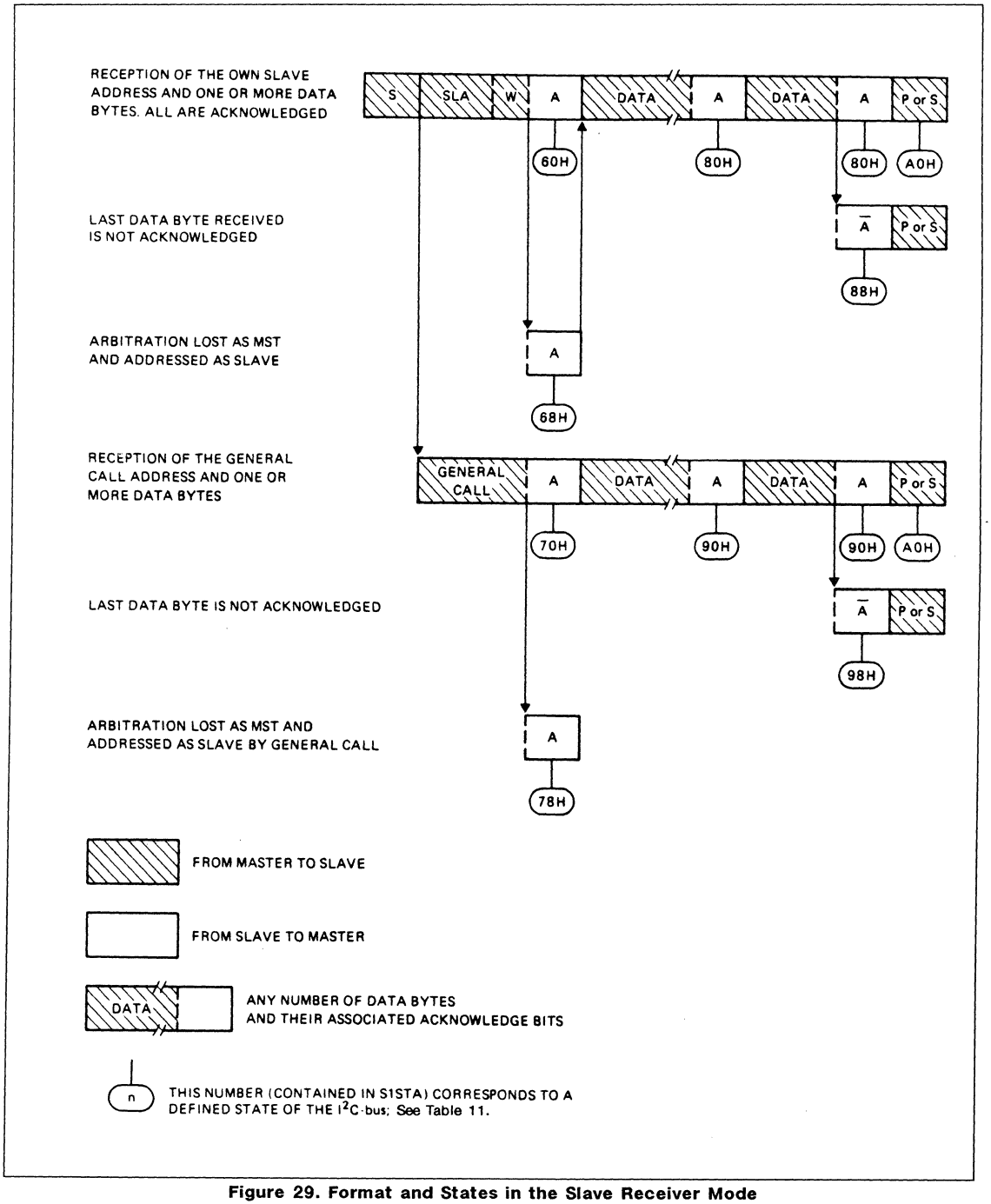


Figure 29. Format and States in the Slave Receiver Mode

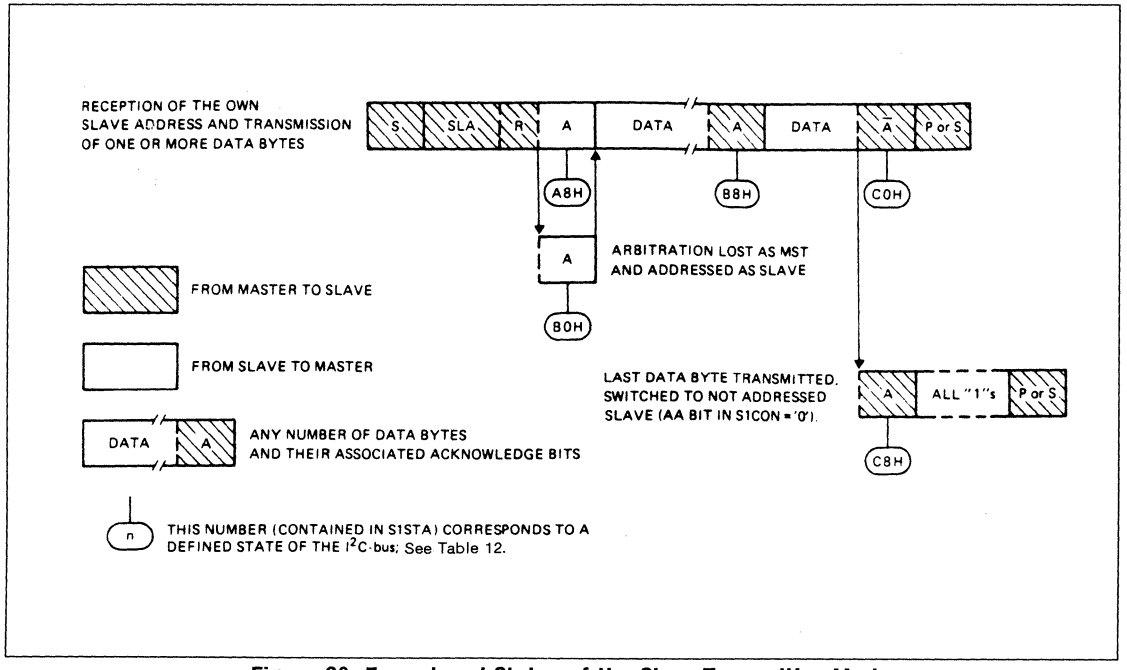


Figure 30. Format and States of the Slave Transmitter Mode

When the slave address and the direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again and a number of status codes in S1STA are possible. There are 18H, 20H or 38H for the master mode and also 68H, 78H, or B0H if the slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 9. ENS1, CR1 and CR0 are not affected by the serial transfer and are not referred to in Table 9. After a repeated start condition (state 10H). SIO1 may switch to the master receiver mode by loading S1DAT with SLA+R.

Master Receiver Mode

In the master receiver mode, a number of data bytes are received from a slave transmitter (see Figure 28). The transfer is initialized as in the master transmitter mode. When the start condition has been transmitted, the interrupt service routine must load S1DAT with the 7-bit slave address and the data direction bit (SLA+R). The SI bit in S1CON must then be cleared before the serial transfer can continue.

When the slave address and the data direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again and a number of status codes in S1STA are possible. These are 40H, 48H or 38H for the master mode and also 68H, 78H, or B0H if the slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 10. ENS1, CR1 and CR0 are not affected by the serial transfer and are not referred to in Table 10. After a repeated start condition (state 10H), SIO1 may switch to the master transmitter mode by loading S1DAT with SLA+W.

Slave Receiver Mode

In the slave receiver mode, a number of data bytes are received from a master transmitter (see Figure 29). To initiate the slave receiver mode. S1ADR and S1CON must be loaded as follows:

	7	6	5	4	3	2	1	0
S1ADR (DBH)	X	X	X	X	X	X	X	GC
	own slave address							

The upper 7 bits are the address to which SIO1 will respond when addressed by a master. If the LSB (GC) is set, SIO1 will respond to the general call address (00H); other wise it ignores the general call address.

	7	6	5	4	3	2	1	0
S1CON (D8H)	–	ENS1	STA	STO	SI	AA	CR1	CR0
	X	1	0	0	0	1	X	X

Table 9. Master Transmitter Mode

Status Code (S1STA)	Status of the I^2C bus and SIO1 Hardware	Application Software Response					Next Action Taken by SIO1 Hardware
		To/From S1DAT	To S1CON				
			STA	STO	SI	AA	
08H	A START condition has been transmitted	Load SLA+W	X	0	0	X	SLA+W will be transmitted; ACK bit will be received
10H	A repeated START condition has been transmitted	Load SLA+W or	X	0	0	X	As above
		Load SLA+R	X	0	0	X	SLA+R will be transmitted; SIO1 will be switched to MST/REC mode
18H	SLA+W has been transmitted; ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
20H	SLA+W has been transmitted; NOT ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
28H	Data byte in S1DAT has been transmitted; ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
30H	Data byte in S1DAT has been transmitted; NOT ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no S1DAT action or	1	0	0	X	Repeated START will be transmitted;
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
38H	Arbitration lost in SLA+R/$\overline{W}$ or Data bytes	No S1DAT action or	0	0	0	X	I^2C bus will be released; not addressed slave will be entered
		no S1DAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free

Table 10. Master Receiver Mode

Status Code (S1STA)	Status of the I²C bus and SIO1 Hardware	Application Software Response					Next Action Taken by SIO1 Hardware
		To/From S1DAT	To S1CON				
			STA	STO	SI	AA	
08H	A START condition has been transmitted	Load SLA+R	X	0	0	X	SLA+R will be transmitted; ACK bit will be received
10H	A repeated START condition has been transmitted	Load SLA+R or	X	0	0	X	As above
		Load SLA+W	X	0	0	X	SLA+W will be transmitted; SIO1 will be switched to MST/TRX mode
38H	Arbitration lost in NOT ACK bit	No S1DAT action or	0	0	0	X	I²C bus will be released; SIO1 will enter a slave mode
		no S1DAT action	1	0	0	X	A START will be transmitted when the bus becomes free
40H	SLA+R has been transmitted; ACK has been received	No S1DAT action or	0	0	0	0	Data byte will be received; NOT ACK bit will be returned
		no S1DAT action	0	0	0	1	Data byte will be received; ACK bit will be returned
48H	SLA+R has been transmitted; NOT ACK has been received.	No S1DAT action or	1	0	0	X	Repeated START condition will be transmitted
		no S1DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no S1DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
50H	Data byte has been received; ACK has been returned	Read data byte or	0	0	0	0	Data byte will be received; NOT ACK bit will be returned
		read data byte	0	0	0	1	Data byte will be received; ACK bit will be returned
58H	Data byte has been received; NOT ACK has been returned	Read data byte or	1	0	0	X	Repeated START condition will be transmitted
		read data byte or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		read data byte	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset

Table 11. Slave Receiver Mode

Status Code (S1STA)	Status of the I²C bus and SIO1 Hardware	Application Software Response					Next Action Taken by SIO1 Hardware
		To/From S1DAT	To S1CON				
			STA	STO	SI	AA	
60H	Own SLA+W has been received; ACK has been returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
68H	Arbitration lost in SLA+R/$\overline{W}$ as master; Own SLA+W has been received, ACK returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
70H	General call address (00H) has been received; ACK has been returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
78H	Arbitration lost in SLA+R/$\overline{W}$ as master; General call address has been received; ACK has been returned	No S1DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		no S1DAT action	X	0	0	1	Data byte will be received and ACK will be returned
80H	Previously addressed with own SLV address; DATA has been received; ACK has been returned	Read data byte or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		read data byte	X	0	0	1	Data byte will be received and ACK will be returned
88H	Previously addressed with own SLA; DATA byte has been received; NOT ACK has been returned	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1
		read data byte or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		read data byte	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1. A START condition will be transmitted when the bus becomes free.
90H	Previously addressed with General Call; DATA byte has been received; ACK has been returned	Read data byte or	X	0	0	0	Data byte will be received and NOT ACK will be returned
		read data byte	X	0	0	1	Data byte will be received and ACK will be returned
98H	Previously addressed with General Call; DATA byte has been received; NOT ACK has been returned	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1
		read data byte or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		read data byte	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1. A START condition will be transmitted when the bus becomes free.
A0H	A STOP condition or repeated START condition has been received while still addressed as SLV/REC or SLV/TRX	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1
		read data byte or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		read data byte	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1. A START condition will be transmitted when the bus becomes free.

Table 12. Slave Transmitter Mode

Status Code (S1STA)	Status of the I²C bus and SIO1 Hardware	Application Software Response					Next Action Taken by SIO1 Hardware
		To/From S1DAT	To S1CON				
			STA	STO	SI	AA	
A8H	Own SLA+R has been received; ACK has been returned	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received
		load data byte	X	0	0	1	Data byte will be transmitted; ACK bit will be received
B0H	Arbitration lost in SLA+R/$\overline{W}$ as master; own SLA+R has been received; ACK has been returned	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received
		load data byte	X	0	0	1	Data byte will be transmitted; ACK bit will be received
B8H	Data byte in S1DAT has been transmitted; ACK has been received	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK bits will be
		load data byte	X	0	0	1	received Data byte will be transmitted; ACK bit will be received
C0H	Data byte in S1DAT has been transmitted; NOT ACK has been received	No S1DAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		no S1DAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1
		no S1DAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		no S1DAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1. A START condition will be transmitted when the bus becomes free.
C8H	Last data byte in S1DAT has been transmitted (AA = 0); ACK has been received	No S1DAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address
		no S1DAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1
		no S1DAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		no S1DAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if S1ADR. 0 = logic 1. A START condition will be transmitted when the bus becomes free.

CR0 and CR1 do not affect SIO1 in the slave mode. ENS1 must be set to logic 1 to enable SIO1. The AA bit must be set to enable SIO1 to acknowledge its own slave address or the general call address. STA, STO and SI must be reset.

When S1ADR and S1CON have been initialized, SIO1 waits until it is addressed by its own slave address followed by the data direction bit which must be '0' (W) for SIO1 to operate in the slave receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag (I) is set and a valid status code can be read from S1STA. This status code is used to vector to an interrupt service routine and the appropriate action to be taken for each of these status codes is detailed in Table 11. The slave receiver mode may also be entered if arbitration is lost while SIO1 is in the master mode (see status 68H and 78H).

If the AA bit is reset during a transfer, SIO1 will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, SIO1 does not respond to its own slave address or a general call address. However, the I2C bus is still monitored and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate SIO1 from the I2C bus.

Slave Transmitter Mode

In the slave transmitter mode, a number of data bytes are transmitted to a master receiver (see Figure 30). Data transfer is initialized as in the slave receiver mode. When S1ADR and S1CON have been initialized, SIO1 waits until it is addressed by its own slave address followed by the data direction bit which must be '1' (R) for SIO1 to operate in the slave transmitter mode. After its own slave address and the R bit have been received, the serial interrupt flag (SI) is set and a valid status code can be read from S1STA. This status code is used to vector to an interrupt service routine and the appropriate action to be taken for each of these status codes is detailed in Table 12. The slave transmitter mode may also be entered if arbitration is lost while SIO1 is in the master mode (see state B0H).

If the AA bit is reset during a transfer, SIO1 will transmit the last byte of the transfer and enter state C0H or C8H. SIO1 is switched to the not addressed slave mode and will ignore the master receiver if it continues the transfer. Thus the master receiver receives all 1s as serial data. While AA is reset, SIO1 does not respond to its own slave address or a general call address. However, the I2C bus is still monitored and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate SIO1 from the I2C bus.

Miscellaneous States

There are two S1STA codes that do not correspond to a defined SIO1 hardware state (see Table 13). These are discussed below.

S1STA = F8H:

This status code indicates that no relevant information is available because the serial interrupt flag, SI is not yet set. This occurs between other states and when SIO1 is not involved in a serial transfer.

S1STA = 00H:

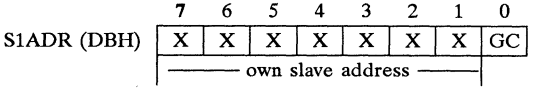
This status code indicates that a bus error has occurred during an SIO1 serial transfer. A bus error is caused when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte or an acknowledge bit. A bus error may also be caused when external interference disturbs the internal SIO1 signals. When a bus error occurs, SI is set. To recover from a bus error, the STO flag must be set and SI must be cleared. This causes SIO1 to enter the 'not addressed' slave mode (a defined state) and to clear the STO flag (no other bits in S1CON are affected). The SDA and SCL lines are released (a STOP condition is not transmitted).

The Four SIO1 Special Function Registers

The microcontroller interfaces to SIO1 via four special function registers. These four SFR's (S1ADR, S1DAT, S1CON and S1STA) are described individually in the following sections.

The Address Register, S1ADR

The CPU can read from and write to this 8-bit, directly addressable SFR. S1ADR is not affected by the SIO1 hardware. The contents of this register are irrelevant when SIO1 is in a master mode. In the slave modes, the seven most significant bits must be loaded with the microcontroller's own slave address and, if the least significant bit is set, the general call address (00H) is recognized; otherwise it is ignored.



The most significant bit corresponds to the first bit received from the I2C bus after a start condition. A logic 1 in S1ADR corresponds to a high level on the I2C bus and a logic 0 corresponds to a low level on the bus.

Table 13. Miscellaneous States

Status Code (S1STA)	Status of the I²C bus and SIO1 Hardware	Application Software Response					Next Action Taken by SIO1 Hardware
		To/From S1DAT	To S1CON				
			STA	STO	SI	AA	
F8H	No relevant state information available; S1 = 0	No S1DAT action	No S1CON action				Wait or proceed current transfer.
00H	Bus error during MST or selected slave modes, due to an illegal START or STOP condition. State 00H can also occur when interference causes SIO1 to enter an undefined state.	No S1DAT action	0	1	0	X	Only the internal hardware is affected in the MST or addressed SLV modes. In all cases, the bus is released and SIO1 is switched to the not addressed SLV mode. STO is reset.

The Data Register, S1DAT

S1DAT contains a byte of serial data to be transmitted or a byte which has just been received. The CPU can read from and write to this 8-bit, directly addressable SFR while it is not in the process of shifting a byte. This occurs when SIO1 is in a defined state and the serial interrupt flag is set. Data in S1DAT remains stable as long as SI is set. Data in S1DAT is always shifted from right to left: the first bit to be transmitted is the MSB (bit 7) and, after a byte has been received, the first bit of received data is located at the MSB of S1DAT. While data is being shifted out, data on the bus is simultaneously being shifted in; S1DAT always contains the last data byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in S1DAT.

	7	6	5	4	3	2	1	0
S1DAT (DAH)	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0

← shift direction ←

SD7 – SD0:

8-bits to be transmitted or just received. A logic 1 in S1DAT corresponds to a high level on the I2C bus and a logic 0 corresponds to a low level on the bus. Serial data shifts through S1DAT from right to left. Figure 31 shows how data in S1DAT is serially transferred to and from the SDA line.

S1DAT and the ACK flag form a 9-bit shift register which shifts in or shifts out an 8-bit byte, followed by an acknowledge bit. The ACK flag is controlled by the SIO1 hardware and cannot be accessed by the CPU. Serial data is shifted through the ACK flag into S1DAT on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into S1DAT, the serial data is available in S1DAT and the acknowledge bit is returned by the control logic during the ninth clock pulse.

Serial data is shifted out from S1DAT via a buffer (BSD7) on the falling edges of clock pulses on the SCL line.

When the CPU writes to S1DAT, BSD7 is loaded with the content of S1DAT.7 which is the first bit to be transmitted to the SDA line (see Figure 32). After nine serial clock pulses, the eight bits in S1DAT will have been transmitted to the SDA line and the acknowledge bit will be present in ACK. Note that the eight transmitted bits are shifted back into S1DAT.

The Control Register, S1CON

The CPU can read from and write to this 8-bit, directly addressable SFR. The most significant bit is not used and a logic 1 will be returned if S1CON.7 is read. Two bits are affected by the SIO1 hardware: the SI bit is set when a serial interrupt is requested and the STO bit is cleared when a STOP condition is present on the I2C bus. The STO bit is also cleared when ENS1 = '0'.

	7	6	5	4	3	2	1	0
S1CON (D8H)	–	ENS1	STA	STO	SI	AA	CR1	CR0

ENS1, the SIO1 Enable Bit

ENS1 = '0': when ENS1 is '0', the SDA and SCL outputs are in a high impedance state. SDA and SCL input signals are ignored, SIO1 is in the 'not addressed' slave state and the STO bit in S1CON is forced to '0'. No other bits are affected. P1.6 and P1.7 may be used as open drain I/O ports.

ENS1 = '1': when ENS1 is '1', SIO1 is enabled. The P1.6 and P1.7 port latches must be set to logic 1.

ENS1 should not be used to temporarily release SIO1 from the I2C bus since, when ENS1 is reset, the I2C bus status is lost. The AA flag should be used instead (see description of the AA flag in the following text).

In the following text, it is assumed that ENS1 = '1'.

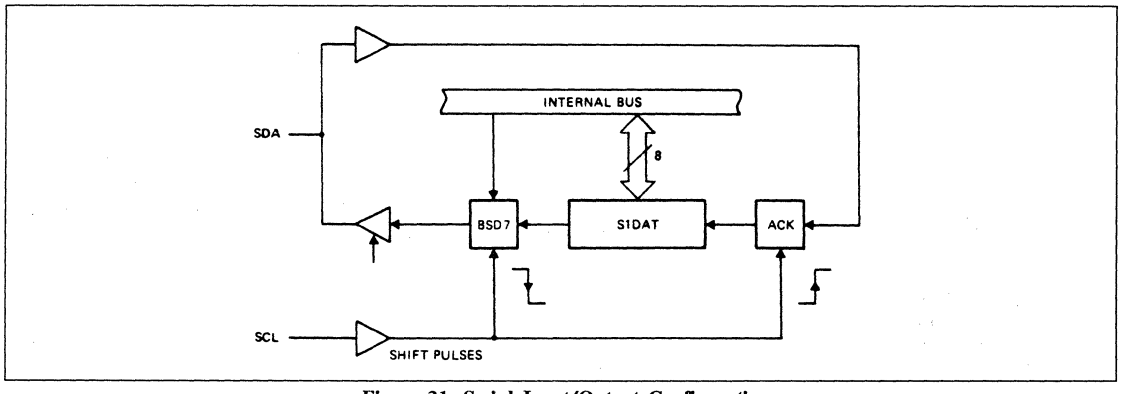


Figure 31. Serial Input/Output Configuration

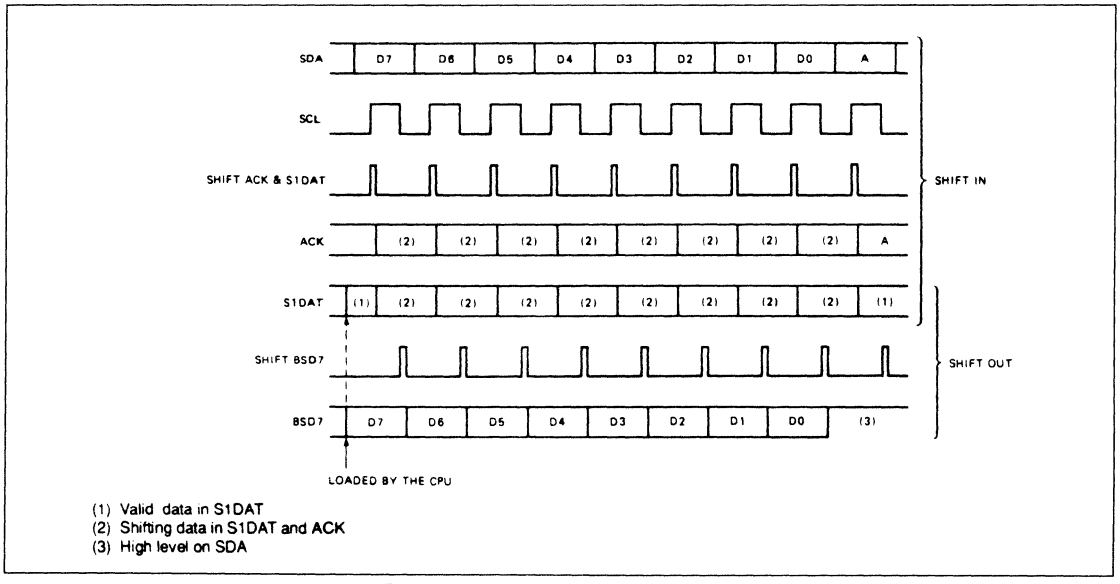


Figure 32. Shift-in and Shift-out Timing

STA, the START Flag

STA = '1': when the STA bit is set to enter a master mode the SIO1 hardware checks the status of the I2C bus and generates a START condition if the bus is free. If the bus is not free, then SIO1 waits for a STOP condition (which will free the bus) and generates a START condition after a delay of a half clock period of the internal serial clock generator.

If STA is set while SIO1 is already in a master mode and one or more bytes are transmitted or received, SIO1 transmits a repeated START condition. STA may be set at any time. STA may also be set when SIO1 is an addressed slave.

STA = '0': when the STA bit is reset, no START condition or repeated START condition will be generated.

STO, the STOP Flag

STO = '1': when the STO bit is set while SIO1 is in a master mode, a STOP condition is transmitted to the I2C bus. When the STOP condition is detected on the bus, the SIO1 hardware clears the STO flag. In a slave mode, the STO flag may be set to recover from an error condition. In this case, no STOP condition is transmitted to the I2C bus. However, the SIO1 hardware behaves as if a STOP condition has been received and switches to the defined 'not addressed' slave receiver mode. The STO flag is automatically cleared by hardware.

If the STA and STO bits are both set, then a STOP condition is transmitted to the I2C bus if SIO1 is in a master mode (in a slave mode, SIO1 generates an internal STOP condition which is not transmitted). SIO1 then transmits a START condition.

STO = '0': when the STO bit is reset, no STOP condition will be generated.

SI, the Serial Interrupt Flag

SI = '1': when the SI flag is set, then, if the EA and ES1 (interrupt enable register) bits are also set, a serial interrupt is requested. SI is set by hardware when one of 25 of the 26 possible SIO1 states is entered. The only state that does not cause SI to be set is state F8H which indicates that no relevant state information is available.

While SI is set, the low period of the serial clock on the SCL line is stretched and the serial transfer is suspended. A high level on the SCL line is unaffected by the serial interrupt flag. SI must be reset by software.

SI = '0': when the SI flag is reset, no serial interrupt is requested and there is no stretching of the serial clock on the SCL line.

AA, the Assert Acknowledge Flag

AA = '1': if the AA flag is set, an acknowledge (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when:
-the 'own slave address' has been received
-the general call address has been received while the general call bit (GC) in S1ADR is set

–a data byte has been received while SIO1 is in the master receiver mode
–a data byte has been received while SIO1 is in the addressed slave receiver mode

AA = '0': if the AA flag is reset, a not acknowledge (high level to SDA) will be returned during the acknowledge clock pulse on SCL when:
–a data has been received while SIO1 is in the master receiver mode
–a data byte has been received while SIO1 is in the addressed slave receiver mode

When SIO1 is in the addressed slave transmitter mode, state C8H will be entered after the last serial is transmitted (see Figure 30). When SI is cleared, SIO1 leaves state C8H, enters the not addressed slave receiver mode and the SDA line remains at a high level. In state C8H, the AA flag can be set again for future address recognition.

When SIO1 is in the not addressed slave mode, its own slave address and the general call address are ignored. Consequently, no acknowledge is returned and a serial interrupt is not requested. Thus, SIO1 can be temporarily released from the I²C bus while the bus status is monitored. While SIO1 is released from the bus, START and STOP conditions are detected and serial data is shifted in. Address recognition can be resumed at any time by setting the AA flag. If the AA flag is set when the part's own slave address or the general call address has been partly received, the address will be recognized at the end of the byte transmission.

CR0 and CR1, the Clock Rate Bits

These two bits determine the serial clock frequency when SIO1 is in a master mode. The various serial rates are shown in Table 14.

A 12.5kHz bit rate may be used by devices that interface to the I²C bus via standard I/O port lines which are software driven and slow. 100kHz is usually the maximum bit rate and can be derived from a 12MHz oscillator or a 6MHz oscillator. A variable bit rate (0.5kHz to 62.5kHz) may also be used if Timer 1 is not required for any other purpose while SIO1 is in a master mode.

The frequencies shown in Table 14 are unimportant when SIO1 is in a slave mode. In the slave modes, SIO1 will automatically synchronize with any clock frequency up to 100kHz.

The Status Register, S1STA

S1STA is an 8-bit read-only special function register. The three least significant bits are always zero. The five most significant bits contain the status code. There are 26 possible status codes. When S1STA contains F8H, no relevant state information is available and no serial interrupt is requested. All other S1STA values correspond to defined SIO1 states. When each of these states is entered, a serial interrupt is requested (SI = '1'). A valid status code is present in S1STA one machine cycle after SI is set by hardware and is still present one machine cycle after SI has been reset by software.

Some Special Cases

The SIO1 hardware has facilities to handle the following special cases that may occur during a serial transfer:

Simultaneous Repeated START Conditions from 2 Masters

A repeated START condition may be generated in the master transmitter or master receiver modes. A special case occurs if another master simultaneously generates a repeated START condition (see Figure 33). Until this occurs, arbitration is not lost by either master since they were both transmitting the same data.

If the SIO1 hardware detects a repeated START condition on the I²C bus before generating a repeated START condition itself, it will release the bus and no interrupt request is generated. If another master frees the bus by generating a STOP condition, SIO1 will transmit a normal START condition (state 08H) and a retry of the total serial data transfer can commence.

Data Transfer After Loss of Arbitration

Arbitration may be lost in the master transmitter and master receiver modes (see Figure 25). Loss of arbitration is indicated by the following states in S1STA: 38H, 68H, 78H and B0H (see Figures 27 and 28).

If the STA flag in S1CON is set by the routines which service these states, then, if the bus is free again, a START condition (state 08H) is transmitted without intervention by the CPU and a retry of the total serial transfer can commence.

Table 14. Serial Clock Rates

CR1	CR0	f_{osc} divided by	bit frequency
0	0	960	12.5 kHz @ f_{osc} = 12 MHz
0	1	120	100 kHz @ f_{osc} = 12 MHz
1	0	60	100 kHz @ f_{osc} = 6 MHz
1	1	96 × (256 – reload value of Timer 1) (reload value range is 0 to 254 in mode 2)	0.5 kHz to 62.5 KHz at f_{osc} = 12 MHz

Forced Access to the I2C bus

In some applications, it may be possible for an uncontrolled source to cause a bus hang-up. In such situations, the problem may be caused by interference, temporary interruption of the bus or a temporary short-circuit between SDA and SCL.

If an uncontrolled source generates a superfluous START or masks a STOP condition, then the I2C bus stays busy indefinitely. If the STA flag is set and bus access is not obtained within a reasonable amount of time, then a forced access to the I2C bus is possible. This is achieved by setting the STO flag while the STA flag is still set. No STOP condition is transmitted. The SIO1 hardware behaves as if a STOP condition was received and is able to transmit a START condition. The STO flag is cleared by hardware (see Figure 34).

I2C Bus Obstructed by a Low Level on SCL or SDA

An I2C bus hang-up occurs if SDA or SCL is pulled LOW by an uncontrolled source. If the SCL line is obstructed (pulled LOW) by a device on the bus, no further serial transfer is possible and the SIO1 hardware cannot resolve this type of problem. When this occurs, the problem must be resolved by the device that is pulling the SCL bus line LOW.

If the SDA line is obstructed by another device on the bus (e.g. a slave device out of bit synchronization), the problem can be solved by transmitting additional clock pulses on the SCL line (see Figure 35). The SIO1 hardware transmits additional clock pulses when he STA flag is set but no START condition can be generated because the SDA line is pulled LOW while the I2C bus is considered free. The SIO1 hardware attempts to generate a

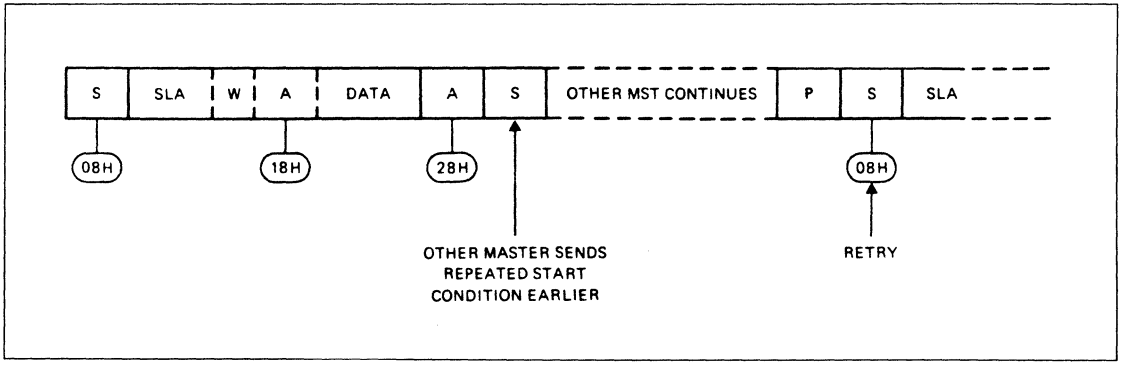


Figure 33. Simultaneous Repeated START Conditions from 2 Masters

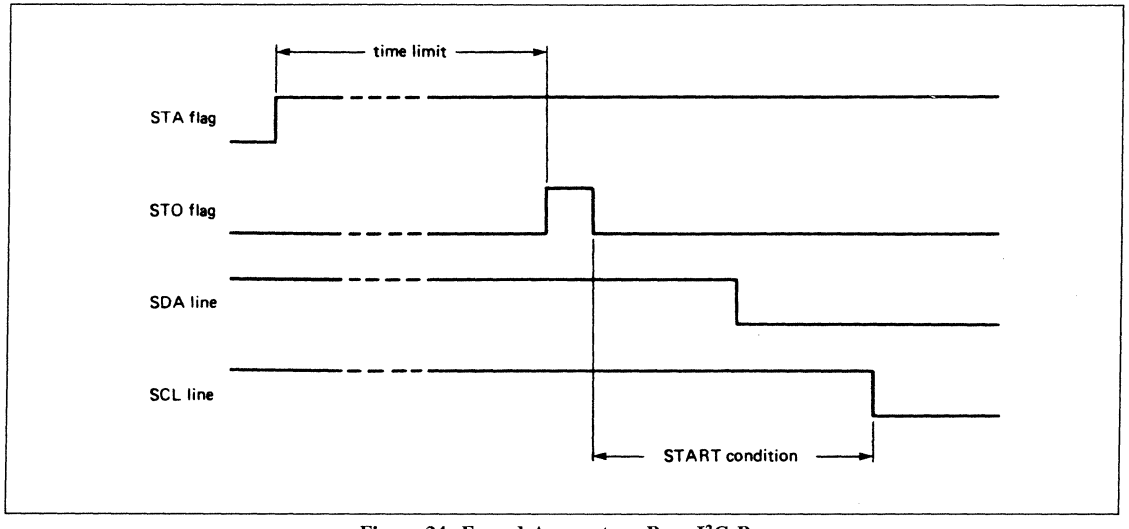
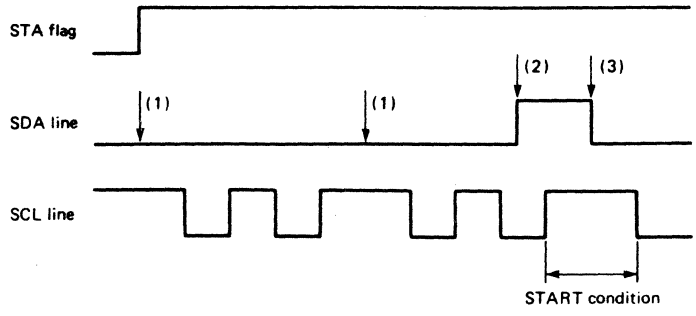


Figure 34. Forced Access to a Busy I2C Bus



(1) Unsuccessful attempt to send a START condition
(2) SDA line released
(3) Successful attempt to send a START condition; state 08H is entered

Figure 35. Recovering from a Bus Obstruction Caused by a Low Level on SDA

START condition after every two additional clock pulses on the SCL line. When the SDA line is eventually released, a normal START condition is transmitted, state 08H is entered and the serial transfer continues.

If a forced bus access occurs or a repeated START condition is transmitted while SDA is obstructed (pulled LOW), the SIO1 hardware performs the same action as described above. In each case, state 08H is entered after a successful START condition is transmitted and normal serial transfer continues. Note that the CPU is not involved in solving these bus hang-up problems.

Bus Error

A bus error occurs when a START or STOP condition is present at an illegal position in the format frame. Examples of illegal positions are during the serial transfer of an address byte, a data or an acknowledge bit.

The SIO1 hardware only reacts to a bus error when it is involved in a serial transfer either as a master or an addressed slave. When a bus error is detected, SIO1 immediately switches to the not addressed slave mode, releases the SDA and SCL lines, sets the interrupt flag and loads the status register with 00H. This status code may be used to vector to a service routine which either attempts the aborted serial transfer again or simply recovers form the error condition as shown in Table 13.

Software Examples of SIO1 Service Routines

This section consists of a software example for:

–Initialization of SIO1 after a RESET
–Entering the SIO1 interrupt routine
–The 26 state service routines for the
  –master transmitter mode
  –master receiver mode
  –slave receiver mode
  –slave transmitter mode

Initialization

In initialization routine, SIO1 is enabled for both master and slave modes. For each mode, a number of bytes of internal data RAM are allocated to the SIO to act as either a transmission or reception buffer. In this example 8 bytes of internal data RAM are reserved for different purposes. The data memory map is shown in Figure 36. The initialization routine performs the following functions:

–S1ADR is loaded with the part's own slave address and the general call bit (GC)
–P1.6 and P1.7 bit latches are loaded with logic 1s
–Ram location HADD is loaded with the high order address byte of the service routines
–The SIO1 interrupt enable and interrupt priority bits are set
–The slave mode is enabled by simultaneously setting the ENS1 and AA bits in S1CON and the serial clock frequency (for master modes) is defined by loading CR0 and CR1 in S1CON. The master routines must be started in the main program.

The SIO1 hardware now begins checking the I2C bus for its own slave address and general call. If the general call or the own slave address is detected, an interrupt is requested and S1STA is loaded with the appropriate state information. The following text describes a fast method of branching to the appropriate service routine.

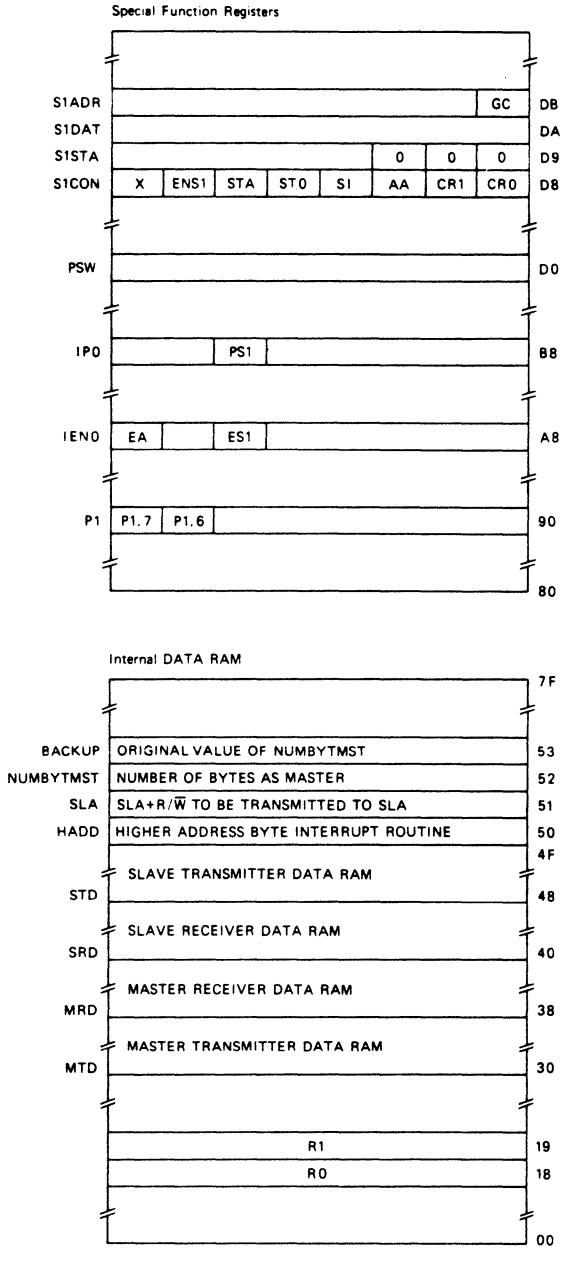


Figure 36. SIO1 Data Memory Map

SIO1 Interrupt Routine

When the SIO1 interrupt is entered, the PSW is first pushed on the stack. Then S1STA and HADD (loaded with the high order address byte of the 26 service routines by the initialization routine) are pushed on to the stack. S1STA contains a status code which is the lower byte of one of the 26 service routines. The next instruction is RET which is the return from subroutine instruction. When this instruction is executed, the high and low order address bytes are popped from stack and loaded into the program counter.

The next instruction to be executed is the first instruction of the state service routine. Seven bytes of program code (which execute in eight machine cycles) are required to branch to one of the 26 state service routines.

```
SI   PUSH  PSW     Save PSW
     PUSH  S1STA   Push status code
                   (low order address byte)
     PUSH  HADD    Push high order address
                   byte
     RET           Jump to state service
                   routine
```

The state service routines are located in a 256-byte page of program memory. The location of this page is defined in the initialization routine. The page can be located anywhere in program memory by loading data RAM register HADD with the page number. Page 01 is chosen in this example and the service routines are located between addresses 0100H and 01FFH.

The State Service Routines

The state service routines are located 8 bytes from each other. 8 bytes of code are sufficient for most of the service routines. A few of the routines require more than 8 bytes and have to jump to other locations to obtain more bytes of code. Each state routine is a part of the SIO1 interrupt routine and handles one of the 26 states. It ends with a RETI instruction which causes a return to the main program.

Master Transmitter and Master Receiver Modes

The master mode is entered in the main program. To enter the master transmitter mode, the main program must first load the internal data RAM with the slave address, data bytes and the number of data bytes to be transmitted. To enter the master receiver mode, the main program must first load the internal data RAM with the slave address and the number of data bytes to be received. The R/$\overline{W}$ bit determines whether SIO1 operates in the master transmitter or master receiver mode.

Master mode operation commences when the STA bit in S1CION is set by the SETB instruction and data transfer is controlled by the master state service routines in accordance with Table 9, Table 10, Figure 27 and Figure 28. In the example below, 4 bytes are transferred. There is no repeated START condition. In the event of lost arbitration, the transfer is restarted when the bus becomes free. If a bus error occurs, the I²C bus is released and SIO1 enters the not selected slave receiver mode. If a slave device returns a not acknowledge, a STOP condition is generated.

A repeated START condition can be included in the serial transfer if the STA flag is set instead of the STO flag in the state service routines vectored to by status codes 28H and 58H. Additional software must be written to determine which data is transferred after a repeated START condition.

Slave Transmitter and Slave Receiver Modes

After initialization, SIO1 continually tests the I²C bus and branches to one of the slave state service routines if it detects its own slave address or the general call address (see Table 11, Table 12, Figure 29, and Figure 30). If arbitration was lost while in the master mode, the master mode is restarted after the current transfer. If a bus error occurs, the I²C bus is released and SIO1 enters the not selected slave receiver mode.

In the slave receiver mode, a maximum of 8 received data bytes can be stored in the internal data RAM. A maximum of 8 bytes ensures that other RAM locations are not overwritten if a master sends more bytes. If more than 8 bytes are transmitted, a not acknowledge is returned and SIO1 enters the not addressed slave receiver mode. A maximum of one received data byte can be stored in the internal data RAM after a general call address is detected. If more than one byte is transmitted, a not acknowledge is returned and SIO1 enters the not addressed slave receiver mode.

In the slave transmitter mode, data to be transmitted is obtained from the same locations in the internal data RAM that were previously loaded by the main program. After a not acknowledge has been returned by a master receiver device, SIO1 enters the not addressed slave mode.

Adapting the Software for Different Applications

The following software example shows the typical structure of the interrupt routine including the 26 state service routines and may be used as a base for user applications. If one or more of the four modes are not used, the associated state service routines may be removed but care should be taken that a deleted routine can never be invoked.

This example does not include any time-out routines. In the slave modes, time-out routines are not very useful since, in these modes, SIO1 behaves essentially as a passive device. In the master modes, an internal timer may be used to cause a time-out if a serial transfer is not complete after a defined period of time. This time period is defined by the system connected to the I²C bus.

```
                !************************************************************
                ! SIO1 EQUATE LIST
                !************************************************************
                !************************************************************
                ! LOCATIONS OF THE SIO1 SPECIAL FUNCTION REGISTERS
                !************************************************************
00D8            S1CON  =0xd8
00D9            S1STA  =0xd9
00DA            S1DAT  =0xda
00DB            S1ADR  =0xdb

00A8            IEN0   =0xa8
00B8            IP0    =0xb8

                !************************************************************
                !BIT LOCATIONS
                !************************************************************

00DD            STA    =0xdd                !STA bit in S1CON
00BD            SIO1HP =0xbd                !IP0, SIO1  Priority bit


                !************************************************************
                !IMMEDIATE DATA TO WRITE INTO REGISTER S1CON
                !************************************************************
00D5            ENS1_NOTSTA_STO_NOTSI_AA_CR0       =0xd5   !Genearates STOP
                                                           !(CR0=100 Khz)
00C5            ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0    =0xc5   !Releases BUS and
                                                           !ACK
00C1            ENS1_NOTSTA_NOTSTO_NOTSI_NOTAA_CR0 =0xc1   !Releases BUS and
                                                           !NOT ACK
00E5            ENS1_STA_NOTSTO_NOTSI_AA_CR0       =0xe5   !Release BUS and
                                                           !set STA

                !************************************************************
                !GENERAL IMMEDIATE DATA
                !************************************************************
0031            OWNSLA  =0x31                  !Own SLA+Generall Call
                                               !must be written into S1ADR
00A0            ENSIO1  =0xa0                  !EA+ES1, enable SIO1 interrupt
                                               !must be written into IEN0
0001            PAG1    =0x01                  !select PAG1 as HADD
00C0            SLAW    =0xc0                  !S1A+W to be transmitted
00C1            SLAR    =0xc1                  !SLA+R to be transmitted
0018            SELRB3  =0x18                  !Select Register Bank 3

                !************************************************************
                !LOCATIONS IN DATA RAM
                !************************************************************
0030            MTD     =0x30                  !MST/TRX/DATA base address
0038            MRD     =0x38                  !MST/REC/DATA base address
0040            SRD     =0x40                  !SLV/REC/DATA base address
0048            STD     =0x48                  !SLV/TRX/DATA base address

0053            BACKUP   =0x53                 !Backup from NUMBYTMST.
                                               !To restore NUMBYTMST in case
                                               !of an Arbitration Lost.
0052            NUMBYTMST=0x52                 !Number of bytes to transmit
                                               !or receive as MST.
0051            SLA      =0x51                 !Contains SLA+R/W to be
                                               !transmitted.
0050            HADD     =0x50                 !High Address byte for STATE 0
                                               !till STATE 25.
```

```
                        !************************************************************
                        !INITIALISATION ROUTINE
                        !Example to initialize IIC Interface as slave receiver or
                        !slave transmitter and start a MASTER TRANSMIT or a MASTER
                        !RECEIVE function. 4 bytes will be transmitted or received.
                        !************************************************************
                        .sect strt
                        .base 0x00
0000 4100                       ajmp INIT               !RESET

                        .sect initial
                        .base 0x200
0200 75DB31             INIT:   mov S1ADR,#OWNSLA       !Load own SLA + enable
                                                        !general call recognition
0203 D296                       setb p1(6)              !P1.6 High level.
0205 D297                       setb p1(7)              !P1.7 High level.
0207 755001                     mov HADD,#PAG1
020A 43A8A0                     orl IEN0,#ENSI01        !Enable SI01 interrupt
020D C2BD                       clr SI01HP              !SI01 interrupt low
                                                        !priority
020F 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                        !Initialize SLV funct.

                        !************************************************************

                        !------------------------------------------------------------
                        !START MASTER TRANSMIT FUNCTION
                        !------------------------------------------------------------

0212 755204                     mov NUMBYTMST,#0x4      !Transmit 4 bytes.
0215 7551C0                     mov SLA,#SLAW           !SLA+W,Transmit funct.
0218 D2DD                       setb STA                !set STA in S1CON

                        !------------------------------------------------------------
                        !START MASTER RECEIVE FUNCTION
                        !------------------------------------------------------------

021A 755204                     mov NUMBYTMST,#0x4      !Receive 4 bytes.
021D 7551C1                     mov SLA,#SLAR           !SLA+R, Receive funct.
0220 D2DD                       setb STA                !set STA in S1CON

                        !------------------------------------------------------------
```

```
                        !*************************************************************
                        !SIO1 INTERRUPT ROUTINE
                        !*************************************************************
                        .sect intvec
                        .base 0x2b                              !SIO1 interrupt vector

                        !S1STA and HADD are pushed onto the stack. They serve as
                        !return address for the RET instruction.
                        !The RET instruction sets the Program Counter to address
                        !HADD,S1STA and jumps to the rigth subroutine.


002B C0D0                       push psw                        !save psw
002D C0D9                       push S1STA
002F C050                       push HADD
0031 22                         ret                             !JMP to address
                                                                !HADD,S1STA.

                        !-------------------------------------------------------
                        ! STATE : 00, Bus error.
                        ! ACTION: Enter not adressed SLV mode and release bus.
                        !         STO reset.
                        !-------------------------------------------------------
                        .sect st0
                        .base 0x100

0100 75D8D5                     mov S1CON,#ENS1_NOTSTA_STO_NOTSI_AA_CR0   !clr SI
                                                                          !set STO,AA
0103 D0D0                       pop psw
0105 32                         reti
```

```
                        !*************************************************************
                        !*************************************************************
                        !MASTER STATE SERVICE ROUTINES
                        !*************************************************************
                        !State 08 and State 10 are both for MST/TRX and MST/REC.
                        !The R/W bit decides whether the next state is within
                        !MST/TRX mode or within MST/REC mode.
                        !*************************************************************

                        !-------------------------------------------------------
                        ! STATE : 08, A START condition has been transmitted.
                        ! ACTION: SLA+R/W are transmitted, ACK bit is received.
                        !-------------------------------------------------------
                        .sect mts8
                        .base 0x108

0108 8551DA                         mov S1DAT,SLA                   !Load SLA+R/W
010B 75D8C5                         mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                                    !clr SI
010E 01A0                           ajmp INITBASE1

                        !-------------------------------------------------------
                        ! STATE : 10, A repeated START condition has been
                        !         transmitted.
                        ! ACTION: SLA+R/W are transmitted, ACK bit is received.
                        !-------------------------------------------------------
                        .sect mts10
                        .base 0x110

0110 8551DA                         mov S1DAT,SLA                   !Load SLA+R/W
0113 75D8C5                         mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                                    !clr SI
0116 01A0                           ajmp INITBASE1

                        .sect ibase1
                        .base 0xa0
00A0 75D018             INITBASE1:  mov psw,#SELRB3
00A3 7930                           mov r1,#MTD
00A5 7838                           mov r0,#MRD
00A7 855253                         mov BACKUP,NUMBYTMST            !Save initial value
00AA D0D0                           pop psw
00AC 32                             reti
```

```
                        !*************************************************************
                        !*************************************************************
                        !MASTER TRANSMITTER STATE SERVICE ROUTINES
                        !*************************************************************
                        !*************************************************************

                        !-------------------------------------------------------
                        ! STATE : 18,Previous state was STATE 8 or STATE 10,
                        !         SLA+W have been transmitted, ACK has
                        !         been received.
                        ! ACTION: First DATA is transmitted, ACK bit
                        !         is recieved.
                        !-------------------------------------------------------
                        .sect mts18
                        .base 0x118

0118 75D018                         mov psw,#SELRB3
011B 87DA                           mov S1DAT,@r1
011D 01B5                           ajmp CON

                        !-------------------------------------------------------
                        ! STATE : 20, SLA+W have been transmitted, NOT ACK
                        !         has been received
                        ! ACTION: Transmit STOP condition.
                        !-------------------------------------------------------
                        .sect mts20
                        .base 0x120

0120 75D8D5                         mov S1CON,#ENS1_NOTSTA_STO_NOTSI_AA_CR0
                                                              !set STO, clr SI
0123 D0D0                           pop psw
0125 32                             reti

                        !-------------------------------------------------------
                        ! STATE : 28, DATA of S1DAT have been transmitted
                        !         ACK received.
                        ! ACTION: If Transmitted DATA is last DATA then
                        !         transmit a STOP condition, else transmit
                        !         next DATA.
                        !-------------------------------------------------------
                        .sect mts28
                        .base 0x128

0128 D55285                         djnz NUMBYTMST,NOTLDAT1     !JMP if NOT last DATA
012B 75D8D5                         mov S1CON,#ENS1_NOTSTA_STO_NOTSI_AA_CR0
                                                              !clr SI, set AA
012E 01B9                           ajmp RETmt


                        .sect mts28sb
                        .base 0x0b0
00B0 75D018             NOTLDAT1:   mov psw,#SELRB3
00B3 87DA                           mov S1DAT,@r1
00B5 75D8C5             CON:        mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                              !clr SI, clr AA
00B8 09                             inc r1
00B9 D0D0               RETmt   :   pop psw
00BB 32                             reti
```

```
                        !-------------------------------------------------------
                        ! STATE : 30, DATA of S1DAT have been transmitted,
                        !         NOT ACK received.
                        ! ACTION: Transmit a STOP condition.
                        !-------------------------------------------------------
                        .sect mts30
                        .base 0x130
0130 75D8D5                       mov S1CON,#ENS1_NOTSTA_STO_NOTSI_AA_CR0
                                                        !set STO, clr SI
0133 D0D0                         pop psw
0135 32                           reti

                        !-------------------------------------------------------
                        ! STATE : 38, Arbitration lost in SLA+W or DATA.
                        ! ACTION: Bus is released, not adressed SLV mode is
                        !         entered. A new START condition is transmitted
                        !         when the IIC bus is free again.
                        !-------------------------------------------------------
                        .sect mts38
                        .base 0x138

0138 75D8E5                       mov S1CON,#ENS1_STA_NOTSTO_NOTSI_AA_CR0
013B 855352                       mov NUMBYTMST,BACKUP
013E 01B9                         ajmp RETmt




                        !*********************************************************
                        !*********************************************************
                        !MASTER RECEIVER STATE SERVICE ROUTINES
                        !*********************************************************
                        !*********************************************************


                        !-------------------------------------------------------
                        ! STATE : 40, Previous state was STATE 08 or STATE 10,
                        !         SLA+R have been transmitted, ACK received.
                        ! ACTION: DATA will be received, ACK returned.
                        !-------------------------------------------------------
                        .sect mrs40
                        .base 0x140

0140 75D8C5                       mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                      !clr STA,STO,SI set AA
0143 01CB                         ajmp RETmr


                        !-------------------------------------------------------
                        ! STATE : 48, SLA+R have been transmitted, NOT ACK
                        !         received.
                        ! ACTION: STOP condition will be generated.
                        !-------------------------------------------------------
                        .sect mrs48
                        .base 0x148

0148 75D8D5             STOP:     mov S1CON,#ENS1_NOTSTA_STO_NOTSI_AA_CR0
                                                        !set STO, clr SI
014B D0D0                         pop psw
014D 32                           reti
```

```
                        !-------------------------------------------------------
                        ! STATE : 50, DATA have been received, ACK returned.
                        ! ACTION: Read DATA of S1DAT.
                        !         DATA will be received, if it is last
                        !         DATA then NOT ACK wil be returned else ACK
                        !         will be returned.
                        !-------------------------------------------------------
                        .sect mrs50
                        .base 0x150

0150 75D018                       mov psw,#SELRB3
0153 A6DA                         mov @r0,S1DAT              !Read received DATA
0155 01C0                         ajmp REC1


                        .sect mrs50s
                        .base 0xc0

00C0 D55205             REC1:     djnz NUMBYTMST,NOTLDAT2
00C3 75D8C1                       mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_NOTAA_CR0
                                                           !clr SI,AA
00C6 8003                         sjmp RETmr
00C8 75D8C5             NOTLDAT2: mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                           !clr SI, set AA
00CB 08                 RETmr:    inc r0
00CC D0D0                         pop psw
00CE 32                           reti



                        !-------------------------------------------------------
                        ! STATE : 58, DATA have been received, NOT ACK
                        !         returned.
                        ! ACTION: Read DATA of S1DAT and generate a STOP
                        !         condition.
                        !-------------------------------------------------------
                        .sect mrs58
                        .base 0x158
0158 75D018                       mov psw,#SELRB3
015B A6DA                         mov @r0,S1DAT
015D 80E9                         sjmp STOP
```

```
                        !*************************************************************
                        !*************************************************************
                        ! SLAVE RECEIVER STATE SERVICE ROUTINES
                        !*************************************************************
                        !*************************************************************

                        !-------------------------------------------------------------
                        ! STATE : 60, Own SLA+W have been received,ACK
                        !         returned.
                        ! ACTION: DATA will be received and ACK returned.
                        !-------------------------------------------------------------
                        .sect srs60
                        .base 0x160
0160 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                        !clr SI, set AA
0163 75D018                     mov psw,#SELRB3
0166 01D0                       ajmp INITSRD
                        .sect insrd
                        .base 0xd0

00D0 7840               INITSRD:    mov r0,#SRD
00D2 7908                           mov r1,#8
00D4 D0D0                           pop psw
00D6 32                             reti

                        !-------------------------------------------------------------
                        ! STATE : 68, Abitration lost in SLA and R/W as MST
                        !         Own SLA+W have been received, ACK returned
                        ! ACTION: DATA will be received and ACK returned.
                        !         STA is set to restart MST mode after the
                        !         bus is free again.
                        !-------------------------------------------------------------
                        .sect srs68
                        .base 0x168
0168 75D8E5                     mov S1CON,#ENS1_STA_NOTSTO_NOTSI_AA_CR0
016B 75D018                     mov psw,#SELRB3
016E 01D0                       ajmp INITSRD

                        !-------------------------------------------------------------
                        ! STATE : 70, General call has been received, ACK
                        !         returned.
                        ! ACTION: DATA will be received and ACK returned.
                        !-------------------------------------------------------------
                        .sect srs70
                        .base 0x170

0170 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                        !clr SI, set AA
0173 75D018                     mov psw,#SELRB3         !Initialize SRD counter
0176 01D0                       ajmp INITSRD

                        !-------------------------------------------------------------
                        ! STATE : 78, Arbitration lost in SLA+R/W as MST.
                        !         General call has been received, ACK returned.
                        ! ACTION: DATA will be received and ACK returned.
                        !         STA is set to restart MST mode after the
                        !         bus is free again.
                        !-------------------------------------------------------------
                        .sect srs78
                        .base 0x178

0178 75D8E5                     mov S1CON,#ENS1_STA_NOTSTO_NOTSI_AA_CR0
017B 75D018                     mov psw,#SELRB3         !Initialize SRD counter
017E 01D0                       ajmp INITSRD
```

```
                        !-------------------------------------------------------
                        ! STATE : 80, Previously addressed with own SLA.
                        !         DATA received, ACK returned.
                        ! ACTION: read DATA.
                        !         IF received DATA was the last THEN superfluous
                        !         DATA will be received and NOT ACK returned ELSE
                        !         next DATA will be received and ACK returned.
                        !-------------------------------------------------------
                        .sect srs80
                        .base 0x180

0180 75D018                       mov psw,#SELRB3
0183 A6DA                         mov @r0,S1DAT              !Read received DATA
0185 01D8                         ajmp REC2

                        .sect srs80s
                        .base 0xd8

00D8 D906               REC2:     djnz r1,NOTLDAT3
00DA 75D8C1             LDAT:     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_NOTAA_CR0
                                                       !clr SI,AA
00DD D0D0                         pop psw
00DF 32                           reti
00E0 75D8C5             NOTLDAT3: mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                       !clr SI, set AA
00E3 08                           inc r0
00E4 D0D0               RETsr:    pop psw
00E6 32                           reti



                        !-------------------------------------------------------
                        ! STATE : 88, Previously adressed with own SLA.
                        !         DATA received NOT ACK returned.
                        ! ACTION: No save of DATA, Enter NOT adressed SLV
                        !         mode. Recognition of own SLA. General call
                        !         recognized, if S1ADR.0=1.
                        !-------------------------------------------------------
                        .sect srs88
                        .base 0x188

0188 75D8C5                       mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                          !clr SI, set AA
018B 01E4                         ajmp RETsr

                        !-------------------------------------------------------
                        ! STATE : 90, Previously addressed with general call.
                        !         DATA has been received, ACK has been returned.
                        ! ACTION: Read DATA.
                        !         After General call only one byte will be
                        !         received with ACK the second DATA will be
                        !         received with NOT ACK.
                        !         DATA will be received and NOT ACK
                        !         returned.
                        !-------------------------------------------------------
                        .sect srs90
                        .base 0x190

0190 75D018                       mov psw,#SELRB3
0193 A6DA                         mov @r0,S1DAT              !Read received DATA
0195 01DA                         ajmp LDAT
```

```
                        !-------------------------------------------------------
                        ! STATE : 98, Previously adressed with general call.
                        !         DATA has been received, NOT ACK has been
                        !         returned.
                        ! ACTION: No save of DATA, Enter NOT adressed SLV
                        !         mode. Recognition of own SLA. General call
                        !         recognized, if S1ADR.0=1.
                        !-------------------------------------------------------
                        .sect srs98
                        .base 0x198

0198 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                          !clr SI, set AA
019B D0D0                       pop psw
019D 32                         reti


                        !-------------------------------------------------------
                        ! STATE : A0, A STOP condition or repeated START has
                        !         been received, while still adressed as
                        !         SLV/REC or SLV/TRX.
                        ! ACTION: No save of DATA, Enter NOT adressed SLV
                        !         mode. Recognition of own SLA. General call
                        !         recognized, if S1ADR.0=1.
                        !-------------------------------------------------------
                        .sect srsA0
                        .base 0x1a0

01A0 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                          !clr SI, set AA
01A3 D0D0                       pop psw
01A5 32                         reti
```

```
                        !*********************************************************
                        !*********************************************************
                        ! SLAVE TRANSMITTER STATE SERVICE ROUTINES
                        !*********************************************************
                        !*********************************************************

                        !---------------------------------------------------------
                        ! STATE : A8, Own SLA+R received, ACK returned.
                        ! ACTION: DATA will be transmitted, A bit received.
                        !---------------------------------------------------------
                        .sect stsa8
                        .base 0x1a8

01A8 8548DA                     mov S1DAT,STD               !load DATA in S1DAT
01AB 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                            !clr SI, set AA
01AE 01E8                       ajmp INITBASE2

                        .sect ibase2
                        .base 0xe8
00E8 75D018             INITBASE2:  mov psw,#SELRB3
00EB 7948                       mov r1, #STD
00ED 09                         inc r1
00EE D0D0                       pop psw
00F0 32                         reti

                        !---------------------------------------------------------
                        ! STATE : B0, Arbitration lost in SLA and R/W as MST.
                        !         Own SLA+R received, ACK returned.
                        ! ACTION: DATA will be transmitted, A bit received.
                        !         STA is set to restart MST mode after the
                        !         bus is free again.
                        !---------------------------------------------------------
                        .sect stsb0
                        .base 0x1b0

01B0 8548DA                     mov S1DAT,STD               !load DATA in S1DAT
01B3 75D8E5                     mov S1CON,#ENS1_STA_NOTSTO_NOTSI_AA_CR0
01B6 01E8                       ajmp INITBASE2

                        !---------------------------------------------------------
                        ! STATE : B8, DATA has been transmitted,ACK received.
                        ! ACTION: DATA will be transmitted, ACK bit is received.
                        !---------------------------------------------------------
                        .sect stsb8
                        .base 0x1b8

01B8 75D018                     mov psw,#SELRB3
01BB 87DA                       mov S1DAT,@r1
01BD 01F8                       ajmp SCON

                        .sect scn
                        .base 0xf8

00F8 75D8C5             SCON:       mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                                !clr SI, set AA
00FB 09                         inc r1
00FC D0D0                       pop psw
00FE 32                         reti
```

```
                        !-------------------------------------------------------
                        ! STATE : C0, DATA has been transmitted, NOT ACK
                        !         received.
                        ! ACTION: Enter not adressed SLV mode.
                        !-------------------------------------------------------
                        .sect stsc0
                        .base 0x1c0
01C0 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                              !clr SI, set AA
01C3 D0D0                       pop psw
01C5 32                         reti

                        !-------------------------------------------------------
                        ! STATE : C8, Last DATA has been transmitted (AA=0),
                        !         ACK received.
                        ! ACTION: Enter not adressed SLV mode.
                        !-------------------------------------------------------
                        .sect stsc8
                        .base 0x1c8
01C8 75D8C5                     mov S1CON,#ENS1_NOTSTA_NOTSTO_NOTSI_AA_CR0
                                                              !clr SI, set AA
01CB D0D0                       pop psw
01CD 32                         reti


                        !*************************************************************
                        !*************************************************************
                        !END OF SIO1 INTERRUPT ROUTINE
                        !*************************************************************
                        !*************************************************************
```

RESET CIRCUITRY

The reset circuitry for the 83C552 is connected to the reset pin RST. A Schmitt trigger is used at the input for noise rejection (see Figure 37). The output of the Schmitt trigger is sampled by the reset circuitry every machine cycle.

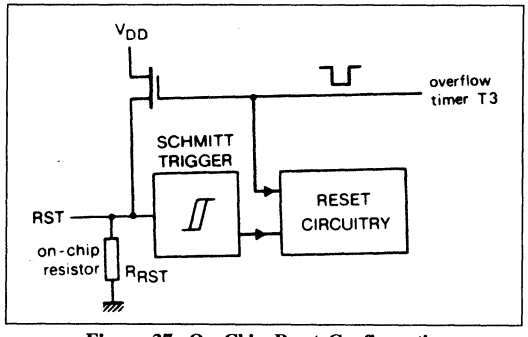


Figure 37. On-Chip Reset Configuration

A reset is accomplished by holding the RST pin HIGH for at least two machine cycles (24 oscillator periods) while the oscillator is running. The CPU responds by executing an internal reset. During reset ALE and $\overline{PSEN}$ output a HIGH level. In order to perform a correct reset, this level must not be affected by external elements. The RST line can also be pulled HIGH internally by a pull-up transistor activated by the watchdog timer T3. The length of the output pulse from T3 is 3 machine cycles. A pulse of such short duration is necessary in order to recover from a processor or system fault as fast as possible.

Note that the short reset pulse from Timer T3 cannot discharge the power-on reset capacitor (see Figure 38). Consequently, when the watchdog timer is also used to reset external devices this capacitor arrangement should not be connected to the RST pin, and a different circuit should be used to perform the power-on reset operation. A Timer T3 overflow, if enabled, will force a reset condition to the 83C552 by an internal connection, whether the output RTS is tied LOW or not.

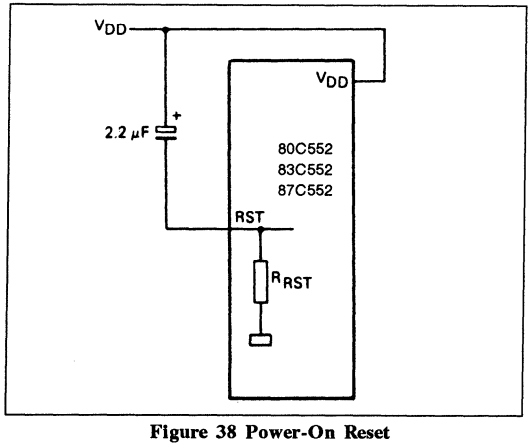


Figure 38 Power-On Reset

The internal reset is executed during the second cycle in which RST is HIGH and is repeated every cycle until RST goes low. It leaves the internal registers as follows:

Register	Content	
ACC	0000	0000
ADCON	xx00	0000
ADCH	xxxx	xxxx
B	0000	0000
CML0 – CML2	0000	0000
CMH0 – CMH2	0000	0000
CTCON	0000	0000
CTL0 – CTL3	xxxx	xxxx
CTH0 – CTH3	xxxx	xxxx
DPL	0000	0000
DPH	0000	0000
IEN0	0000	0000
IEN1	0000	0000
IP0	x000	0000
IP1	0000	0000
PCH	0000	0000
PCL	0000	0000
PCON	0xx0	0000
PSW	0000	0000
PWM0	0000	0000
PWM1	0000	0000
PWMP	0000	0000
P0 - P4	1111	1111
P5	xxxx	xxxx
RTE	0000	0000
SOBUF	xxxx	xxxx
SOCON	0000	0000
S1ADR	0000	0000
S1CON	x000	0000
S1DAT	0000	0000
S1STA	1111	1000
SP	0000	0111
STE	1100	0000
TCON	0000	0000
TH0, TH1	0000	0000
TMH2	0000	0000
TL0, TL1	0000	0000
TML2	0000	0000
TMOD	0000	0000
TM2CON	0000	0000
TM2IR	0000	0000
T3	0000	0000

The internal RAM is not affected by reset. At power-on, the RAM content is indeterminate.

INTERRUPTS

The 83C552 has fifteen interrupt sources, each of which can be assigned one of two priority levels, as shown in Figure 39. The five interrupt sources common to the 80C51 are the external interrupts ($\overline{INT0}$ and $\overline{INT1}$), the timer 0 and timer 1 interrupts (IT0 and IT1) and the serial I/O interrupt (RI or TI). In the 83C552, the standard serial interrupt is called SIO0. Since the subsystems which create these interrupts are identical on both parts, their functionality is likewise identical. The only differences are the locations of the enable and priority register configurations and the priority structure. This is detailed below along with the specifics of the interrupts unique to the 83C552.

The eight Timer T2 interrupts are generated by flags CTI0–CT13, CMI0–CMI2 and by the logical OR of flags T2OV and T2BO. Flags CTI0 to CT13 are set by input signals CT0I to CT3I. Flags CMI0 to CMI2 are set when a match occurs between Timer T2 and the compare registers CM0, CM1 and CM2. When an 8-bit or 16-bit overflow occurs, flags T2BO and T2OV are set respectively. These nine flags are not cleared by hardware and must be reset by software to avoid recurring interrupts.

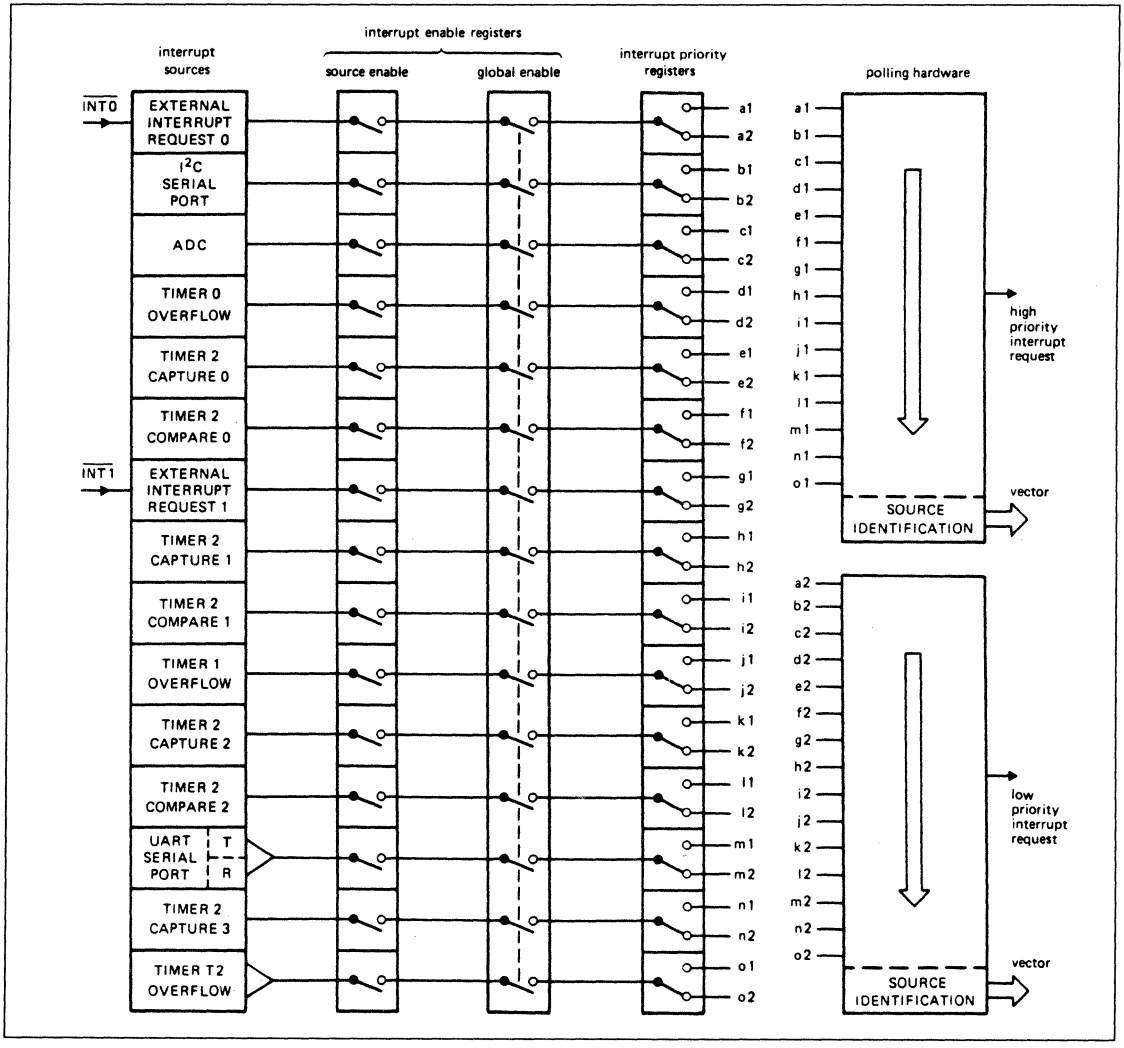


Figure 39. The Interrupt System

The ADC interrupt is generated by the ADCI flag in the ADC control register (ADCON). This flag is set when an ADC conversion result is ready to be read. ADCI is not cleared by hardware and must be reset by software to avoid recurring interrupts.

The SIO1 (I^2C) interrupt is generated by the SI flag in the SIO1 control register (S1CON). This flag is set when S1STA is loaded with a valid status code.

The ADCI flag may be reset by software. It cannot be set by software. All other flags that generate interrupts may be set or cleared by software and the effect is the same as setting or resetting the flags by hardware. Thus, interrupts may be generated by software and pending interrupts can be cancelled by software.

Interrupt Enable Registers

Each interrupt source can be individually enabled or disabled by setting or clearing a bit in the interrupt enable special function registers IEN0 and IEN1. All interrupt sources can also be globally enabled or disabled by setting or clearing bit EA in IEN0. The interrupt enable registers are described in Figures 40 and 41.

Interrupt Priority Structure

Each interrupt source can be assigned one of two priority levels. Interrupt priority levels are defined by the interrupt priority special function registers IP0 and IP1. IP0 and IP1 are described in Figure 42 and 43.

Interrupt priority levels are as follows:
'0' – low priority
'1' – high priority

	7	6	5	4	3	2	1	0
IEN0 (A8H)	EA	EAD	ES1	ES0	ET1	EX1	ET0	EX0
	MSB							LSB

Bit	Symbol	Function
IEN0.7	EA	Global enable/disable control 0 = No interrupt is enabled 1 = Any individually enabled interrupt will be accepted
IEN0.6	EAD	Enable ADC interrupt
IEN0.5	ES1	Enable SIO1 (I^2C) interrupt
IEN0.4	ES0	Enable SIO0 (UART) interrupt
IEN0.3	ET1	Enable Timer 1 interrupt
IEN0.2	EX1	Enable External interrupt 1
IEN0.1	ET0	Enable Timer 0 interrupt
IEN0.0	EX0	Enable External interrupt 0

Figure 40. Interrupt Enable Register (IEN0)

	7	6	5	4	3	2	1	0
IEN1 (E8H)	ET2	ECM2	ECM1	ECM0	ECT3	ECT2	ECT1	ECT0
	MSB							LSB

Bit	Symbol	Function
IEN1.7	ET2	Enable T2 overflow interrupt(s)
IEN1.6	ECM2	Enable T2 comparator 2 interrupt
IEN1.5	ECM1	Enable T2 comparator 1 interrupt
IEN1.4	ECM0	Enable T2 comparator 0 interrupt
IEN1.3	ECT3	Enable T2 capture register 3 interrupt
IEN1.2	ECT2	Enable T2 capture register 2 interrupt
IEN1.1	ECT1	Enable T2 capture register 1 interrupt
IEN1.0	ECT0	Enable T2 capture register 0 interrupt

In all cases, if the enable bit is '0', then the interrupt is disabled and if the enable bit is '1', then the interrupt is enabled.

Figure 41. Interrupt Enable Register (IEN1)

	7	6	5	4	3	2	1	0
IP0 (B8H)	—	PAD	PS1	PS0	PT1	PX1	PT0	PX0
	MSB							LSB

Bit	Symbol	Function
IP.7	-	Unused
IP.6	PAD	ADC interrupt priority level
IP.5	PS1	SIO1 (I^2C) interrupt priority level
IP.4	PS0	SIO0 (UART) interrupt priority level
IP.3	PT1	Timer 1 interrupt priority level
IP.2	PX1	External interrupt 1 priority level
IP.1	PT0	Timer 0 interrupt priority level
IP.0	PX0	External interrupt 0 priority level

Figure 42. Interrupt Priority Register (IP0)

	7	6	5	4	3	2	1	0
IP1 (F8H)	PT2	PCM2	PCM1	PCM0	PCT3	PCT2	PCT1	PCT0

Bit	Symbol	Function
IP1.7	PT2	T2 overflow interrupt(s) priority level
IP1.6	PCM2	T2 comparator 2 interrupt priority level
IP1.5	PCM1	T2 comparator 1 interrupt priority level
IP1.4	PCM0	T2 comparator 0 interrupt priority level
IP1.3	PCT3	T2 capture register 3 interrupt priority level
IP1.2	PCT2	T2 capture register 2 interrupt priority level
IP1.1	PCT1	T2 capture register 1 interrupt priority level
IP1.0	PCT0	T2 capture register 0 interrupt priority level

Figure 43. Interrupt Priority Register (IP1)

A low priority interrupt may be interrupted by a high priority interrupt. A high priority interrupt cannot be interrupted by any other interrupt source. If two requests of different priority occur simultaneously, the high priority level request is serviced. If requests of the same priority are received simultaneously, an internal polling sequence determines which request is serviced. Thus, within each priority level, there is a second priority structure determined by the polling sequence. This second priority structure is shown in Table 15.

Table 15. Interrupt Priority Structure

Source	Name	Priority Within Level
External interrupt 0	X0	(highest)
SIO1 (I^2C)	S1	
ADC completion	ADC	
Timer 0 overflow	T0	
T2 capture 0	CT0	
T2 compare 0	CM0	
External interrupt 1	X1	
T2 capture 1	CT1	
T2 compare 1	CM1	
Timer 1 overflow	T1	
T2 capture 2	CT2	
T2 compare 2	CM2	
SIO0 (UART)	S0	
T2 capture 3	CT3	
Timer T2 overflow	T2	(lowest)

The above Priority Within Level structure is only used when there are simultaneous requests of the same priority level.

Interrupt Handling

The interrupt sources are sampled at S5P2 of every machine cycle. The samples are polled during the following machine cycle. If one of the flags was in a set condition at S5P2 of the previous machine cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this hardware generated LCALL is not blocked by any of the following conditions:

1. An interrupt of higher or equal priority level is already in progress.
2. The current machine cycle is not the final cycle in the execution of the instruction in progress. (no interrupt request will be serviced until the instruction in progress is completed).
3. The instruction in progress is RETI or any access to the interrupt priority or interrupt enable registers. (No interrupt will be serviced after RETI or after a read or write to IP0, IP1, IE0, or IE1 until at least one other instruction has been subsequently executed).

The polling cycle is repeated with every machine cycle, and the values polled are the values present at S5P2 of the previous machine cycle. Note that if an interrupt flag is active but is not being responded to because of one of the above conditions, and if the flag is inactive when the blocking condition is removed, then the blocked interrupt will not be serviced. Thus, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle is new.

The processor acknowledges an interrupt request by executing a hardware-generated LCALL to the appropriate service routine. In some cases it also clears the flag which generated the interrupt, and in others it does not. It clears the Timer 0, Timer 1 and external interrupt flags. An external interrupt flag (IE0 or IE1) is cleared only if it was transition-activated. All other interrupt flags are not cleared by hardware and must be cleared by the software. The LCALL pushes the contents of the program counter on to the stack (but it does not save the PSW) and reloads the PC with an address that depends on the source of the interrupt being vectored to as shown in Table 16.

Table 16. Interrupt Vector Addresses

Source	Name	Vector Address
External interrupt 0	X0	0003H
Timer 0 overflow	T0	000BH
External interrupt 1	X1	0013H
Timer 1 overflow	T1	001BH
SIO0 (UART)	S0	0023H
SIO1 (I^2C)	S1	002BH
T2 capture 0	CT0	0033H
T2 capture 1	CT1	003BH
T2 capture 2	CT2	0043H
T2 capture 3	CT3	004BH
ADC completion	ADC	0053H
T2 compare 0	CM0	005BH
T2 compare 1	CM1	0063H
T2 compare 2	CM2	006BH
T2 overflow	T2	0073H

Execution proceeds from the vector address until the RETI instruction is encountered. The RETI instruction clears the 'priority level active' flip-flop that was set when this interrupt was acknowledged. It then pops the top two bytes from the stack and reloads the program counter. Execution of the interrupted program continues from where it was interrupted.

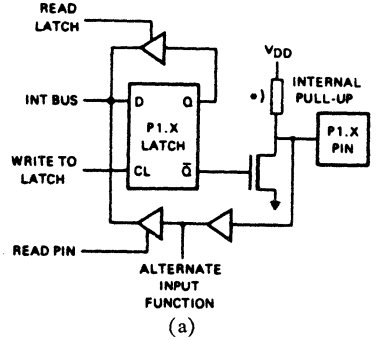
I/O PORT STRUCTURE

The 83C552 has six 8-bit ports. Each port consists of a latch (special function registers P0 to P5), an input buffer and an output driver (Port 0 to 4 only). Ports 0–3 are the same as in the 80C51, with the exception of the additional functions of Port 1. The parallel I/O function of Port 4 is equal to that of Ports 1, 2 and 3. Port 5 may be used as an input port only.

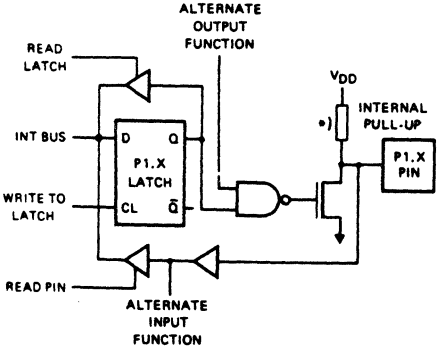
Figure 44 shows the bit latch and I/O buffer functional diagrams of the unique 83C552 ports. A bit latch corresponds to one bit in a port's SFR and is represented as a D Type flip-flop. A 'write to latch' signal from the CPU latches a bit from the internal bus and a 'read latch' signal from the CPU places the Q output of the flip–flop on the internal bus. A 'read pin' signal from the CPU places the actual port pin level on the internal bus. Some instructions that read a port read the actual port pin levels and other instructions read the latch (SFR) contents.

PORT 1 OPERATION

Port 1 operates the same as it does in the 8051 with the exception of port lines P1.6 and P1.7 which may be selected as the SCL and SDA lines of serial port SIO1 (I2C). Because the I2C bus may be active while the device is disconnected from V_{DD}, these pins are provided with open drain drivers. Therefore pins P1.6 and P1.7 do not have internal pull-ups.

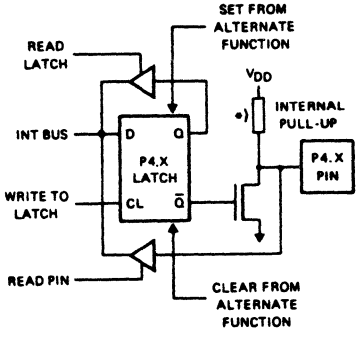


(a)

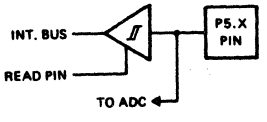


(b)

NOTE:
Pull-up not present on P1.6 and P1.7.
*Two period active pull-up as in the 80C51.



(c)



(d)

Figure 44. Port Bit Latches and I/O Buffers

PORT 5 OPERATION

Port 5 may be used to input up to 8 analog signals to the ADC. Unused ADC inputs may be used to input digital inputs. These inputs have an inherent hysteresis to prevent the input logic from drawing excessive current from the power lines when driven by analog signals. Channel to channel crosstalk (Ct) should be taken into consideration when both analog and digital signals are simultaneously input to Port 5 (see D.C. characteristics in data sheet).

Port 5 is not bidirectional and may not be configured as an output port. All six ports are multifunctional and their alternate functions are listed in Table 17. A more detailed description of these features can be found in the relevant parts of this section.

PULSE WIDTH MODULATED OUTPUTS

The 83C552 contains two pulse width modulated output channels (see figure 45). These channels generate pulses of programmable length and interval. The repetition frequency is defined by an 8-bit prescaler PWMP which supplies the clock for the counter. The prescaler and counter are common to both PWM channels. The 8-bit counter counts modulo 255 i.e. from 0 to 254 inclusive. The value of the 8-bit counter is compared to the contents of two registers: PWM0 and PWM1. Provided the contents of either of these registers is greater than the counter value, the corresponding $\overline{\text{PWM0}}$ or $\overline{\text{PWM1}}$ output is set LOW. If the contents of these registers are equal to, or less than the counter value, the output will be HIGH. The pulse-width-ratio is therefore defined by the contents of the registers PWM0 and PWM1. The pulse-width-ratio is in the range of 0 to 1 and may be programmed in increments of 1/255.

Buffered PWM outputs may be used to drive DC motors. The rotation speed of the motor would be proportional to the contents of PWMn. The PWM outputs may also be configured as a dual DAC. In this application, the PWM outputs must be integrated using conventional operational amplifier circuitry. If the resulting output voltages have to be accurate, external buffers with their own analog supply should be used to buffer the PWM outputs before they are integrated. The repetition frequency fpwm, at the PWMn outputs is given by:

$$\text{fpwm} = \frac{\text{fosc}}{2 \times (1 + \text{PWMP}) \times 255}$$

This gives a repetition frequency range of 92Hz to 23.5kHz (fosc = 12MHz). By loading the PWM registers with either 00H or FFH, the PWM channels will output a constant HIGH or LOW level respectively. Since the 8-bit counter counts modulo 255, it can never actually reach the value of the PWM registers when they are loaded with FFH.

Table 17. Input/Output Ports

Port Pin	Alternate Function	
P0.0	AD0	Multiplexed lower order address/data bus used during external memory accesses
P0.1	AD1	
P0.2	AD2	
P0.3	AD3	
P0.4	AD4	
P0.5	AD5	
P0.6	AD6	
P0.7	AD7	
P1.0	CT0I	Capture timer input signals for timer T2
P1.1	CT1I	
P1.2	CT2I	
P1.3	CT3I	
P1.4	T2	T2 event input
P1.5	RT2	T2 timer reset signal. Rising edge triggered
P1.6	SCL	Serial port clock line I^2C bus
P1.7	SDA	Serial port data line I^2C bus
P2.0	A8	High order address byte used during external memory accesses
P2.1	A9	
P2.2	A10	
P2.3	A11	
P2.4	A12	
P2.5	A13	
P2.6	A14	
P2.7	A15	
P3.0	RXD	Serial input port (UART)
P3.1	$\overline{\text{TXD}}$	Serial output port (UART)
P3.2	$\overline{\text{INT0}}$	External interrupt 0
P3.3	$\overline{\text{INT1}}$	External interrupt 1
P3.4	T0	Timer 0 external input
P3.5	$\overline{\text{T1}}$	Timer 1 external input
P3.6	$\overline{\text{WR}}$	External data memory write strobe
P3.7	$\overline{\text{RD}}$	External data memory read strobe
P4.0	CMSR0	Timer T2: compare and set/reset outputs on a match with timer T2
P4.1	CMSR1	
P4.2	CMSR2	
P4.3	CMSR3	
P4.4	CMSR4	
P4.5	CMSR5	
P4.6	CMT0	Timer T2: compare and toggle outputs on a match with timer T2
P4.7	CMT1	
P5.0	ADC0	Eight analogue ADC inputs
P5.1	ADC1	
P5.2	ADC2	
P5.3	ADC3	
P5.4	ADC4	
P5.5	ADC5	
P5.6	ADC6	
P5.7	ADC7	

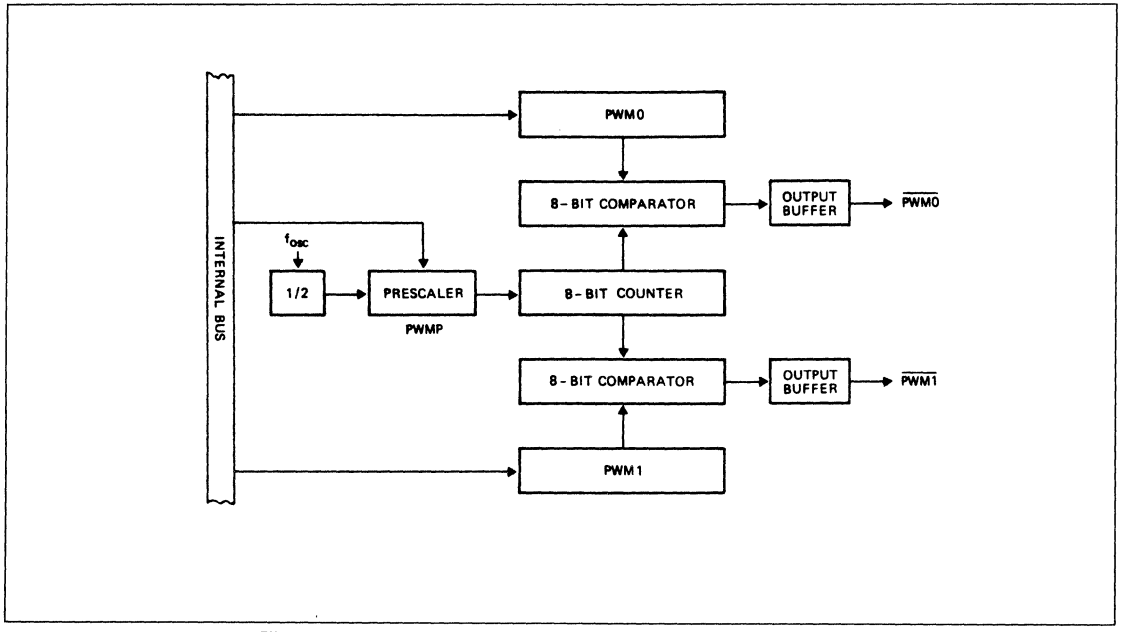


Figure 45. Functional Diagram of Pulse Width Modulated Outputs

When a compare register (PWM0 or PWM1) is loaded with a new value, the associated output is updated immediately. It does not have to wait until the end of the current counter period. Both $\overline{PWMn}$ output pins are driven by push-pull drivers. These pins are not used for any other purpose.

Prescaler frequency control register PWMP

PWMP (FEH)	7	6	5	4	3	2	1	0
	MSB							LSB

PWMP.0–7 Prescaler division factor = PWMP + 1.

Reading PWMP gives the current reload value. The actual count of the prescaler cannot be read.

PWM0 (FCH) PWM1 (FDH)	7	6	5	4	3	2	1	0
	MSB							LSB

PWM0/1.0–7} Low/high ratio of $\overline{PWMn}$ =

$$\frac{(PWMn)}{255 - (PWMn)}$$

ANALOG-TO-DIGITAL CONVERTER

The analog input circuitry consists of an 8-input analog multiplexer and a 10-bit, straight binary, successive approximation ADC. The analog reference voltage and analog power supplies are connected via separate input pins. The conversion takes 50 machine cycles i.e. 50µs at an oscillator frequency of 12MHz. Input voltage swing is from 0V to + 5V. Because the internal DAC employs a ratiometric potentiometer, there are no discontinuities in the converter characteristic. Figure 46 shows a functional diagram of the analog input circuitry.

Analog-to-Digital Conversion

Figure 47 shows the elements of a successive approximation (SA) ADC. The ADC contains a DAC which converts the contents of a successive approximation register to a voltage (VDAC) which is compared to the analog input voltage (Vin). The output of the comparator is fed to the successive approximation control logic which controls the successive approximation register. A conversion is initiated by setting ADCS in the ADCON register. ADCS can be set by software only or by either hardware or software

The software only start mode is selected when control bit ADCON.5 (ADEX) = 0. A conversion is then started by setting control bit ADCON.3 (ADCS). The hardware or software start mode is selected when ADCON.5 = 1 and a conversion may be started by setting ADCON.3 as above or by applying a rising edge to external pin STADC. When a conversion is started by applying a rising edge, a low level must be applied to STADC for at least one machine cycle followed by a high level for at least one machine cycle.

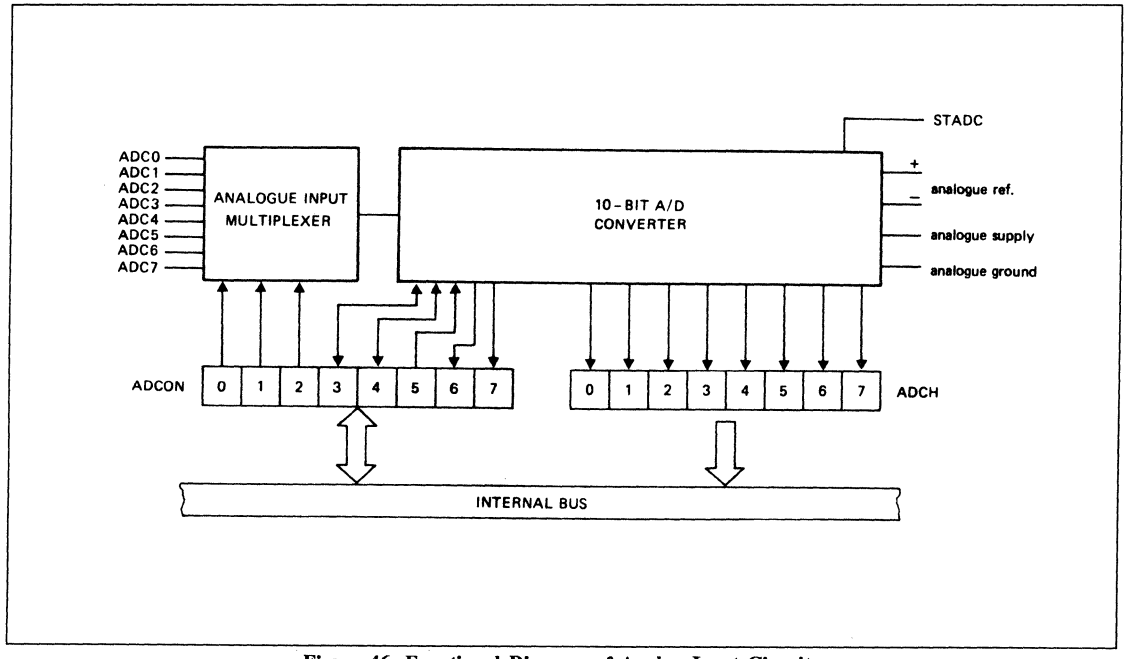


Figure 46. Functional Diagram of Analog Input Circuitry

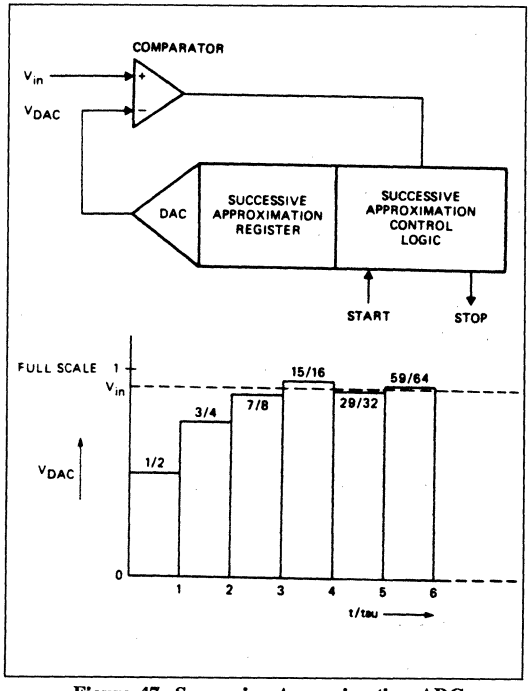


Figure 47. Successive Approximation ADC

The low-to-high transition of STADC is recognized at the end of a machine cycle and the conversion commences at the beginning of the next cycle. When a conversion is initiated by software, the conversion starts at the beginning of the machine cycle which follows the instruction that sets ADCS. ADCS is actually implemented with two flip-flops: a command flip-flop which is affected by set operations, and a status flag which is accessed during read operations.

The next two machine cycles are used to initiate the converter. At the end of the first cycle, the ADCS status flag is set and a value of '1' will be returned if the ADCS flag is read while the conversion is in progress. Sampling of the analog input commences at the end of the second cycle.

During the next eight machine cycles, the voltage at the previously selected pin of Port 5 is sampled and this input voltage should be stable in order to obtain a useful sample. In any event, the input voltage slew rate must be less than 10V/ms in order to prevent an undefined result.

The successive approximation control logic first sets the most significant bit and clears all other bits in the successive approximation register (10 0000 0000B). The output of the DAC (50% full scale) is compared to the input voltage Vin. If the input voltage is greater than VDAC, then the bit remains set; otherwise it is cleared.

The successive approximation control logic now sets the next most significant bit (11 0000 0000B or 01 0000 0000B depending on the previous result) and VDAC is compared to Vin again. If the input voltage is greater than VDAC, then the bit being tested remains set; otherwise the bit being tested is cleared. This process is repeated until all ten bits have been tested at which stage the result of the conversion is held in the successive approximation register. Figure 48 shows a conversion flow chart. The bit pointer identifies the bit under test. The conversion takes four machine cycles per bit.

The end of the 10-bit conversion is flagged by control bit ADCON.4 (ADCI). The upper 8 bits of the result are held in special function register ADCH and the two remaining bits are held in ADCON.7 (ADC.1) and ADCON.6 (ADC.0). The user may ignore the two least significant bits in ADCON and use the ADC as an 8-bit converter (8 upper bits in ADCH). In any event, the total actual conversion time is 50 machine cycles. ADCI will be set and the ADCS status flag will be reset 50 cycles after the command flip-flop (ADCS) is set.

Control bits ADCON.0, ADCON.1 and ADCON.2 are used to control an analog multiplexer which selects one of eight analog channels (see Figure 49). An ADC conversion in progress is unaffected by an external or software ADC start. The result of a completed conversion remains unaffected provided ADCI = logic 1. While ADCS = logic 1 or ADCI = logic 1, a new ADC START will be blocked and consequently lost. An ADC conversion already in progress is aborted when the Idle or Power-down mode is entered. The result of a completed conversion (ADCI = logic 1) remains unaffected when entering the Idle mode.

ADC Resolution and Analog Supply

Figure 50 shows how the ADC is realized. The ADC has its own supply pins (AV_{DD} and AV_{SS}) and two pins (Vref+ and Vref–) connected to each end of the DAC's resistance-ladder. The ladder has 1023 equally spaced taps, separated by a resistance of 'R'. The first tap is located 0.5 x R above Vref– and the last tap is located 1.5 x R below Vref+. This gives a total ladder resistance of 1024 x R. This structure ensures that the DAC is monotonic and results in a symmetrical quantization error as shown in Figure 51.

For input voltages between Vref– and (Vref–) + 1/2 LSB, the 10-bit result of an A/D conversion will be 00 0000 0000B = 000H. For input voltages between (Vref+) – 3/2 LSB and Vref+, the result of a conversion will be 11 1111 1111B = 3FFH. AVref+ and AVref– may be between AV_{DD} + 0.2V and AV_{SS}– 0.2V. AVref+ should be positive with respect to AVref– and the input voltage (Vin) should be between AVref+ and AVref–. If the analog input voltage range is from 2V to 4V, then ten bit resolution can be obtained over this range if AVref+ = 4V and AVref– = 2V.

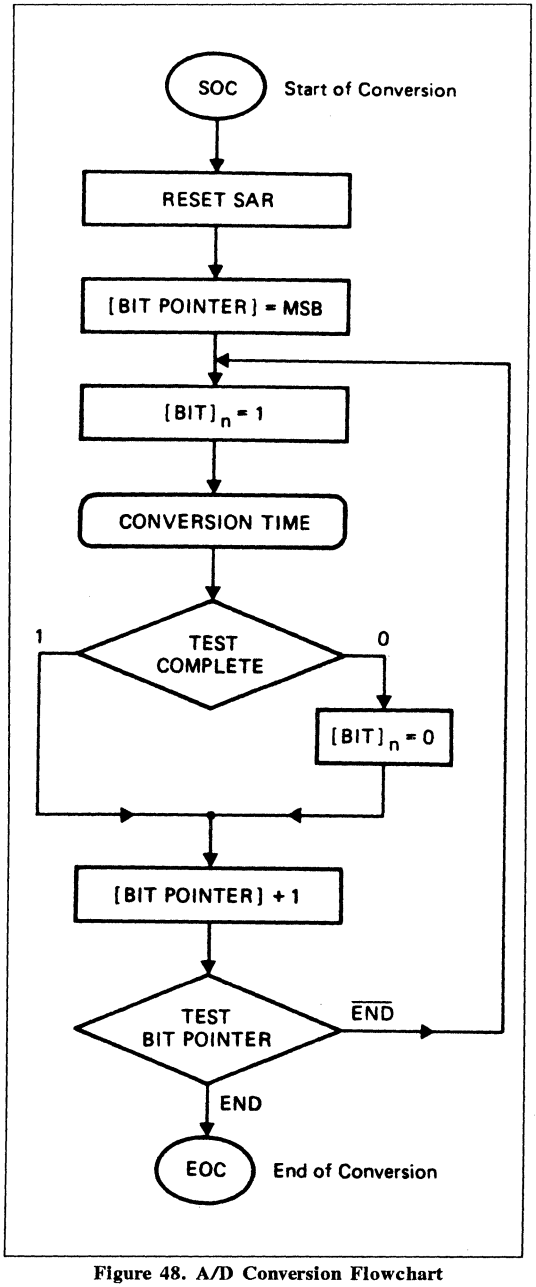


Figure 48. A/D Conversion Flowchart

	7	6	5	4	3	2	1	0
ADCON (C5H)	ADC.1	ADC.0	ADEX	ADCI	ADCS	AADR2	AADR1	AADR0

Bit	Symbol	Function
ADCON.7	ADC.1	Bit 1 of ADC result
ADCON.6	ADC.0	Bit 0 of ADC result
ADCON.5	ADEX	Enable external start of conversion by STADC 0 = Conversion can be started by software only (by setting ADCS) 1 = Conversion can be started by software or externally (by a rising edge on STADC)
ADCON.4	ADCI	ADC interrupt flag: this flag is set when an A/D conversion result is ready to be read. An interrupt is invoked if it is enabled. The flag may be cleared by the interupt service routine. While this flag is set, the ADC can not start a new conversion. ADCI can not be set by software.
ADCON.3	ADCS	ADC start and status: setting this bit starts an A/D conversion. It may be set by software or by the external signal STADC. The ADC logic ensures that this signal is HIGH while the ADC is busy. On completion of the conversion, ADCS is reset at the same time the interrupt flag is set. ADCS can not be reset by software. A new conversion may not be started while either ADCS or ADCI is high.

ADCI	ADCS	ADC Status
0	0	ADC not busy, a conversion can be started
0	1	ADC busy, start of a new conversion is blocked
1	0	Conversion completed, start of a new conversion is blocked
1	1	Not possible

ADCON.2	AADR2	Analogue input select: this binary coded address selects one of the eight analogue port bits of P5 to be input to the converter. It can only be changed when ADCI and ADCS are both LOW.
ADCON.1	AADR1	
ADCON.0	AADR0	

AADR2	AADR1	AADR0	Selected Analogue Channel
0	0	0	ADC0 (P5.0)
0	0	1	ADC1 (P5.1)
0	1	0	ADC2 (P5.2)
0	1	1	ADC3 (P5.3)
1	0	0	ADC4 (P5.4)
1	0	1	ADC5 (P5.5)
1	1	0	ADC6 (P5.6)
1	1	1	ADC7 (P5.7)

Figure 49. ADC Control Register (ADCON)

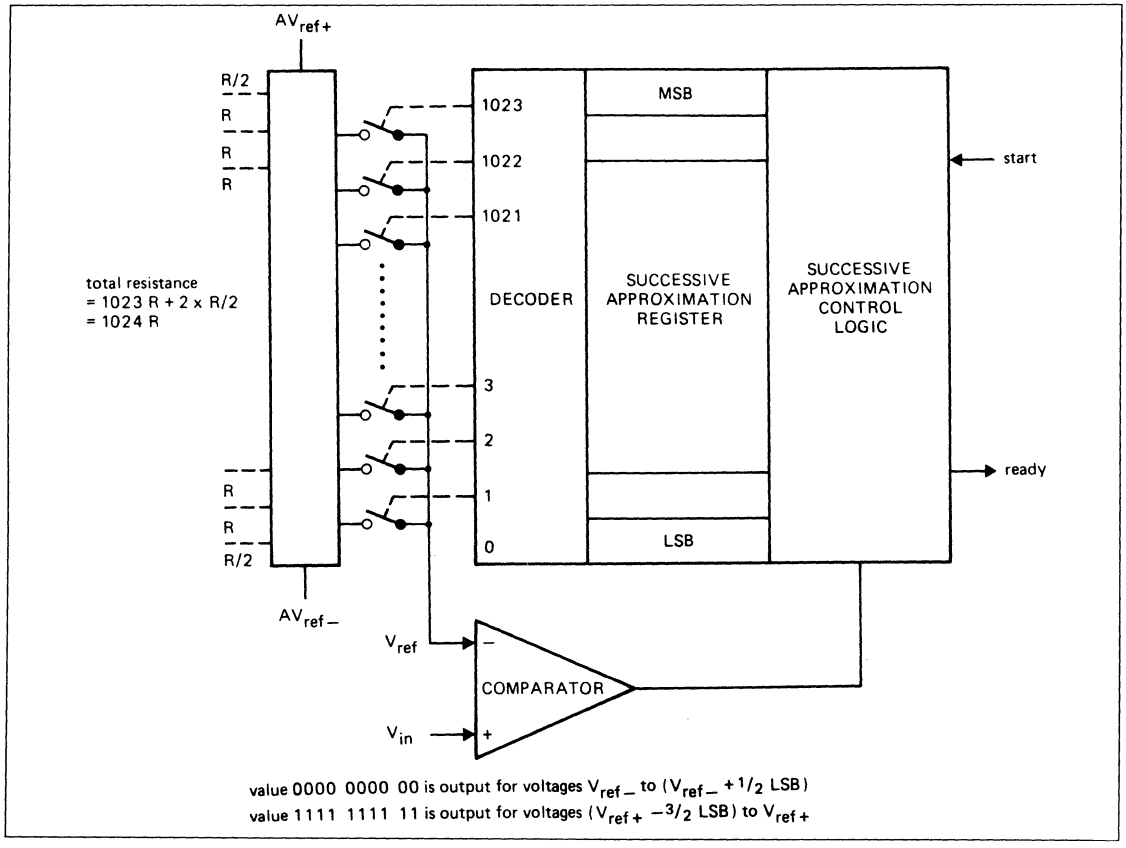


Figure 50. ADC Realization

The result can always be calculated from the following formula:

$$\text{Result} = 1024 \times \frac{\text{Vin} - \text{AVref–}}{\text{AVref+} - \text{AVref–}}$$

POWER REDUCTION MODES

The 83C552 has two reduced power modes of operation: the Idle mode and the Power-down mode. These modes are entered by setting bits in the PCON special function register. When the 83C552 enters the Idle mode, the following functions are disabled:

CPU	(halted)
Timer T2	(halted and reset)
PWM0,PWM1	(reset; outputs are high)
ADC	(conversion aborted if in progress)

In Idle mode, the following functions remain active:
Timer 0
Timer 1
Timer T3
SIO0 SIO1
External interrupts

When the 83C552 enters the Power-down mode, the oscillator is stopped. The Power-down mode is entered by setting the PD bit in the PCON register. The PD bit can only be set if the $\overline{EW}$ input is tied HIGH.

Power-Down Mode

The instruction that sets PCON.1 will be the last instruction executed in the normal operating mode before the Power-down mode is entered. In the Power-down mode, the on-chip oscillator is stopped. This freezes all functions; only the on-chip RAM and special function registers are held. The port pins output the contents of their respective special function registers. A hardware reset is the only way to terminate the Power-down mode. Reset re-defines all the special function registers, but does not change the on-chip RAM.

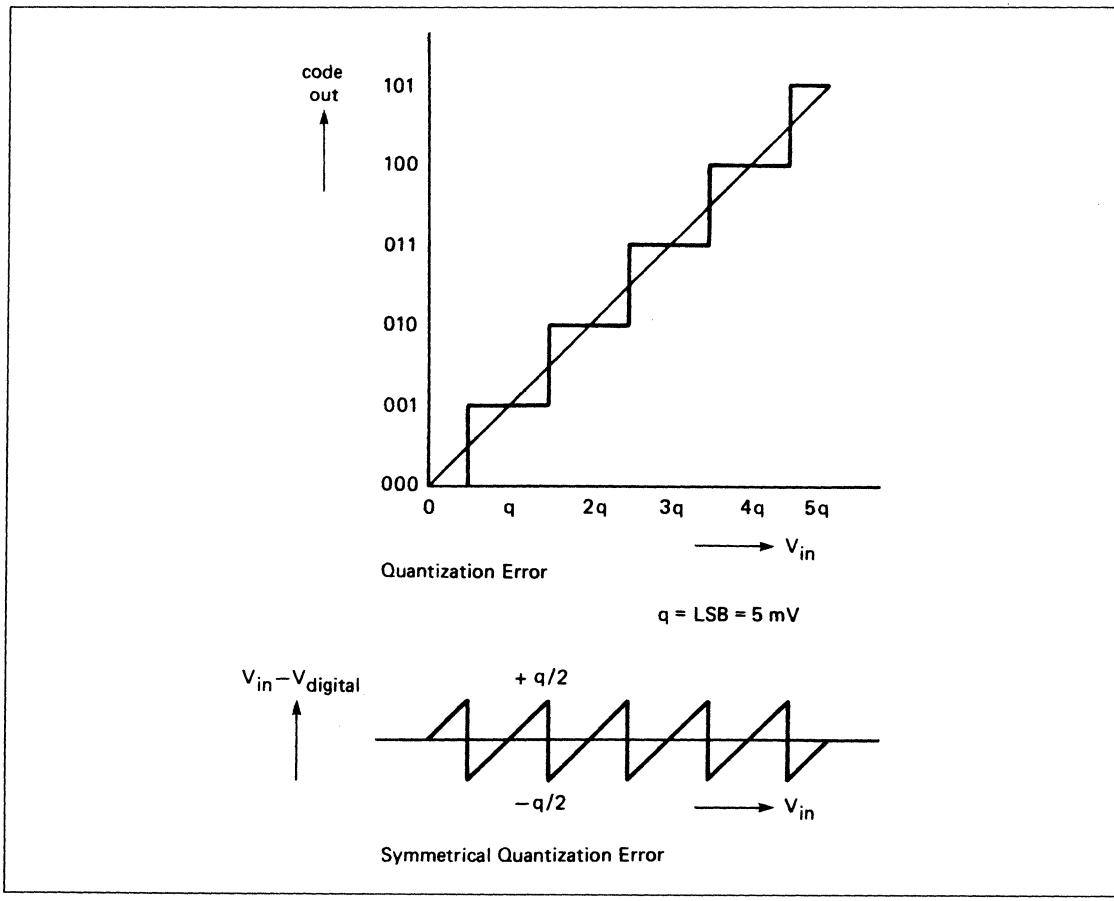


Figure 51. Effective Conversion Characteristic

In the Power-down mode, V_{DD} and AV_{DD} can be reduced to minimize power consumption. V_{DD} and AV_{DD} must not be reduced before the Power-down mode is entered and must be restored to the normal operating voltage before the Power-down mode is terminated. The reset that terminates the Power–down mode also freezes the oscillator. The reset should not be activated before V_{DD} and AV_{DD} are restored to their normal operating level, and must be held active long enough to allow the oscillator to restart and stabilize (normally less than 10ms).

The status of the external pins during Power-down is shown in Table 18. If the Power-down mode is entered while the 83C552 is executing out of external program memory, the port data that is held in the P2 special function register is restored to Port 2. If a port latch contains a '1', the port pin is held HIGH during the Power–down mode by the strong pull-up transistor.

Power Control Register PCON

The Idle and Power-down modes are entered by writing to bits in PCON. PCON is not bit addressable. See Figure 52.

MEMORY ORGANIZATION

The memory organization of the 83C552 is the same as in the 80C51, with the exception that the 83C552 has 8K ROM, 256 bytes RAM and additional SFR's. Addressing modes are the same in the 83C552 and the 80C51. Details of the differences are given in the following paragraphs.

In the 83C552, the lower 8K of the 64K program memory address space is filled by internal ROM. By tying the $\overline{EA}$ pin high, the processor fetches instructions from in-

Table 18. External Pin Status During Idle and Power-Down Modes

mode	memory	ALE	$\overline{PSEN}$	Port 0	Port 1	Port 2	Port 3	Port 4	$\overline{PWM0}$/$\overline{PWM1}$
Idle (1)	internal	1	1	port data	port data	port data	port data	port data	HIGH
Idle (1)	external	1	1	floating	port data	address	port data	port data	HIGH
Power-down	internal	0	0	port data	port data	port data	port data	port data	HIGH
Power-down	external	0	0	floating	port data	port data	port data	port data	HIGH

	7	6	5	4	3	2	1	0
PCON (87H)	SMOD	–	–	WLE	GF1	GF0	PD	IDL

Bit	Symbol	Function
PCON.7	SMOD	Double Baud rate bit. When set to logic 1 the baud rate is doubled when the serial port SIO0 is being used in Modes 1, 2 or 3.
PCON.6	–	(reserved)
PCON.5	–	(reserved)
PCON.4	WLE	Watchdog load enable. This flag must be set by software prior to loading Timer T3 (watchdog timer). It is cleared when Timer T3 is loaded.
PCON.3	GF1	General-purpose flag bit
PCON.2	GF0	General-purpose flag bit
PCON.1	PD	Power-down bit. Setting this bit activates the Power-down mode. It can only be set if input $\overline{EW}$ is high.
PCON.0	IDL	Idle mode bit. Setting this bit activates the Idle mode.

If logic 1s are written to PD and IDL at the same time, PD takes precedence.The reset value of PCON is (0XX00000).

Figure 52. Power Control Register (PCON)

ternal program ROM. Bus expansion for accessing program memory from 8K upwards is automatic since external instruction fetches occur automatically when the program counter exceeds 8191. If the $\overline{EA}$ pin is tied low all program memory fetches are from external memory. The execution speed of the 83C552 is the same regardless of whether fetches are from external or internal program memory. If all storage is on-chip, then byte location 8191 should be left vacant to prevent an undesired pre-fetch from external program memory address 8192.

Certain locations in program memory are reserved for specific programs. Locations 0000H to 0002H are reserved for the initialization program. Following reset, the CPU always begins execution at locations 0000H. Locations 0003H to 0075H are reserved for the fifteen interrupt request service routines.

Functionally, the internal data memory is the most flexible of the address spaces. The internal data memory space is subdivided into a 256-byte internal data RAM address space and a 128-byte special function register (SFR) address space, as shown in Figure 53.

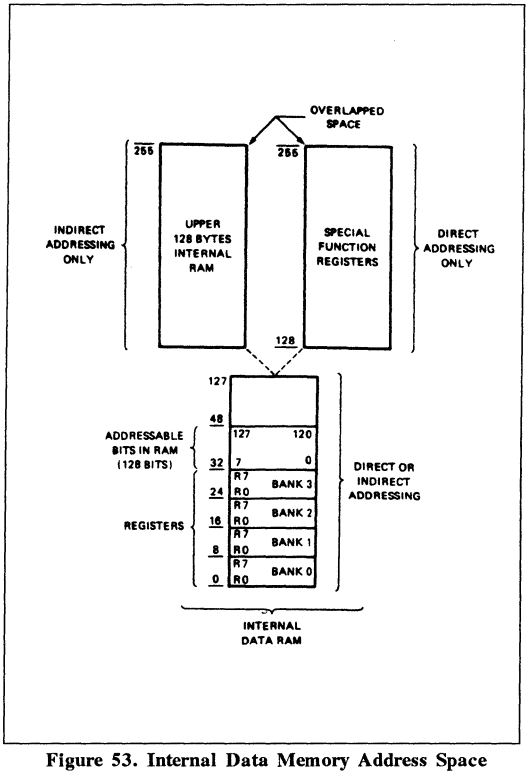


Figure 53. Internal Data Memory Address Space

The internal data RAM address space is 0 to 255. Four 8-bit register banks occupy locations 0 to 31. 128 bit locations of the internal data RAM are accessible through direct addressing. These bits reside in 16 bytes of internal data RAM at locations 20H to 2FH. The stack can be located anywhere in the internal data RAM address space by loading the 8-bit stack pointer. The stack depth may be 256 bytes maximum.

The SFR address space is 128 to 255. All registers except the program counter and the four 8-bit register banks reside in this address space. Memory mapping the SFRs allows them to be accessed as easily as internal RAM and as such, they can be operated on by most instructions. The 56 SFRs are listed in Figure 54 and their mapping in the SFR address space is shown in Figures 55 and 56. RAM bit addresses are the same as in the 80C51 and are summarized in Figure 57. The special function bit addresses are summarized in Figure 58.

ARITHMETIC REGISTERS:
ACCumulator*, B register*,
Program Status Word*

POINTERS:
Stack Pointer,
Data Pointer (High and Low)

PARALLEL I/O PORTS:
Port 5*, Port 4*, Port 3*,
Port 2*, Port 1*, Port 0*

INTERRUPT SYSTEM:
Interrupt Priority 0*,
Interrupt Priority 1*,
Interrupt Enable 0*,
Interrupt Enable 1*

PULSE WIDTH MODULATED O/P's
Pulse Width Modutation Prescaler
Pulse Width Modulation Register 0,
Pulse Width Modulation Register 1

SERIAL I/O PORTS:
Serial 0 CONtrol*, Serial 0 data BUFfer,
Serial 1 CONtrol*, Serial 1 DATa,
Serial 1 STAtus, Serial 1 ADDress, PCON

TIMERS:
Timer MODe, Timer CONtrol*,
Timer Low 0, Timer High 0,
Timer Low 1, Timer High 1,
TiMer T2 CONtrol, TiMer Low 2,
Timer High 2, Timer T3

CAPTURE AND COMPARE LOGIC
CapTure CONtrol,
TiMer T2 Interrupt flag Register,
CapTure Low 0, CapTure High 0,
CapTure Low 1, CapTure High 1,
CapTure Low 2, CapTure High 2,
CapTure Low 3, CapTure High 3,
CoMpare Low 0, CoMpare High 0,
CoMpare Low 1, CoMpare High 1,
CoMpare Low 2, CoMpare High 2
SeT Enable, ReseT Enable

ADC
ADC cONtrol, ADC High byte

Note: * Bit and byte addressable

Figure 54. Special Function Registers

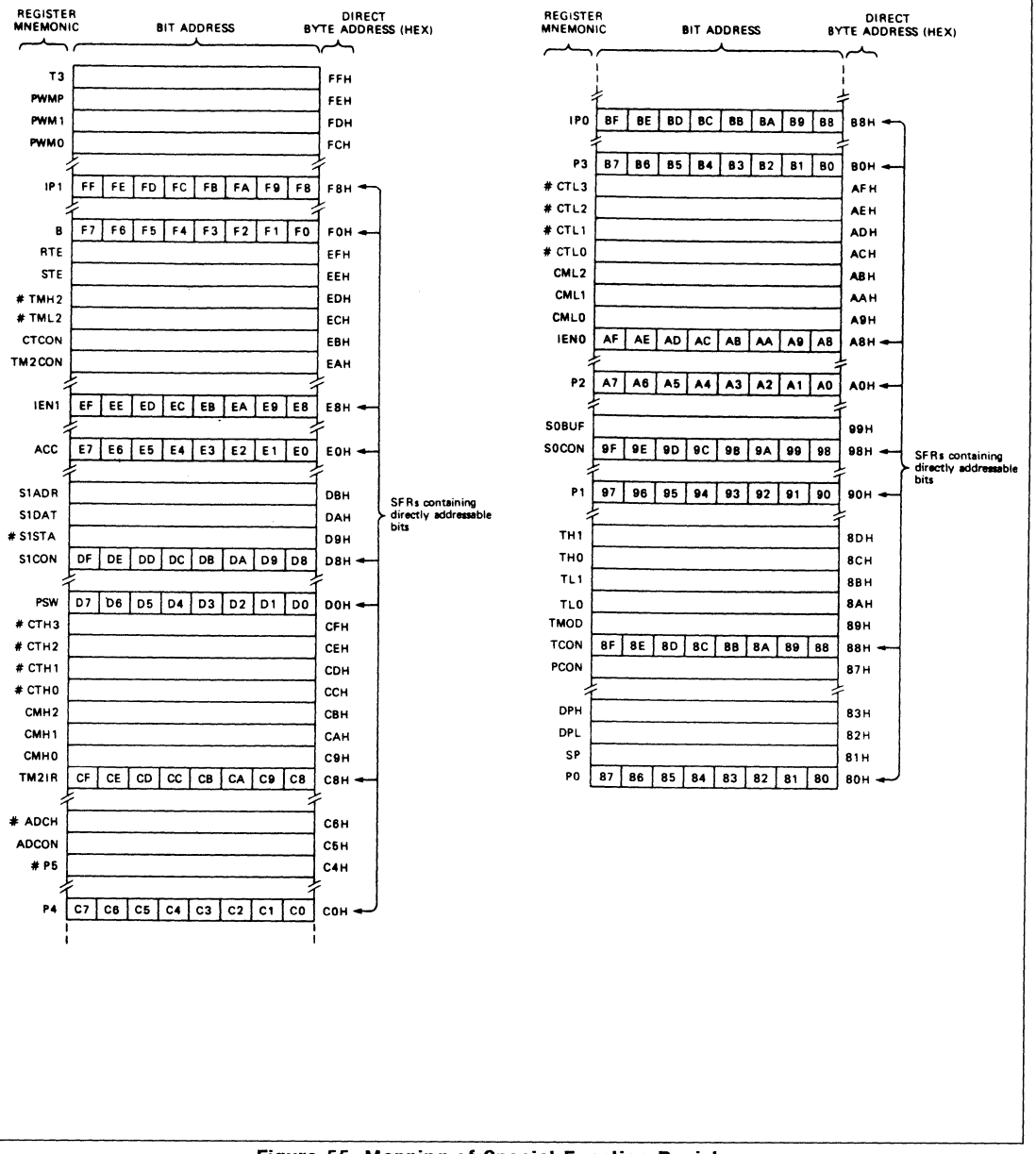


Figure 55. Mapping of Special Function Registers

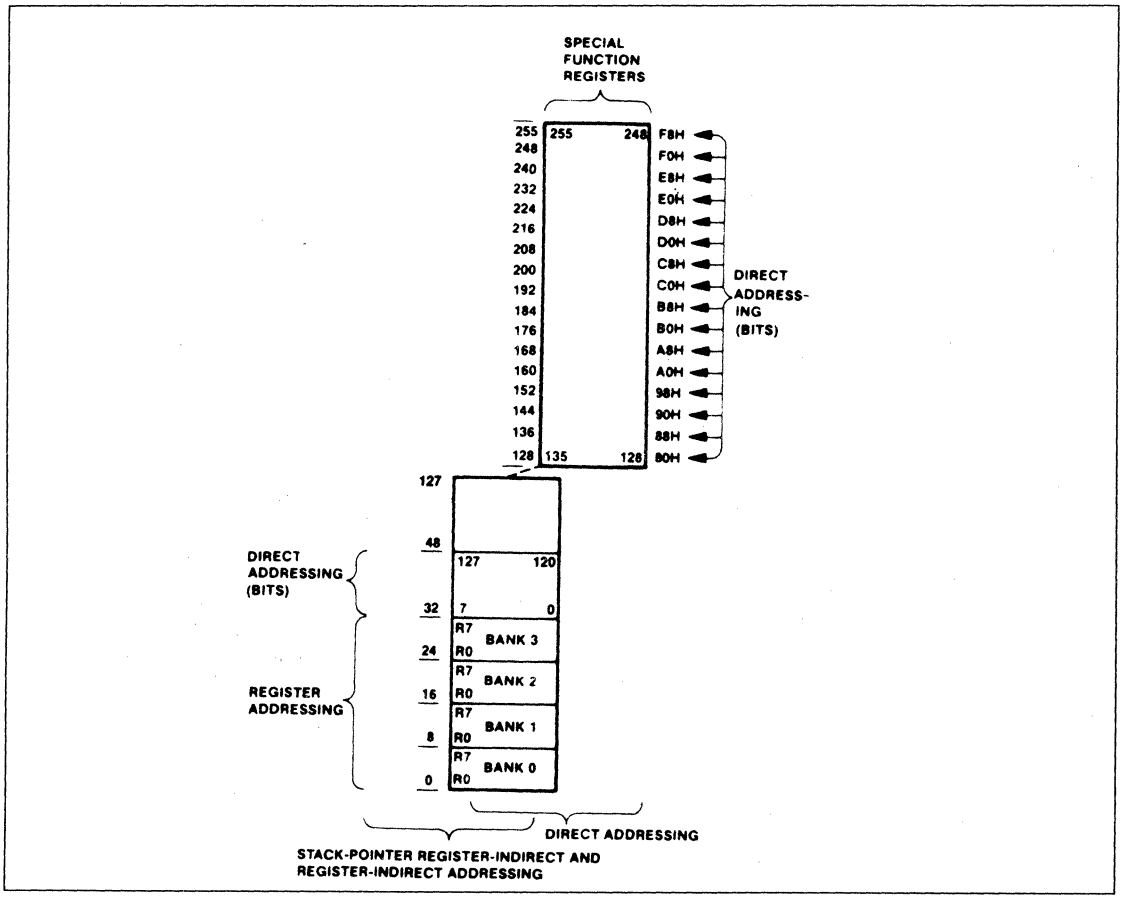


Figure 56. Special Function Register Bit Address

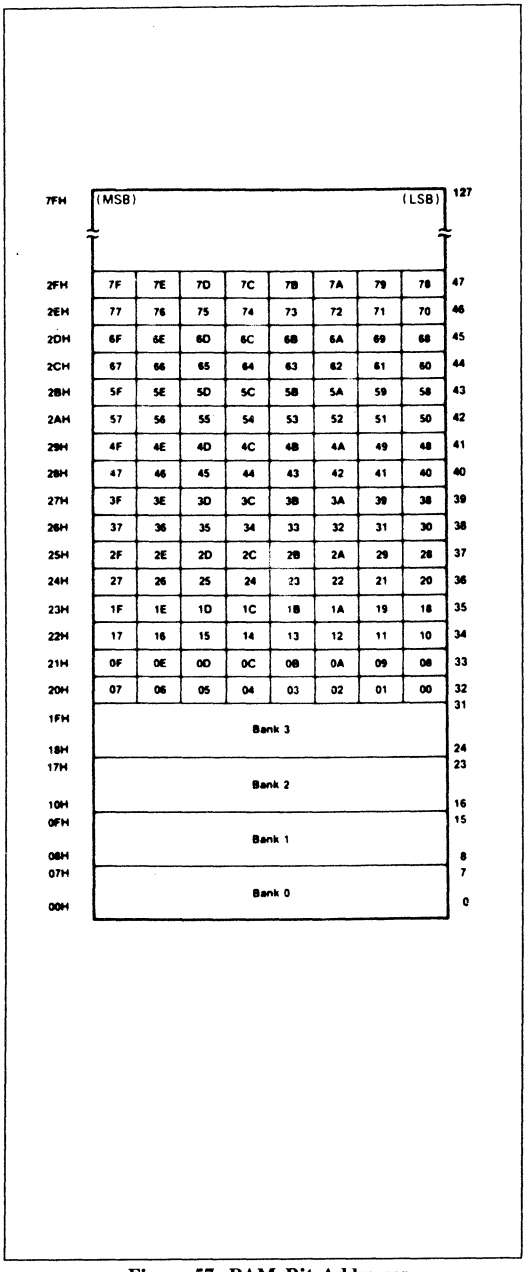


Figure 57. RAM Bit Addresses

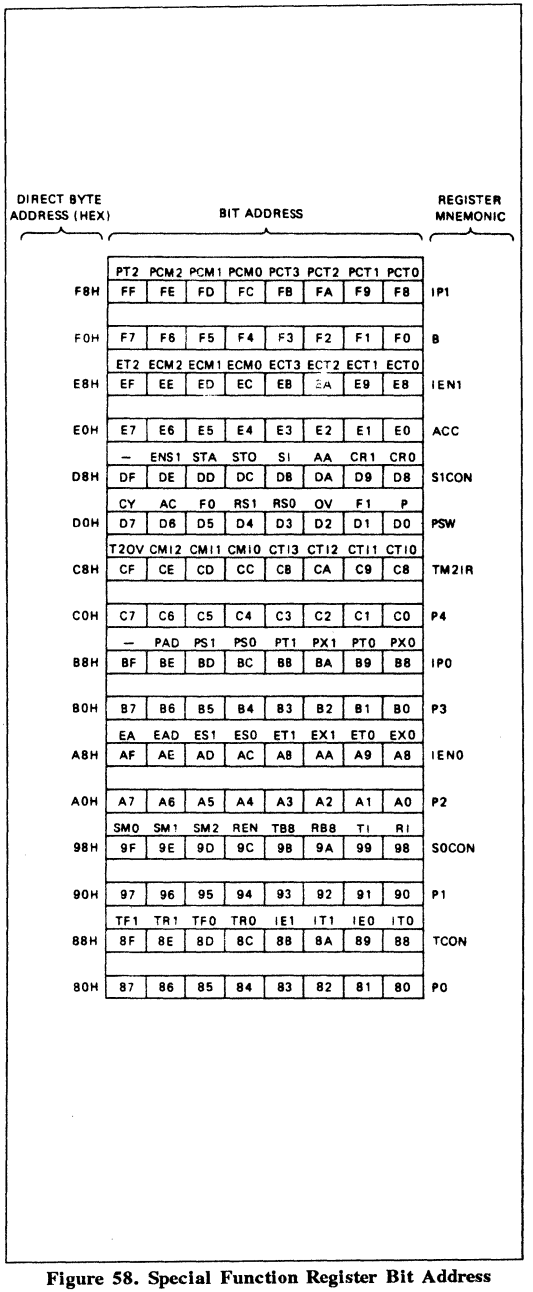


Figure 58. Special Function Register Bit Address

Signetics

S83C552/S80C552

Single-Chip 8-Bit Microcontroller With A/D, Capture/Compare Timer, With High-Speed Outputs, PWM

Product Specification

Microprocessor Division

DESCRIPTION

The S83C552/S80C552 Single-Chip 8-Bit Microcontroller is manufactured in an advanced CMOS process and is a derivative of the SC80C51 microcontroller family. The S83C552/S80C552 has the same instruction set as the 80C51. Two versions of the derivative exist:

- S83C552 – 8K bytes mask programmable ROM
- S80C552 – ROMless version of the S83C552

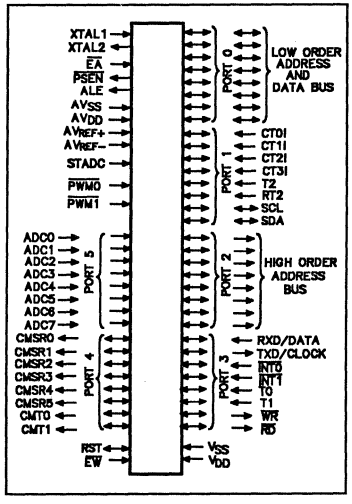
The S83C552 contains a non-volatile 8K x 8 read-only program memory, a volatile 256 x 8 read/write data memory, six 8-bit I/O ports, two 16-bit timer/event counters (identical to the timers of the SC80C51), an additional 16-bit timer coupled to capture and compare latches, a 15-source, two-priority-level, nested interrupt structure, an 8-input ADC, a dual DAC pulse width modulated interface, two serial interfaces (UART and I2C-bus), a 'watchdog' timer and on-chip oscillator and timing circuits. For systems that require extra capability, the S83C552 can be expanded using standard TTL compatible memories and logic.

The device also functions as an arithmetic processor having facilities for both binary and BCD arithmetic plus bit-handling capabilities. The instruction set consists of over 100 instructions: 49 one-byte, 45 two-byte and 17 three-byte. With a 12MHz crystal, 58% of the instructions are executed in 1μs and 40% in 2μs. Multiply and divide instructions require 4μs.

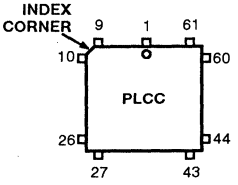
FEATURES

- **SC80C51B central processing unit**
- **8K x 8 ROM (83C552), expandable externally to 64K bytes**
- **256 x 8 RAM, expandable externally to 64K bytes**
- **Two standard 16-bit timer/counters**
- **An additional 16-bit timer/counter coupled to four capture registers and three compare registers**
- **Capable of producing 8 synchronized, timed outputs**
- **A 10-bit ADC with 8 multiplexed analog inputs**
- **Two 8-bit resolution, pulse width modulation outputs**
- **Five 8-bit I/O ports plus one 8-bit input port shared with analog inputs**
- **I2C-bus serial I/O port with byte oriented master and slave functions**
- **Full-duplex UART compatible with the standard 80C51**
- **On-chip watchdog timer**

LOGIC SYMBOL



PIN CONFIGURATION



TOP VIEW

Pin	Function	Pin	Function
1	P5.0/ADC0	35	XTAL1
2	VDD	36	Vss
3	STADC	37	Vss
4	PWM0	38	NC
5	PWM1	39	P2.0/A08
6	EW	40	P2.1/A09
7	P4.0/CMSR0	41	P2.2/A10
8	P4.1/CMSR1	42	P2.3/A11
9	P4.2/CMSR2	43	P2.4/A12
10	P4.3/CMSR3	44	P2.5/A13
11	P4.4/CMSR4	45	P2.6/A14
12	P4.5/CMSR5	46	P2.7/A15
13	P4.6/CMT0	47	PSEN
14	P4.7/CMT1	48	ALE
15	RST	49	EA
16	P1.0/CT0I	50	P0.7/AD7
17	P1.1/CT1I	51	P0.6/AD6
18	P1.2/CT2I	52	P0.5/AD5
19	P1.3/CT3I	53	P0.4/AD4
20	P1.4/T2	54	P0.3/AD3
21	P1.5/RT2	55	P0.2/AD2
22	P1.6/SCL	56	P0.1/AD1
23	P1.7/SDA	57	P0.0/AD0
24	P3.0/RXD	58	AVref–
25	P3.1/TXD	59	AVref+
26	P3.2/INT0	60	AVSS
27	P3.3/INT1	61	AVDD
28	P3.4/T0	62	P5.7/ADC7
29	P3.5/T1	63	P5.6/ADC6
30	P3.6/WR	64	P5.5/ADC5
31	P3.7/RD	65	P5.4/ADC4
32	NC	66	P5.3/ADC3
33	NC	67	P5.2/ADC2
34	XTAL2	68	P5.1/ADC1

Single-Chip 8-Bit Microcontroller | S83C552/S80C552

ORDERING INFORMATION

S8□C552-□□□□ (CPxxxx)

Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

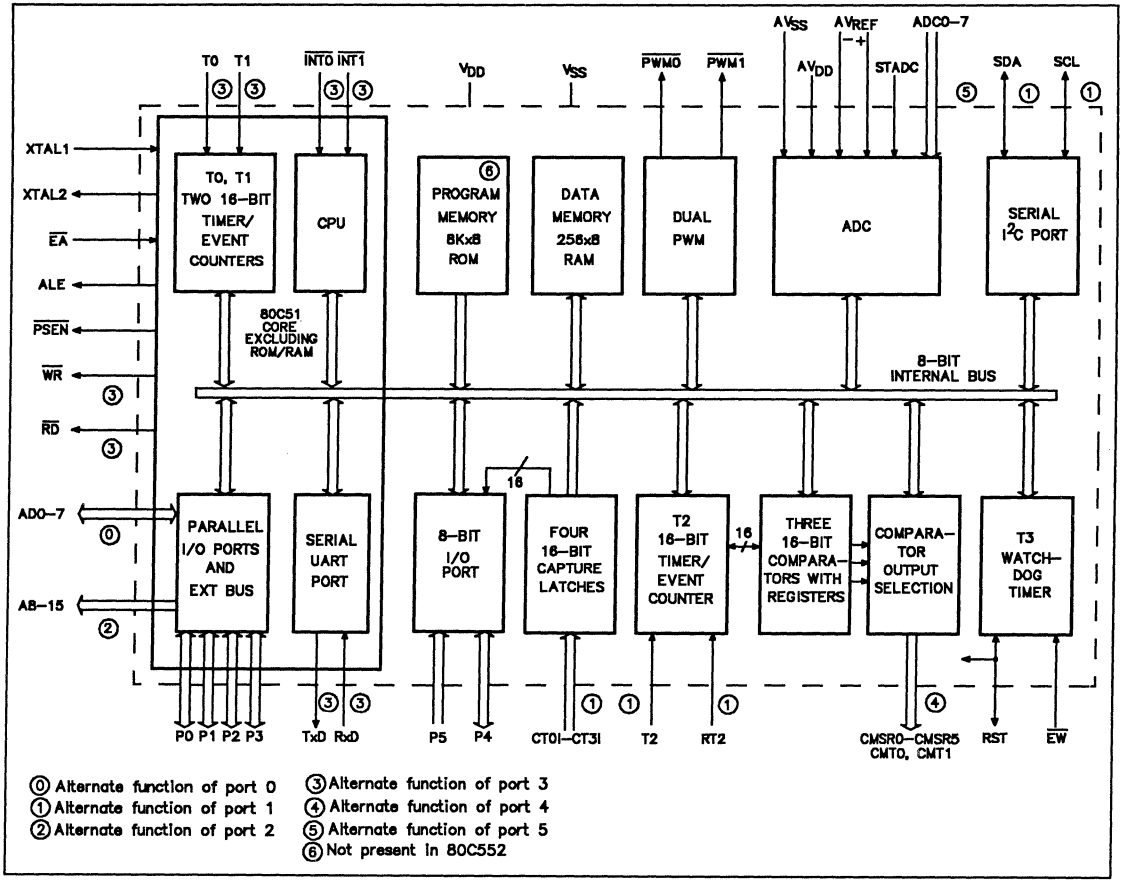
Package and Pins
A68 = 68-Pin Plastic PLCC

Speed and Temperature Range
1 = 0 to +70° C, 1.2 to 12MHz

ROMless/ROM
0 = ROMless
3 = ROM

PART NUMBER SELECTION			
ROMless Version	ROM Version	Temperature & Package	Frequency
S80C552-1A68	S83C552-1A68	0 to +70°C plastic PLCC	1.2 to 12MHz

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO.	TYPE	NAME AND FUNCTION
V_{CC}	2	I	**Digital Power Supply:** +5V power supply pin during normal operation, idle and power-down mode.
STADC	3	I	**Start ADC Operation:** Input starting analog to digital conversion (ADC operation can also be started by software).
$\overline{PWM0}$	4	O	**Pulse Width Modulation:** Output 0.
$\overline{PWM1}$	5	O	**Pulse Width Modulation:** Output 1.
$\overline{EW}$	6	I	**Enable Watchdog Timer:** Enable for T3 watchdog timer and disable power-down mode.
P0.0–P0.7	57–50	I/O	**Port 0:** Port 0 is an 8-bit open-drain bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application it uses strong internal pull-ups when emitting 1s.
P1.0–P1.7	16–23	I/O	**Port 1:** 8-bit I/O port. Alternate functions include:
	16–21	I/O	**(P1.0–P1.5):** Quasi-bidirectional port pins.
	22–23	I/O	**(P1.6, P1.7):** Open drain port pins
	16–19	I	**CT0I –CT3I (P1.0–P1.3):** Capture timer input signals for timer T2
	20	I	**T2 (P1.4):** T2 event input
	21	I	**RT2 (P1.5):** T2 timer reset signal. Rising edge triggered
	22	I/O	**SCL (P1.6):** Serial port clock line I2C-bus
	23	I/O	**SDA (P1.7):** Serial port data line I2C-bus
P2.0–P2.7	39–46	I/O	**Port 2:** 8-bit quasi-bidirectional I/O port. Alternate Function: High-order address byte for external memory (A08–A15).
P3.0–P3.7	24–31	I/O	**Port 3:** 8-bit quasi-bidirectional I/O port. Alternate functions include:
	24	I	**RxD (P3.0):** Serial input port
	25	O	**TxD (P3.1):** Serial output port
	26	I	**INT0 (P3.2):** External interrupt
	27	I	**INT1 (P3.3):** External interrupt
	28	I	**T0 (P3.4):** Timer 0 external input
	29	I	**T1 (P3.5):** Timer 1 external input
	30	O	**WR (P3.6):** External data memory write strobe
	31	O	**RD (P3.7):** External data memory read strobe
P4.0–P4.7	7–14	I/O	**Port 4:** 8-bit quasi-bidirectional I/O port. Alternate Functions include:
	7–12	O	**CMSR0–CMSR5 (P4.0–P4.5):** Timer T2 compare and set/reset outputs on a match with timer T2.
	13, 14	O	**CMT0, CMT1 (P4.6, P4.7):** Timer T2 compare and toggle outputs on a match with timer T2.
P5.0–P5.7	68–62, 1	I	**Port 5:** 8-bit input port. **ADC0–ADC7 (P5.0–P5.7):** Alternate Function: Eight input channels to ADC.
RST	15	I/O	**Reset:** Input to reset the S83C552. It also provides a reset pulse as output when timer T3 overflows.
XTAL1	35	I	**Crystal Input 1:** Input to the inverting amplifier that forms the oscillator, and input to the internal clock generator. Receives the external clock signal when an external oscillator is used.
XTAL2	34	O	**Crystal Input 2:** Output of the inverting amplifier that forms the oscillator. Left open-circuit when an external clock is used.
V_{SS}	36, 37	I	**Digital Ground**
$\overline{PSEN}$	47	O	**Program Store Enable:** Active-low read strobe to external program memory.
ALE	48	O	**Address Latch Enable:** Latches the low byte of the address during accesses to external memory. It is activated every six oscillator periods. During an external data memory access, one ALE pulse is skipped. ALE can drive up to eight LS TTL inputs and handles CMOS inputs without an external pull-up.
$\overline{EA}$	49	I	**External Access:** When $\overline{EA}$ is held at TTL level high, the CPU executes out of the internal program ROM provided the program counter is less than 8192. When $\overline{EA}$ is held at TTL low level, the CPU executes out of external program memory. $\overline{EA}$ is not allowed to float.
AV_{REF-}	58	I	**Analog to Digital Conversion Reference Resistor:** Low-end.
AV_{REF+}	59	I	**Analog to Digital Conversion Reference Resistor:** High-end.
AV_{SS}	60	I	**Analog Ground**
AV_{CC}	61	I	**Analog Power Supply**

NOTE:
To avoid 'latch-up' effect at power-on, the voltage on any pin at any time must not be higher or lower than V_{CC} + 0.5V or V_{SS} – 0.5V respectively.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol, page 1.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. To insure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

Table 1 shows the state of I/O ports during low current operating modes.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Program Memory	ALE	$\overline{\text{PSEN}}$	Port 0	Port 1	Port 2	Port 3	Port 4	PWM0/ PWM1
Idle	Internal	1	1	Data	Data	Data	Data	Data	High
Idle	External	1	1	Float	Data	Address	Data	Data	High
Power-down	Internal	0	0	Data	Data	Data	Data	Data	High
Power-down	External	0	0	Float	Data	Data	Data	Data	High

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Storage temperature range	−65 to +150	°C
Voltage on any other pin to V_{SS}	−0.5 to + 6.5	V
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.0	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C, V_{CC}, AV_{CC} = 5V ±10%, V_{SS}, AV_{SS} = 0V

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			Min	Typical[1]	Max	
V_{CC}	Supply voltage		4.5		5.5	V
I_{CC}	Power supply current: Active mode @ 12MHz Idle mode @ 12MHz Power down mode	See note 5		 11.5 1.3 3	 30 7 50	 mA mA µA
Inputs						
V_{IL}	Input low voltage, except $\overline{EA}$, P1.6/SCL, P1.7/SDA		−0.5		$0.2V_{CC}$−0.1	V
V_{IL1}	Input low voltage to $\overline{EA}$		−0.5		$0.2V_{CC}$−0.3	V
V_{IL2}	Input low voltage to P1.6/SCL, P1.7/SDA[6]		−0.5		$0.3V_{CC}$	V
V_{IH}	Input high voltage, except XTAL1, RST, P1.6/SCL, P1.7/SDA		$0.2V_{CC}$+0.9		V_{CC}+0.5	V
V_{IH1}	Input high voltage, XTAL1, RST		$0.7V_{CC}$		V_{CC}+0.5	V
V_{IH2}	Input high voltage, P1.6/SCL, P1.7/SDA[6]		$0.7V_{CC}$		6.0	V
$-I_{IL}$	Logical 0 input current, ports 1, 2, 3, 4, except P1.6/SCL, P1.7/SDA	V_{IN} = 0.45V			−50	µA
$-I_{TL}$	Logical 1−to−0 transition current, ports 1, 2, 3, 4, except P1.6/SCL, P1.7/SDA	See note 4			−650	µA
$\pm I_{IL1}$	Input leakage current, port 0, $\overline{EA}$, STADC, $\overline{EW}$	0.45V<V_I< V_{CC}			10	µA
$\pm I_{IL2}$	Input leakage current, P1.6/SCL, P1.7/SDA	0V<V_I<6V 0V<V_{CC}<5.5V			10	µA
Outputs						
V_{OL}	Output low voltage, ports 1, 2, 3, 4 except P1.6/SCL, P1.7/SDA	I_{OL} = 1.6mA[2]			0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$, $\overline{PWM0}$, $\overline{PWM1}$	I_{OL} = 3.2mA[2]			0.45	V
V_{OL2}	Output low voltage, P1.6/SCL, P1.7/SDA	I_{OL} = 3.0mA[2]			0.4	V
V_{OH}	Output high voltage, ports 1, 2, 3, 4, except P1.6/SCL, P1.7/SDA	$-I_{OH}$ = 60µA V_{CC} = 5V±10% $-I_{OH}$ = 25µA $-I_{OH}$ = 10µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V V
V_{OH1}	Output high voltage (port 0 in external bus mode, ALE, $\overline{PSEN}$, $\overline{PWM0}$, $\overline{PWM1}$)[3]	$-I_{OH}$ = 400µA V_{CC} = 5V±10% $-I_{OH}$ = 150µA $-I_{OH}$ = 40µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V V

DC ELECTRICAL CHARACTERISTICS (Continued)

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			Min	Typical[1]	Max	
Outputs (Continued)						
V_{OH2}	High level output voltage (RST)	$-I_{OH}$ = 350µA $-I_{OH}$ = 60µA	2.4 0.75V_{CC}			V
R_{RST}	Internal reset pulldown resistor		50		150	kΩ
C_{IO}	Pin capacitance	Test freq = 1MHz, T_A = 25°C			10	pF
Analog Inputs						
AV_{CC}	Analog supply voltage[7]	AV_{CC} = V_{CC}±0.2V	4.5		5.5	V
AI_{CC} AI_{ID} AI_{PD}	Analog supply current Operating Idle mode Power-down	Port 5 = 1.4V AV_{CC} = 2–5.5V			 1.0 50 50	 mA µA µA
AV_{IN}	Analog input voltage		AV_{SS}–0.2		AV_{CC}+0.2	V
AV_{ref}	Reference voltage: AV_{REF-} AV_{REF+}		 AV_{SS}–0.2		 AV_{CC}+0.2	 V V
R_{REF}	Resistance between AV_{REF+} and AV_{REF-}		10		50	kΩ
C_{IA}	Analog input capacitance				15	pF
t_{ADS}	Sampling time				$8t_{CY}$	µs
t_{ADC}	Conversion time (including sampling time)				$50t_{CY}$	µs
DL_e	Differential non-linearity[8]		–1		+2	LSB
IL_e	Integral non-linearity[8]				±2	LSB
OS_e	Offset error[8]				±10	mV
G_e	Gain error[8]				0.4	%
M_{CTC}	Channel to channel matching				±1	LSB
C_t	Crosstalk[9]	0–100kHz			–60	dB

NOTES:

1. Typical ratings are based on a limited number of samples taken from early manufacturing lots and are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1–to–0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the 0.9V_{CC} specification when the address bits are stabilizing.
4. Pins of ports 1 (except P1.6, P1.7), 2, 3 and 4 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. See Figures 8 through 11 for I_{CC} test conditions.
6. The input threshold voltage of P1.6 and P1.7 (SIO1) meets the I2C specification, so an input voltage below 1.5V will be recognized as a logic 0 while an input voltage above 3.0V will be recognized as a logic 1.
7. The following condition must not be exceeded: $V_{CC} - 0.2V < AV_{CC} < V_{CC} + 0.2V$.
8. Conditions: AV_{REF-} = 0V; AV_{CC} = 5.0V, AV_{REF+} = 5.12V. ADC is monotonic with no missing codes.
9. This should be considered when both analog and digital signals are simultaneously input to port 5.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C, V_{CC}, AV_{CC} = 5V ±10%, V_{SS}, AV_{SS} = 0V

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	1	Oscillator frequency			1.2	12	MHz
t_{LHLL}	1	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	1	Address valid to ALE low	28		$t_{CLCL}-55$		ns
t_{LLAX}	1	Address hold after ALE low	48		$t_{CLCL}-35$		ns
t_{LLIV}	1	ALE low to valid instruction in		234		$4t_{CLCL}-100$	ns
t_{LLPL}	1	ALE low to $\overline{PSEN}$ low	43		$t_{CLCL}-40$		ns
t_{PLPH}	1	$\overline{PSEN}$ pulse width	205		$3t_{CLCL}-45$		ns
t_{PLIV}	1	$\overline{PSEN}$ low to valid instruction in		145		$3t_{CLCL}-105$	ns
t_{PXIX}	1	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	1	Input instruction float after $\overline{PSEN}$		59		$t_{CLCL}-25$	ns
t_{AVIV}	1	Address to valid instruction in		312		$5t_{CLCL}-105$	ns
t_{PLAZ}	1	$\overline{PSEN}$ low to address float		10		10	ns
Data Memory							
t_{AVLL}	2, 3	Address valid to ALE low	43		$t_{CLCL}-40$		ns
t_{RLRH}	2, 3	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	2, 3	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	2, 3	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	2, 3	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	2, 3	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	2, 3	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	2, 3	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	2, 3	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	2, 3	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	2, 3	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{WHQX}	2, 3	Data hold after $\overline{WR}$	33		$t_{CLCL}-50$		ns
t_{RLAZ}	2, 3	$\overline{RD}$ low to address float		12		12	ns
t_{WHLH}	2, 3	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns
External Clock							
t_{CHCX}	5	High time[3]	20		20		ns
t_{CLCX}	5	Low time[3]	20		20		ns
t_{CLCH}	5	Rise time[3]		20		20	ns
t_{CHCL}	5	Fall time[3]		20		20	ns
Shift Register							
t_{XLXL}	4	Serial port clock cycle time[3]	1.0		$12t_{CLCL}$		µs
t_{QVXH}	4	Output data setup to clock rising edge[3]	700		$10t_{CLCL}-133$		ns
t_{XHQX}	4	Output data hold after clock rising edge[3]	50		$2t_{CLCL}-117$		ns
t_{XHDX}	4	Input data hold after clock rising edge[3]	0		0		ns
t_{XHDV}	4	Clock rising edge to input data valid[3]		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.
3. These values are characterized but not 100% production tested.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P– $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.

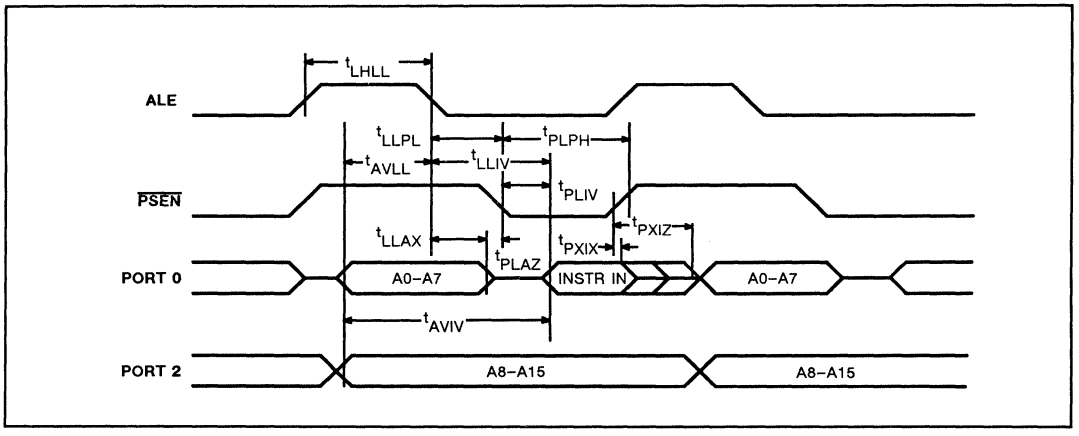


Figure 1. External Program Memory Read Cycle

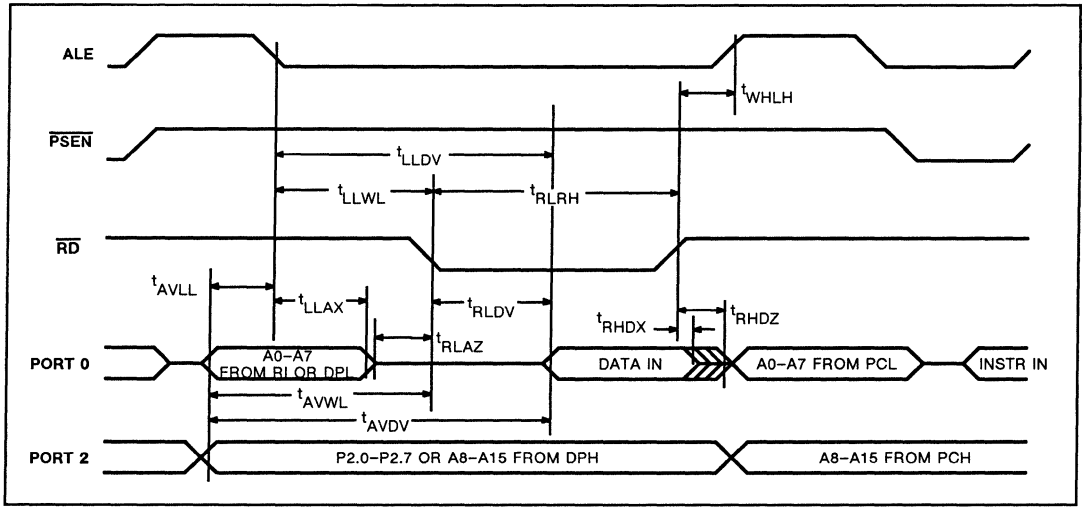


Figure 2. External Data Memory Read Cycle

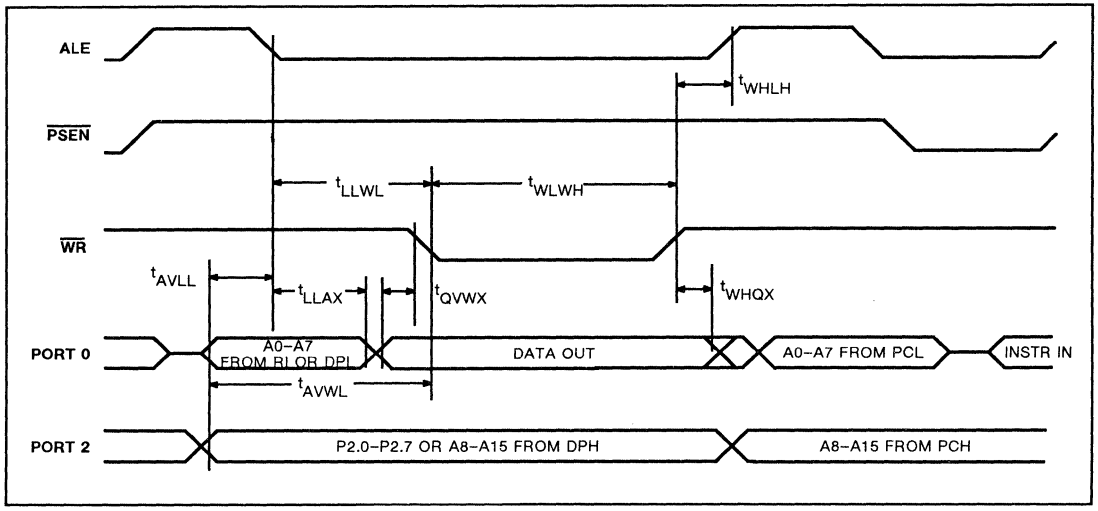


Figure 3. External Data Memory Write Cycle

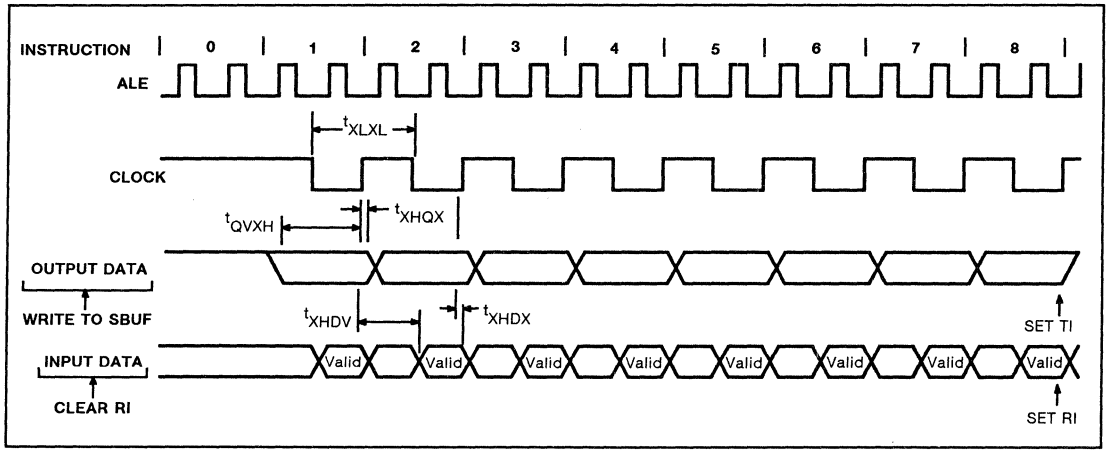


Figure 4. Shift Register Mode Timing

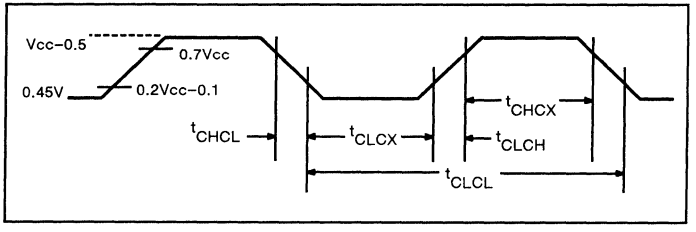


Figure 5. External Clock Drive

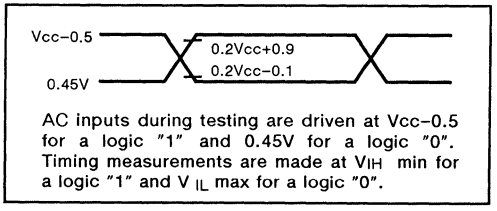


Figure 6. AC Testing Input/Output

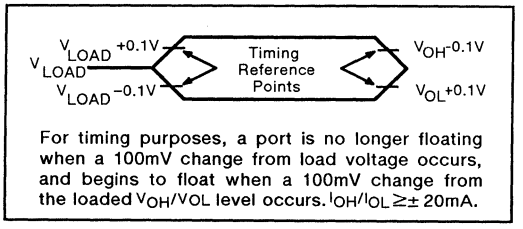


Figure 7. Float Waveform

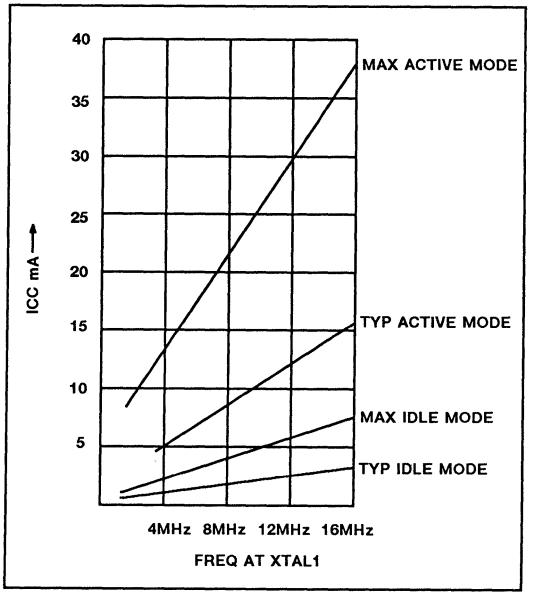


Figure 8. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test.
Maximum values taken at V_{CC} = 5.5V and worst case temperature.
Typical Values taken at V_{CC} = 5.0V and 25°C.

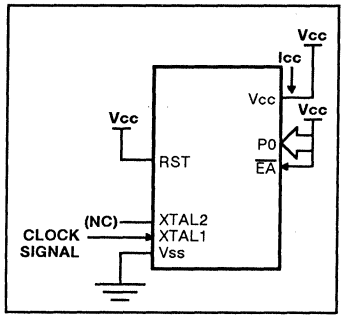


Figure 9. I_{CC} Test Condition, Active Mode
All other pins are disconnected

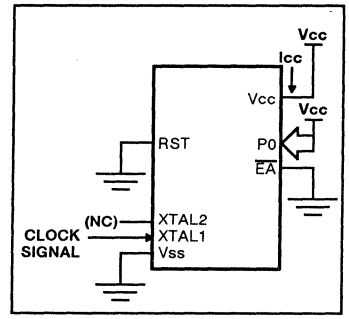


Figure 10. I_{CC} Test Condition, Idle Mode
All other pins are disconnected

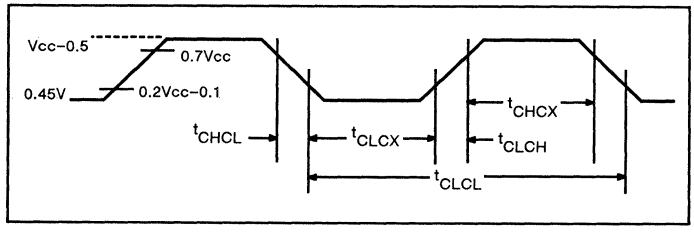


Figure 11. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

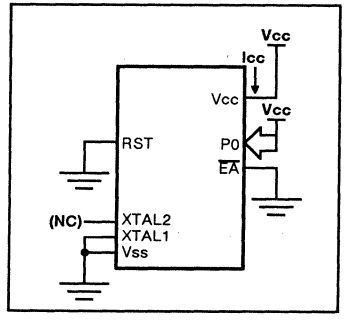


Figure 12. I_{CC} Test Conditions, Power Down Mode
All other pins are disconnected. V_{CC} = 2V to 5.5V

8XC652 OVERVIEW

The 8XC652 is a derivative of the 80C51 8-bit CMOS microcontroller. The 8XC652 contains all of the features of the 80C51 (that is the standard counter/timers T0 and T1, the standard serial I/O (UART), and four 8-bit I/O ports). In addition, the 8XC652 has the following:

- 8K bytes of ROM
- 256 bytes of RAM
- I2C bus serial I/O

The 8XC652 is pin-for-pin compatible and fully code compatible with the 80C51. There are some differences in the P1.6 and P1.7 pin functions that are described in detail later in this section. All of the 80C51 functions are present including the external 64K program and data memory expansion, Boolean processing, and two reduced power modes.

Differences from the 80C51

The data and program memory are organized similar to the 80C51. The 8XC652 program memory differs in that it has 8K bytes of on-chip ROM. When $\overline{EA}$ is high the 8XC652 fetches instructions from the internal ROM unless the address exceeds 1FFFH. Locations 2000H to FFFFH are fetched from external program memory. When $\overline{EA}$ is held low, all instruction fetches are from external memory.

The organization of the data memory is similar to the 80C51 except that the 8XC652 has an additional 128 bytes of RAM overlapped with the special function register space. This additional RAM is addressed using indirect addressing only and is available as stack space. (This memory addition is the same as in the 8052 and 83C552. See Figure 59 for a memory map.)

Special Function Registers

The 8XC652 special function register space is the same as that on the 80C51 except that it contains four additional SFRs. The added registers are: S1CON, S1STA, S1DAT, and S1ADR. In addition to these the standard UART special function registers, SCON and SBUF have been renamed S0CON and S0BUF for clarity.

Since the standard 80C51 on-chip functions are the same on the 8XC652, the SFR locations, bit locations, and operation are unchanged. The only exception is in the interrupt enable and interrupt priority SFRs. These have been changed to include the interrupt from the I2C serial port. Table 19 shows the special function registers, their direct address, the bit addresses, and the value in the register after a reset.

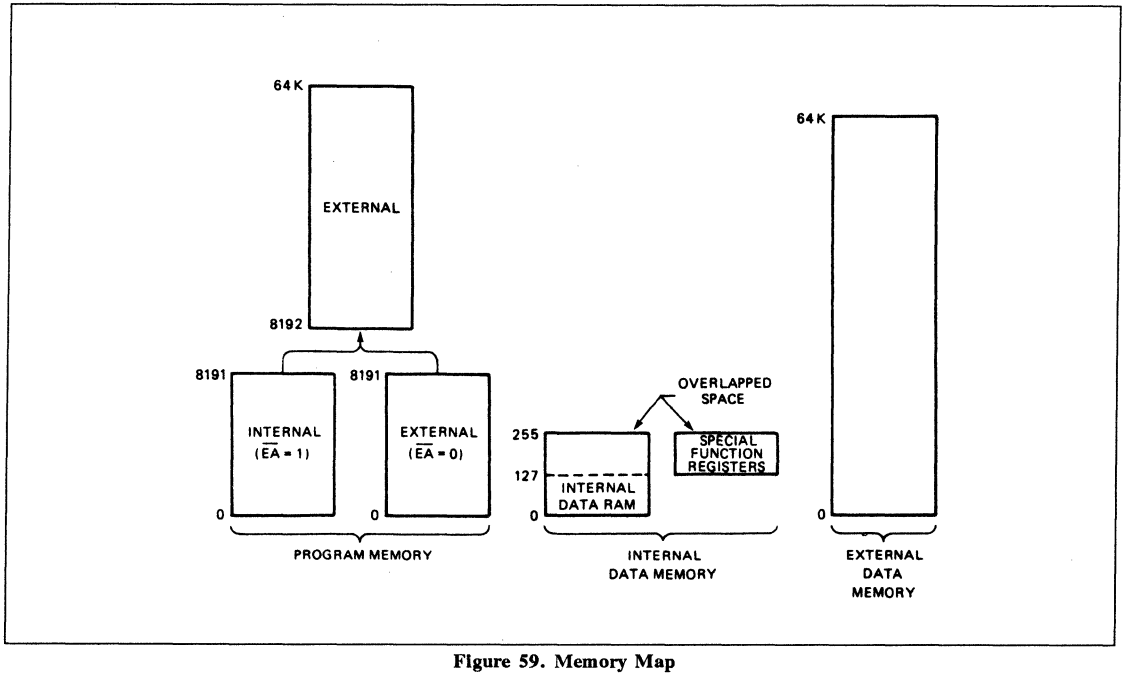


Figure 59. Memory Map

Table 19. 8XC652 Special Function Registers

Symbol	Description	Direct Address	Bit Address, Symbol or Alternative Port Function MSB							LSB	Reset Value
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
DPTR:	Data pointer (2 bytes):										
DPH	Data pointer high	83H									00H
DPL	Data pointer low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*	Interrupt enable	A8H	EA		ES1	ES0	ET1	EX1	ET0	ET1	0X000000B
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt priority	B8H	–		PS1	PS0	PT1	PX1	PT0	PX0	xx000000B
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	SDA	SCL							FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	A15	A14	A13	A12	A11	A10	A9	A8	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T1	T0	$\overline{INT1}$	$\overline{INT0}$	TXD	RXD	FFH
PCON	Power control	87H	SMOD	–	–	–	GF1	GF0	PD	IDL	0xxx0000B
			9F	9E	9D	9C	9B	9A	99	98	
S0CON*#	Serial 0 port control	98H	SM0	SM1	SM2	REN	TBB	RBB	TI	RI	00H
S0BUF#	Serial 0 data buffer	99H									xxxxxxxxB
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	00H
S1DAT#	Serial 1 data	DAH									00H
SP	Stack pointer	81H									07H
S1ADR#	Serial 1 address	DBH	SLAVE ADDRESS							GC	00H
S1STA#	Serial 1 status	D9H	SC4	SC3	SC2	SC1	SC0	0	0	0	F8H
			DF	DE	DD	DC	DB	DA	D9	D8	
S1CON*#	Serial 1 control	D8H	–	ENS1	STA	STO	SI	AA	CR1	CR0	x0000000B
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00H
TH1	Timer high 1	8DH									00H
TH0	Timer high 0	8CH									00H
TL1	Timer low 1	8BH									00H
TL0	Timer low 0	8AH									00H
TMOD	Timer mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H

* = SFRs are bit addressable
\# = SFRs are modified from or added to the 80C51 SFRs.

I2C Serial Communication – SIO1

The I2C Serial port is identical to the I2C serial port on the 83C552. The operation of this subsystem is described in detail in the 83C552 section of this manual.

Note that in both the 8XC652 and the 83C552 the I2C pins are alternate functions to port pins P1.6 and P1.7. Because of this P1.6 and P1.7 on these parts do not have a pull-up structure as found on the 80C51. Therefore P1.6 and P1.7 have open drain outputs on the 8XC652.

Idle and Power-down Operation

Idle mode operation permits the interrupt, serial ports, and timer blocks to continue to function while the CPU is halted. The following functions remain active during idle mode. These functions may generate an interrupt or reset and thus end the idle mode:

- Timer 0, Timer 1
- SIO0, SIO1
- External interrupt

In idle mode, port pins P1.6 and P1.7 function as SCL and SDA respectively if the I2C serial port is enabled. The power-down operation freezes the oscillator. The power-down mode can only be activated by setting the PD bit in the PCON register. The power-down mode in the 8XC652 operates exactly the same as in the 80C51.

Interrupt System

The interrupt system is the same as in the 80C51 except that the 8XC652 acknowledges interrupt requests from six sources as follows:

- $\overline{\text{INT0}}$ external interrupt 0
- $\overline{\text{INT1}}$ external interrupt 1
- Timer 0 overflow
- Timer 1 overflow
- I2C serial I/O interrupt
- UART serial I/O interrupt

See Figure 60 for a function diagram of the 8XC652 interrupt structure. Each interrupt vectors to a separate location in program memory for its service program. Each source can be individually enabled or disabled by a corresponding bit in the IE register, moreover each interrupt may be programmed to a high or low priority level using a corresponding bit in the IP register. Also all enabled sources can be globally disabled or enabled.

Both external interrupts can be programmed to be level-activated or transition-activated; an active LOW level allows "Wire-ORing" of several input sources to the input pin.

Each interrupt source can be set for either high priority or low priority. If two separate interrupts are requested simultaneously the processor will branch to the vector associated with the interrupt that has the higher priority. If there are simultaneous requests from sources that have the same priority, then the interrupts will be serviced in the following order:

1 – $\overline{\text{INT0}}$ external interrupt 0
2 – I2C serial I/O interrupt
3 – Timer 0 overflow
4 – $\overline{\text{INT1}}$ external interrupt 1
5 – Timer 1 overflow
6 – UART serial I/O interrupt

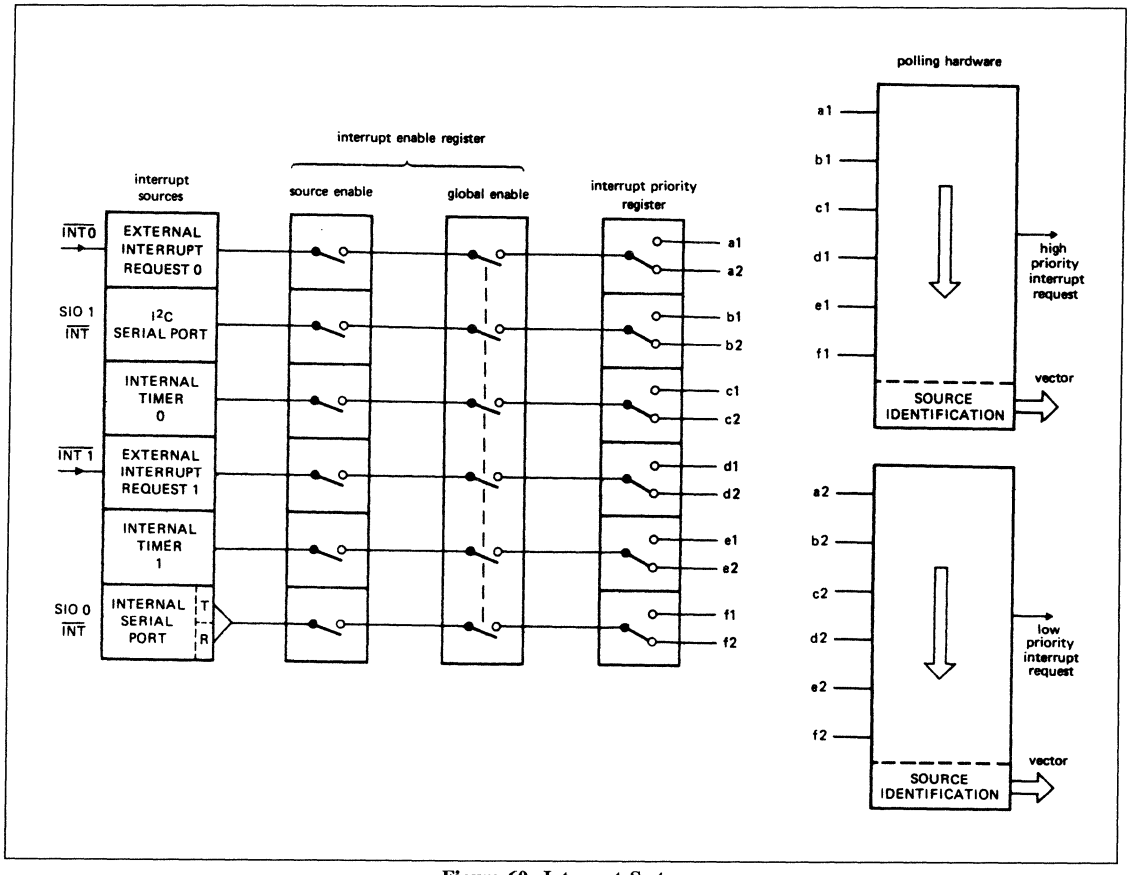


Figure 60. Interrupt System

A low priority interrupt routine can be interrupted by an interrupt having a higher priority. A high priority interrupt can not be interrupted. All of the features of the 8XC652 that have not been discussed in this section are the same as those on the 80C51.

Interrupt Enable Register

	7	6	5	4	3	2	1	0
IE (A8H)	EA	–	ES1	ES0	ET1	EX1	ET0	EX0

Bit	Symbol	Function
IE.7	EA	General enable/disable control 0 = No interrupt enabled 1 = Any individually enabled interrupt will be accepted
IE.6	–	Unused
IE.5	ES1	Enable SIO1 (I2C) interrupt
IE.4	ES0	Enable SIO0 (UART) interrupt
IE.3	ET1	Enable timer 1 interrupt
IE.2	EX1	Enable external 1 interrupt
IE.1	ET0	Enable timer 0 interrupt
IE.0	EX0	Enable external 0 interrupt 0 = interrupt disabled 1 = interrupt enabled

Interrupt Priority Register

	7	6	5	4	3	2	1	0
IP (B8H)	–	–	PS1	PS0	PT1	PX1	PT0	PX0

Bit	Symbol	Function
IP.7	–	Unused
IP.6	–	Unused
IP.5	PS1	SIO1 (I2C) interrupt priority level
IP.4	PS0	SIO0 (UART interrupt priority level
IP.3	PT1	Timer 1 interrupt priority level
IP.2	PX1	Enable interrupt 1 priority level
IP.1	PT0	Timer 0 interrupt priority level
IP.0	PX0	External interrupt 0 priority level 0 = Low priority 1 = High priority

The following vectors indicate the ROM location where the appropriate interrupt service routine starts.

Source	Vector
External 0 (X0)	0003H
Timer 0 overflow (T0)	000BH
External 1 (X1)	0013H
Timer 1 overflow (T1)	001BH
Serial I/O 0 (UART) (S0)	0023H
Serial I/O 1 (I2C) (S1)	002BH

Signetics

S83C652/S80C652
CMOS Single-Chip 8-Bit Microcontroller

Product Specification

Microprocessor Division

DESCRIPTION

The S83C652/S80C652 Single-Chip 8-Bit Microcontroller is manufactured in an advanced CMOS process and is a derivative of the SC80C51 microcontroller family. The S83C652/S80C652 has the same instruction set as the 80C51. Two versions of the derivative exist:

- S83C652 – 8K bytes mask programmable ROM, 256 bytes RAM
- S80C652 – ROMless version of the S83C652

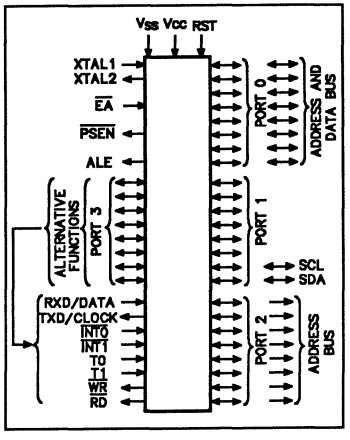
This device provides architectural enhancements that make it applicable in a variety of applications for general control systems. The S83C652 contains a non-volatile 8K x 8 read-only program memory, a volatile 256 x 8 read/write data memory, four 8-bit I/O ports, two 16-bit timer/event counters (identical to the timers of the S80C51), a multi-source, two-priority-level, nested interrupt structure, an I2C interface, UART and on-chip oscillator and timing circuits. For systems that require extra capability, the S83C652 can be expanded using standard TTL compatible memories and logic.

The device also functions as an arithmetic processor having facilities for both binary and BCD arithmetic plus bit-handling capabilities. The instruction set consists of over 100 instructions: 49 one-byte, 45 two-byte and 17 three-byte. With a 12MHz crystal, 58% of the instructions are executed in 1μs and 40% in 2μs. Multiply and divide instructions require 4μs.

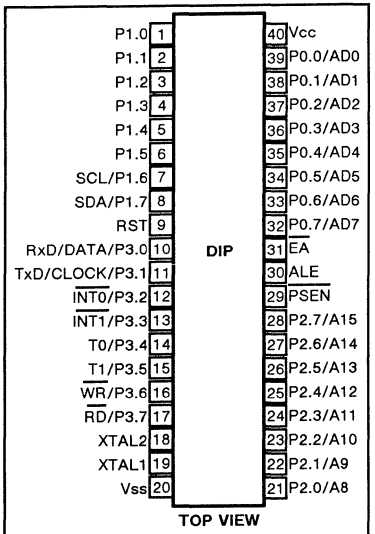
FEATURES

- **SC80C51 central processing unit**
- **8K x 8 ROM, expandable externally to 64K bytes**
- **256 x 8 RAM, expandable externally to 64K bytes**
- **Two standard 16-bit timer/counters**
- **Four 8-bit I/O ports**
- **I2C-bus serial I/O port with byte oriented master and slave functions**
- **Full-duplex UART facilities**

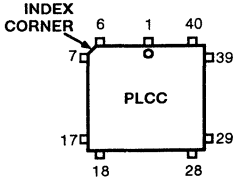
LOGIC SYMBOL



PIN CONFIGURATION



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	SCL/P1.6	30	P2.6/A14
9	SDA/P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/DATA/P3.0	33	ALE
12	NC	34	NC
13	TxD//CLOCK/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

CMOS Single-Chip 8-Bit Microcontroller — S83C652/S80C652

S8□C652-□□□□ (CPxxxx)

Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

Package and Pins
A44 = 44-Pin Plastic LCC
N40 = 40-Pin Plastic DIP

Speed and Temperature Range
1 = 0 to +70°C, 1.2 to 12MHz
2 = −40 to +85°C, 1.2 to 12MHz
4 = 0 to +70°C, 1.2 to 16MHz
5 = −40 to +85°C, 1.2 to 16MHz
6 = −40 to +110°C, 1.2 to 12MHz

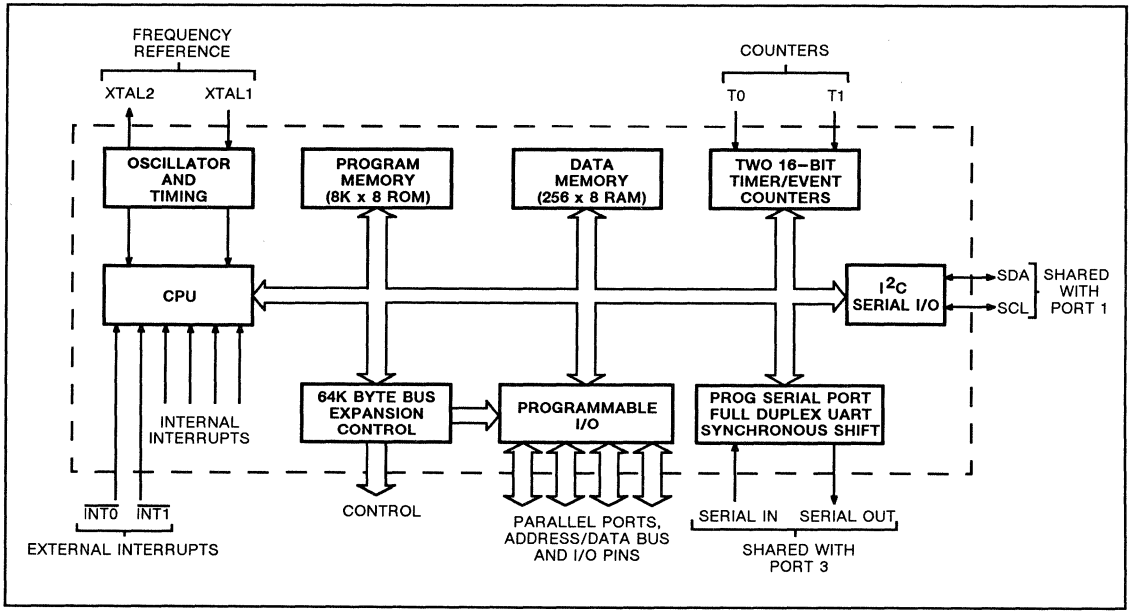
ROMless/ROM
0 = ROMless
3 = ROM

PART NUMBER SELECTION

ROMless	ROM Version	Temperature and Package	Frequency
S80C652-1N40	S83C652-1N40	0 to +70°C plastic DIP	1.2 to 12MHz
S80C652-1A44	S83C652-1A44	0 to +70°C plastic LCC	1.2 to 12MHz
S80C652-2N40	S83C652-2N40	−40 to +85°C plastic DIP	1.2 to 12MHz
S80C652-2A44	S83C652-2A44	−40 to +85°C plastic LCC	1.2 to 12MHz
S80C652-4N40	S83C652-4N40	0 to +70°C plastic DIP*	1.2 to 16MHz
S80C652-4A44	S83C652-4A44	0 to +70°C plastic LCC*	1.2 to 16MHz
S80C652-5N40	S83C652-5N40	−40 to +85°C plastic DIP*	1.2 to 16MHz
S80C652-5A44	S83C652-5A44	−40 to +85°C plastic LCC*	1.2 to 16MHz
S80C652-6N40	S83C652-6N40	−40 to +110°C plastic DIP	1.2 to 12MHz
S80C652-6A44	S83C652-6A44	−40 to +110°C plastic LCC	1.2 to 12MHz

*Preliminary specification

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. LCC	TYPE	NAME AND FUNCTION
P0.7–P0.0	39–32	43–46	I/O	**Port 0:** 8-bit open-drain bidirectional I/O port. It is also the multiplexed low-order address and data bus during accesses to external memory (during these accesses it activates internal pullups). Port 0 can sink/source 8 LS TTL inputs.
P1.0–P1.7	1–8	2–9	I/O	**Port 1:** 8-bit quasi-bidirectional I/O port. Port 1 can sink/source one TTL (= 4 LS TTL) input. It can drive CMOS inputs without external pullups, except P1.6 and P1.7 which have open drain outputs. Alternate functions include:
P1.6	7	8	I/O	**SCL:** I2C-bus serial port clock line.
P1.7	8	9	I/O	**SDA:** I2C-bus serial port data line.
P2.0–P2.7	21–28	24–31	I/O	**Port 2:** 8-bit quasi-bidirectional I/O port with internal pullups. During access to external memories (RAM/ROM) that use 16-bit addresses (MOVX @DPTR), port 2 emits the high-order address byte. When external RAM is accessed with an 8-bit address (MOVX @Ri), port 2 emits the contents of the P2 function register. Port 2 can sink/source one TTL (=4 LS TTL) input. It can drive CMOS inputs without external pullups.
P3.0–P3.7	10–17	11, 13–19	I/O	**Port 3:** 8-bit quasi-bidirectional I/O port with internal pullups. It also serves the following alternative functions:
P3.0	10	11	I	**RxD/DATA:** Serial port receiver data input (asynchronous) or data input/output (synchronous).
P3.1	11	13	O	**TxD/CLOCK:** Serial port transmitter data output (asynchronous) or clock output (synchronous).
P3.2	12	14	I	**$\overline{INT0}$:** External interrupt 0 or gate control input for timer/event counter 0.
P3.3	13	15	I	**$\overline{INT1}$:** External interrupt 1 or gate control input for timer/event counter 1.
P3.4	14	16	I	**T0:** External input for timer/event counter 0.
P3.5	15	17	I	**T1:** External input for timer/event counter 1.
P3.6	16	18	O	**$\overline{WR}$:** External data memory write strobe.
P3.7	17	19	O	**$\overline{RD}$:** External data memory read strobe. The generation or use of a port pin as an alternative function is carried out automatically by the S83C652, provided the associated special function register bit is set high. Port 3 can sink/source one TTL (=4 LS TTL) input. It can drive CMOS inputs without external pullups.
RST	9	10	I	**Reset:** A high level on this pin for two machine cycles, while the oscillator is running, resets the device. An internal pull-down resistor permits power-on reset using only a capacitor to V_{CC}.
XTAL1	19	21	I	**Crystal Input 1:** Input to the inverting amplifier that forms the oscillator. Left open-circuit when an external oscillator clock is used.
V_{SS}	20	22		**Ground:** Circuit ground potential.
$\overline{PSEN}$	29	32	O	**Program Store Enable:** Read strobe to the external program memory via port 0 and 2. It is activated twice each machine cycle during fetches from external program memory. When executing out of external program memory, two activations of PSEN are skipped during each access to external data memory. PSEN is not activated (remains high) during fetches from external program memory. PSEN can sink/source 8 LS TTL inputs. It can drive CMOS inputs without an external pullup.
ALE	30	33	O	**Address Latch Enable:** Latches the low byte of the address during accesses to external memory in normal operation. It is activated every six oscillator periods except during an external data memory access. ALE can sink/source 8 LS TTL inputs. It can drive CMOS inputs without an external pullup.
$\overline{EA}$	31	35	I	**External Access:** When $\overline{EA}$ is held at a TTL high level, the CPU executes out of the internal program ROM, provided the program counter is less than 8192. When $\overline{EA}$ is held at a TTL low level, the CPU executes out of external program memory via port 0 and port 2. $\overline{EA}$ is not allowed to float.
V_{CC}	40	44		**Power Supply:** +5V power supply pin during normal operation, idle mode and power-down mode.

NOTE:
To avoid a 'latch-up' effect at power-on, the voltage on any pin at any time must not be higher or lower than V_{CC} + 0.5V or V_{SS} – 0.4V respectively.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator, as shown in the logic symbol, page 1.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. To insure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

Table 1 shows the state of I/O ports during low current operating modes.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	Port 0	Port 1	Port 2	Port 3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

ABSOLUTE MAXIMUM RATINGS[1, 2, 3]

PARAMETER	RATING	UNIT
Storage temperature range	−65 to +150	°C
Voltage on any pin to V_{SS}	−0.5 to + 6.5	V
Power dissipation (based on package heat transfer limitations, not device power consumption)	1.0	W

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions other than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from the damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages are with respect to V_{SS} unless otherwise noted.

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C/110°C, V_{CC} = 5V ±10%, V_{SS} = 0V

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			Min	Typical[1]	Max	
V_{CC}	Supply voltage		4.5		5.5	V
I_{CC}	Power supply current: Active mode @ 16MHz Idle mode @ 16MHz Power down mode	See note 5		11.5 1.3 3	30 6.5 50	mA mA µA
Inputs						
V_{IL}	Input low voltage, except $\overline{EA}$, P1.6/SCL, P1.7/SDA		−0.5		$0.2V_{CC}-0.1$	V
V_{IL1}	Input low voltage to $\overline{EA}$		−0.5		$0.2V_{CC}-0.3$	V
V_{IL2}	Input low voltage to P1.6/SCL, P1.7/SDA[7]		−0.5		1.5	V
V_{IH}	Input high voltage, except XTAL1, $\overline{RST}$, P1.6/SCL, P1.7/SDA		$0.2V_{CC}+0.9$		$V_{CC}+0.5$	V
V_{IH1}	Input high voltage, XTAL1, $\overline{RST}$		$0.7V_{CC}$		$V_{CC}+0.5$	V
V_{IH2}	Input high voltage, P1.6/SCL, P1.7/SDA[6]		3.0		6.0	V
$-I_{IL}$	Logical 0 input current, ports 1, 2, 3, except P1.6/SCL, P1.7/SDA	V_{IN} = 0.45V			50	µA
$-I_{TL}$	Logical 1−to−0 transition current, ports 1, 2, 3, except P1.6/SCL, P1.7/SDA	See note 4			−650	µA
$\pm I_{IL1}$	Input leakage current, port 0, $\overline{EA}$	$0.45V<V_I<V_{CC}$			10	µA
$\pm I_{IL2}$	Input leakage current, P1.6/SCL, P1.7/SDA	$0V<V_I<6V$ $0V<V_{CC}<5.5V$			10	µA
Outputs						
V_{OL}	Output low voltage, ports 1, 2, 3, except P1.6/SCL, P1.7/SDA	I_{OL} = 1.6mA[2]			0.45	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$	I_{OL} = 3.2mA[2]			0.45	V
V_{OL2}	Output low voltage, P1.6/SCL, P1.7/SDA	I_{OL} = 3.0mA[2]			0.4	V
V_{OH}	Output high voltage, ports 1, 2, 3	$-I_{OH}$ = 60µA $-I_{OH}$ = 25µA $-I_{OH}$ = 10µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V
V_{OH1}	Output high voltage (port 0 in external bus mode, ALE, $\overline{PSEN}$)[3]	$-I_{OH}$ = 400µA $-I_{OH}$ = 150µA $-I_{OH}$ = 40µA	2.4 $0.75V_{CC}$ $0.9V_{CC}$			V V V

DC ELECTRICAL CHARACTERISTICS (Continued)

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			Min	Typical[1]	Max	
Outputs (Continued)						
R_{RST}	Internal reset pulldown resistor		50		150	kΩ
C_{IO}	Pin capacitance	Test freq = 1MHz, T_A = 25°C			10	pF

NOTES:
1. Typical ratings are based on a limited number of samples taken from early manufacturing lots and are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the $0.9V_{CC}$ specification when the address bits are stabilizing.
4. Pins of ports 1, 2, and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. See Figures 7 through 10 for I_{CC} test conditions.
6. The input threshold voltage of P1.6 and P1.7 (SIO1) meets the I2C specification, so an input voltage below 1.5V will be recognized as a logic 0 while an input voltage above 3.0V will be recognized as a logic 1.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C, or T_A = −40°C to +85°C/110°C, V_{CC} = 5V ±10%, V_{SS} = 0V[1, 2]

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
Program Memory							
$1/t_{CLCL}$	1	Oscillator frequency			1.2	16	MHz
t_{LHLL}	1	ALE pulse width	127		$2t_{CLCL}-40$		ns
t_{AVLL}	1	Address valid to ALE low	28		$t_{CLCL}-55$		ns
t_{LLAX}	1	Address hold after ALE low	43		$t_{CLCL}-35$		ns
t_{LLIV}	1	ALE low to valid instruction in		234		$4t_{CLCL}-100$	ns
t_{LLPL}	1	ALE low to $\overline{PSEN}$ low	43		$t_{CLCL}-40$		ns
t_{PLPH}	1	$\overline{PSEN}$ pulse width	205		$3t_{CLCL}-45$		ns
t_{PLIV}	1	$\overline{PSEN}$ low to valid instruction in		145		$3t_{CLCL}-105$	ns
t_{PXIX}	1	Input instruction hold after $\overline{PSEN}$	0		0		ns
t_{PXIZ}	1	Input instruction float after $\overline{PSEN}$		59		$t_{CLCL}-25$	ns
t_{AVIV}	1	Address to valid instruction in		312		$5t_{CLCL}-105$	ns
t_{PLAZ}	1	$\overline{PSEN}$ low to address float		10		10	ns
Data Memory							
t_{AVLL}	2, 3	Address valid to ALE low	43		$t_{CLCL}-40$		ns
t_{RLRH}	2, 3	$\overline{RD}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{WLWH}	2, 3	$\overline{WR}$ pulse width	400		$6t_{CLCL}-100$		ns
t_{RLDV}	2, 3	$\overline{RD}$ low to valid data in		252		$5t_{CLCL}-165$	ns
t_{RHDX}	2, 3	Data hold after $\overline{RD}$	0		0		ns
t_{RHDZ}	2, 3	Data float after $\overline{RD}$		97		$2t_{CLCL}-70$	ns
t_{LLDV}	2, 3	ALE low to valid data in		517		$8t_{CLCL}-150$	ns
t_{AVDV}	2, 3	Address to valid data in		585		$9t_{CLCL}-165$	ns
t_{LLWL}	2, 3	ALE low to $\overline{RD}$ or $\overline{WR}$ low	200	300	$3t_{CLCL}-50$	$3t_{CLCL}+50$	ns
t_{AVWL}	2, 3	Address valid to $\overline{WR}$ low or $\overline{RD}$ low	203		$4t_{CLCL}-130$		ns
t_{QVWX}	2, 3	Data valid to $\overline{WR}$ transition	23		$t_{CLCL}-60$		ns
t_{WHQX}	2, 3	Data hold after $\overline{WR}$	33		$t_{CLCL}-50$		ns
t_{RLAZ}	2, 3	$\overline{RD}$ low to address float		12		12	ns
t_{WHLH}	2, 3	$\overline{RD}$ or $\overline{WR}$ high to ALE high	43	123	$t_{CLCL}-40$	$t_{CLCL}+40$	ns

AC ELECTRICAL CHARACTERISTICS (Continued)

SYMBOL	FIGURE	PARAMETER	12MHz CLOCK		VARIABLE CLOCK		UNIT
			Min	Max	Min	Max	
External Clock							
t_{CHCX}	4	High time[3]	20		20		ns
t_{CLCX}	4	Low time[3]	20		20		ns
t_{CLCH}	4	Rise time[3]		20		20	ns
t_{CHCL}	4	Fall time[3]		20		20	ns
Shift Register							
t_{XLXL}	5	Serial port clock cycle time[3]	1.0		$12t_{CLCL}$		µs
t_{QVXH}	5	Output data setup to clock rising edge[3]	700		$10t_{CLCL}-133$		ns
t_{XHQX}	5	Output data hold after clock rising edge[3]	50		$2t_{CLCL}-117$		ns
t_{XHDX}	5	Input data hold after clock rising edge[3]	0		0		ns
t_{XHDV}	5	Clock rising edge to input data valid[3]		700		$10t_{CLCL}-133$	ns

NOTES:
1. Parameters are valid over operating temperature range unless otherwise specified.
2. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.
3. These values are characterized but not 100% production tested.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

A – Address
C – Clock
D – Input data
H – Logic level high
I – Instruction (program memory contents)
L – Logic level low, or ALE
P – $\overline{PSEN}$
Q – Output data
R – $\overline{RD}$ signal
t – Time
V – Valid
W – $\overline{WR}$ signal
X – No longer a valid logic level
Z – Float

Examples: t_{AVLL} = Time for address valid to ALE low.
t_{LLPL} = Time for ALE low to $\overline{PSEN}$ low.

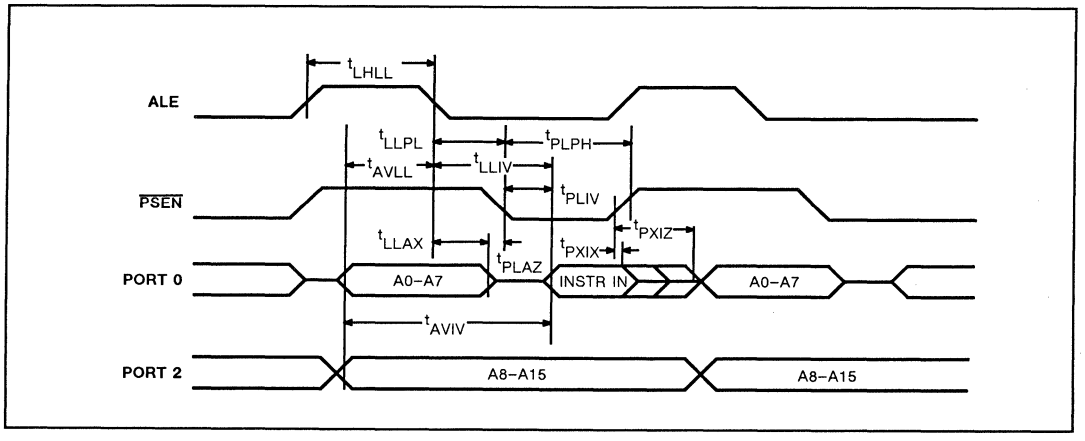


Figure 1. External Program Memory Read Cycle

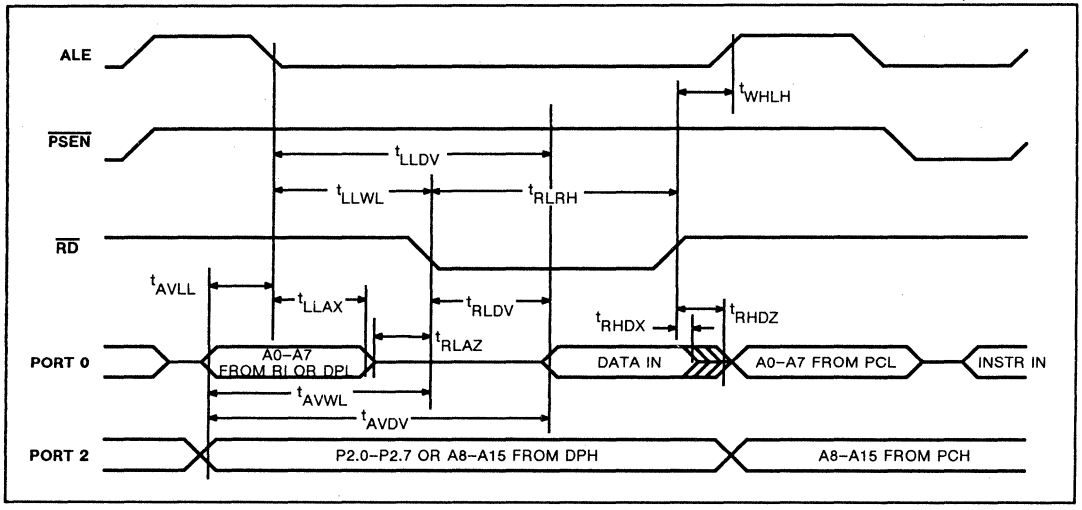


Figure 2. External Data Memory Read Cycle

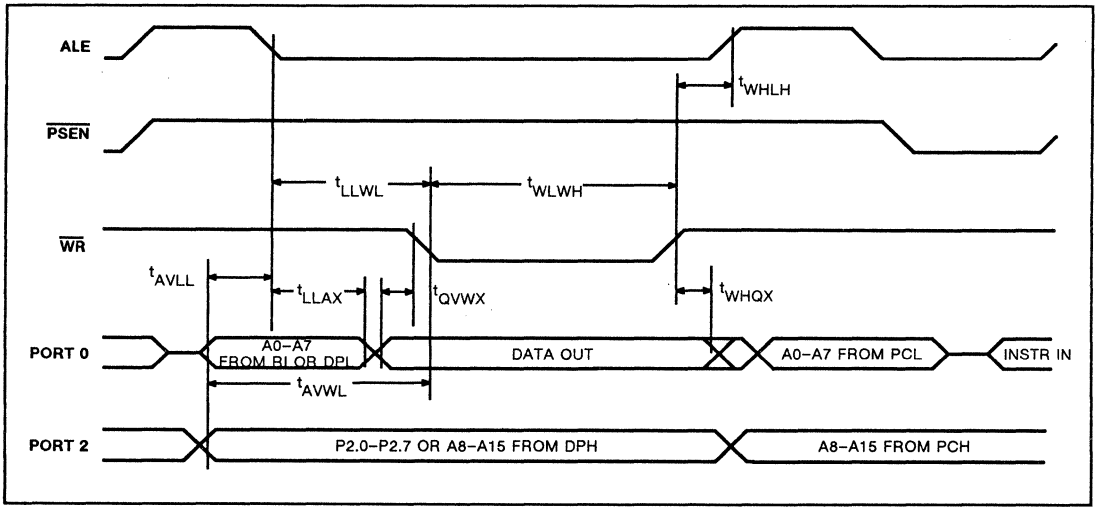


Figure 3. External Data Memory Write Cycle

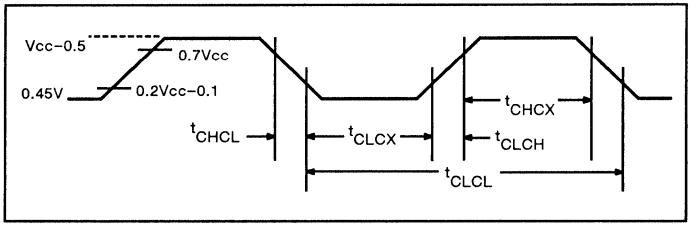


Figure 4. External Clock Drive

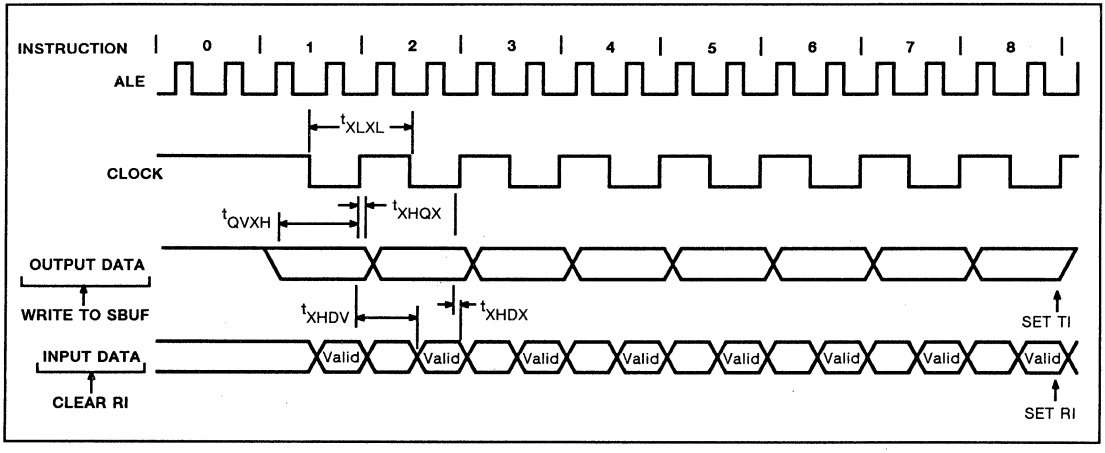


Figure 5. Shift Register Mode Timing

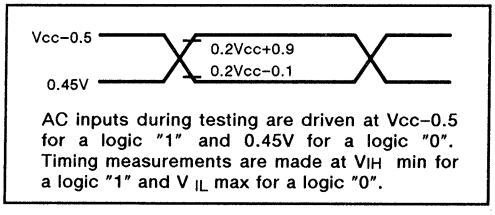


Figure 6. AC Testing Input/Output

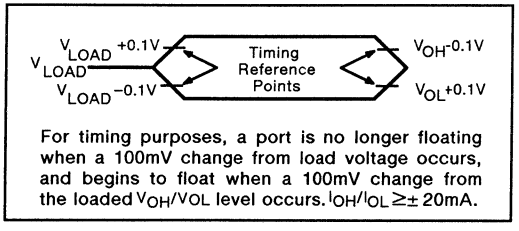


Figure 7. Float Waveform

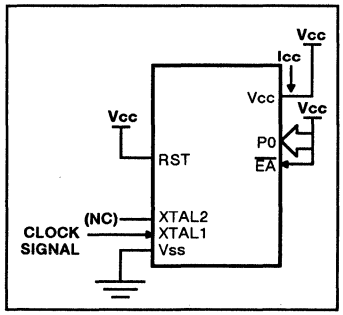


Figure 8. I_{CC} Test Condition, Active Mode
All other pins are disconnected

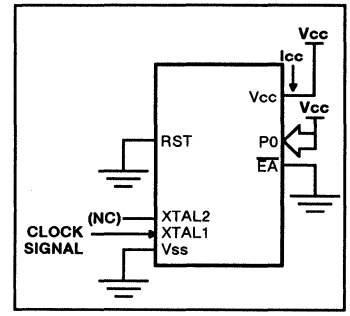


Figure 9. I_{CC} Test Condition, Idle Mode
All other pins are disconnected

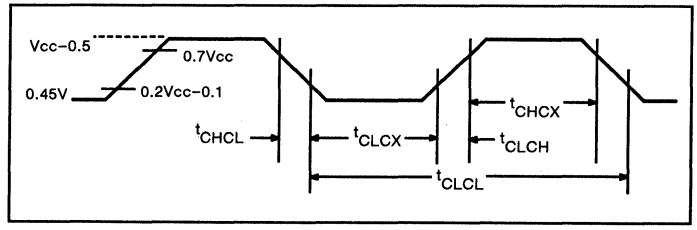


Figure 10. Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes
$t_{CLCH} = t_{CHCL} = 5ns$

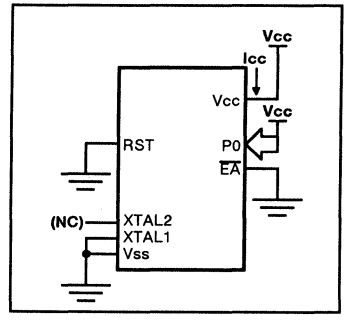


Figure 11. I_{CC} Test Conditions, Power Down Mode
All other pins are disconnected. V_{CC} = 2V to 5.5V

8XC751 OVERVIEW

The Signetics 83C751/87C751 offers the advantages of the SC80C51 architecture in a small package and at a low cost. This microcontroller is fabricated with Signetics high-density CMOS technology. Signetics epitaxial substrate minimizes CMOS lach-up sensitivity. The 83C751/87C751 (hereafter referred to collectively as the 83C751) contains a 2K x 8 ROM/EPROM, a 64 x 8 RAM, 19 I/O lines, a 16-bit auto-reload counter/timer, a fixed rate timer, a five source fixed priority interrupt structure, a bidirectional Inter-Integrated Circuit (I2C) serial bus interface, and an on-chip oscillator. The on-board inter-integrated circuit (I2C) bus interface allows the 83C751 to operate as a master or slave device on the I2C small area network. This capability facilitates I/O and RAM expansion, access to EPROM, processor to processor communication, and efficient interface to a wide variety of dedicated I2C peripherals. The 83C751 has the following features:

- SC80C51 based architecture
- Boolean processor
- Inter-integrated Circuit (I2C) serial bus interface
- Fixed-rate timer
- 16-bit auto reloadable counter/timer
- Small package sizes
  - 24 pin DIP (300 mil "skinny DIP")
  - 28 pin PLCC
- 2K x 8 ROM/EPROM
- Available in erasable quartz lid (87C751), one-time programmable (87C751), or mask programmable versions (83C751)
- Wide oscillator frequency range
- Low power consumption:
  - Normal operation: less than 11mA @ 5V, 12MHz
- Idle mode
- Power-down mode
- CMOS and TTL compatible

This part is well suited for logic replacement in consumer and industrial applications.

DIFFERENCES FROM THE 80C51

Memory Organization

The central processing unit (CPU) manipulates operands in 2 address spaces as shown in Figure 61. The part's internal memory space consists of 2K bytes of program memory, and 64 bytes of data RAM overlapped with the 128 byte special function register area. The differences from the 80C51 are in RAM size (64 bytes vs 128 bytes), in external RAM access (not available on the 83C751), in internal ROM size (2K bytes vs 4k bytes), and in external program memory expansion (not available on the 83C751). The 128 byte special function register (SFR) space is accessed as on the 80C51 with some of the registers having been changed to reflect changes in the 83C751 peripheral functions. The stack may be located anywhere in internal RAM by loading the 8-bit stack pointer (SP). It should be noted that stack depth is limited to 64 bytes, the amount of available RAM. A reset loads the stack pointer with 07F (which is pre-incremented on a PUSH instruction).

Special Function Registers

The 83C751 contains many of the Special Function Registers (SFR) that are found on the 80C51. Due to the different peripheral features on the 83C751, there are several additional SFRs and several that have been changed. There is no port 2 on the 83C751 so the P2 SFR isn't used. The standard UART found on the 80C51 has been replaced by the I2C serial interface, so the UART SFRs, SCON and SBUF, have been replaced by I2CON and I2DAT, and two additional I2C registers have been added (I2STA and I2CFG).

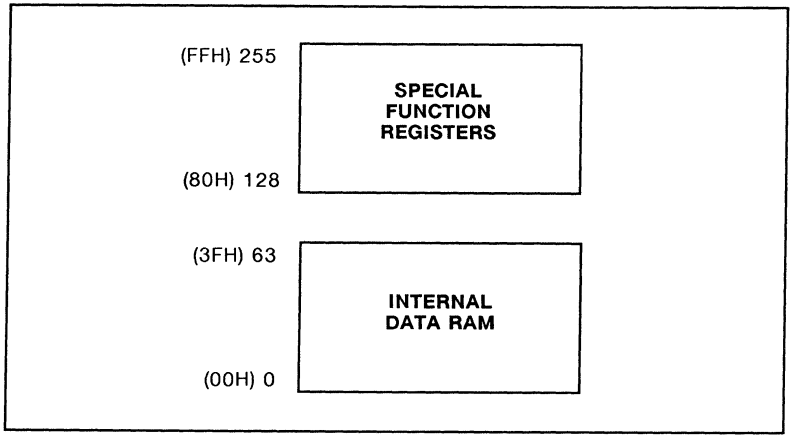


Figure 61. Memory Map

Because the interrupt structure is single level on the 83C751 there is no need for the IP SFR, so it is not used. The counter/timer has only one mode of operation so the TMOD SFR is not used. There is also only one counter/timer so there is no need for the TL1 and TH1 SFRs found on the 80C51. These have been replaced on the 83C751 by RTL and RTH the counter/timer reload registers. Table 20 shows the special function registers, their locations, and reset values.

Data Pointer (DPTR)

The Data Pointer (DPTR) consists of a high byte (DPH) and a low byte (DPL). In the 80C51 this register allows the access of external data memory using the MOVX instruction. Since the 83C751 does not support MOVX or external memory accesses this register is generally used as a 16-bit offset pointer of the accumulator in a MOVC instruction. DPTR may also be manipulated as two independent 8-bit registers.

I/O Port Latches (P0, P1, P3)

The port latches function the same as those on the 80C51. Since there is no Port 2 on the 83C751 the P2 latch is not used. Port 0 on the 83C751 has only 3 bits, so only 3 bits of the P0 SFR have a useful function.

I/O Port Structure

The 8XC751 has 2 eight-bit ports (ports 1 and 3) and 1 three-bit port (port 0). All three ports on the 8XC751 are bi-directional. Each consists of a latch (special function register P0, P1, P3), an output driver, and an input buffer. Three Port 1 pins and two Port 0 pins are multifunctional. In addition to being port pins, these pins serve the function of special features as follows:

Port Pin	Alternate Function
P0.0	I2C clock (SCL)
P0.1	I2C data (SDA)
P1.5	$\overline{\text{INT0}}$ (external interrupt 0 input)
P1.6	$\overline{\text{INT1}}$ (external interrupt 1 input)
P1.7	T0 (timer 0 external input)

Ports 1 and 3 are identical in structure to the same ports on the 80C51. The structure of Port 0 on the 8XC751 is similar to that of the 80C51 but does not include address/data input and output circuitry. As on the 80C51, ports 1 and 3 are quasi-bidirectional while port 0 is bi-directional with no internal pullups.

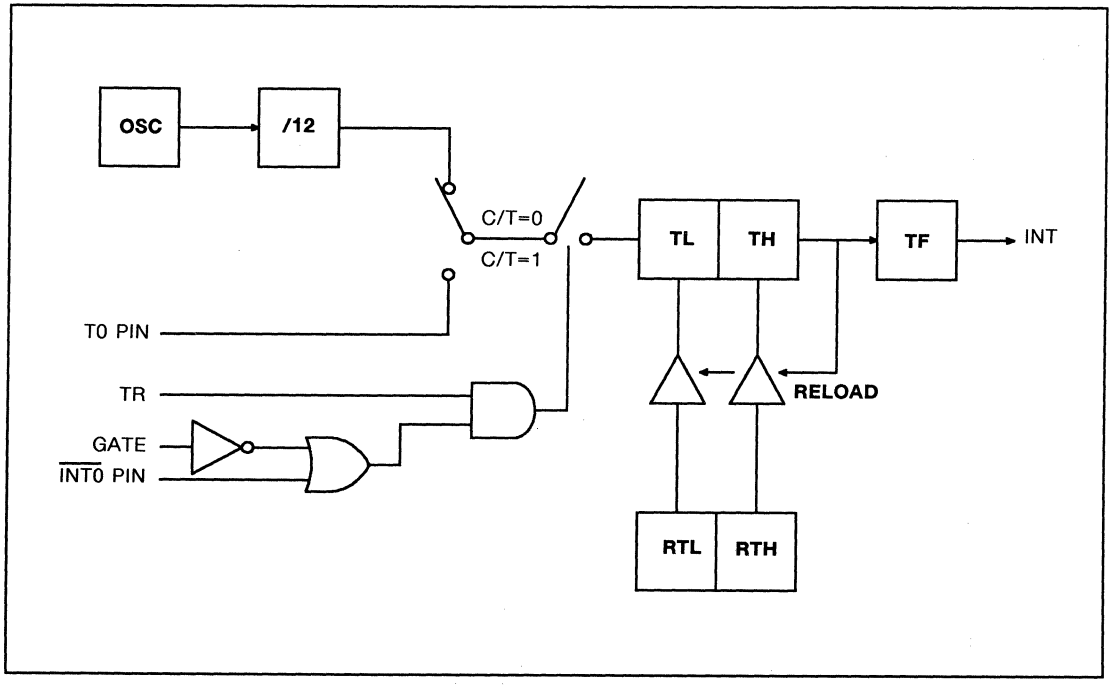


Figure 62. 83C751 Counter/Timer Block Diagram

Table 20. 8XC751 Special Function Registers

Symbol	Description	Direct Address	Bit Address, Symbol or Alternative Port Function MSB LSB								Reset Value
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
DPTR:	Data pointer (2 bytes):										
DPH	High byte	83H									00H
DPL	Low byte	82H									00H
			DF	DE	DD	DC	DB	DA	D9	D8	
I2CFG*#	I2C configuration	D8H/RD	SLAVEN	MASTRQ	0	TIRUN	–	–	CT1	CT0	0000xx00B
		WR	SLAVEN	MASTRQ	CLRTI	TIRUN	–	–	CT1	CT0	
			9F	9E	9D	9C	9B	9A	99	98	
I2CON*#	I2C control	98H/RD	RDAT	ATN	DRDY	ARL	STR	STP	MASTER	–	81H
		WR	CXA	IDLE	CDR	CARL	CSTR	CSTP	XSTR	XSTP	
I2DAT#	I2C data	99H/RD	RDAT	0	0	0	0	0	0	0	80H
		WR	XDAT	X	X	X	X	X	X	X	
			FF	FE	FD	FC	FB	FA	F9	F8	
I2STA*#	I2C control	F8H	–	IDLE	XDATA	XACTV	MAKSTR	MAKSTP	XSTR	XSTP	x0100000B
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*#	Interrupt enable	A8H	EA	–	–	EI2	ETI	EX1	ET0	EX0	00H
P0*#	Port 0	80H	–	–	–	–	–	82	81	80	xxxxx111B
P1*	Port 1	90H	97	96	95	94	93	92	91	90	FFH
P3*	Port 3	B0H	B7	B6	B5	B4	B3	B2	B1	B0	FFH
PCON	Power control	87H	–	–	–	–	–	–	PD	IDL	xxxxxx00B
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	-	P	00H
SP	Stack pointer	81H									07H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*#	Timer/counter control	88H	GATE	C/T	TF	TR	IE0	IT0	IE1	IT1	00H
TL#	Timer low byte	8AH									00H
TH#	Timer high byte	8CH									00H
RTL#	Timer low reload	8BH									00H
RTH#	Timer high reload	8DH									00H

* = SFRs are bit addressable.
\# = SFRs are modified from or added to the 80C51 SFRs.

Timer/Counter

The 8XC751 has two timers: a 16-bit Timer/Counter and a 10-bit fixed rate timer. The 16-bit Timer/Counter's operation is similar to Mode 2 operation on the 80C51, but is extended to 16 bits. The Timer/Counter is clocked by either 1/12 the oscillator frequency or by transitions on the T0 pin. The C/T pin in special function register TCON selects between these two modes. When the TCON TR bit is set, the timer/counter is enabled. Register pair TH and TL are incremented by the clock source. When the register pair overflows, the register pair is reloaded with the values in registers RTH and RTL. The value in the reload registers is left unchanged. See the 83C751 counter/timer block diagram in Figure 62. The TF bit in special function register TCON is set on counter overflow and, if the interrupt is enabled, will generate an interrupt.

TCON Register

7	6	5	4	3	2	1	0
GATE	C/T	TF	TR	IE0	IT0	IE1	IT1
MSB							LSB

GATE 1 – Timer/counter is enabled only when $\overline{INT0}$ pin is high, and TR is 1.
0 – Timer/counter is enabled when TR is 1.
C/T 1 – Counter/timer operation from T0 pin.
0 – Timer operation from internal clock.
TF 1 – Set on overflow of TH.
0 – Cleared when processor vectors to interrupt routine and by reset.
TR 1 – Timer/counter enabled.
0 – Timer/counter disabled.
IE0 1 – Edge detected in $\overline{INT0}$.

IT0 1 – $\overline{INT0}$ is edge triggered.
0 – $\overline{INT0}$ is level sensitive.
IE1 1 – Edge detected on $\overline{INT1}$.
IT1 1 – $\overline{INT1}$ is edge triggered.
0 – $\overline{INT1}$ is level sensitive.

These flags are functionally identical to the corresponding 80C51 flags, except that there is only one timer on the 83C751 and the flags are therefore combined into one register.

The I2C watchdog timer, timer I, is also available as a general purpose fixed rate timer when the I2C interface is not being used. A clock rate of 1/12 the oscillator frequency forms the input to a prescaler. This prescaler can be programmed for 1 of 4 values to give a range of timeout periods (see more discussion in I2C section). Timer I has a timeout interval of 1024 machine cycles. An external reset on this timer is initiated by a transition on the SCL(P0.0) pin.

I2C Serial Interface

The I2C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main technical features of the bus are:

- Bidirectional data transfer between masters and slaves
- Serial addressing of slaves
- Acknowledgment after each transferred byte
- Multimaster bus
- Arbitration between simultaneously transmitting masters without corruption of serial data on bus

A large family of I2C compatible ICs is available. See the I2C section of this manual for more details on the bus and available ICs.

The 83C751 I2C subsystem includes hardware to simplify the software required to drive the I2C bus. The hardware is a single bit interface which in addition to including the necessary arbitration and framing error checks, includes clock stretching and a bus timeout timer. The interface is synchronized to software either through polled loops or interrupts. Six time spans are important in I2C operation and are insured by Timer I:

- The MINIMUM HIGH time for SCL when this device is the master.
- The MINIMUM LOW time for SCL when this device is a master. This is not very important for a single-bit hardware interface like this one, because the SCL low time is stretched until the software responds to the I2C flags. The software response time normally meets or exceeds the MIN LO time. In cases where the software responds within (MIN HI + MIN LO) time, Timer I will insure that the minimum time is met.
- The MINIMUM SCL HIGH TO SDA HIGH time in a stop condition.
- The MINIMUM SDA HIGH TO SDA LOW time between I2C stop and start conditions. (4.7μs see spec.)
- The MINIMUM SDA LOW TO SCL LOW time in a start condition.
- The MAXIMUM SCL CHANGE time while an I2C frame is in progress. A frame is in progress between a Start condition and the following Stop condition. This time span serves to detect a lack of software response on this 8XC751 as well as external I2C problems. SCL "stuck low" indicates a faulty Master or Slave. SCL "stuck high" may mean a faulty device, or that noise induced onto the I2C caused all Masters to withdraw from I2C arbitration.

The first 5 of these times are 4.7μs (see I2C specification) and are covered by the low order 3 bits of Timer I. Timer I is clocked by the 8XC751 oscillator which can vary in frequency from 0.5 to 16MHz. A prescaler with one of 4 divisor values allows Timer I values to be optimized for different oscillator frequencies. At lower frequencies, software response time is increased and will degrade maximum performance of the I2C bus. For 100Khz bus performance, the oscillator rate should be limited to the 8–16MHz range which is the range that the Timer I prescaler was designed for. See special function register I2CFG description for prescale values (CT0, CT1).

The MAXIMUM SCL CHANGE time is important but its exact span is not critical. The complete 10 bits of Timer I are used to count out the maximum time. When I2C operation is enabled, this counter is cleared by transitions on the SCL pin. The timer does not run between I2C frames (i.e. whenever Reset or Stop occurred more recently than the last Start). When this counter is running, it will carry out after 1024 machine cycles have elapsed since a change on SCL. A carry out causes a hardware reset of the 83C751 I2C interface and generates an interrupt if the Timer I interrupt is enabled. In cases where the bus hangup is due to a lack of software response by this 83C751, the reset releases SCL and allows I2C operation among other devices to continue.

I2C Register I2CON

	7	6	5	4	3	2	1	0
Read	RDAT	ATN	DRDY	ARL	STR	STP	MASTER	–
Write	CXA	IDLE	CDR	CSTL	CSTR	CSTP	XSTR	XSTP

Reading I2CON

RDAT The data from SDA is captured into "Receive DATa" whenever a rising edge occurs on SCL. RDAT is also available (with 7 low-order zeros) in the I2DAT register. The difference between reading it here and there is that reading I2DAT clears DRDY, allowing the I2C to proceed on to another bit. Typically, the first 7 bits of a received byte are read from I2DAT, while the 8th is read here. Then, I2DAT can be written to send the Ack bit and clear DRDY.

ATN "ATteNtion" is 1 when one or more of DRDY, ARL, STR, or STP is 1. Thus, ATN comprises a single bit that can be tested to release the I2C service routine from a "wait loop".

DRDY "Data ReaDY" (and thus ATN) is set when a rising edge occurs on SCL, except at idle Slave. DRDY is cleared by writing CDR = 1, or by writing or reading the I2DAT register. The following low period on SCL is stretched until the program responds by clearing DRDY.

Checking ATN and DRDY

When a program detects ATN = 1, it should next check DRDY. If DRDY = 1, then if it receives the last bit it should capture the data from RDAT (in I2DAT or I2CON). Next, if the next bit is to be sent it should be written to I2DAT. One way or another it should clear DRDY and then return to monitoring ATN. Note that if any of ARL, STR, or STP is set, clearing DRDY will not release SCL to high, so that the I2C will not go on to the next bit. If a program detects ATN = 1, and DRDY = 0, it should go on to examine ARL, STR, and STP.

ARL "Arbitration Loss" is 1 when transmit Active was set, but this 83C751 lost arbitration to another transmitter. Transmit Active is cleared when ARL is 1. There are 4 separate cases in which ARL is set.
1. If the program sent a 1 or repeated start, but another device sent a 0, or a stop, so that SDA is 0 at the rising edge of SCL. (If the other device sent a Stop, the setting of ARL will be followed shortly by STP being sent.)
2. If the program sent a 1, but another device sent a repeated start, and it drove SDA low before the 83C751 could drive SCL low. (This type of ARL is always accompanied by STR = 1.)
3. In Master mode, if the program sent a repeated Start, but another device sent a 1, and it drove SCL low before this 83C751 could drive SDA low.
4. In Master mode, if the program sent Stop, but it could not be sent because another device sent a 0.

STR "STaRt" is set to a 1 when an I2C Start condition is detected at a non-idle Slave or at a Master. (STR is not set when an idle Slave becomes active due to a Start bit; the Slave has nothing useful to do until the rising edge of SCL sets DRDY.)

STP "SToP" is set to 1 when an I2C Stop condition is detected at a non-idle Slave or at a Master. (STP is not set for a Stop condition at an idle Slave.)

MASTER "MASTER" is 1 if this 83C751 is currently a Master on the I2C. MASTER is set when MASTRQ is 1 and the bus is not busy (i.e., if a start bit hasn't been received since Reset or a "Timer I" time-out, or if a Stop has been received since the last Start). MASTER is cleared when ARL is set, or after the software writes MASTRQ = 0 and then XSTP = 1.

Writing I2CON

Typically, for each bit in an I2C message, a service routine waits for ATN = 1. Based on DRDY, ARL, STR, and STP, and on the current bit position in the message, it may then write I2CON with one or more of the following bits, or it may read or write the I2DAT register.

CXA Writing a 1 to "Clear Xmit Active" clears the Transmit Active state. (Reading the I2DAT register also does this.)

Regarding Transmit Active

Transmit Active is an internal state in the I2C interface and is not directly testable. Transmit Active is set by writing the I2DAT register, or by writing I2CON with XSTR = 1 or XSTP = 1. The I2C interface will only drive the SDA line low when Transmit Active is set, and the ARL bit will only be set to 1 when Transmit Active is set. Transmit Active is cleared by reading the I2DAT register, or by writing I2CON with CXA = 1. Transmit Active is automatically cleared when ARL is 1.

IDLE Writing 1 to "IDLE" causes a Slave's I2C hardware to ignore the I2C until the next Start condition (but if MASTRQ is 1 then a Stop condition will make the 83C751 into a Master).

CDR Writing a 1 to "Clear Data Ready" clears DRDY. (Reading or writing the I2DAT register also does this.)

CARL Writing a 1 to "Clear Arbitration Loss" clears the ARL bit.

CSTR Writing a 1 to "Clear STaRt" clears the STR bit.

CSTP Writing a 1 to "Clear SToP" clears the STP bit. Note that if one or more of DRDY, ARL, STR, or STP is 1, the low time of SCL is stretched until the service routine responds by clearing them.

XSTR Writing 1's to "Xmit repeated STaRt" and CDR tells the I2C hardware to send a Repeated Start condition. This should only be at a Master. Note that XSTR need not and should not be used to send an "initial" (non-repeated) Start; it is sent automatically by the I2C hardware. Writing XSTR = 1 includes the effect of writing I2DAT with XDAT = 1; it sets Transmit Active and releases SDA to high during the SCL low time. After SCL goes high, the I2C hardware waits for the suitable minimum time and then drives SDA low to make the Start condition.

XSTP Writing 1's to "Xmit SToP" and CDR tells the I2C hardware to send a Stop condition. This should only be done at a Master. If there are no more messages to initiate, the service routine should clear the MASTRQ bit in I2CFG to 0 before writing XSTP with 1. Writing XSTP = 1 includes the effect of writing I2DAT with XDAT = 0; it sets Transmit Active and drives SDA low during the SCL low time. After SCL goes high, the I2C hardware waits for the suitable minimum time and then releases SDA to high to make the Stop condition.

I2C Register I2DAT

	7	6	5	4	3	2	1	0
Read	RDAT	0	0	0	0	0	0	0
Write	XDAT	X	X	X	X	X	X	X

RDAT "Receive DATa" is captured from SDA every rising edge of SCL. Reading I2DAT also clears DRDY and the Transmit Active state.

XDAT "Xmit Data" sets the data for the next bit. Writing I2DAT also clears DRDY and sets the Transmit Active state.

Regarding Software Response Time

Because the 83C751 can run at 16MHz, and because the I2C interface is optimized for high-speed operation, it is quite likely that an I2C service routine will sometimes respond to DRDY (which is set at a rising edge of SCL) and write I2DAT before SCL has gone low again. If XDAT were applied directly to SDA, this situation would produce an I2C protocol violation. The programmer need not worry about this possibility because XDAT is applied to SDA only when SCL is low.

Conversely, a program that includes an I2C service routine may take a long time to respond to DRDY. Typically, an I2C routine operates on a flag-polling basis during a message, with interrupts from other peripheral functions enabled. If an interrupt occurs it will delay the response of the I2C service routine. The programmer need not worry about this very much either, because the I2C hardware stretches the SCL low time until the service routine responds. The only constraint on the response is that it must not exceed the Timer I time-out, which is at least 768 microseconds.

I2C Register I2CFG

	7	6	5	4	3	2	1	0
Read	SLAVEN	MASTRQ	0	TIRUN	–	–	CT1	CT0
Write	SLAVEN	MASTRQ	CLRT1	TIRUN	–	–	CT1	CT0

SLAVEN Writing a 1 to "SLAVe ENable" enables the Slave functions of the I2C subsystem. If SLAVEN and MASTRQ are 0, the I2C hardware is disabled. This bit is cleared to 0 by reset and by an I2C time-out.

MASTRQ Writing a 1 to "MASTRQ" requests mastership of the I2C. If a frame from another Master is in progress when this bit is changed from 0 to 1, action is delayed until a stop condition is detected. Then, or immediately if a frame is not in progress, a start condition is sent and DRDY is set (thus making ATN 1 and generating an I2C interrupt). When a Master wishes to release mastership status of the I2C, it writes a 1 to XSTP in I2CON. MASTRQ is cleared by Reset and by an I2C time-out.

CLRTI Writing a 1 to this bit clears the Timer I interrupt flag. This bit position always reads as a 0.

TIRUN Writing a 1 to this bit lets Timer I run; a zero stops and clears it. Together with SLAVEN, MASTRQ, and MASTER, this bit determines operational modes as shown in Table 21.

CT1,0 These two bits are programmed as a function of the OSC rate, to optimize the MIN HI and LO time of SCL when this 83C751 is a master on the I2C. The time value determined by these bits controls both of these parameters, and also the timing for Stop and Start conditions. These bits are cleared to 00 by reset.

The value that should be programmed in these bits is given in Table 22, for integer-MHz crystal values. For other values, the controlling factor is that the MIN TIME must be greater than or equal to 4.7μsec. (MIN TIME is a range because I2C events are not synchronized to osc/12.)

The maximum oscillator frequency (MHz) for a given CT1, 0 value is given by:

$$\text{fosc max} = \frac{(\text{osc/12 count} - 0.42) * 12}{4.7}$$

The 4.7 in the denominator is the minimum LOW time for I2C in microseconds.

Table 21. Interaction of TIRUN with SLAVEN, MASTRQ, and MASTER

SLAVEN, MASTRQ, MASTER	TIRUN	Operating Mode
All 0	0	The I2C interface is disabled. Timer I is cleared and does not run. This is the state assumed after a reset. If an I2C application wants to ignore the I2C at certain times, it should write SLAVEN, MASTRQ, and TIRUN all to zero.
All 0	1	The I2C interface is disabled. Timer I operates as a free-running time base. Use this mode only in non-I2C applications.
Any or all 1	0	The I2C interface is enabled. The 3 low-order bits of Timer I run for min-time generation. but the hi-order bits do not, so that there is no checking for I2C being 'hung'. This configuration can be used for very slow I2C operation.
Any or all 1	1	The I2C interface is enabled. Timer I runs during frames on the I2C, and is cleared by transitions on SCL, and by Start and Stop conditions. This is the normal state for I2C operation.

Table 22. CT1, CT0 Values

CT1, CT0 Values	Osc/12 Count
10	7
01	6
00	5
11	4

I2C Register I2STA

	7	6	5	4	3	2	1	0
Read	–	IDLE	XDATA	XACTV	MAKSTR	MAKSTP	XSTR	XSTP
	MSB							LSB

This register is read only and reflects the internal status of the I2C hardware. IDLE, XSTR, and XSTP reflect the status of the like named bits in the I2CON register.

XDATA	The content of the transmitter buffer.
XACTV	Transmitter active.
MAKSTR	This bit is high while the hardware is effecting a Start condition.
MAKSTP	This bit is high while the hardware is effecting a Stop condition.
XSTR	This bit is active while the hardware is effecting a repeated Start Condition.
XSTP	This bit is active while the hardware is effecting a repeated Stop Condition.

Instruction Set

The instruction set of the 83C751 is identical to the 80C51 except for the instructions: MOVX, LCALL, and LJUMP which are not implemented.

Interrupts

The 8XC751 has 5 interrupt sources with fixed priority levels. Interrupt sources common to the 80C51 are the external interrupts ($\overline{INT0}$, $\overline{INT1}$) and the timer/counter interrupt (ET0). The I2C interrupt (EI2) and Timer I interrupt (ETI) are the other two interrupt sources.

Upon interrupt or reset the program counter is loaded with specific values for the appropriate interrupt service routine in program memory. These values are:

Event	Program Memory Address	Priority
Reset	000	Highest
$\overline{INT0}$	003	
Counter/Timer 0	00B	
$\overline{INT1}$	013	
Timer I	01B	
I2C	023	Lowest

The interrupt enable register (IE) is used to individually enable or disable the 5 sources. Bit $\overline{EA}$ in the interrupt enable register can be used to globally enable or disable all interrupt sources. The interrupt enable register is described below. All other interrupt details are based on the 80C51 interrupt architecture.

Interrupt Enable Register IE

$\overline{EA}$	X	X	EI2	ETI	EX1	ET0	EX0

Symbol	Position	Function
$\overline{EA}$	IE.7	Disables all interrupts. If $\overline{EA}$ = 0, no interrupt will be acknowledged. If $\overline{EA}$ = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
–	IE.6	Reserved
–	IE.5	Reserved
EI2	IE.4	Enables or disables the I2C interrupt. If ES = 0, the I2C interrupt is disabled.
ETI	IE.3	Enables or disables the Timer I overflow interrupt. IF ETI = 0, the Timer I interrupt is disabled.
EX1	IE.2	Enables or disables external interrupt 1. If EX1 = 0, external interrupt 1 is disabled.
ET0	IE.1	Enables or disables the Timer 0 overflow interrupt. If ET0 = 0, the Timer 0 interrupt is disabled.
EX0	IE.0	Enables or disables external interrupt 0. If EX0 = 0, external interrupt 0 is disabled.

Signetics

Microprocessor Division

S83C751
CMOS Single-Chip 8-Bit Microcontroller

Product Specification

DESCRIPTION

The Signetics S83C751 offers many of the advantages of the SC80C51 architecture in a small package and at low cost.

The S83C751 Microcontroller is fabricated with Signetics high-density CMOS technology. Signetics' epitaxial substrate minimizes CMOS latch-up sensitivity.

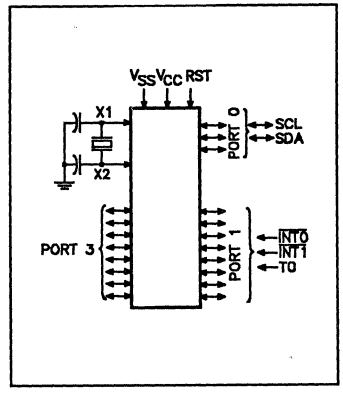
The S83C751 contains a 2K x 8 ROM, a 64 x 8 RAM, 19 I/O lines, a 16-bit counter/timer, a fixed-rate timer, a five-source fixed-priority interrupt structure, a bidirectional Inter-Integrated Circuit (I2C) serial bus interface, and an on-chip oscillator.

The onboard inter-integrated circuit (I2C) bus interface allows the S83C751 to operate as a master or slave device on the I2C small area network. This capability facilitates I/O and RAM expansion, access to EEPROM, processor-to-processor communication, and efficient interface to a wide variety of dedicated I2C peripherals.

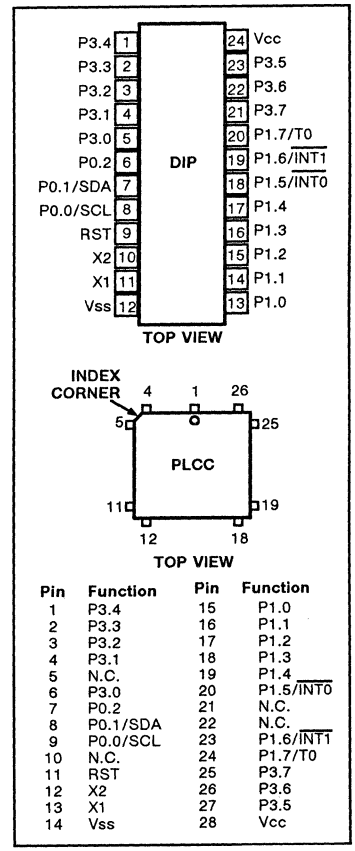
FEATURES

- **SC80C51 based architecture**
- **Inter-Integrated Circuit (I2C) serial bus interface**
- **Small package sizes**
  - **24-pin DIP (300 mil "skinny DIP")**
  - **28-pin PLCC**
- **Wide power supply voltage range**
- **Wide oscillator frequency range**
- **Low power consumption:**
  - **Normal operation: less than 24mA @ 5V, 12MHz**
  - **Idle mode**
  - **Power-down mode**
- **2K x 8 ROM, 64 x 8 RAM**
- **16-bit counter/timer**
- **Fixed-rate timer**
- **Boolean processor**
- **CMOS and TTL compatible**
- **Well suited for logic replacement, consumer and industrial applications**

LOGIC SYMBOL



PIN CONFIGURATION



Pin	Function	Pin	Function
1	P3.4	15	P1.0
2	P3.3	16	P1.1
3	P3.2	17	P1.2
4	P3.1	18	P1.3
5	N.C.	19	P1.4
6	P3.0	20	P1.5/INT0
7	P0.2	21	N.C.
8	P0.1/SDA	22	N.C.
9	P0.0/SCL	23	P1.6/INT1
10	N.C.	24	P1.7/T0
11	RST	25	P3.7
12	X2	26	P3.6
13	X1	27	P3.5
14	Vss	28	Vcc

ORDERING INFORMATION

S83C751– □□ (CVxxxx)

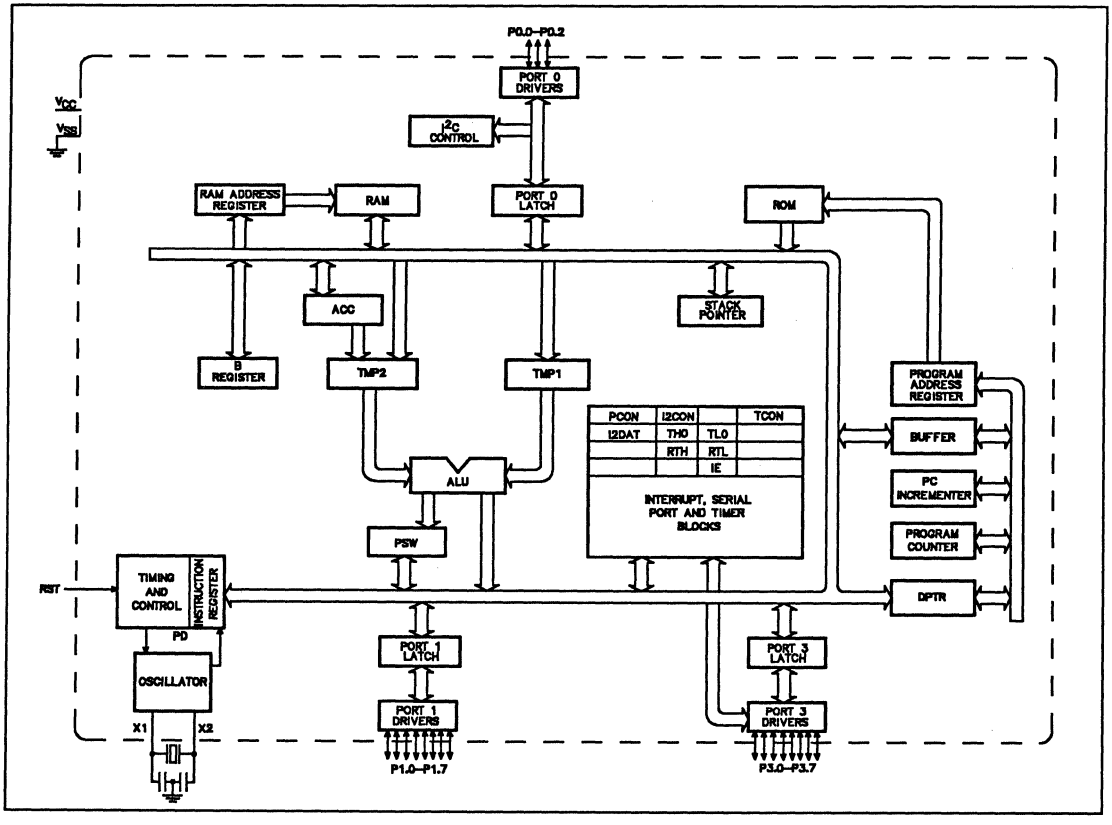
Custom ROM Pattern No.
Applies to masked ROM versions only. Number will be assigned by Signetics. Contact Signetics sales office for ROM pattern submission requirements.

Package Codes:
N24 = Plastic DIP
A28 = Plastic PLCC

Speed and Temperature Range:
1 = 3.5 to 12MHz, 0°C to +70°C
2 = 3.5 to 12MHz, –40°C to +85°C
3 = 0.5 to 12MHz, 0°C to +70°C
4 = 3.5 to 16MHz, 0°C to +70°C
5 = 3.5 to 16MHz, –40°C to +85°C

PART NUMBER SELECTION		
Part Number	**Speed**	**Temperature and Package**
S83C751–1N24	3.5 to 12MHz	0 to +70°C, Plastic DIP
S83C751–2N24	3.5 to 12MHz	–40 to +85°C, Plastic DIP
S83C751–3N24	0.5 to 12MHz	0 to +70°C, Plastic DIP
S83C751–4N24	3.5 to 16MHz	0 to +70°C, Plastic DIP
S83C751–5N24	3.5 to 16MHz	–40 to +85°C, Plastic DIP
S83C751–1A28	3.5 to 12MHz	0 to +70°C, Plastic LCC
S83C751–2A28	3.5 to 12MHz	–40 to +85°C, Plastic LCC
S83C751–3A28	0.5 to 12MHz	0 to +70°C, Plastic LCC
S83C751–4A28	3.5 to 16MHz	0 to +70°C, Plastic LCC
S83C751–5A28	3.5 to 16MHz	–40 to +85°C, Plastic LCC

BLOCK DIAGRAM



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. PLCC	TYPE	NAME AND FUNCTION
V_{SS}	12	14	I	**Circuit ground potential.**
V_{CC}	24	28	I	**Supply voltage during normal, idle, and power–down operation.**
P0.0–P0.2	8–6	9–7	I/O	**Port 0:** Port 0 is a 3-bit open-drain bidirectional port. Port 0 pins that have ones written to them float, and in that state can be used as high-impedance inputs. Port 0 also serves as the serial I2C interface as shown in the pinout diagram. When this feature is activated by software, SCL and SDA are driven low in accordance with the I2C protocol. These pins are driven low if the port register bit is written with a 0 or if the I2C subsystem presents a 0. The state of the pin can always be read from the port register by the program. To comply with the I2C specification, P0.0 and P0.1 are open-drain bidirectional I/O pins with the electrical characteristics listed in the tables that follow. While these differ from 'standard TTL' characteristics, they are close enough for the pins to still be used as general-purpose I/O in non-I2C applications.
	7	8	I/O	**SDA (P0.1)** I2C data.
	8	9	I/O	**SCL (P0.0)** I2C clock.
P1.0–P1.7	13–20	15–20, 23, 24	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pullups. Port 1 pins that have ones written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pullups. (See DC electrical characteristics: I_{IL}). Port 1 also serves the special function features of the SC80C51 family as listed below:
	18	20		**$\overline{INT0}$ (P1.5):** External interrupt
	19	23		**$\overline{INT1}$ (P1.6):** External interrupt
	20	24		**T0 (P1.7):** Timer 0 external input
P3.0–P3.7	5–1 23–21	4–1, 6, 27–25	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 3 pins that have ones written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pullups (See DC electrical characteristics: I_{IL}).
RST	9	11	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal diffused resistor to V_{SS} permits a power-on RESET using only an external capacitor to V_{CC}.
X1	11	13	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits.
X2	10	12	O	**Crystal 2:** Output from the inverting oscillator amplifier.

OSCILLATOR CHARACTERISTICS

X1 and X2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in the logic symbol.

To drive the device from an external clock source, X1 should be driven while X2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. To insure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Port 0	Port1	Port 2
Idle	Data	Data	Data
Power–down	Data	Data	Data

DIFFERENCES BETWEEN THE S83C751 AND THE SC80C51

Program Memory

On the S83C751, program memory is 2048 bytes long and is not externally expandable. Program memory can contain S83C751 instructions and constant data. The only fixed allocations in program memory are the addresses at which execution is taken up in response to reset and to interrupts, which are as follows:

Event	Program Memory Address
Reset	000
External INT0	003
Counter/timer 0	00B
External INT1	013
Timer I	01B
I2C serial	023

Counter/Timer Subsystem

The S83C751 has one counter/timer called timer/counter 0. Its operation is similar to mode 2 operation on the SC80C51, but is extended to 16 bits with 16 bits of autoload. The controls for this counter are centralized in a single register called TCON.

A watchdog timer, called Timer I, is for use with the I2C subsystem. In I2C applications, this timer is dedicated to time-generation and bus monitoring of the I2C. In non-I2C applications, it is available for use as a fixed timebase.

Interrupt Subsystem – Fixed Priority

The IP register and the 2-level interrupt system of the SC80C51 are eliminated. Simultaneous interrupt conditions are resolved by a single-level, fixed priority as follows:

Highest priority:	Pin INT0
	Counter/timer flag 0
	Pin INT1
	Timer I
Lowest priority:	Serial I2C

Serial Communications

The S83C751 contains an I2C serial communications port instead of the SC80C51 UART. The I2C serial port is a single bit hardware interface with all of the hardware necessary to support multimaster and slave operations. Also included are receiver digital filters and timer (timer I) for communication watch-dog purposes. The I2C serial port is controlled through four special function registers; I2C control, I2C data, I2C status, and I2C configuration.

Special Function Register Addresses

Special function register addresses for the S83C751 are identical to those of the SC80C51, except for the changes listed below:

SC80C51 special function registers not present in the S83C751 are TMOD (89), P2 (A0) and IP (B8). The SC80C51 registers TH1, TL1, SCON, and SBUF are replaced with the S83C751 registers RTH, RTL, I2CON, and I2DAT, respectively. Additional special function registers are I2CFG (D8) and I2STA (F8).

Table 2. I2C Special Function Register Addresses

Register Address			Bit Address							
Name	Symbol	Address	MSB							LSB
I2C control	I2CON	98	9F	9E	9D	9C	9B	9A	99	98
I2C data	I2DAT	99	–	–	–	–	–	–	–	–
I2C configuration	I2CFG	D8	DF	DE	DD	DC	DB	DA	D9	D8
I2C status	I2STA	F8	FF	FE	FD	FC	FB	FA	F9	F8

ABSOLUTE MAXIMUM RATINGS

SYMBOL	PARAMETER	RATING	UNIT
T_{STG}	Storage temperature range	−65 to +150	°C
V_{CC}	Voltage from V_{cc} to V_{SS}	−0.5 to +6.5	V
V_S	Voltage from any pin to V_{SS}	−0.5 to V_{CC} + 0.5	V
P_D	Power dissipation	1.0	W

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −45°C to +85°C, V_{CC} = 4.0V to 6.0V, V_{SS} = 0V[3]

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS		UNIT
			Min	Max	
V_{IL} V_{IH} V_{IH1}	Input low voltage, except SDA, SCL Input high voltage, except X1, RST Input high voltage, X1, RST		−0.5 0.2V_{CC}+0.9 0.7V_{CC}	0.2V_{CC}−0.1 V_{CC}+0.5 V_{CC}+0.5	V V V
 V_{IL1} V_{IH2}	SDA, SCL: Input low voltage Input high voltage		 −0.5 0.7V_{CC}	 0.3V_{CC} V_{CC}+0.5	 V V
V_{OL} V_{OL1}	Output low voltage, ports 1 and 3 Output low voltage, port 0.2	I_{OL} = 1.6mA I_{OL} = 3.2mA		0.45 0.45	V V
V_{OH}	Output high voltage, ports 1 and 3	I_{OH} = −60µA I_{OH} = −25µA I_{OH} = −10µA	2.4 0.75V_{CC} 0.9V_{CC}		V V V
V_{OL2} C	Port 0.0 and 0.1 (I2C) – Drivers Output low voltage Driver, receiver combined: Capacitance	I_{OL} = 3mA (over V_{CC} range)		0.4 10	V pF
I_{IL} I_{TL} I_{LI}	Logical 0 input current, ports 1 and 3 Logical 1 to 0 transition current, ports 1 and 3[8] Input leakage current, port 0	V_{IN} = 0.45V V_{IN} = 2V 0.45 < V_{IN} < V_{CC}		−50 −650 ±10	µA µA µA
R_{RST}	Reset pull-down resistor		25	175	kohm
C_{IO}	Pin capacitance	Test freq = 1MHz, T_A = 25°C		10	pF
I_{PD}	Power-down current[4]	V_{CC} = 2 to 6.0V		50	µA
I_{CC}	Supply current (see Figure 3)				

AC SYMBOL DESIGNATIONS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

C – Clock
D – Input data
H – Logic level high
L – Logic level low
Q – Output data
T – Time
V – Valid
X – No longer a valid logic level
Z – Float

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −45°C to +85°C , V_{CC} = 5V±20%, V_{SS} = 0V[3, 7]

SYMBOL	PARAMETER	12MHz		VARIABLE CLOCK		UNIT
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator frequency			3.5 3.5 0.5	12 16 12	MHz
	External clock (Figure 1)					
t_{CHCH}	High time	20				ns
t_{CLCL}	Low time	20				ns
t_{CLCH}	Rise time		20			ns
t_{CHCL}	Fall time		20			ns

NOTES:

1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages with respect to V_{SS} unless otherwise noted.
4. Power-down I_{CC} is measured with all output pins disconnected; port 0 = V_{CC}; X2, X1 n.c.; RST = V_{SS}.
5. Active I_{CC} is measured with all output pins disconnected; X1 driven with t_{CLCH}, t_{CHCL} = 5ns, V_{IL} = V_{SS} + 0.5V, V_{IH} = V_{CC} − 0.5V; X2 n.c.; RST = port 0 = V_{CC}. I_{CC} will be slightly higher if a crystal oscillator is used.
6. Idle I_{CC} is measured with all output pins disconnected; X1 driven with t_{CLCH}, t_{CHCL} = 5ns, V_{IL} = V_{SS} + 0.5V, V_{IH} = V_{CC} − 0.5V; X2 n.c.; port 0 = V_{CC}; RST = Vss.
7. Load capacitance for ports = 80pF.
8. Pins of ports 1 and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.

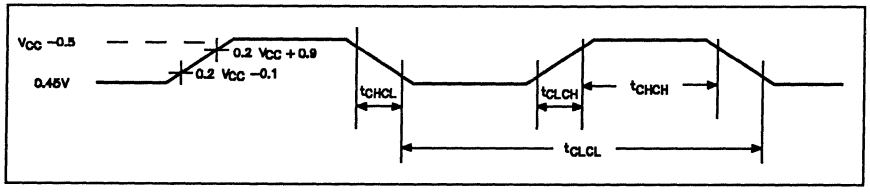


Figure 1. External Clock Drive Waveform

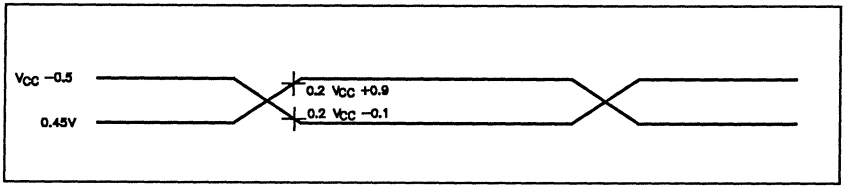


Figure 2. AC Testing Input/Output Waveform

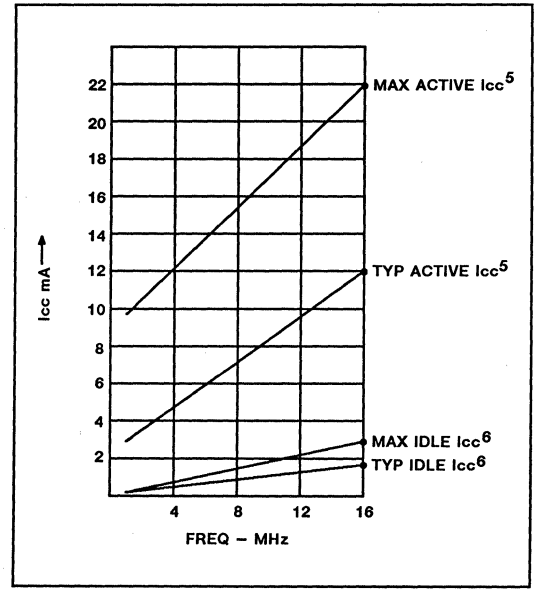


Figure 3. I_{CC} vs. FREQ
Maximum I_{CC} values taken at V_{CC} = 6.0V and worst case temperature.
Typical I_{CC} values taken at V_{CC} = 5.0V and 25°C.
Notes 5 and 6 refer to AC Electrical Characteristics.

Signetics

Microprocessor Division

S87C751
CMOS Single-Chip 8-Bit Microcontroller

Product Specification

DESCRIPTION

The Signetics S87C751 offers many of the advantages of the SC80C51 architecture in a small package and at low cost.

The S87C751 Microcontroller is fabricated with Signetics high-density CMOS technology. Signetics epitaxial substrate minimizes CMOS latch-up sensitivity.

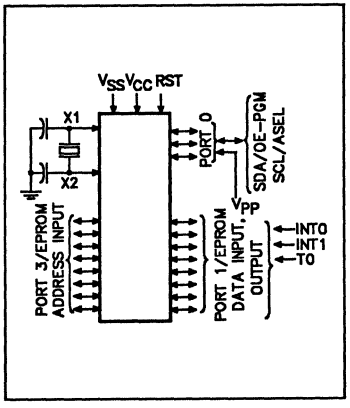
The S87C751 contains a 2K x 8 EPROM, a 64 x 8 RAM, 19 I/O lines, a 16-bit auto-reload counter/timer, a fixed-rate timer, a five-source fixed-priority interrupt structure, a bidirectional Inter-Integrated Circuit (I2C) serial bus interface, and an on-chip oscillator.

The onboard inter-integrated circuit (I2C) bus interface allows the S87C751 to operate as a master or slave device on the I2C small area network. This capability facilitates I/O and RAM expansion, access to EEPROM, processor-to-processor communication, and efficient interface to a wide variety of dedicated I2C peripherals.

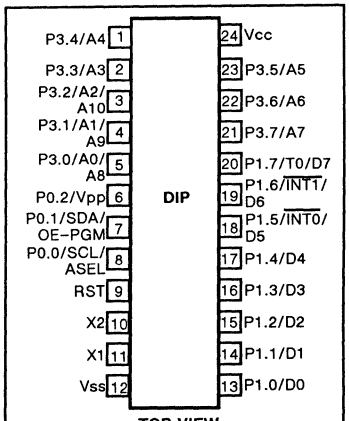
FEATURES

- **EPROM version of S83C751**
- **SC80C51 based architecture**
- **Inter-Integrated Circuit (I2C) serial bus interface**
- **Small package sizes**
  - **24 pin DIP (300 mil "skinny DIP")**
  - **28 pin PLCC**
- **Available in erasable quartz lid or one-time programmable plastic packages**
- **Wide oscillator frequency range**
- **Low power consumption:**
  - **Normal operation: less than 11mA @ 5V, 12MHz**
  - **Idle mode**
  - **Power-down mode**
- **2K x 8 EPROM, 64 x 8 RAM**
- **16-bit auto reloadable counter/timer**
- **Fixed-rate timer**
- **Boolean processor**
- **CMOS and TTL compatible**
- **Well suited for logic replacement, consumer and industrial applications**

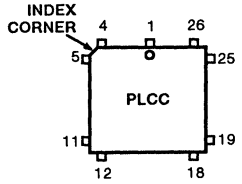
LOGIC SYMBOL



PIN CONFIGURATION



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	P3.4/A4	15	P1.0/D0
2	P3.3/A3	16	P1.1/D1
3	P3.2/A2/A10	17	P1.2/D2
4	P3.1/A1/A9	18	P1.3/D3
5	N.C.	19	P1.4/D4
6	P3.0/A0/A8	20	P1.5/INT0/D5
7	P0.2/Vpp	21	N.C.
8	P0.1/SDA/OE–PGM	22	N.C.
		23	P1.6/INT1/D6
9	P0.0/SCL/ASEL	24	P1.7/T0/D7
		25	P3.7/A7
10	N.C.	26	P3.6/A6
11	RST	27	P3.5/A5
12	X2	28	Vcc
13	X1		
14	Vss		

ORDERING INFORMATION

S87C751– □□

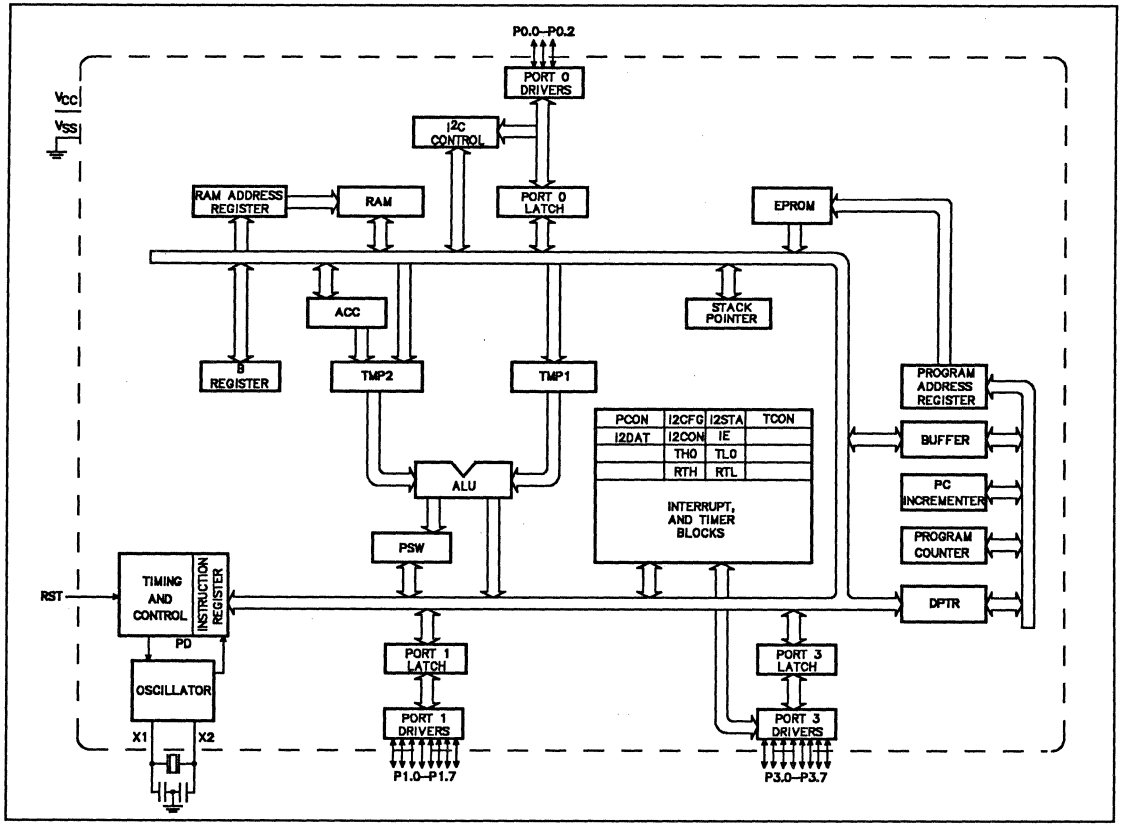
Package Codes:
F24 = Ceramic DIP
N24 = Plastic DIP (OTP)
A28 = Plastic PLCC (OTP)

Speed and Temperature Range:
1 = 3.5 to 12MHz, 0°C to +70°C
2 = 3.5 to 12MHz, –40°C to +85°C
3 = 0.5 to 12MHz, 0°C to +70°C
4 = 3.5 to 16MHz, 0°C to +70°C
5 = 3.5 to 16MHz, –40°C to +85°C

PART NUMBER SELECTION		
Part Number	**Speed**	**Temperature and Package**
S87C751–1F24	3.5 to 12MHz	0 to +70°C, Ceramic DIP
S87C751–3F24	0.5 to 12MHz	0 to +70°C, Ceramic DIP
S87C751–4F24	3.5 to 16MHz	0 to +70°C, Ceramic DIP
S87C751–1N24	3.5 to 12MHz	0 to +70°C, Plastic DIP
S87C751–2N24	3.5 to 12MHz	–40 to +85°C, Plastic DIP
S87C751–3N24	0.5 to 12MHz	0 to +70°C, Plastic DIP
S87C751–4N24	3.5 to 16MHz	0 to +70°C, Plastic DIP
S87C751–5N24	3.5 to 16MHz	–40 to +85°C, Plastic DIP*
S87C751–1A28	3.5 to 12MHz	0 to +70°C, Plastic LCC*
S87C751–2A28	3.5 to 12MHz	–40 to +85°C, Plastic LCC*
S87C751–3A28	0.5 to 12MHz	0 to +70°C, Plastic LCC*
S87C751–4A28	3.5 to 16MHz	0 to +70°C, Plastic LCC*
S87C751–5A28	3.5 to 16MHz	–40 to +85°C, Plastic LCC*

*Preliminary Specification

BLOCK DIAGRAM



PIN CONFIGURATION

MNEMONIC	PIN NO. DIP	PIN NO. PLCC	TYPE	NAME AND FUNCTION
V_{SS}	12	14	I	**Circuit ground potential.**
V_{CC}	24	28	I	**Supply voltage during normal, idle, and power-down operation.**
P0.0–P0.2	8–6	9–7	I/O	**Port 0:** Port 0 is a 3-bit open-drain bidirectional port. Port 0 pins that have ones written to them float, and in that state can be used as high-impedance inputs. Port 0 also serves as the serial I2C interface as shown in the pinout diagram. When this feature is activated by software, SCL and SDA are driven low in accordance with the I2C protocol. These pins are driven low if the port register bit is written with a 0 or if the I2C subsystem presents a 0. The state of the pin can always be read from the port register by the program. To comply with the I2C specification, P0.0 and P0.1 are open drain bidirectional I/O pins with the electrical characteristics listed in the tables that follow. While these differ from 'standard TTL' characteristics, they are close enough for the pins to still be used as general-purpose I/O in non-I2C applications. Port 0 also provides alternate functions for programming the EPROM memory as follows:
	6	7	N/A	**V_{pp} (P0.2)** – Programming voltage input.
	7	8	I	**OE/PGM (P0.1)** – input, OE/PGM, which specifies verify mode (output enable) or the program mode. OE/PGM = 1 output enabled (verify mode) OE/PGM = 0 program mode.
	8	9	I	**ASEL (P0.0)** – input which indicates which bits of the EPROM address are applied to port 3. ASEL = 0 low address byte available on port 3. ASEL = 1 high address byte is available on port 3 (only the three least significant bits are used).
	7	8	I/O	**SDA (P0.1)** I2C data.
	8	9	I/O	**SCL (P0.0)** I2C clock.
P1.0–P1.7	13–20	15–20, 23, 24	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pullups. Port 1 pins that have ones written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pullups. (See DC electrical characteristics: I_{IL}). Port 1 serves to output the addressed EPROM contents in the verify mode and accepts as inputs the value to program into the selected address during the program mode. Port 1 also serves the special function features of the SC80C51 family as listed below:
	18	20		**$\overline{INT0}$ (P1.5):** External interrupt
	19	23		**$\overline{INT1}$ (P1.6):** External interrupt
	20	24		**T0 (P1.7):** Timer 0 external input
P3.0–P3.7	5–1 23–21	4–1, 6, 27–25	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 3 pins that have ones written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pullups (See DC electrical characteristics: I_{IL}). Port 3 also functions as the address input for the EPROM memory location to be programmed (or verified). The 11-bit address is multiplexed into this port as specified by P0.0/ASEL.
RST	9	11	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running resets the device. An internal diffused resistor to V_{SS} permits a power-on RESET using only an external capacitor to V_{CC}. After the device is reset, a 10-bit serial sequence, sent LSB first, applied to RESET, places the device in the programming state allowing programming address, data and V_{PP} to be applied for programming or verification purposes. The RESET serial sequence must be synchronized with the X1 input.
X1	11	13	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits. X1 also serves as the clock to strobe in a serial bit stream into RESET to place the device in the programming state.
X2	10	12	O	**Crystal 2:** Output from the inverting oscillator amplifier.

OSCILLATOR CHARACTERISTICS

X1 and X2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator.

To drive the device from an external clock source, X1 should be driven while X2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (24 oscillator periods), while the oscillator is running. To insure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Port 0	Port1	Port 2
Idle	Data	Data	Data
Power-down	Data	Data	Data

DIFFERENCES BETWEEN THE S87C751 AND THE SC80C51

Program Memory

On the S87C751, program memory is 2048 bytes long and is not externally expandable. Program memory can contain S87C751 instructions and constant data. The only fixed locations in program memory are the addresses at which execution is taken up in response to reset and to interrupts, which are as follows:

Event	Program Memory Address
Reset	000
External INT0	003
Counter/timer 0	00B
External INT1	013
Timer I	01B
I2C serial	023

Counter/Timer Subsystem

The S87C751 has one counter/timer called timer/counter 0. Its operation is similar to mode 2 operation on the SC80C51, but is extended to 16 bits with 16 bits of autoload. The controls for this counter are centralized in a single register called TCON.

A watchdog timer, called Timer I, is for use with the I2C subsystem. In I2C applications, this timer is dedicated to time-generation and bus monitoring of the I2C. In non-I2C applications, it is available for use as a fixed timebase.

Interrupt Subsystem — Fixed Priority

The IP register and the 2-level interrupt system of the SC80C51 are eliminated. Simultaneous interrupt conditions are resolved by a single-level, fixed priority as follows:

Highest priority:	Pin INT0
	Counter/timer flag 0
	Pin INT1
	Timer I
Lowest priority:	Serial I2C

Serial Communications

The S87C751 contains an I2C serial communications port instead of the SC80C51 UART. The I2C serial port is a single bit hardware interface with all of the hardware necessary to support multimaster and slave operations. Also included are receiver digital filters and timer (timer I) for communication watch-dog purposes. The I2C serial port is controlled through four special function registers; I2C control, I2C data, I2C status, and I2C configuration.

Special Function Register Addresses

Special function register addresses for the S87C751 are identical to those of the SC80C51, except for the changes listed below:

SC80C51 special function registers not present in the S87C751 are TMOD (89), P2 (A0) and IP (B8). The SC80C51 registers TH1, TL1, SCON, and SBUF are replaced with the S87C751 registers RTH, RTL, I2CON, and I2DAT, respectively. Additional special function registers are I2CFG (D8) and I2STA (F8).

Table 2. I2C Special Function Register Addresses

Register Address			Bit Address							
Name	Symbol	Address	MSB							LSB
I2C control	I2CON	98	9F	9E	9D	9C	9B	9A	99	98
I2C data	I2DAT	99	–	–	–	–	–	–	–	–
I2C configuration	I2CFG	D8	DF	DE	DD	DC	DB	DA	D9	D8
I2C status	I2STA	F8	FF	FE	FD	FC	FB	FA	F9	F8

ABSOLUTE MAXIMUM RATINGS

SYMBOL	PARAMETER	RATING	UNIT
T_{STG}	Storage temperature range	−65 to +150	°C
V_{CC}	Voltage from V_{CC} to V_{SS}	−0.5 to +6.5	V
V_S	Voltage from any pin to V_{SS}	−0.5 to V_{CC} + 0.5	V
P_D	Power dissipation	1.0	W
V_{PP}	Voltage on V_{PP} pin to V_{SS}	0 to +13.0	V

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 4.5V to 5.5V, V_{SS} = 0V[3]

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS		UNIT
			Min	Max	
V_{IL} V_{IH} V_{IH1}	Input low voltage, except SDA, SCL Input high voltage, except X1, RST Input high voltage, X1, RST		−0.5 0.2V_{CC}+0.9 0.7V_{CC}	0.2V_{CC}−0.1 V_{CC}+0.5 V_{CC}+0.5	V V V
 V_{IL1} V_{IH2}	SDA, SCL: Input low voltage Input high voltage		 −0.5 0.7V_{CC}	 0.3V_{CC} V_{CC}+0.5	 V V
V_{OL} V_{OL1}	Output low voltage, ports 1 and 3 Output low voltage, port 0.2	I_{OL} = 1.6mA I_{OL} = 3.2mA		0.45 0.45	V V
V_{OH}	Output high voltage, ports 1 and 3	I_{OH} = −60µA I_{OH} = −25µA I_{OH} = −10µA	2.4 0.75V_{CC} 0.9V_{CC}		V V V
 V_{OL2} C	Port 0.0 and 0.1 (I^2C) – Drivers Output low voltage Driver, receiver combined: Capacitance	 I_{OL} = 3mA (over V_{CC} range)		 0.4 10	 V pF
I_{IL} I_{TL} I_{LI}	Logical 0 input current, ports 1 and 3 Logical 1 to 0 transition current, ports 1 and 3[8] Input leakage current, port 0	V_{IN} = 0.45V V_{IN} = 2V 0.45 < V_{IN} < V_{CC}		−50 −650 ±10	µA µA µA
R_{RST}	Reset pull-down resistor		25	175	kohm
C_{IO}	Pin capacitance	Test freq = 1MHz, T_A = 25°C		10	pF
I_{PD}	Power-down current[4]	V_{CC} = 2 to 5.5V		50	µA
V_{PP}	V_{PP} program voltage	V_{SS} = 0V V_{CC} = 5V±10% T_A = 21°C–27°C	12.5	13.0	V
I_{PP}	Program current	V_{PP} = 13.0V		10	mA
I_{CC}	Supply current (see Figure 3)				

AC SYMBOL DESIGNATIONS

Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:

C – Clock
D – Input data
H – Logic level high
L – Logic level low
Q – Output data
T – Time
V – Valid
X – No longer a valid logic level
Z – Float

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = –40°C to +85°C, V_{CC} = 5V±10%, V_{SS} = 0V[3, 7]

SYMBOL	PARAMETER	12MHz		VARIABLE CLOCK		UNIT
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator frequency			3.5 3.5 0.5	12 16 12	MHz
	External clock (Figure 1)					
t_{CHCH}	High time	20				ns
t_{CLCL}	Low time	20				ns
t_{CLCH}	Rise time		20			ns
t_{CHCL}	Fall time		20			ns

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages with respect to V_{SS} unless otherwise noted.
4. Power-down Icc is measured with all output pins disconnected; port 0 = VCC; X2, X1 n.c.; RST = VSS.
5. Active ICC is measured with all output pins disconnected; X1 driven with tCLCH, tCHCL = 5ns, V_{IL} = Vss + 0.5V, V_{IH} = VCC – 0.5V; X2 n.c.; RST = port 0 = VCC. ICC will be slightly higher if a crystal oscillator is used.
6. Idle Icc is measured with all output pins disconnected; X1 driven with tCLCH, tCHCL = 5ns, V_{IL} = Vss + 0.5V, V_{IH} = VCC – 0.5V; X2 n.c.; port 0 = VCC; RST = VSS.
7. Load capacitance for ports = 80pF.
8. Pins of ports 1 and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.

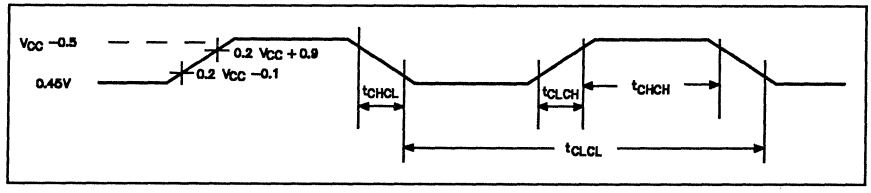


Figure 1. External Clock Drive Waveform

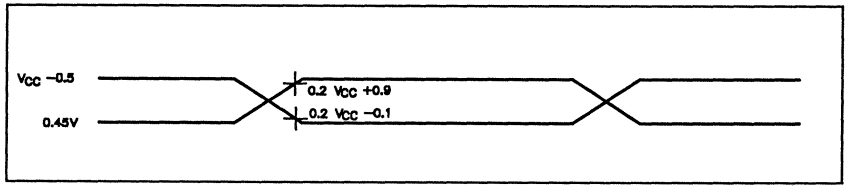


Figure 2. AC Testing Input/Output Waveform

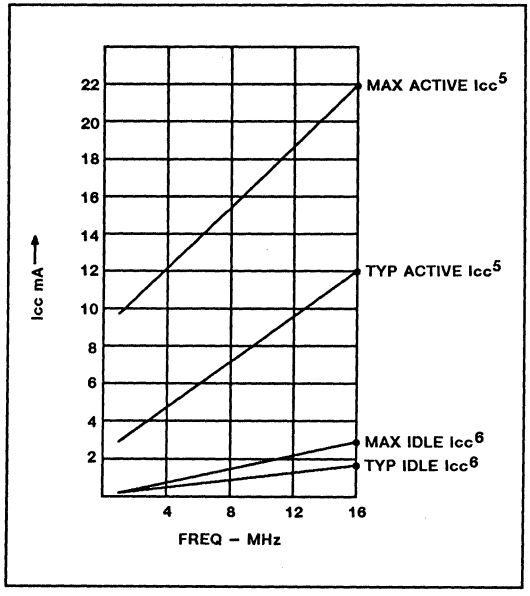


Figure 3. I_{CC} vs. FREQ
Maximum I_{CC} values taken at V_{CC} = 5.5V and worst case temperature.
Typical I_{CC} values taken at V_{CC} = 5.0V and 25°C.
Notes 5 and 6 refer to AC Electrical Characteristics.

PROGRAMMING CONSIDERATIONS

EPROM Characteristics

The 87C751 is programmed by using a modified Quick-Pulse Programming algorithm similar to that used for devices such as the 87C451 and 87C51. It differs from these devices in that a serial data stream is used to place the 87C751 in the programming mode.

Figure 4 shows a block diagram of the programming configuration for the 87C751. Port pin P0.2 is used as the programming voltage supply input (V_{PP} signal). Port pin P0.1 is used as the program (PGM/) signal. This pin is used for the 25 programming pulses.

Port 3 is used as the address input for the byte to be programmed and accepts both the high and low components of the eleven bit address. Multiplexing of these address components is performed using the ASEL input. The user should drive the ASEL input high and then drive port 3 with the high order bits of the address. ASEL should remain high for at least 13 clock cycles. ASEL may then be driven low which latches the high order bits of the address internally. The high address should remain on port 3 for at least two clock cycles after ASEL is driven low. Port 3 may then be driven with the low byte of the address. The low address will be internally stable 13 clock cycles later. The address will remain stable provided that the low byte placed on port 3 is held stable and ASEL is kept low. **Note:** ASEL needs to be pulsed high only to change the high byte of the address.

Port 1 is used as a bidirectional data bus during programming and verify operations. During the programming mode, it accepts the byte to be programmed. During the verify mode, it provides the contents of the EPROM location specified by the address which has been supplied to Port 3.

The XTAL1 pin is the oscillator input and receives the master system clock. This clock should be between 1.2 and 6 MHz.

The RESET pin is used to accept the serial data stream that places the 87C751 into various programming modes. This pattern consists of a 10-bit code with the LSB sent first. Each bit is synchronized to the clock input, X1.

Programming Operation

Figures 5 and 6 show the timing diagrams for the program/verify cycle. RESET should initially be held high for at least two machine cycles. P0.1 (PGM/) and P0.2 (V_{PP}) will be at V_{OH} as a result of the RESET operation. At this point, these pins function as normal quasi-bidirectional I/O ports and the programming equipment may pull these lines low. However, prior to sending the 10 bit code on the reset pin, the programming equipment should drive these pins high (V_{IH}). The RESET pin may now be used as the serial data input for the data stream which places the 87C751 in the programming mode. Data bits are sampled during the clock high time and thus should only change during the time that the clock is low. Following transmission of the last data bit, the RESET pin should be held low.

Next the address information for the location to be programmed is placed on port 3 and ASEL used to perform the address multiplexing, as previously described. At this time, port 1 functions as an output.

A high voltage V_{PP} level is then applied to the V_{PP} input (P0.2). (This sets Port 1 as an input port.). The data to be programmed into the EPROM array is then placed on Port 1. This is followed by a series of programming pulses applied to the PGM/ pin (P0.1). These pulses are created by driving P0.1 low and then high. This pulse is repeated until a total of twenty-five programming pulses have occurred. At the conclusion of the last pulse, the PGM/ signal should remain high.

The V_{PP} signal may now be driven to the V_{OH} level, placing the 87C751 in the verify mode. (Port 1 is now used as an output port.). After 4 machine cycles (48 clock periods) the contents of the addressed location in the EPROM array will appear on Port 1.

The next programming cycle may now be initiated by placing the address information at the inputs of the multiplexed buffers, driving the V_{PP} pin to the V_{PP} voltage level, providing the byte to be programmed to Port 1 and issuing the 25 programming pulses on the PGM/ pin, bringing V_{PP} back down to the V_{CC} level and verifying the byte.

Programming Modes

The 87C751 has four programming features incorporated within its EPROM array. These include the USER EPROM for storage of the application's code, a sixteen byte encryption KEY array and two security bits. Programming and verification of these four elements are selected by a combination of the serial data stream applied to the RESET pin and the voltage levels applied to port pins P0.1 and P0.2. The various combinations are shown in Table 3.

Encryption Key Table

The 87C751 includes a 16 byte EPROM array that is programmable by the end user. The contents of this array can then be used to encrypt the program memory contents during a program memory verify operation. When a program memory verify operation is performed, the contents of the program memory location is XNOR'ed with one of the bytes in the 16 byte encryption table. The resulting data pattern is then provided to port 1 as the verify data. The encryption mechanism can be disabled, in essence, by leaving the bytes in the encryption table in their erased state (FFH) since the XNOR product of a bit with a logical one will result in the original bit. The encryption bytes are mapped with the code memory in 16 byte groups. The first byte in code memory will be encrypted with the first byte in the encryption table; the second byte in code memory will be encrypted with the second byte in the encryption table and so forth up to and including the sixteenth byte. The encryption repeats in 16 byte groups; the seventeenth byte in the code memory will be encrypted with the first byte in the encryption table, and so forth.

Security Bits

Two security bits, security bit 1 and security bit 2, are provided to limit access to the USER EPROM and encryption key arrays. Security bit 1 is the program inhibit bit, and once programmed performs the following functions:

1. Additional programming of the USER EPROM is inhibited.
2. Additional programming of the encryption key is inhibited.
3. Verification of the encryption key is inhibited.
4. Verification of the USER EPROM and the security bit levels may still be performed.

(If the encryption key array is being used, this security bit should be programmed by the user to prevent unauthorized parties from reprogramming the encryption key to all logical zero bits. Such programming would provide data during a verify cycle that is the logical compliment of the USER EPROM contents.)

Security bit 2, the verify inhibit bit, prevents verification of both the USER EPROM array and the encryption key arrays. The security bit levels may still be verified.

Programming and Verifying Security Bits

Security bits are programmed employing the same techniques used to program the USER EPROM and KEY arrays using serial data streams and logic levels on port pins indicated in Table 3. When programming either security bit, it is not necessary to provide address or data information to the 87C751 on ports 1 and 3.

Verification occurs in a similar manner using the RESET serial stream shown in Table 3. Port 3 is not required to be driven and the results of the verify operation will appear on ports 1.6 and 1.7.

Port 1.7 contains the security bit 1 data and is a logical one if programmed and a logical zero if erased. Likewise, P1.6 contains the security bit 2 data and is a logical one if programmed and a logical zero if erased.

Erasure Characteristics

Erasure of the EPROM begins to occur when the chip is exposed to light with wavelengths shorter than approximately 4,000 angstroms. Since sunlight and fluorescent lighting have wavelengths in this range, exposure to these light sources over an extended time (about 1 week in sunlight, or 3 years in room level fluorescent lighting) could cause inadvertent erasure. **For this and secondary effects, it is recommended that an opaque label be placed over the window.** For elevated temperature or solvent environments, use Kapton tape Fluorglas part number 2345-5 or equivalent.

The recommended erasure procedure is exposure to ultraviolet light (at 2537 angstroms) to an integrated dose of at least 15W-sec/cm^2. Exposing the EPROM to an ultraviolet lamp of 12,000µW/cm^2 rating for 20 to 39 minutes, at a distance of about 1 inch, should be sufficient.

Erasure leaves the array in an all 1s state.

Table 3. Implementing Program/Verify Modes

Operation	Serial Code	P0.1 (PGM/)	P0.2 (V_{PP})
Program user EPROM	296H	-*	V_{PP}
Verify user EPROM	296H	V_{IH}	V_{IH}
Program key EPROM	292H	-*	V_{PP}
Verify key EPROM	292H	V_{IH}	V_{IH}
Program security bit 1	29AH	-*	V_{PP}
Program security bit 2	298H	-*	V_{PP}
Verify security bits	29AH	V_{IH}	V_{IH}

NOTE:
*Pulsed from V_{IH} to V_{IL} and returned to V_{IH}.

EPROM PROGRAMMING AND VERIFICATION CHARACTERISTICS T_A = 21°C to +27°C, V_{CC} = 5V±10%, V_{SS} = 0V

SYMBOL	PARAMETER	MIN	MAX	UNIT
$1/t_{CLCL}$	Oscillator/clock frequency	1.2	6	MHz
t_{AVGL}*	Address setup to P0.1 (PROG-) low	10µs + 24t_{CLCL}		
t_{GHAX}	Address hold after P0.1 (PROG-) high	48t_{CLCL}		
t_{DVGL}	Data setup to P0.1 (PROG-) low	38t_{CLCL}		
t_{GHDX}	Data hold after P0.1 (PROG-) high	36t_{CLCL}		
t_{SHGL}	V_{PP} setup to P0.1 (PROG-) low	10		µs
t_{GHSL}	V_{PP} hold after P0.1 (PROG-)	10		µs
t_{GLGH}	P0.1 (PROG-) width	90	110	µs
t_{AVQV}**	V_{PP} low (V_{CC}) to data valid		48t_{CLCL}	
t_{GHGL}	P0.1 (PROG-) high to P0.1 (PROG-) low	10		µs
t_{SYNL}	P0.0 (sync pulse) low	4t_{CLCL}		
t_{SYNH}	P0.0 (sync pulse) high	8t_{CLCL}		
t_{MASEL}	ASEL high time	13t_{CLCL}		
t_{MAHLD}	Address hold time	2t_{CLCL}		
t_{HASET}	Address setup to ASEL	13t_{CLCL}		
t_{ADSTA}	Low adress to address stable	13t_{CLCL}		

NOTES:
*Address should be valid at least 24t_{CLCL} before the rising edge of P0.2 (V_{PP}).
**For a pure verify mode, i.e., no program mode in between, t_{AVQV} is 14t_{CLCL} maximum.

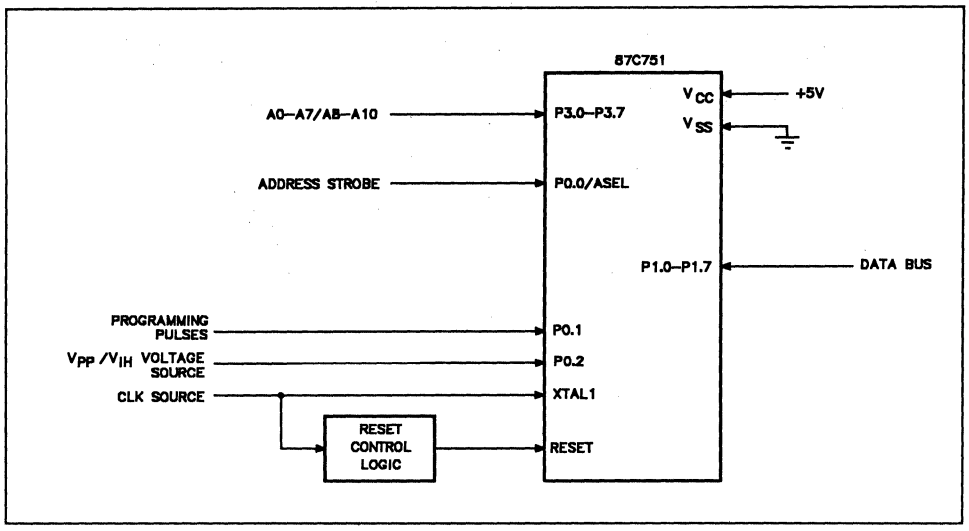


Figure 4. Programming Configuration

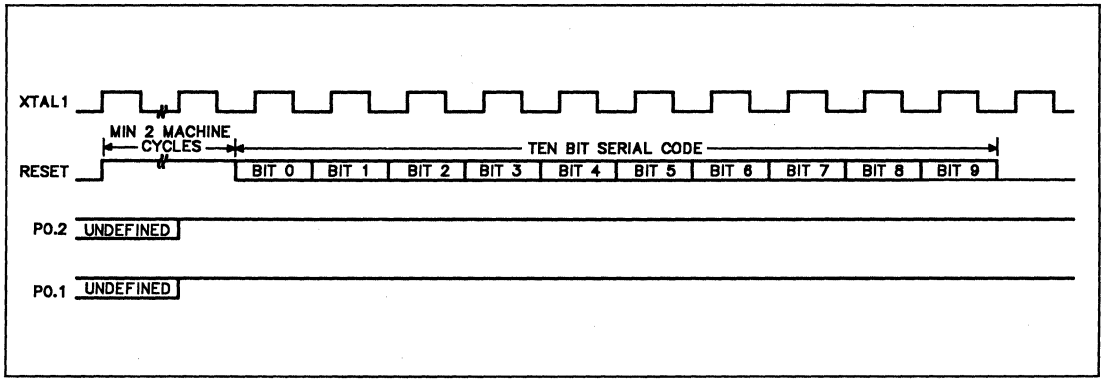


Figure 5. Entry Into Program/Verify Modes

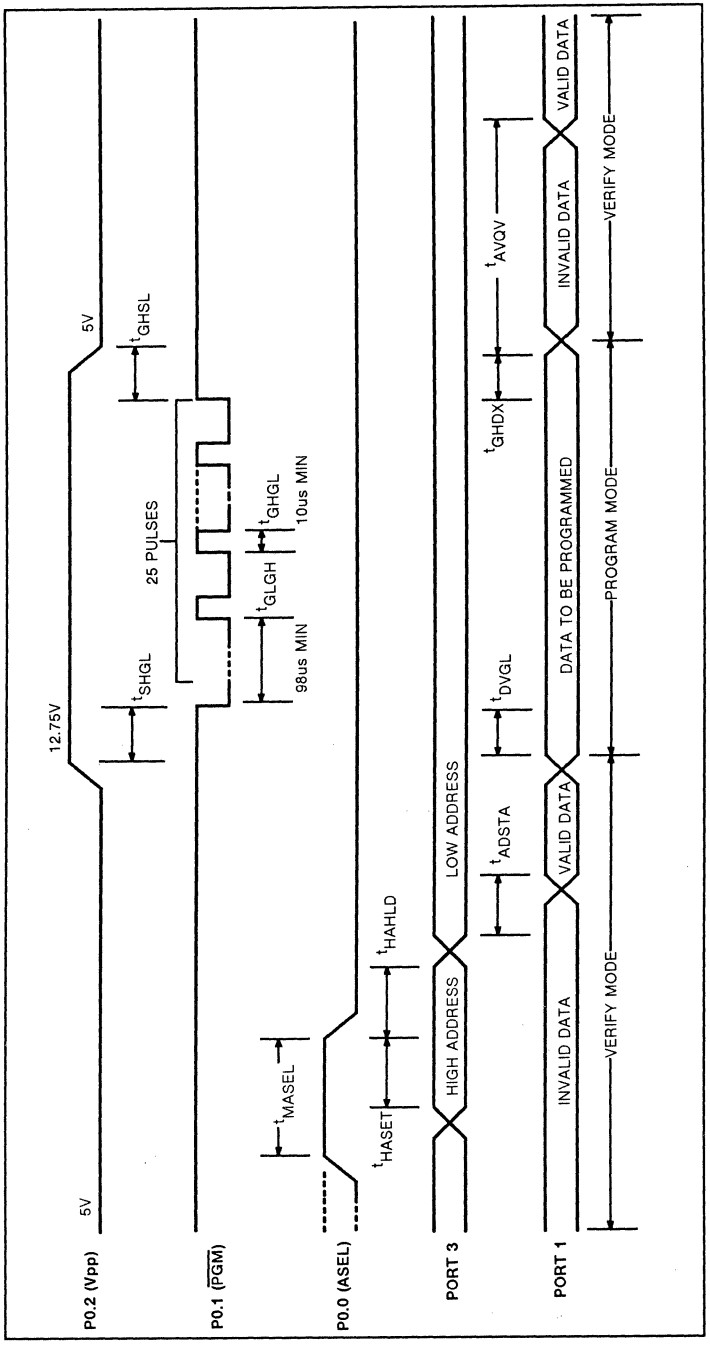


Figure 6. Program/Verify Cycle

8XC752 OVERVIEW

The Signetics 83C752/87C752 is a single-chip control oriented microcontroller fabricated with Signetics high-density CMOS technology minimizing CMOS latch-up sensitivity. Being a member of the 80C51 family, the 83C752 has a powerful instruction set, and has the same basic architecture as the 80C51. The 83C752 is essentially the popular industry-standard 83C751 with the inclusion of a 5-channel multiplexed 8-bit ADC and a PWM output.

The 83C752 contains a 2K X 8 masked ROM, 64 bytes of RAM, 21 I/O lines, a 16-bit auto-reload timer/counter, a fixed rate timer, a seven source fixed priority interrupt structure, a bi-directional Inter-Integrated Circuit (I2C) serial bus interface, and an on-chip oscillator. This device also includes a 5-channel multiplexed 8-bit A/D converter and an 8-bit PWM output.

The on-board I2C bus interface allows the 83C752 to operate as a Master or Slave device on the I2C small area network. This capability facilitates I/O and RAM expansion, access to EEPROM, processor-to-processor communications, and efficient interface to a wide variety of dedicated I2C peripherals.

The EPROM version of this device, the 87C752, is also available in both quartz-lid erasable and plastic one-time programmable (OTP) packages. Once the array has been programmed, it is functionally equivalent to the masked ROM 83C752. Thus, unless explicitly stated otherwise, all references made to the 83C752 apply equally to the 87C752.

The 83C752 supports two power reduction modes of operation referred to as the idle mode and the power-down mode.

Idle Mode

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active, except for the A/D converter and the PWM output. The instruction which places the 83C752 into the idle mode is the last instruction executed in normal operation before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the SFRs remain intact during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset. An interrupt will cause processing to begin with the interrupt service routine.

Power-Down Mode

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM is preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the PCON register.

Memory Organization

The 83C752 manipulates operands in three memory address spaces. The first is the program memory space which contains program instructions as well as constants such as look-up tables. The program memory space contains 2Kbytes in the 83C752.

The second memory space is the data memory array which has a logical address space of 128 bytes. However, only the first 64 bytes (0 to 3FH) are implemented in the 83C752.

The third memory space is the special function register array having a 128 byte address space (80H to FFH). Only selected locations in this memory space are used (see Table 23). Note that the architecture of these memory spaces (internal program memory, internal data memory, and special function registers) is identical to the 80C51 and the 83C752 varies only in the amount of memory physically implemented.

The 83C752 does not directly address any external data or program memory spaces. For this reason, the MOVX instructions in the 80C51 instruction set are not implemented in the 83C752, nor are the alternate I/O pin functions $\overline{RD}$ and $\overline{WR}$.

I/O Ports

The I/O pins provided by the 83752 consists of port 0, port 1, and port 3.

Port 0

Port 0 is a 5-bit bi-directional I/O port and includes alternate functions on some pins of this port. Pins P0.3 and P0.4 are provided with internal pullups while the remaining pins (P0.0, P0.1 and P0.2) have open drain output structures. The alternate functions for port 0 are:

P0.0 SCL – the I2C bus clock
P0.1 SDA – the I2C bus data
P0.4 PWM – the PWM output

If the alternate functions, I2C and PWM, are not being used then these pins may be used as I/O ports.

Port 1

Port 1 is an 8-bit bi-directional I/O port whose structure is identical to the 80C51, but also includes alternate input functions on all pins. The alternate pin functions for port 1 are:

P1.0–P1.4 – ADC0–ADC4 – A/D converter analog inputs
P1.5 $\overline{INT0}$ – external interrupt 0 input
P1.6 $\overline{INT1}$ – external interrupt 1 input
P1.7 – T0 – timer 0 external input

Table 23. 8XC752 Special Function Registers

Symbol	Description	Direct Address	Bit Address, Symbol or Alternative Port Function MSB							LSB	Reset Value
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
ADAT#	A/D result	84H									00H
ADCON#	A/D control	A0H	–	–	ENADC	ADCI	ADCS	AADR2	AADR1	AADR0	C0H
B*	B Register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
DPTR:	Data pointer (2 bytes):										
DPL	Data pointer low	82H									00H
DPH	Data pointer high	83H									00H
			DF	DE	DD	DC	DB	DA	D9	D8	
I2CFG*#	I²C configuration	D8H/RD	SLAVEN	MASTRQ	0	TIRUN	–	–	CT1	CT0	0000xx00B
		WR	SLAVEN	MASTRQ	CLRTI	TIRUN	–	–	CT1	CT0	
			9F	9E	9D	9C	9B	9A	99	98	
I2CON*#	I²C control	98H/RD	RDAT	ATN	DRDY	ARL	STR	STP	MASTER	–	81H
		WR	CXA	IDLE	CDR	CARL	CSTR	CSTP	XSTR	XSTP	
I2DAT*#	I²C data	99H/RD	RDAT	0	0	0	0	0	0	0	80H
		WR	XDAT	X	X	X	X	X	X	X	
			FF	FE	FD	FC	FB	FA	F9	F8	
I2STA*#	I²C status	F8H	–	IDLE	XDATA	XACTV	MAKSTR	MAKSTP	XSTR	XSTP	x0100000B
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*#	Interrupt enable	A8H	EA	EAD	ETI	ES	EPWM	EX1	ET0	EX0	00H
P0*#	Port 0	80H	–	–	–	84	83	82	81	80	xxx11111B
P1*#	Port 1	90H	97	96	95	94	93	92	91	90	FFH
P3*#	Port 3	B0H	B7	B6	B5	B4	B3	B2	B1	B0	FFH
PCON#	Power control	87H	–	–	–	–	–	–	PD	IDL	xxxx0000B
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	–	P	00H
PWCM#	PWM compare	8EH									xxxxxxxxB
PWENA#	PWM enable	FEH	–	–	–	–	–	–	–	PWE	FEH
PWMP#	PWM prescaler	8FH									00H
RTL#	Timer low reload	8BH									00H
RTH#	Timer high reload	8DH									00H
SP	Stack pointer	81H									07H
TL#	Timer low	8AH									00H
TH#	Timer high	8CH									00H
TCON*#	Timer control	88H	8F	8E	8D	8C	8B	8A	89	88	00H
			GATE	C/T	TF	TR	IE0	IT0	IE1	IT1	00H

* = SFRs are bit addressable.
\# = SFRs are modified from or added to the 80C51 SFRs.

If the alternate functions $\overline{INT0}$, $\overline{INT1}$, or T0 are not being used, these pins may be used as standard I/O ports. If the A/D converter is not enabled, pins P1.0–P1.4 can be used as standard I/O pins.

Port 3

Port 3 is an 8-bit bi-directional I/O port whose structure is identical to the 80C51. Note that the alternate functions associated with port 3 of the 80C51 have been moved to port 1 of the 83C752 (as applicable). See Figure 63 for port bit configurations.

PWM Outputs

The single PWM output is an alternate function assigned to P0.4, and can be used to output pulses of programmable length and interval. The repetition frequency is defined by an 8-bit prescaler which generates the clock for the counter. This prescaler is contained in the PWMP register.

The 8-bit counter counts from 0 to 254 inclusive. The value of the 8-bit counter is compared to the contents of the compare register, PWM. When the content's value matches that of the PWM register, the PWM output is set high. When the counter reaches zero, the PWM output is set low. The pulse width ratio (duty cycle) is defined by the contents of the compare register and is in the range of 0 to 1 programmed in increments of 1/255.

The PWM output can be set continuously high by loading the compare register with 00H and continuously low by loading the compare register with FFH. The PWM output is enabled by setting the PWE bit in the PWM enable register, PWENA. When enabled, the output is driven with a fully active strong pullup. When disabled, the pin behaves as a normal bi-directional I/O pin. When disabled, the counter remains active. The PWM function is disabled by a reset condition. The PWM output is high during power-down and idle modes and the counter is disabled.

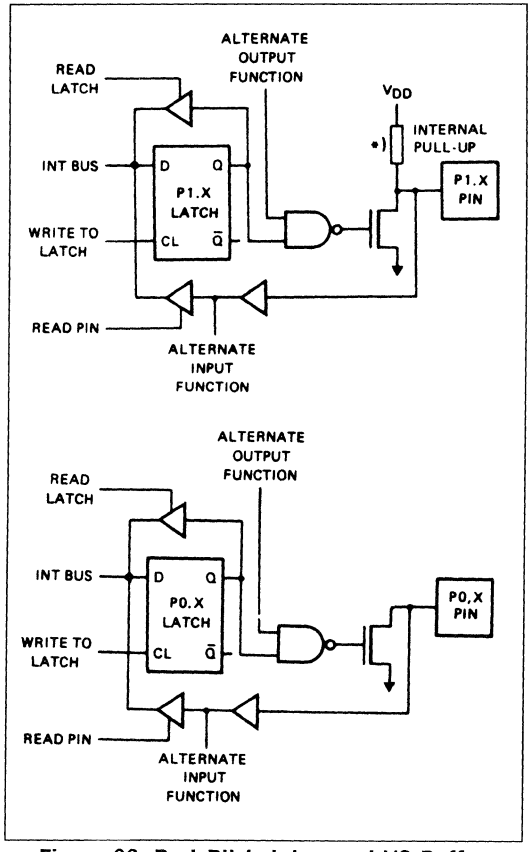


Figure 63. Port Bit Latches and I/O Buffers

The repetition frequency is given by:

$$f_{PWM} = \frac{f_{osc}}{2 \times (1 + PWMP)\ 255}$$

An oscillator frequency of 12MHz results in a repetition range of 92Hz to 23.5KHz.

The low/high ratio of the PWM output is PWM / (255 – PWM) for PWM values except 255. A PWM value of 255 results in the low PWM output.

If enabled, a PWM interrupt will occur when the PWM counter overflows.

In order for the PWM output to be used as a standard I/O pin, the PWM function needs to be disabled. The PWM counter can still be used as an internal timer by enabling the PWM interrupt.

A/D Converter

The 83C752 contains a 5-channel multiplexed 8-bit A/D converter. The conversion requires 40 machine cycles (40μs at 12MHz oscillator frequency).

The A/D converter is controlled by the A/D control register, ADCON. Input channels are selected by the analog multiplexer by bits ADCON.0 through ADCON.2. The ADCON register is not bit addressable.

ADCON Register

MSB LSB

X	X	ENADC	ADCI	ADCS	AADR2	AADR1	AADR0

ADCI	ADCS	Operation
0	0	ADC not busy, a conversion can be started.
0	1	ADC busy, start of a new conversion is blocked.
1	0	Conversion completed, start of a new conversion is blocked.
1	1	Not possible.

Input Channel Selection			
ADDR2	ADDR1	ADDR0	Input Pin
0	0	0	P1.0
0	0	1	P1.1
0	1	0	P1.2
0	1	1	P1.3
1	0	0	P1.4

Position	Symbol	Function
ADCON.5	ENADC	Enable A/D function when ENADC = 1. Disable A/D function when ENADC = 0. Reset forces ENADC = 0.
ADCON.4	ADCI	ADC interrupt flag. This flag is set when an ADC conversion is complete. If IE.6 = 1, an interrupt is requested when ADCI = 1. The ADCI flag is cleared when conversion data is read. This flag is read only.
ADCON.3	ADCS	ADC start and status. Setting this bit starts an A/D conversion. Once set, ADCS remains high throughout the conversion cycle. On completion of the conversion, it is reset at the same time the ADCI interrupt flag is set. ADCS cannot be reset by software.
ADCON.2	AADR2	Analog input select.
ADCON.1	AADR1	Analog input select.
ADCON.0	AADR0	Analog input select. This binary coded address selects one of the five analog input port pins of P1 to be input to the converter. It can only be changed when ADCI and ADCS are both low. AADR2 is the most significant bit.

The completion of the 8-bit ADC conversion is flagged by ADCI in the ADCON register and the result is stored in the special function register ADAT.

An ADC conversion in progress is unaffected by an ADC start. The result of a completed conversion remains unaffected provided ADCI remains at a logic 1. While ADCS is a logic 1 or ADCI is a logic 1, a new ADC START will be blocked and consequently lost. An ADC

conversion in progress is aborted when the Idle or Power-down mode is entered. The result of a completed conversion (ADCI = logic 1) remains unaffected when entering the Idle mode. See Figure 64 for an A/D input equivalent circuit.

The analog input pins ADC0–ADC4 may be used as digital inputs and outputs when the A/D converter is disabled by a 0 in the ENADC bit in ADCON. When the A/D is enabled, the analog input channel that is selected by the ADDR2–ADDR0 bits in ADCON cannot be used as a digital input. Reading the selected A/D channel as a digital input will always return a 1. The unselected A/D inputs may always be used as digital inputs.

Counter/Timer

The 8XC752 counter/timer is designated Timer 0 and is separate from Timer I of the I2C serial port and from the PWM. Its operation is similar to mode 2 of the 80C51 counter/timer, extended to 16 bits. When Timer 0 is used in the external counter mode, the T0 input (P1.7) is sampled every S4P1. The counter/timer function is controlled using the timer control register (TCON)

TCON Register

MSB							LSB
GATE	C/T	TF	TR	IE0	IT0	IE1	IT1

Position	Symbol	Function
TCON.7	GATE	1 – Timer 0 is enabled only when INT0 pin is high and TR is 1 0 – Timer 0 is enabled only when TR is 1
TCON.6	C/T	1 – Counter operation from T0 pin 0 – Timer operation from internal clock
TCON.5	TF	1 – Set on overflow of T0 0 – Cleared when processor vectors to interrupt routine and by reset
TCON.4	TR	1 – Enable timer 0 0 – Disable timer 0
TCON.3	IE0	1 – Edge detected on INT0
TCON.2	IT0	1 – INT0 is edge triggered 0 – INT0 is level sensitive
TCON.1	IE1	1 – Edge detected on INT1
TCON.0	IT1	1 – INT1 is edge triggered 0 – INT1 is level sensitive

These flags are functionally identical to the corresponding 80C51 flags except that there is only one of the 80C51 style timers and the flags are combined into one register.

A communications watchdog timer Timer I, is described in the I2C section. In I2C applications, this timer is dedicated to time generation and bus monitoring for the I2C. In non-I2C applications, it is available for use as a fixed time base.

The 16-bit timer/counter's operation is similar to mode 2 operation on the 80C51, but is extended to 16 bits. The timer/counter is clocked by either 1/12 the oscillator frequency or by transitions on the T0 pin. The C/T pin in special function register TCON selects between these two modes. When the TCON TR bit is set, the timer/counter is enabled. Register pair TH and TL are incremented by the clock source. When the register pair overflows, the register pair is reloaded with the values in registers RTH and RTL. The value in the reload registers is left unchanged. The TF bit in special function register TCON is set on counter overflow and, if the interrupt is enabled, will generate an interrupt (see Figure 65).

I2C Serial I/O

The I2C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main technical features of the bus are:

- Bidirectional data transfer between masters and slaves
- Serial addressing of slaves
- Acknowledgment after each transferred byte
- Multimaster bus
- Arbitration between simultaneously transmitting master without corruption of serial data on bus

A large family of I2C compatible ICs is available. See the I2C section for more details on the bus and available ICs.

The 83C752 I2C subsystem includes hardware to simplify the software required to drive the I2C bus. This circuitry is the same as that on the 83C751. (See the 83C751 section for a detailed discussion of this subsystem).

Interrupts

The interrupt structure is a seven source, one level interrupt system similar to the 8XC751. The interrupt sources are listed below in their order of polling sequence priority (highest to lowest):

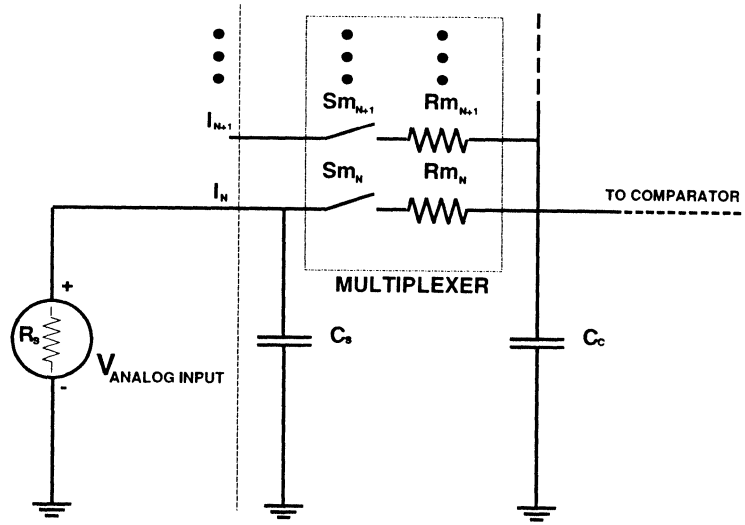
Priority	Source	Function
Highest	INT0	External interrupt 0
	TF0	Timer flag 0
	INT1	External interrupt 1
	PWM	PWM counter overflow
	TI	I2C timer overflow
	SIO	Serial port interrupt
Lowest	ADC	A/D conversion complete

The vector addresses are as follows:

Source	Vector Address
INT0	0003H
TF0	000BH
INT1	0013H
TIMER I	001BH
SIO	0023H
ADC	002BH
PWM	0033H

Interrupt Control Registers

The 80C51 interrupt enable register is modified to take into account the different interrupt sources of the 8XC752.



Rm = 0.5–3Kohms
Cs + Cc = 15pF maximum
Rs = Recommended < 9.6Kohms for 1LSB @ 12MHz

NOTE:
Because the analog to digital converter has a sampled–data comparator, the input looks capacitive to a source. When a conversion is initiated, switch Sm closes for 8tcy (8μs @ 12MHz crystal frequency) during which time capacitance Cs + Cc is charged. It should be noted that the sampling causes the analog input to present a varying load to an analog source.

Figure 64. A/D Input: Equivalent Circuit

Interrupt Enable Register

MSB							LSB
EA	EAD	ETI	ES	EPWM	EX1	ET0	EX0

Position	Symbol	Function
IE.7	EA	Global interrupt disable when EA = 0
IE.6	EAD	A/D conversion complete
IE.5	ETI	Timer I
IE.4	ES	I2C serial port
IE.3	EPWM	PWM counter overflow
IE2	EX1	External interrupt 1
IE.1	ET0	Timer 0 overflow
IE.0	EX0	External interrupt 0

Power-Down and Idle Modes

The 8XC752 includes the 80C51 power-down and idle mode features. The functions that continue to run while in the idle mode are Timer 0, the I2C interface including Timer I, the counter portion of the PWM, and the interrupts. Upon powering-up the circuit, or exiting from idle mode, sufficient time must be allowed for stabilization of the internal analog reference voltages before an A/D conversion is started.

Instruction Set

The instruction set of the 83C752 is identical to the 80C51 except that:

MOVX, LCALL and LJUMP are not implemented.

If these instructions are executed, the appropriate number of instruction cycles will take place along with external fetches, however, no operation will take place. The LJMP may not respond to all program address bits.

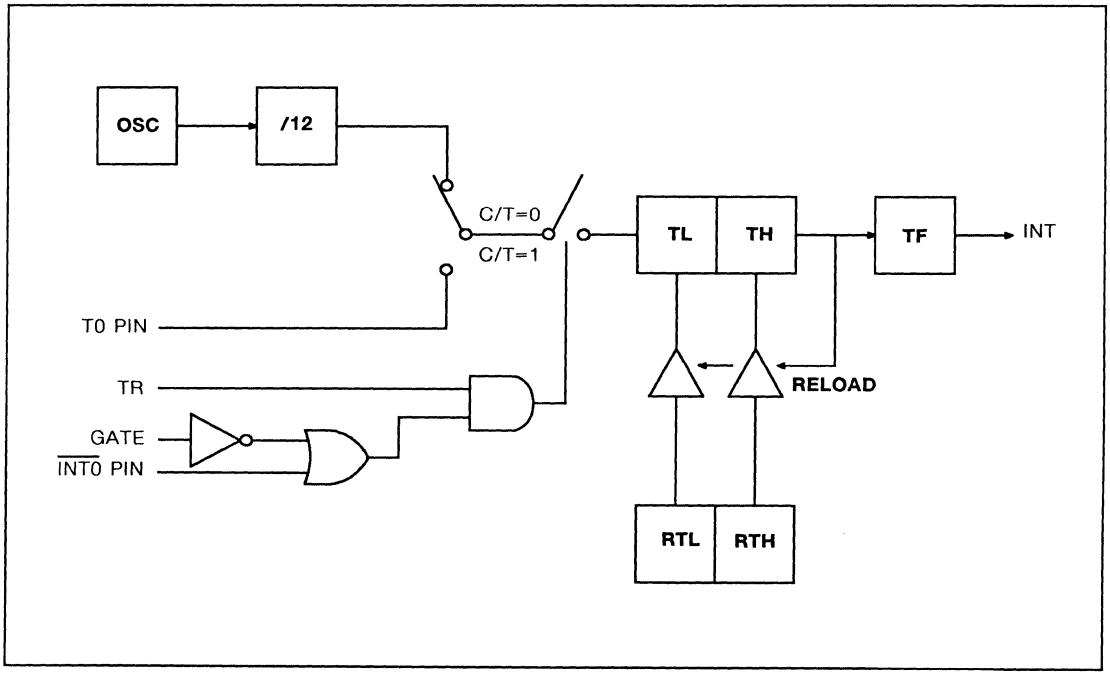


Figure 65. 83C752 Counter/Timer Block Diagram

Special Function Registers

The special function registers (directly addressable only) contain all of the 8XC751 registers except the program counter and the four register banks. Most of the 21 special function registers are used to control the on-chip peripheral hardware. Other registers include arithmetic registers (ACC, B, PSW), stack pointer (SP) and data pointer registers (DPH, DPL). Nine of the SFRs are bit addressable.

Data Pointer
The Data Pointer DPTR) consists of a high byte (DPH) and a low byte (DPL). In the 80C51, this register allows the access of external data memory using the MOVX instruction.

Ports 0, 1, 2

P0, P1, P2 are the latches of Ports 0, 1, and 2, respectively. P1 and P2 are each 8 bits wide while P0 has only five valid bits since it represents a 5-bit port.

Signetics

Microprocessor Division

S87C752
CMOS Single-Chip 8-Bit Microcontroller

Preliminary Specification

DESCRIPTION

The Signetics S87C752 offers many of the advantages of the SC80C51 architecture in a small package and at low cost.

The S87C752 Microcontroller is fabricated with Signetics high-density CMOS technology. Signetics epitaxial substrate minimizes CMOS latch-up sensitivity.

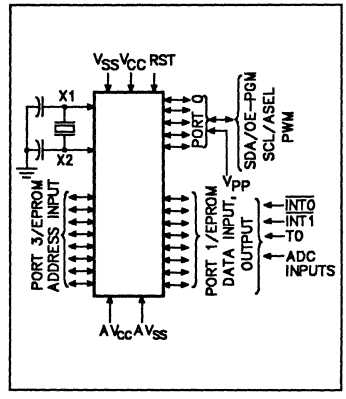
The S87C752 contains a 2K x 8 E-PROM, a 64 x 8 RAM, 21 I/O lines, a 16-bit auto-reload counter/timer, a fixed-rate timer, a seven-source fixed-priority interrupt structure, a bidirectional Inter-Integrated Circuit (I2C) serial bus interface, an on-chip oscillator, a five channel multiplexed 8-bit A/D converter, and an 8-bit PWM output.

The onboard inter-integrated circuit (I2C) bus interface allows the S87C752 to operate as a master or slave device on the I2C small area network. This capability facilitates I/O and RAM expansion, access to EEPROM, processor-to-processor communication, and efficient interface to a wide variety of dedicated I2C peripherals.

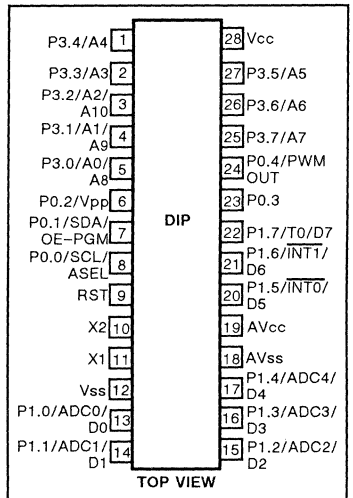
FEATURES

- **EPROM version of S83C752**
- **Available in erasable quartz lid or One-Time Programmable plastic packages**
- **SC80C51 based architecture**
- **Inter-Integrated Circuit (I2C) serial bus interface**
- **Small package sizes**
  - **28-pin DIP**
  - **28-pin PLCC**
- **Wide oscillator frequency range**
- **Low power consumption:**
  - **Normal operation: less than 11mA @ 5V, 12MHz**
  - **Idle mode**
  - **Power-down mode**
- **2K x 8 EPROM, 64 x 8 RAM**
- **16-bit auto reloadable counter/timer**
- **5 channel 8-bit A/D converter**
- **8-bit PWM output/timer**
- **Fixed-rate timer**
- **Boolean processor**
- **CMOS and TTL compatible**
- **Well suited for logic replacement, consumer and industrial applications**

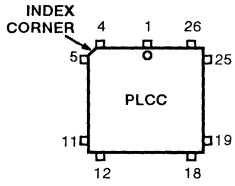
LOGIC SYMBOL



PIN CONFIGURATION



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	P3.4/A4	15	P1.2/ADC2/D2
2	P3.3/A3	16	P1.3/ADC3/D3
3	P3.2/A2/A10	17	P1.4/ADC4/D4
4	P3.1/A1/A9	18	AVss
5	P3.0/A0/A8	19	AVcc
6	P0.2/Vpp	20	P1.5/INT0/D5
7	P0.1/SDA/OE–PGM	21	P1.6/INT1/D6
8	P0.0/SCL/ASEL	22	P1.7/T0/D7
9	RST	23	P0.3
10	X2	24	P0.4/PWM OUT
11	X1	25	P3.7/A7
12	Vss	26	P3.6/A6
13	P1.0/ADC0/D0	27	P3.5/A5
14	P1.1/ADC1/D1	28	Vcc

NOTE:
A0–A10 and D0–D7 available for EPROM verify only.

ORDERING INFORMATION

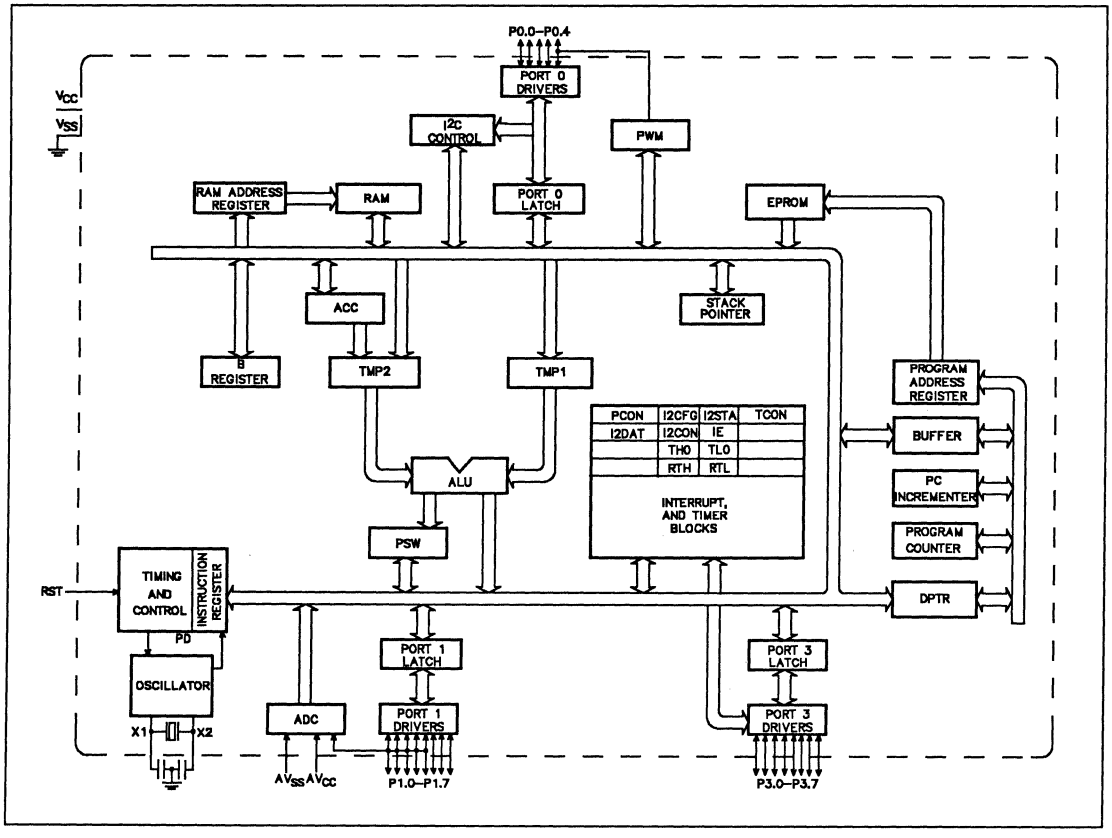
S87C752– □□

Package Codes:
A28 = Plastic PLCC (OTP)
F28 = Ceramic DIP
N28 = Plastic DIP (OTP)

Speed and Temperature Range:
1 = 3.5 to 12MHz, 0°C to +70°C
2 = 3.5 to 12MHz, −40°C to +85°C
3 = 0.5 to 12MHz, 0°C to +70°C
4 = 3.5 to 16MHz, 0°C to +70°C
5 = 3.5 to 16MHz, −40°C to +85°C

PART NUMBER SELECTION		
Part Number	**Speed**	**Temperature and Package**
S87C752-1F28	3.5 to 12MHz	0 to +70°C, Ceramic DIP
S87C752-3F28	0.5 to 12MHz	0 to +70°C, Ceramic DIP
S87C752-4F28	3.5 to 16MHz	0 to +70°C, Ceramic DIP
S87C752-1N28	3.5 to 12MHz	0 to +70°C, Plastic DIP
S87C752-2N28	3.5 to 12MHz	−40 to +85°C, Plastic DIP
S87C752-3N28	0.5 to 12MHz	0 to +70°C, Plastic DIP
S87C752-4N28	3.5 to 16MHz	0 to +70°C, Plastic DIP
S87C752-5N28	3.5 to 16MHz	−40 to +85°C, Plastic DIP
S87C752-1A28	3.5 to 12MHz	0 to +70°C, Plastic LCC
S87C752-2A28	3.5 to 12MHz	−40 to +85°C, Plastic LCC
S87C752-3A28	0.5 to 12MHz	0 to +70°C, Plastic LCC
S87C752-4A28	3.5 to 16MHz	0 to +70°C, Plastic LCC
S87C752-5A28	3.5 to 16MHz	−40 to +85°C, Plastic LCC

BLOCK DIAGRAM



PIN CONFIGURATION (DIP and PLCC)

MNEMONIC	PIN NO.	TYPE	NAME AND FUNCTION
V_{SS}	12	I	**Circuit ground potential.**
V_{CC}	28	I	**Supply voltage during normal, idle, and power-down operation.**
P0.0–P0.4	8–6, 23, 24	I/O	**Port 0:** Port 0 is a 5-bit bidirectional port. Port 0.0 – P0.2 are open drain. Port 0.0 – P0.2 pins that have ones written to them float, and in that state can be used as high-impedance inputs. P0.3–P0.4 are bi-directional I/O port pins with internal pullups. Port 0 also serves as the serial I2C interface as shown in the pinout diagram. When this feature is activated by software, SCL and SDA are driven low in accordance with the I2C protocol. These pins are driven low if the port register bit is written with a 0 or if the I2C subsystem presents a 0. The state of the pin can always be read from the port register by the program. Port 0.3 and 0.4 have internal pull-ups that function identically to port 3. Pins that have ones written to them are pulled high by the internal pull-ups and can be used as inputs. To comply with the I2C specification, P0.0 and P0.1 are open drain bidirectional I/O pins with the electrical characteristics listed in the tables that follow. While these differ from 'standard TTL' characteristics, they are close enough for the pins to still be used as general-purpose I/O in non-I2C applications.
	7	I/O	**SDA (P0.1)** I2C data.
	8	I/O	**SCL (P0.0)** I2C clock.
	24	O	**PWM OUT (P0.4) –** This pin also functions as the pulse width modulated output. When the PWM is enabled, the output (P0.4) has a strong active pull-up and cannot be used as an input. (see DC Electrical Characteristics).
			Port 0 also provides alternate functions for programming the EPROM memory as follows:
	6	I	**V_{PP} (P0.2)** – Programming voltage input.
	7	I	**OE/PGM (P0.1)** – Input, OE/PGM, which specifies verify mode (output enable) or the prrogram mode.
			OE/PGM = 1 output enabled (verify mode)
			OE/PGM = 0 program mode.
	8	I	**ASEL (P0.0)** – input which indicates which bits of the EPROM address are applied to port 3.
			ASEL = 0 low address byte available on port 3.
			ASEL = 1 high address byte is available on port 3 (only the three least significant bits are used).
P1.0–P1.7	13–17, 20–22	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pullups. Port 1 pins that have ones written to them are pulled high by the internal pullups and can be used as inputs. P0.3–P0.4 pins are bi-directional I/O port pins with internal pullups. As inputs, port 1 pins that are externally pulled low will source current because of the internal pullups. (See DC electrical characteristics: I_{IL}). Port 1 also serves the special function features of the SC80C51 family as listed below:
	20		**$\overline{INT0}$ (P1.5):** External interrupt
	21		**$\overline{INT1}$ (P1.6):** External interrupt
	22		**T0 (P1.7):** Timer 0 external input
	13–17	I	**ADC0 (P1.0) – ADC4 (P1.4) –** Port 1 also functions as the inputs to the five channel multiplexed A/D converter. These pins can be used as outputs only if the A/D function has been disabled. These pins may be used as inputs while the A/D converter is enabled. Port 1 serves to output the addressed EPROM contents in the verify mode and accepts as inputs the value to program into the selected address during the program mode.
P3.0–P3.7	5–1 27–25	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pullups. Port 3 pins that have ones written to them are pulled high by the internal pullups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pullups (See DC electrical characteristics: I_{IL}). Port 3 also functions as the address input for the EPROM memory location to be programmed (or verified). The 11-bit address is multiplexed into this port as specified by P0.0/ASEL.
RST	9	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running resets the device. An internal diffused resistor to V_{SS} permits a power-on RESET using only an external capacitor to V_{CC}. After the device is reset, a 10-bit serial sequence, sent LSB first, applied to RESET, places the device in the programming state allowing programming address, data and V_{PP} to be applied for programming or verification purposes. The RESET serial sequence must be synchronized with the X1 input.
X1	11	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits. X1 also serves as the clock to strobe in a serial bit stream into RESET to place the device in the programming state.
X2	10	O	**Crystal 2:** Output from the inverting oscillator amplifier.
AVcc	19	I	**Analog supply voltage and reference input.**
AVss	18	I	**Analog supply and reference ground.**

OSCILLATOR CHARACTERISTICS

X1 and X2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator.

To drive the device from an external clock source, X1 should be driven while X2 is left unconnected. There are no requirements on the duty cycle of the external clock signal, because the input to the internal clock circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the data sheet must be observed.

IDLE MODE

In the idle mode, the CPU puts itself to sleep while all of the on-chip peripherals stay active, except for the A/D converter and the PWM output. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

POWER-DOWN MODE

In the power-down mode, the oscillator is stopped and the instruction to invoke power-down is the last instruction executed. Only the contents of the on-chip RAM are preserved. A hardware reset is the only way to terminate the power-down mode. The control bits for the reduced power modes are in the special function register PCON.

Table 1. External Pin Status During Idle and Power-Down Modes

Mode	Port 0*	Port1	Port 2
Idle	Data	Data	Data
Power-down	Data	Data	Data

*Except for PWM output (P0.4)

DIFFERENCES BETWEEN THE S87C752 AND THE SC80C51

Program Memory

On the S87C752, program memory is 2048 bytes long and is not externally expandable. Program memory can contain S87C752 instructions and constant data. The only fixed locations in program memory are the addresses at which execution is taken up in response to reset and to interrupts, which are as follows:

Event	Program Memory Address
Reset	000
External INT0	003
Counter/timer 0	00B
External INT1	013
Timer I	01B
I2C serial	023
ADC	02B
PWM	033

Counter/Timer Subsystem

The S87C752 has one counter/timer called timer/counter 0. Its operation is similar to mode 2 operation on the SC80C51, but is extended to 16 bits with 16 bits of autoload. The controls for this counter are centralized in a single register called TCON.

A watchdog timer, called Timer I, is for use with the I2C subsystem. In I2C applications, this timer is dedicated to time-generation and bus monitoring of the I2C. In non-I2C applications, it is available for use as a fixed timebase.

Interrupt Subsystem – Fixed Priority

The IP register and the 2-level interrupt system of the SC80C51 are eliminated. Simultaneous interrupt conditions are resolved by a single-level, fixed priority as follows:

Highest priority:	Pin INT0
	Counter/timer flag 0
	Pin INT1
	PWM
	Timer I
	Serial I2C
Lowest priority:	ADC

Serial Communications

The S87C752 contains an I2C serial communications port instead of the SC80C51 UART. The I2C serial port is a single bit hardware interface with all of the hardware necessary to support multimaster and slave operations. Also included are receiver digital filters and timer (timer I) for communication watch-dog purposes. The I2C serial port is controlled through four special function registers; I2C control, I2C data, I2C status, and I2C configuration.

Pulse Width Modulation Output (P0.4)

The PWM outputs pulses of programmable length and interval. The repetition frequency is defined by an 8-bit prescaler which generates the clock for the counter. The prescaler register is PWMP. The prescaler and counter are not associated with any other timer. The 8-bit counter counts modulo 255, that is from 0 to 254 inclusive. The value of the 8-bit counter is compared to the contents of a compare register, PWM. When the counter value matches the contents of this register, the output of PWM is set high. When the counter reaches zero, the output of PWM is set low. The pulse width ratio (duty cycle) is defined by the contents of the compare register and is in the range of 0 to 1 programmed in increments of 1/255. The PWM output can be set to be continuously high by loading the compare register with 0 and the output can be set to be continuously low by loading the compare register with 255. The PWM output is enabled by a bit in a special function register, PWENA. When enabled, the pin output is driven with a fully active pull-up. That is, when the output is high, a strong pull-up is continuously applied. When disabled, the pin behaves as a normal bidirectional I/O pin, however, the counter remains active.

The PWM function is disable during RESET and remains disabled after reset is removed until re-enabled by software. The PWM output is high during power down and idle. The counter is disabled during idle. The repetition frequency of the PWM is given by:

$$f_{PWM} = f_{OSC} / 2 (1 + PWMP) 255$$

The low/high ratio of the PWM signal is PWM / (255 – PWM) for PWM not equal to 255. For PWM = 255, the output is always low.

The repetition frequency range is 92Hz to 23.5kHz for an oscillator frequency of 12MHz.

An interrupt will be asserted upon PWM counter overflow if the interrupt is not masked off.

The PWM output is an alternative function of P0.4. In order to use this port as a bidirectional I/O port, the PWM output must be disabled by clearing the enable/disable bit in PWENA. In this case, the PWM subsystem can be used as an interval timer by enabling the PWM interrupt.

A/D Converter

The analog input circuitry consists of a 5-input analog multiplexer and an A to D converter with 8-bit resolution. The conversion takes 40 machine cycles, i.e., 40μs at 12MHz oscillator frequency. The A/D converter is controlled using the ADCON control register. Input channels are selected by the analog multiplexer through ADCON register bits 0–2.

Special Function Register Addresses
Special function register addresses for the S87C752 are identical to those of the SC80C51, except for the changes listed below:

SC80C51 special function registers not present in the S87C752 are TMOD (89), P2 (A0) and IP (B8). The SC80C51 registers TH1, TL1, SCON, and SBUF are replaced with the S87C752 registers RTH, RTL, I2CON, and I2DAT, respectively. Additional special function registers are I2CFG (D8), I2STA (FB), ADCON (A0), ADAT (84), PWM (8E), PWMP (8F), and PWENA (FE).

Table 2. I²C Special Function Register Addresses

Register Address			Bit Address							
Name	Symbol	Address	MSB							LSB
I2C control	I2CON	98	9F	9E	9D	9C	9B	9A	99	98
I2C data	I2DAT	99	–	–	–	–	–	–	–	–
I2C configuration	I2CFG	D8	DF	DE	DD	DC	DB	DA	D9	D8
I2C status	I2STA	F8	FF	FE	FD	FC	FB	FA	F9	F8

ABSOLUTE MAXIMUM RATINGS

SYMBOL	PARAMETER	RATING	UNIT
T_{STG}	Storage temperature range	−65 to +150	°C
V_{CC}	Voltage from V_{CC} to V_{SS}[3]	−0.5 to +6.5	V
V_S	Voltage from any pin to V_{SS}[3]	−0.5 to V_{CC} + 0.5	V
P_D	Power dissipation	1.0	W

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, AV_{CC} = 5V ±10%, AV_{SS} = 0V[3]

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS		UNIT
			Min	Max	
I_{CC}	Supply current (see Figure 3)			TBD	
Inputs					
V_{IL}	Input low voltage, except SDA, SCL		−0.5	$0.2V_{CC}$−0.1	V
V_{IH}	Input high voltage, except X1, RST		$0.2V_{CC}$+0.9	V_{CC}+0.5	V
V_{IH1}	Input high voltage, X1, RST		$0.7V_{CC}$	V_{CC}+0.5	V
	SDA, SCL:				
V_{IL1}	Input low voltage		−0.5	$0.3V_{CC}$	V
V_{IH2}	Input high voltage		$0.7V_{CC}$	V_{CC}+0.5	V
Outputs					
V_{OL}	Output low voltage, ports 1, 3, 0.3, and 0.4 (PWM disabled)	I_{OL} = 1.6mA		0.45	V
V_{OL1}	Output low voltage, port 0.2	I_{OL} = 3.2mA		0.45	V
V_{OH}	Output high voltage,ports 1, 3, 0.3, and 0.4 (PWM disabled)	I_{OH} = −60µA	2.4		V
		I_{OH} = −25µA	$0.75V_{CC}$		V
		I_{OH} = −10µA	$0.9V_{CC}$		V
V_{OH2}	Output high voltage, P0.4 (PWM enabled)	I_{OH} = −400µA	2.4		V
		I_{OH} = −40µA	$0.9V_{CC}$		V
	Port 0.0 and 0.1 (I2C) – Drivers				
V_{OL2}	Output low voltage	I_{OL} = 3mA (over V_{CC} range)		0.4	V
	Driver, receiver combined:				
C	Capacitance			10	pF
I_{IL}	Logical 0 input current, ports 1, 3, 0.3, and 0.4 (PWM disabled)	V_{IN} = 0.45V		−50	µA
I_{TL}	Logical 1 to 0 transition current, ports 1, 3, 0.3 and 0.4	V_{IN} = 2V		−650	µA
I_{LI}	Input leakage current, port 0.0, 0.1 and 0.2	0.45 < V_{IN} < V_{CC}		±10	µA
R_{RST}	Reset pull-down resistor		25	175	kohm
C_{IO}	Pin capacitance	Test freq = 1MHz, T_A = 25°C		10	pF
I_{PD}	Power-down current[6]	V_{CC} = 2 to 5.5V		50	µA
V_{PP}	V_{PP} program voltage	V_{SS} = 0V, V_{CC} = 5V±10%, T_A = 21°C–27°C	12.5	13.0	V
I_{PP}	Program current	V_{PP} = 13.0V		10	mA

DC ELECTRICAL CHARACTERISTICS (Continued)

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			Min	Typical	Max	
Analog Inputs (A/D guaranteed only with quartz window covered.)						
AV_{CC}	Analog supply voltage[9]	AV_{CC} = V_{CC}±0.2	4.5		5.5	V
AI_{CC} AI_{ID} AI_{PD}	Analog supply current Operating Idle mode[8] Power-down[8]	AV_{CC} = 5.12V			 TBD TBD TBD	 mA mA mA
AV_{IN}	Analog input voltage		AV_{SS}–0.2		AV_{CC}+0.2	V
C_{IA}	Analog input capacitance				TBD	pF
t_{ADS}	Sampling time				$8t_{CY}$	s
t_{ADC}	Conversion time				$40t_{CY}$	s
R	Resolution				8	bits
E_{RA}	Relative accuracy[8]				±1	LSB
OS_e	Zero scale offset[8]		TBD			LSB
G_e	Full scale gain error[8]				TBD	%
M_{CTC}	Channel to channel matching				TBD	LSB
C_t	Crosstalk	0 – 100kHz			TBD	dB

NOTES:
1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions than those described in the AC and DC Electrical Characteristics section of this specification is not implied.
2. This product includes circuitry specifically designed for the protection of its internal devices from damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying voltages greater than the rated maxima.
3. Parameters are valid over operating temperature range unless otherwise specified. All voltages with respect to V_{SS} unless otherwise noted.
4. Power-down ICC is measured with all output pins disconnected; port 0 = VCC; X2, X1 n.c.; RST = Vss.
5. Icc is measured with all output pins disconnected; X1 driven with t_{CLCH}, t_{CHCL} = 5ns, V_{IL} = Vss + 0.5V, V_{IH} = Vcc – 0.5V; X2 n.c.; RST = port 0 = VCC. ICC will be slightly higher if a crystal oscillator is used.
6. Idle Icc is measured with all output pins disconnected; X1 driven with t_{CLCH}, t_{CHCL} = 5ns, V_{IL} = VSS + 0.5V, V_{IH} = Vcc – 0.5V; X2 n.c.; port 0 = VCC; RST = VSS.
7. Load capacitance for ports = 80pF.
8. The resistor ladder network is not disconnected in the power down or idle modes. Thus, to conserve power, the user may remove AV_{CC}.
9. If the A/D function is not required, or if the A/D function is only needed periodically, AV_{CC} may be removed without affecting the operation of the digital circuitry. Contents of ADCON and ADAT are not guaranteed to be valid. Digital inputs on P1.0–P1.4 will not function normally.

A/D CONVERTER PARAMETER DEFINITIONS

The following list of definitions are included to clarify the specifications given. These definitions are not meant to be a complete set of A/D parameter definitions.

Absolute Accuracy Error

Absolute accuracy error at a given output is the difference between the theoretical analog input voltage to produce a given output code and the actual analog input voltage required to produce the same code. Since the same output code is produced by a band of input voltages, the "required input voltage" is defined as the midpoint of the band of input voltages that will produce that code. Absolute accuracy error not specified with a code is the maximum over all codes.

Absolute accuracy error includes gain error, offset error and relative accuracy error, and accounts for all deviations from an ideal converter.

Nonlinearity

If a straight line is drawn between the end points of the actual converter characteristic such that zero offset and full scale errors are removed, then nonlinearity is the maximum deviation of the code transitions of the actual characteristic from that of the straight line so constructed. This is also referred to as relative accuracy and also integral nonlinearity.

Differential Nonlinearity

Differential nonlinearity is the maximum difference between the actual and ideal code widths of the converter. The code widths are the differences expressed in LSB between the code transition points, as the input voltage is varied through the range for the complete set of codes.

Gain Error

Gain error is the deviation between the ideal and actual analog input voltage required to cause the final code transition to a full-scale output code after the offset error has been removed. This may sometimes be referred to as full scale error.

Offset Error

Offset error is the difference between the actual input voltage that causes the first code transition and the ideal value to cause the first code transition. This ideal value is 1/2 LSB above V_{ref}–.

Channel to Channel Matching

Channel to channel matching is the maximum difference between the corresponding code transitions of the actual characteristics taken from different channels under the same temperature, voltage and frequency conditions.

Crosstalk

Crosstalk is the measured level of a signal at the output of the converter resulting from a signal applied to one deselected channel.

Total Error
Maximum deviation of any step point from a line connecting the ideal first transition point to the ideal last transition point.

Relative Accuracy
Relative accuracy error is the deviation of the ADC's actual code transition points from the ideal code transition points on a straight line which connects the ideal first code transition point and the final code transition point, after nulling offset error and gain error. It is generally expressed in LSBs or in percent of FSR.

AC ELECTRICAL CHARACTERISTICS T_A = 0°C to +70°C or T_A = −40°C to +85°C, V_{CC} = 5V±10%, V_{SS} = 0V[3, 7]

SYMBOL	PARAMETER	12MHz		VARIABLE CLOCK		UNIT
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator frequency			3.5 3.5 0.5	12 16 12	MHz
	External clock (Figure 1)					
t_{CHCH}	High time	20				ns
t_{CLCL}	Low time	20				ns
t_{CLCH}	Rise time		20			ns
t_{CHCL}	Fall time		20			ns

AC SYMBOL DESIGNATIONS
Each timing symbol has five characters. The first character is always 't' (= time). The other characters, depending on their positions, indicate the name of a signal or the logical status of that signal. The designations are:
C – Clock
D – Input data
H – Logic level high
L – Logic level low
Q – Output data
T – Time
V – Valid
X – No longer a valid logic level
Z – Float

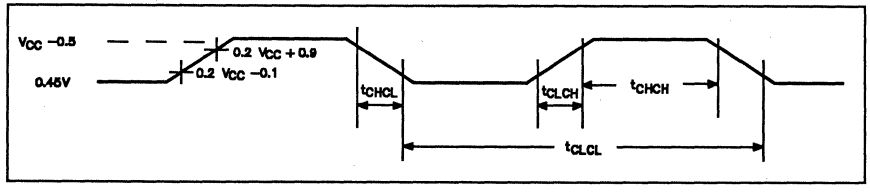


Figure 1. External Clock Drive Waveform

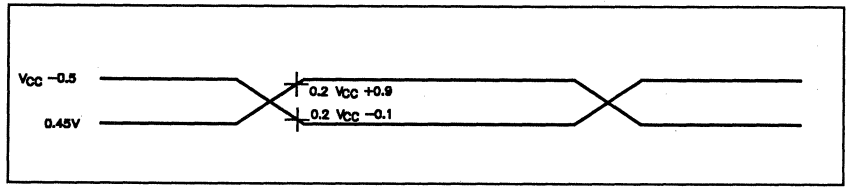


Figure 2. AC Testing Input/Output Waveform

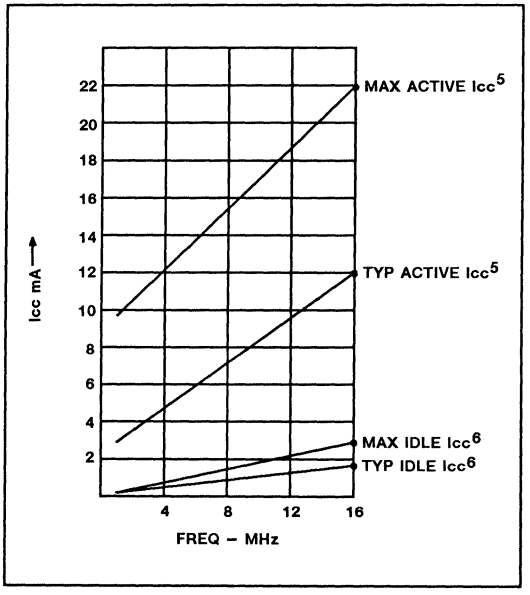


Figure 3. I_{CC} vs. FREQ
Maximum I_{CC} values taken at V_{CC} and Worst Case Temperature
Typical I_{CC} values taken at V_{CC} = 5V and 25°C
Notes 5 and 6 refer to AC Electrical Characteristics

PROGRAMMING CONSIDERATIONS

EPROM Characteristics

The 87C752 is programmed by using a modified Quick-Pulse Programming algorithm similar to that used for devices such as the 87C451 and 87C51. It differs from these devices in that a serial data stream is used to place the 87C752 in the programming mode.

Figure 4 shows a block diagram of the programming configuration for the 87C752. Port pin P0.2 is used as the programming voltage supply input (V_{PP} signal). Port pin P0.1 is used as the program (PGM/) signal. This pin is used for the 25 programming pulses.

Port 3 is used as the address input for the byte to be programmed and accepts both the high and low components of the eleven bit address. Multiplexing of these address components is performed using the ASEL input. The user should drive the ASEL input high and then drive port 3 with the high order bits of the address. ASEL should remain high for at least 13 clock cycles. ASEL may then be driven low which latches the high order bits of the address internally. The high address should remain on port 3 for at least two clock cycles after ASEL is driven low. Port 3 may then be driven with the low byte of the address. The low address will be internally stable 13 clock cycles later. The address will remain stable provided that the low byte placed on port 3 is held stable and ASEL is kept low. **Note:** ASEL needs to be pulsed high only to change the high byte of the address.

Port 1 is used as a bidirectional data bus during programming and verify operations. During the programming mode, it accepts the byte to be programmed. During the verify mode, it provides the contents of the EPROM location specified by the address which has been supplied to Port 3.

The XTAL1 pin is the oscillator input and receives the master system clock. This clock should be between 1.2 and 6 MHz.

The RESET pin is used to accept the serial data stream that places the 87C752 into various programming modes. This pattern consists of a 10-bit code with the LSB sent first. Each bit is synchronized to the clock input, X1.

Programming Operation

Figures 5 and 6 show the timing diagrams for the program/verify cycle. RESET should initially be held high for at least two machine cycles. P0.1 (PGM/) and P0.2 (V_{PP}) will be at V_{OH} as a result of the RESET operation. At this point, these pins function as normal quasi-bidirectional I/O ports and the programming equipment may pull these lines low. However, prior to sending the 10 bit code on the reset pin, the programming equipment should drive these pins high (V_{IH}). The RESET pin may now be used as the serial data input for the data stream which places the 87C752 in the programming mode. Data bits are sampled during the clock high time and thus should only change during the time that the clock is low. Following transmission of the last data bit, the RESET pin should be held low.

Next the address information for the location to be programmed is placed on port 3 and ASEL used to perform the address multiplexing, as previously described. At this time, port 1 functions as an output.

A high voltage V_{PP} level is then applied to the V_{PP} input (P0.2). (This sets Port 1 as an input port.). The data to be programmed into the EPROM array is then placed on Port 1. This is followed by a series of programming pulses applied to the PGM/ pin (P0.1). These pulses are created by driving P0.1 low and then high. This pulse is repeated until a total of twenty-five programming pulses have occurred. At the conclusion of the last pulse, the PGM/ signal should remain high.

The V_{PP} signal may now be driven to the V_{OH} level, placing the 87C752 in the verify mode. (Port 1 is now used as an output port.). After 4 machine cycles (48 clock periods) the contents of the addressed location in the EPROM array will appear on Port 1.

The next programming cycle may now be initiated by placing the address information at the inputs of the multiplexed buffers, driving the V_{PP} pin to the V_{PP} voltage level, providing the byte to be programmed to Port 1 and issuing the 25 programming pulses on the PGM/ pin, bringing V_{PP} back down to the V_{CC} level and verifying the byte.

Programming Modes

The 87C752 has four programming features incorporated within its EPROM array. These include the USER EPROM for storage of the application's code, a sixteen byte encryption KEY array and two security bits. Programming and verification of these four elements are selected by a combination of the serial data stream applied to the RESET pin and the voltage levels applied to port pins P0.1 and P0.2. The various combinations are shown in Table 3.

Encryption Key Table

The 87C752 includes a 16 byte EPROM array that is programmable by the end user. The contents of this array can then be used to encrypt the program memory contents during a program memory verify operation. When a program memory verify operation is performed, the contents of the program memory location is XNOR'ed with one of the bytes in the 16 byte encryption table. The resulting data pattern is then provided to port 1 as the verify data. The encryption mechanism can be disabled, in essence, by leaving the bytes in the encryption table in their erased state (FFH) since the XNOR product of a bit with a logical one will result in the original bit. The encryption bytes are mapped with the code memory in 16 byte groups. The first byte in code memory will be encrypted with the first byte in the encryption table; the second byte in code memory will be encrypted with the second byte in the encryption table and so forth up to and including the sixteenth byte. The encryption repeats in 16 byte groups; the seventeenth byte in the code memory will be encrypted with the first byte in the encryption table, and so forth.

Security Bits

Two security bits, security bit 1 and security bit 2, are provided to limit access to the USER EPROM and encryption key arrays. Security bit 1 is the program inhibit bit, and once programmed performs the following functions:

1. Additional programming of the USER EPROM is inhibited.
2. Additional programming of the encryption key is inhibited.
3. Verification of the encryption key is inhibited.
4. Verification of the USER EPROM and the security bit levels may still be performed.

(If the encryption key array is being used, this security bit should be programmed by the user to prevent unauthorized parties from reprogramming the encryption key to all logical zero bits. Such programming would provide data during a verify cycle that is the logical compliment of the USER EPROM contents.)

Security bit 2, the verify inhibit bit, prevents verification of both the USER EPROM array and the encryption key arrays. The security bit levels may still be verified.

Programming and Verifying Security Bits

Security bits are programmed employing the same techniques used to program the USER EPROM and KEY arrays using serial data streams and logic levels on port pins indicated in Table 3. When programming either security bit, it is not necessary to provide address or data information to the 87C752 on ports 1 and 3.

Verification occurs in a similar manner using the RESET serial stream shown in Table 3. Port 3 is not required to be driven and the results of the verify operation will appear on ports 1.6 and 1.7.

Port 1.7 contains the security bit 1 data and is a logical one if programmed and a logical zero if erased. Likewise, P1.6 contains the security bit 2 data and is a logical one if programmed and a logical zero if erased.

Erasure Characteristics

Erasure of the EPROM begins to occur when the chip is exposed to light with wavelengths shorter than approximately 4,000 angstroms. Since sunlight and fluorescent lighting have wavelengths in this range, exposure to these light sources over an extended time (about 1 week in sunlight, or 3 years in room level fluorescent lighting) could cause inadvertent erasure. **For this and secondary effects, it is recommended that an opaque label be placed over the window.** For elevated temperature or solvent environments, use Kapton tape Fluorglas part number 2345-5 or equivalent. A/D performance is not guaranteed unless window is covered.

The recommended erasure procedure is exposure to ultraviolet light (at 2537 angstroms) to an integrated dose of at least 15W-sec/cm^2. Exposing the EPROM to an ultraviolet lamp of 12,000µW/cm^2 rating for 20 to 39 minutes, at a distance of about 1 inch, should be sufficient.

Erasure leaves the array in an all 1s state.

Table 3. Implementing Program/Verify Modes

Operation	Serial Code	P0.1 (PGM/)	P0.2 (V_{PP})
Program user EPROM	296H	~*	V_{PP}
Verify user EPROM	296H	V_{IH}	V_{IH}
Program key EPROM	292H	~*	V_{PP}
Verify key EPROM	292H	V_{IH}	V_{IH}
Program security bit 1	29AH	~*	V_{PP}
Program security bit 2	298H	~*	V_{PP}
Verify security bits	29AH	V_{IH}	V_{IH}

NOTE:
*Pulsed from V_{IH} to V_{IL} and returned to V_{IH}.

EPROM PROGRAMMING AND VERIFICATION CHARACTERISTICS T_A = 21°C to +27°C, V_{CC} = 5V±10%, V_{SS} = 0V

SYMBOL	PARAMETER	MIN	MAX	UNIT
$1/t_{CLCL}$	Oscillator/clock frequency	1.2	6	MHz
t_{AVGL}*	Address setup to P0.1 (PROG-) low	10µs + 24t_{CLCL}		
t_{GHAX}	Address hold after P0.1 (PROG-) high	48t_{CLCL}		
t_{DVGL}	Data setup to P0.1 (PROG-) low	38t_{CLCL}		
t_{GHDX}	Data hold after P0.1 (PROG-) high	36t_{CLCL}		
t_{SHGL}	V_{PP} setup to P0.1 (PROG-) low	10		µs
t_{GHSL}	V_{PP} hold after P0.1 (PROG-)	10		µs
t_{GLGH}	P0.1 (PROG-) width	90	110	µs
t_{AVQV}**	V_{PP} low (V_{CC}) to data valid		48t_{CLCL}	
t_{GHGL}	P0.1 (PROG-) high to P0.1 (PROG-) low	10		µs
t_{SYNL}	P0.0 (sync pulse) low	4t_{CLCL}		
t_{SYNH}	P0.0 (sync pulse) high	8t_{CLCL}		
t_{MASEL}	ASEL high time	13t_{CLCL}		
t_{MAHLD}	Address hold time	2t_{CLCL}		
t_{HASET}	Address setup to ASEL	13t_{CLCL}		
t_{ADSTA}	Low adress to address stable	13t_{CLCL}		

NOTES:
*Address should be valid at least 24t_{CLCL} before the rising edge of P0.2 (V_{PP}).
**For a pure verify mode, i.e., no program mode in between, t_{AVQV} is 14t_{CLCL} maximum.

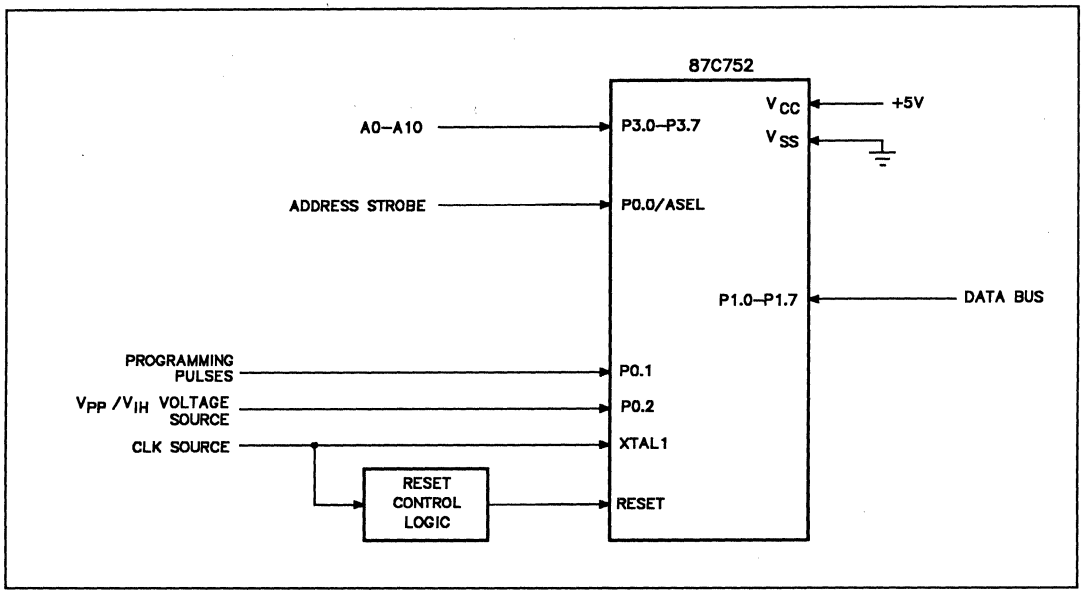


Figure 4. Programming Configuration

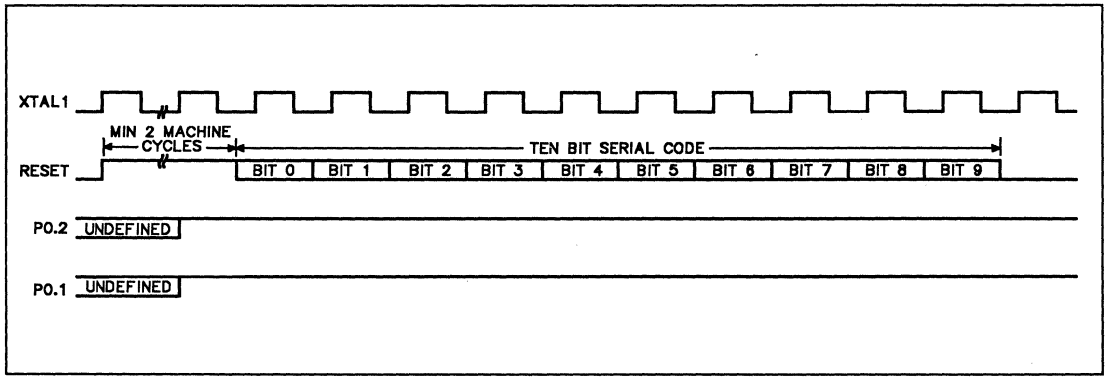


Figure 5. Entry Into Program/Verify Modes

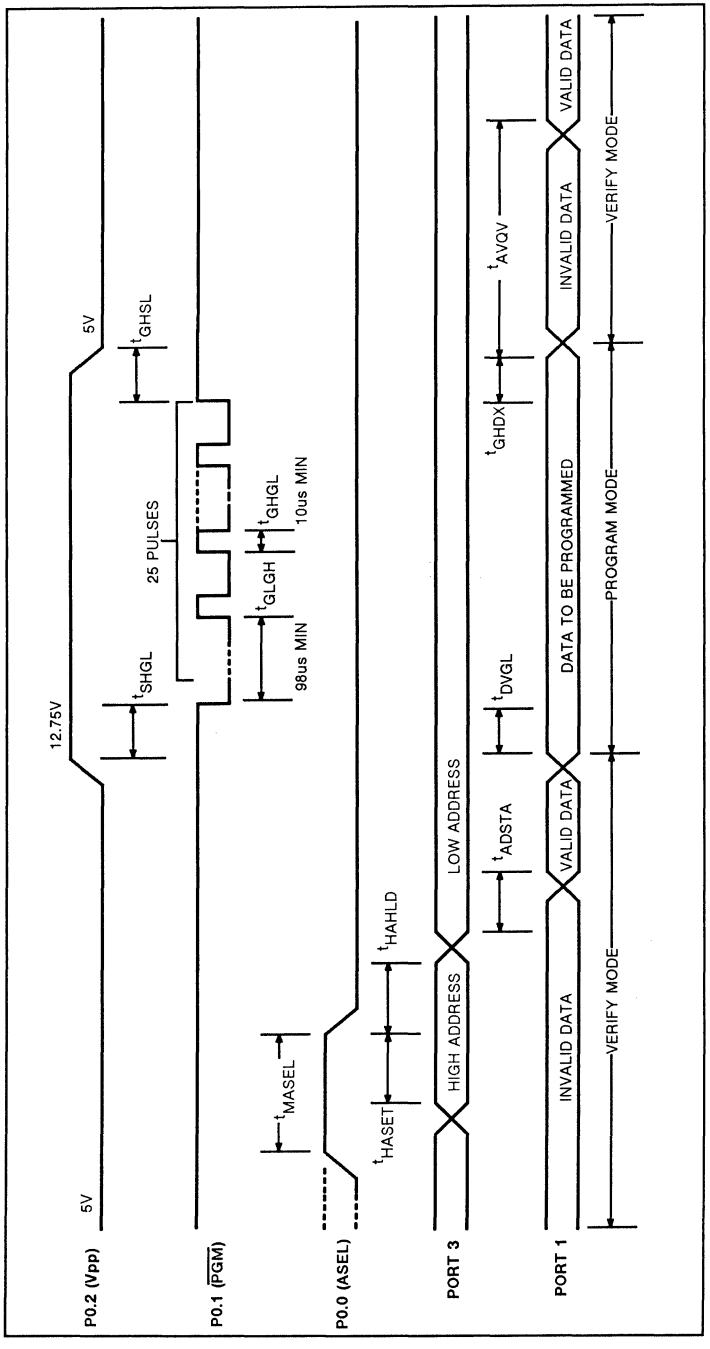


Figure 6. Program/Verify Cycle

Signetics

Microprocessor Products

Section 3
Application Notes

INDEX

Signetics

AN408
SC80C451 Operation of Port 6

Application Note

Microprocessor Division

INTRODUCTION

The features of the 80C451 are shared with the 80C51 or are conventional except for the operation of port 6. The flexibility of this port facilitates high-speed parallel data communications. This application note discusses the use of port 6 and is divided into the following sections:

1. Port 6 as a processor bus interface.
2. Using port 6 as a standard pseudo bi-directional I/O port.
3. Implementation of parallel printer ports.

This information applies to all versions of the part: 80C451, 83C451 and the 87C-451.

PORT 6 AS A PROCESSOR BUS INTERFACE

Port 6 allows use of the 80C451 as an element on a microprocessor type bus. The host processor could be a general purpose MPU or the data bus of a microcontroller like the 80C451 itself. This feature allows single or multiple 80C451 controllers to be used on a bus as flexible peripheral processing elements. Applications could include keyboard scanners, serial I/O controllers, servo controllers, etc.

OPERATION

On reset, port 6 is programmed correctly for use as a bus interface (see Figure 2). This prevents the interface from disrupting data on the bus of the host processor during power-up. Software initialization of the CSR (Control Status Register) is not required. A dummy read of port 6 may be required to clear the IBF (Input Buffer Full) flag since it could be set by turn on transients on the bus of the host processor. On reset the CSR of the 83C451 is programmed to allow the following:

1. AFLAG is an input controlling the port select function. If AFLAG is high, the contents of the CSR is output on port 6 when the port is read by the host. If AFLAG is low, then the contents of the output latch is output when port 6 is read by the host.
2. BFLAG is an input controlling the port enable function. In this mode when BFLAG is high, the input latch and the output drivers are disabled and the flags are not affected by the $\overline{IDS}$ (Input Data Strobe) or $\overline{ODS}$ (Output Data Strobe) signals. When BFLAG is low, the port is enabled for reading and writing under the control of $\overline{IDS}$ and $\overline{ODS}$ pins.

Figure 1 shows one possible example of an 80C451 on a memory bus. This arrangement allows the main processor to query port 6 for flag status without interrupting the 80C451. If the address decoder, shown in Figure 1, enables port 6 on the 80C451 when the address is 8000H or 8001H, and the address line A0 controls the port select feature, then the host processor can read and write to port 6 using address 8000H. Since the port select function is being controlled by the address line A0, the CSR contents can be read by the host processor at address 8001H.

By testing the CSR contents in this way, the host processor can tell if new data has been written to the port 6 output latch since it last read the port or if the 80C451 has read the last byte that the host wrote to the port. Conversely the 80C451 can poll the flags in its CSR to see if the host processor has written to or read from port 6 since the last time it serviced the port.

If desired, an interrupt source for the 80C451 can be derived easily from the port enable source as shown by the dashed line in Figure 1.

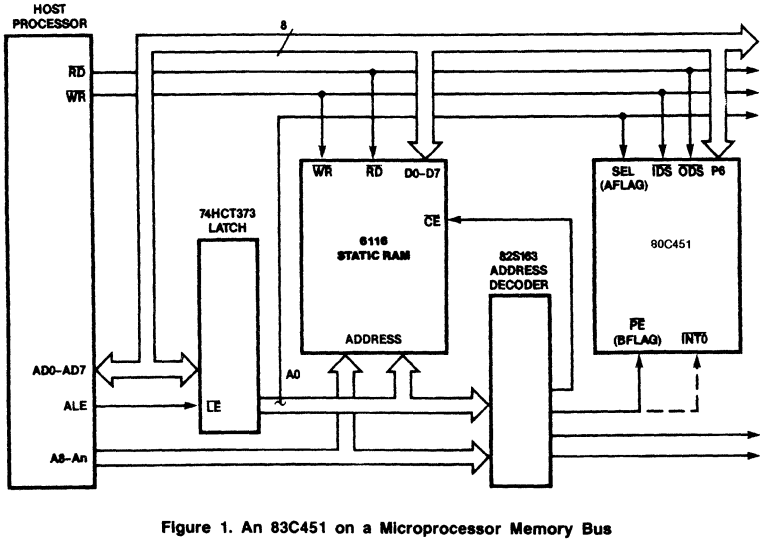


Figure 1. An 83C451 on a Microprocessor Memory Bus

SOFTWARE EXAMPLES

To write to port 6 on the bus shown in Figure 1, the host processor first reads the CSR contents at address 8001H, and tests the input buffer full flag (CSR bit 0). If the flag is clear the host writes a byte to address 8000H. This loads the input buffer latch of port 6 and sets the input buffer full flag. Conversely the 80C451 polls the IBF flag and reads a byte from port 6 when it finds the flag set. The flag is automatically reset when this internal read occurs.

80C451 ROUTINE TO READ ONE BYTE FROM HOST VIA PORT 6

```
RCVR:   JNB CSR.0,RCVR      ;TEST IBF FLAG
        MOV A,P6            ;WHEN FLAG IS SET READ BYTE
        RET
```

80C51 ROUTINE TO WRITE ONE BYTE TO THE 83C451 PORT 6

If the host processor is an 80C51 the following routine will write a byte of data to the 80C451. The data involved is passed to the routine through register 1.

```
XMIT:   MOV DPTR,8001H
TEST:   MOVX A,@DPTR        ;READ THE CSR
        JB ACC.0,TEST       ;TEST IBF FLAG
        MOV DPTR,8000H
        MOV A,R1
        MOVX @DPTR,A        ;WRITE DATA TO THE 451
        RET
```

80C451 ROUTINE TO WRITE ONE BYTE TO HOST VIA PORT 6

Routines for data transfer in the opposite direction are similar to the above two. The 80C451 version is given below.

```
XMIT:   JB CSR.1,XMIT       ;TEST OBF FLAG
        MOV P6,A            ;WRITE DATA
        RET
```

CSR 7	CSR 6	CSR 5	CSR 4	CSR 3	CSR 2	CSR 1	CSR 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
1	1	1	1	1	1		

Figure 2. CSR Programmed to Allow Port 6 as a Bus Interface

USING PORT 6 AS A STANDARD QUASI-BIDIRECTIONAL I/O PORT

To use port 6 as a common I/O port, all of the control pins are tied to ground (see Figure 3). On hardware reset, bits 2 – 7 in the CSR are set to one. Port operation and electrical characteristics become identical to port 1 on the 80C51 and the 80C451 ports 1, 4, and 5. No software initialization is required.

If desired, AFLAG and BFLAG can be used as outputs while port 6 is operating as a standard quasi-bidirectional I/O port (see Figure 4). In this case, only $\overline{IDS}$ and $\overline{ODS}$ are tied to ground and the CSR is initialized to allow operation of AFLAG and BFLAG as simple outputs (see Figure 5).

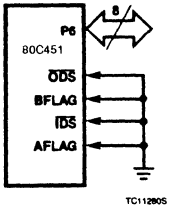


Figure 3. Standard I/O Port on Reset

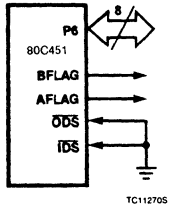


Figure 4. Standard I/O Port on Reset with AFLAG and BFLAG as Outputs

IMPLEMENTATION OF PARALLEL PRINTER PORTS USING PORT 6

The 80C451 is an excellent choice for a printer controller. The 80C451 has the facilities to permit all of the intelligent features of a common printer to be handled by a single chip:

1. The features of port 6 allow a parallel printer port to be designed with only line driving and receiving chips required as additional hardware.
2. The onboard UART allows RS232 interfacing with only level shifting chips added.
3. The 8-bit parallel ports 0 to 6 are ample to drive onboard control functions, even when ports are used for external memory access, interrupts and other functions.
4. The RAM addressing ability of ports 0 and 2 can be used to address up to 64K bytes of a hardware buffer/spooler. AFLAG and BFLAG as simple outputs (see Figure 5).
5. The 64K byte ROM addressing capability allows space for the most sophisticated software.

In addition, either end of a parallel interface can be implemented using port 6, and the interfaces can be interrupt driven or polled in either case.

CSR 7	CSR 6	CSR 5	CSR 4	CSR 3	CSR 2	CSR 1	CSR 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
0	X	0	X	X	1		

Figure 5. CSR Programmed to Allow AFLAG and BFLAG to Operate as Outputs and Port 6 as a Standard I/O Port

DATA TRANSFER SIGNAL PINS		
Pin No.	Ground Return Pin No.	Signal
1	19	$\overline{STROBE}$
2	20	DATA 1
3	21	DATA 2
4	22	DATA 3
5	23	DATA 4
6	24	DATA 5
7	25	DATA 6
8	26	DATA 7
9	27	DATA 8
10	28	$\overline{ACKNLG}$
11	29	BUSY

TYPICAL AUXILIARY PIN FUNCTIONS	
Pin No.	Signal
12	PAPER OUT
14	$\overline{AUTO\ LINE\ FEED}$
16	LOGIC GROUND
17	CHASSIS GND
30	GROUND RETURN
31	$\overline{RESET\ PRINTER}$
32	$\overline{ERROR}$
33	GROUND RETURN
36	$\overline{SLCT\ IN}$

Figure 6. Parallel Printer Interface Pin Functions

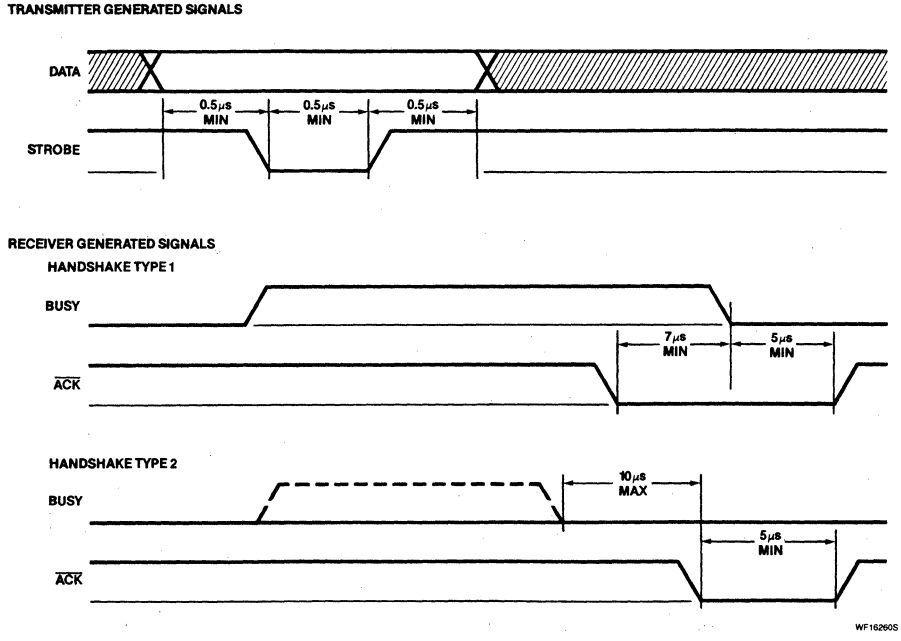


Figure 7. Parallel Printer Interface Signals

THE INTERFACE

Data transfer on a parallel printer interface occurs across eleven signal lines. The other conductors on the standard plug are used as ground returns or for auxiliary functions (see Figure 6). Only the data transfer signals will be considered.

The Data Transfer Format

The parallel printer interfaces are far more standardized in features than their serial counterpart. However at least three significant variations exist in handshake style in printers using generic parallel interfaces. This fact influences the design of both port hardware and software. A good transmitter should be able to drive devices with all three styles of handshakes and a good receiver should generate the handshake most likely compatible with any transmitter.

The Variations

Type 1 — Figure 7 shows a common style of handshake and is the style that will be implemented in the receiver examples. A busy signal and an acknowledge strobe pulse are generated for every byte received.

Type 2 — Another style of handshake generates a busy signal only when the printer will not be able to accept more data for a relatively long time. Acknowledge pulses are created after every byte received. When the busy signal is generated after a byte is received, the associated acknowledge pulse does not occur until **after** the busy signal returns to logic zero. (see Figure 7).

Type 3 — A third handshake style does not generate acknowledge pulses, but a busy signal is produced after every byte is received.

PARALLEL PRINTER INTERFACES USING POLLING

Transmitter Operation

This application illustrates the flexibility of the port 6 logic in solving an applications problem. We need to be able to handle all types of acknowledge signals that might be received by the transmitter. We will use the $\overline{ODS}$ pin and output buffer full flag logic to record the receipt of the acknowledge pulse (see Figure 8), but not all parallel receivers generate acknowledge pulses. We could poll the busy signal line, but not all receivers generate busy signals for each byte received; so lack of a busy signal does not imply that we can send another byte. We can, however, expect an acknowledge pulse very shortly after the end of a busy signal if one is going to arrive at all. So we can send a new data byte after having received either a positive transition on the acknowledge line, or shortly after receiving a negative edge on the busy line.

The CSR is programmed to the output only mode. In this mode the $\overline{ODS}$ pin does not control the output drivers but only the output buffer full flag. The flag serves to record the positive transition of the acknowledge signal. The input latch is not used, but the $\overline{IDS}$ pin is used to set the input buffer full flag. This is used to record the negative transition at the end of the busy signal. Dummy reads by the 80C451 of port 6 will be used to clear the flag. In this example, the AFLAG mode is set only to place the port in the output only mode. The AFLAG pin is not actually used (see Figure 10).

The transmitter's CSR (control status register) is programmed to the following mode (see Figure 9):

1. CSR bit 6 controls the BFLAG output and therefore the strobe line.
2. The OBF (output buffer full) flag controls the AFLAG output.
3. The OBF is cleared on the positive edge of the $\overline{ODS}$ input.
4. The IBF flag is cleared on the negative edge of the $\overline{IDS}$ strobe.

NOTE:
With this combination of modes set, port 6 is in the output only mode.

Receiver Operation

In receiver operation, the $\overline{IDS}$ input is used to latch in the data transmitted on receipt of the strobe pulse. The receiver's CSR is programmed to allow the following:

1. The input buffer full flag is output through the BFLAG pin and is used as the busy signal to the transmitter.
2. The IBF flag is set and data is latched on the positive edge of $\overline{IDS}$.
3. Writing to the CSR bit 4 controls the AFLAG output and therefore the acknowledge line.

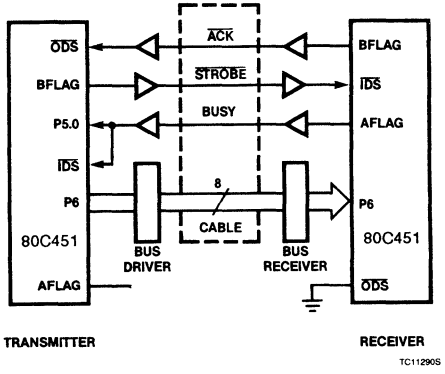


Figure 8. Interconnection for a Parallel Interface Using Polling

CSR 7	CSR 6	CSR 5	CSR 4	CSR 3	CSR 2	CSR 1	CSR 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
0	1	1	0	0	1		

Figure 9. CSR Programmed for Polled Transmitter Operation

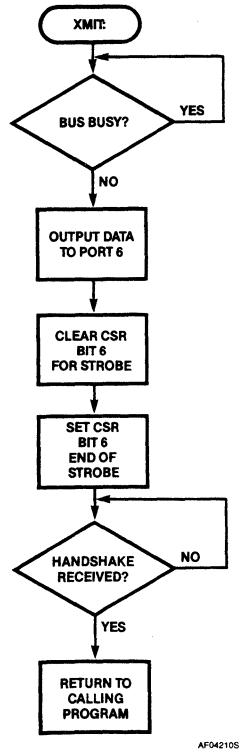


Figure 10. Flow Chart of Polled Parallel Transmitter Operation

CSR 7	CSR 6	CSR 5	CSR 4	CSR 3	CSR 2	CSR 1	CSR 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
1	0	0	1	1	0		

Figure 11. CSR Programmed for Polled Parallel Receiver Operation

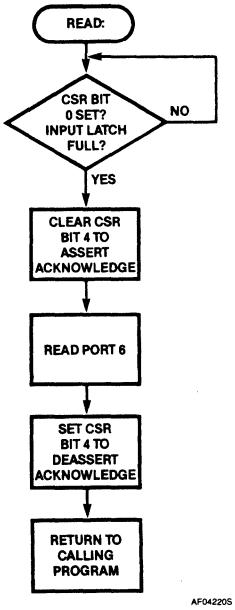


Figure 12. Flow Chart of Polled Parallel Receiver Operation

SOFTWARE EXAMPLES

This polled parallel transmit routine outputs one byte passed to it in the accumulator.

```
P_INIT:   MOV CSR,#064H      ;INITIALIZE PORT 6 OPERATING MODE
P_OUT:    JB P5.0            ;WAIT IF BUSY SIGNAL IS HIGH
          MOV P6,ACC         ;OUTPUT DATA
          MOV R1,P6          ;DUMMY READ TO CLEAR IBF FLAG
          MOV R1,#02H        ;INITIALIZE DELAY COUNTER
          CLEAR CSR.6        ;START STROBE PULSE
          DJNZ R1,$          ;TIME 6 MICROSECOND STROBE PULSE
          SETB CSR.6         ;END STROBE PULSE
WAIT:     JNB CSR.1,OUT      ;EXIT IF ACKNOWLEDGE RCV'D
          JNB CSR.0,WAIT     ;EXIT IF NEGATIVE BUSY EDGE RCV'D
          RET
```

This polled parallel receive routine places one byte in the accumulator each time it is called.

```
P_INIT:   MOV CSR,#09CH      ;INITIALIZE PORT 6 OPERATING MODE
          MOV R7,P6          ;DUMMY READ TO CLEAR IBF FLAG
P_IN:     JNB CSR.0          ;INPUT BUFFER LATCH FULL?
          CLR CSR.4          ;BEGIN ACKNOWLEDGE PULSE
          MOV R7,#02H        ;INITIALIZE DELAY COUNTER
          DJNZ R7,$          ;TIME ACKNOWLEDGE PULSE
          MOV A,P6           ;READ BYTE - CLEAR BUSY SIGNAL
          MOV R7,#02H        ;INITIALIZE DELAY COUNTER
          DJNZ R7,$          ;TIME ACKNOWLEDGE PULSE
          SETB CSR.4         ;END ACKNOWLEDGE PULSE
          RET
```

INTERRUPT DRIVEN PARALLEL PRINTER INTERFACE

Transmitter Operation

The transmitter's CSR (control status register) is programmed to the following mode (see Figure 14):

1. CSR bit 6 controls the BFLAG output and therefore the strobe line.
2. The OBF (output buffer full) flag controls the AFLAG output.
3. The OBF is cleared on the positive edge of the $\overline{ODS}$ (output data strobe) input.
4. The IBF flag is set on the negative edge of the $\overline{IDS}$ (input data strobe) pin.

NOTE:
With this combination of AFLAG and BFLAG modes set, port 6 is in the output only mode. The output drivers are always enabled and the $\overline{ODS}$ input is only used to clear the OBF flag.

INT0 is programmed to be negative edge sensitive and is connected to the OBF flag through the AFLAG pin. The OBF is cleared on the positive edge of $\overline{ODS}$. The net result is that INT0 is triggered on the end of the ACK pulse (a positive edge). This signals the transmitter that another byte may be transmitted. The transmitting 83C451 is free to do other tasks prior to this interrupt.

In this routine, Figure 15, the main program establishes a buffer in data memory ended by an ASCII end of text character. To begin outputting the buffer the routine PSEND is called. The rest of the buffer is emptied by the interrupt vectors to PSEND1.

For printers which generate acknowledge pulses, output rates of 25k transfers per second are achieved. Timer generated interrupts are used to periodically return program execution to the routine to service non-acknowledging printers and to provide a timeout feature. Non acknowledging printers are serviced at a rate of about 2.5k transfers per second. This maximum rate may be varied by adjusting the timer reload value. As written, the time out procedure attempts to retransmit a byte when the printer has not acknowledged for an excessively long time.

Receiver Operation

In receiver operation, the $\overline{IDS}$ input is used to latch in the data transmitted on receipt of the strobe pulse. The receiver's CSR is programmed to allow the following (see Figure 16):

1. The input buffer full flag is output through the BFLAG pin and is used as the busy signal to the transmitter. The IBF flag is set and data is latched on the positive edge of $\overline{IDS}$.
2. Writing to the CSR bit 4 controls the AFLAG output and therefore the acknowledge line.

The receiver is interrupted on the negative edge of the data strobe. Data is latched in on the positive edge of the strobe pulse (see Figure 17). Since the strobe pulse is normally very short there is little time lost between receiving the interrupt and having valid data in the input latch. The receiver is free to do other tasks prior to receiving the $\overline{INT0}$ interrupt.

SOFTWARE EXAMPLES

The software for the interrupt driven parallel receiver is similar to the polled receiver example. However, after an interrupt is received, this routine checks to confirm that data has been latched by the positive edge of the strobe pulse before proceeding with the routine.

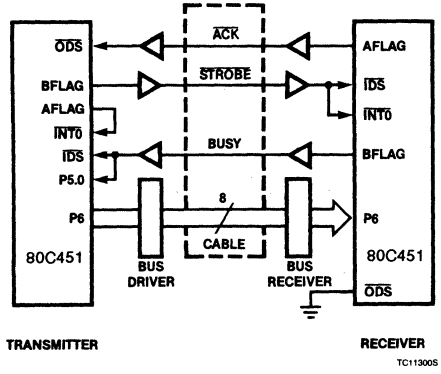


Figure 13. Interrupt Driven Parallel Interfaces Using 80C451 Controllers

CSR 7	CSR 6	CSR 5	CSR 4	CSR 3	CSR 2	CSR 1	CSR 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
0	1	1	0	1	1		

Figure 14. CSR Programming for Use as an Interrupt Driven Parallel Transmitter

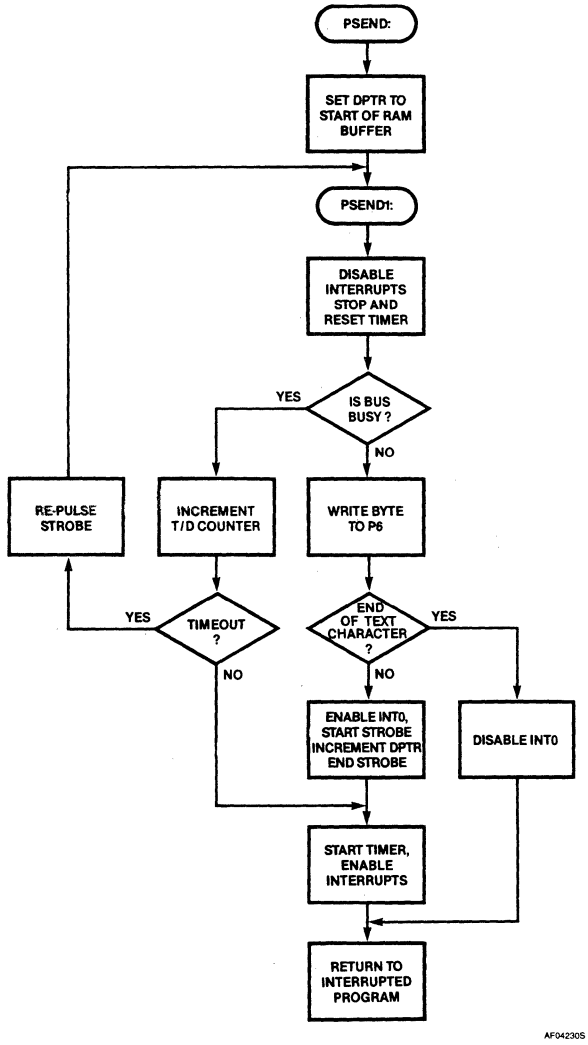


Figure 15. Flow Chart for an Interrupt Driven Parallel Transmitter

CSR 7	CSR 6	CSR 5	CSR 4	CSR 3	CSR 2	CSR 1	CSR 0
MB1	MB0	MA1	MA0	OBFC	IDSM	OBF	IBF
1	0	0	1	1	0		

Figure 16. CSR Programming for Use as an Interrupt Driven Parallel Receiver

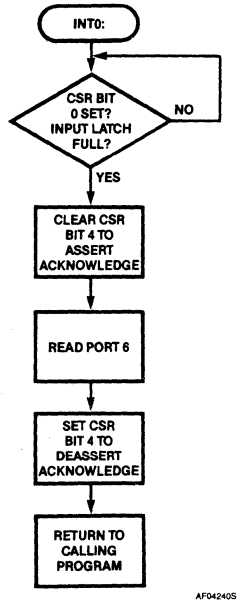


Figure 17. Flow Chart of Interrupt Driven Parallel Receiver Operation

SOFTWARE EXAMPLES

The software for the interrupt driven parallel receiver is similar to the polled receiver example. However, after an interrupt is received, this routine checks to confirm that data has been latched by the positive edge of the strobe pulse before proceeding with the routine.

```
INIT:     MOV CSR,#090H     ;INITIALIZE CSR
          SETB EX0          ;ENABLE INTERRUPT 0
          SETB IT0          ;SET NEG EDGE TRIGGERED INTERRUPTS
          SETB EA           ;ENABLE ALL INTERRUPTS

ORG EXTI0                   ;INTERRUPT 0 VECTOR
          JMP RCVR

RCVR:     JNB CSR.0,$       ;CONFIRM DATA LATCHED
          CLR CSR.4         ;START ACKNOWLEDGE PULSE
          MOV R7,#02H       ;INITIALIZE THE DELAY COUNTER
          DJNZ R7,$         ;TIME ACK PULSE
          MOV A,P6          ;READ BYTE - RESET BUSY LINE
          MOV R7,#02H       ;INITIALIZE THE DELAY COUNTER
          DJNZ R7,$         ;TIME ACK PULSE
          SETB CSR.4        ;END ACK PULSE
          RETI
```

This is the software for the interrupt driven parallel transmitter example.

```
; XMIT ROUTINE DRIVEN BY ACK PULSE GENERATED INTERRUPTS, OR TIMER GENERATED INTERRUPTS
; FOR NON ACKNOWLEDGING PRINTERS. READS DATA BUFFER IN EXTERNAL RAM STARTING AT 100H
; AND READING UNTIL 04H IS FOUND.

        ORG RESET
        JMP 26H
        ORG TIMER0
        JMP PSEND1
        ORG EXTI0
        JMP PSEND1
        ORG 26H
              MOV CSR,#064H          ;PORT 6 MODE
              MOV TMOD,#002H         ;CONFIGURE TIMER 0 TO 16 BITS
              SETB TO0               ;INT0 IS EDGE TRIGGERED
              SETB EA                ;ENABLE INTERRUPTS
PSEND:        MOV DPTR,#0100H        ;SET DPTR TO START OF TEXT
                                     ;BUFFER
PSEND1:       CLR EA                 ;DISABLE INTERRUPTS AND STOP
                                     ;TIMER
              CLR TR0                ;IF ENABLED
              CLR ET0
              MOV R7,00H             ;CLEAR TIMEOUT COUNTER
              MOV R6,00H
              MOV TH0,# - 4          ;SET TIMER INTERRUPT PERIOD
              MOV TL0,#00H
              JB 0C8H,BB             ;BUS BUSY
              MOV ACC,#00H           ;CLEAR ACCUMULATOR
              MOVX A,@DPTR           ;RETRIEVE FIRST BYTE
              MOV P6,ACC             ;OUTPUT FIRST BYTE
              CJNE A,#004H, CONT1    ;LOOK FOR END OF TEXT
              JMP EOTB
CONT1:        SETB EX0               ;ENABLE INT0
              CLR 0EEH               ;START STROBE PULSE
              INC DPTR
              MOV ACC,DPH            ;LOOK FOR PHYSICAL END OF
              JB ACC.2,EOTB          ;TEXT BUFFER
              SETB 0EEH              ;END STROBE PULSE
              JMP CONT
EOTB:         CLR EX0                ;END OF TEXT FOUND, DISABLE
                                     ;INT0
              SETB 0EEH
              SETB EA
              RETI
BB:           INC R7                 ;COUNT TIMER TIMEOUTS ON
                                     ;BUS BUSY
              CJNE R7,#00H, CONT     ;LOOK FOR OVERFLOW
              INC R6                 ;COUNT OVERFLOWS
              CJNE R6,#10H, CONT     ;TIMEOUT APPROX 5 SEC
              JMP TO
CONT:         SETB TR0               ;ENABLE TIMER INTERRUPT
              SETB ET0               ;START TIMER
              SETB EA
              RETI
TO:           CLR 0C9H               ;SEND NEW STROBE PULSE IN
                                     ;RESPONSE TO TIMEOUT
              NOP
              NOP
              MOV R6,#00H            ;RESET TO COUNTER
              MOV R7,#00H
              SETB 0C9H              ;END OF STROBE PULSE
              JMP PSEND1
```

Signetics

AN417
256K Centronics Printer Buffer
Using the SC87C451 Microcontroller

Application Note

Microprocessor Division

DESCRIPTION

This application note describes a stand alone Centronics type parallel printer buffer using the Signetics SC87C451 expanded I/O microcontroller. This type of unit would typically be placed between a personal computer and its printer. It captures the data to be printed at high speed, freeing the personal computer to go on to other tasks, and sends data to the printer as required. As described here, 256K dynamic RAMs are used, providing over one quarter million characters of storage. If desired the design is easily modified to work with 1 megabit DRAMs. Although written with the 87C451 in mind, this design is applicable to the 80C451 and 83C451.

Design Objectives

The objectives kept in mind during the design of this device were: provide a substantial size of buffer, keep the parts count and the power consumption to a minimum, and use readily available components.

A buffer size of 256k bytes was chosen because, although a 64K byte buffer is very easily implemented using the 8051 family's 64K external data storage capabilities, it is a little too small for today's printing applications that print a page of text in graphics mode, using up twenty times as many bytes as standard printing mode. Presenting a method for controlling 256K DRAMs shows off the I/O capabilities of the 87C451, and it is very easy to add the extra address line for one megabit devices if a larger buffer is needed.

The SC8XC451 Microcontroller

The 8XC451 is an 8-bit microcontroller based on the familiar 8051 family of devices. In fact, it is an 80C51 with three added ports: P4, P5 and P6. Ports 4 and 5 give 12 (16 in PLCC) additional quasi-bidirectional I/O lines. Port 6 provides another 8 bits of I/O, plus 4 handshake lines that can be programmed to operate in several useful modes for interfacing. The 8XC451 comes in three versions; ROM-less 80C451, 83C451 with 4K x 8 ROM, and 87C451 with 4K x 8 EPROM.

In this note, port 6 is used in the I/O mode as a Centronics compatible printer output port. Additionally, the /IDS and BFLAG pins normally associated with port 6 are used as part of the input port logic. For a complete discussion of port 6 operating modes and programming, see the Signetics application note AN-408 titled "SC83C451 Microcontroller – Operation of Port 6".

Circuit Description

Figure 1 is a schematic diagram of the printer buffer circuit. Other than the 87C451 (U1), and the eight 256K DRAMs (U5–U12), only two 74LS244 buffers (U2, U3) and a 74HCT374 (U4) octal flip flop are needed. The U2 and U3 buffers are included to provide full drive capability for the output port and some of the handshake signals on the input port, as the output buffers on the 87C451 can only drive 3 LSTTL loads. U4 has 8-bit data strobed into it by the /STB pulse of the input port.

As the code size for this application is quite small (less than 1K bytes), the on chip instruction memory is quite sufficient for program storage. For a production version, the 87C451 could be replaced with the Signetics 83C451 with a 4K X 8 masked ROM on chip. Note that Port 0 and Port 1 are not used in the present design, thus the 80C451 may be used in this application with the addition of an external address latch and E-PROM.

The /RAS, /CAS, and /WR signals for the DRAM array are provided by port 3 bits /WR, /RD, and T1. Note that as in the 80C51, all port 3 signals are mltifunctional. That is, each can be treated a regular quasi-bidirectional port bit, or as having the special function indicated by its name. This feature is an advantage when using /WR and /RD as /RAS and /CAS control signals for a DRAM array. Treated as a normal port bit, the /WR pin is cleared and set by individual CLR and SETB instructions for a normal length RAM read or write cycle. However, when performing a refresh cycle, /RAS (port 3 /WR) can be pulsed low using a dummy MOVX @R0,A (move to external data memory) instruction. This allows DRAM refresh to be done much more quickly than would otherwise be possible.

Port 1 and one bit from port 4 form the 9-bit address required when addressing the DRAM array. The data inputs to the array come from the parallel input data lines which are latched by U4. The RAM data outputs are fed to port 5. By making the data outputs available to the processor, it is possible to add some additional features to the firmware, such as control codes for printing multiple copies of a document, data compression, data conversion, etc. which are not implemented in this design.

Port 6 Operation

The /IDS (input Data Strobe) and BFLAG pins are normally used in conjunction with the port 6 bidirectional mode. In this mode, the /IDS pin is used to strobe data into the port 6 input latches, and BFLAG is used as flag output. In this application however, these two bits are used to good effect as part of the (separate) input port logic. When a byte of data is strobed into U4 by the printer port of the host computer, the /STB signal connected to /IDS sets the input buffer full flag (IBF). BFLAG is programmed to mirror the contents of IBF, and therefore becomes asserted. This makes it ideal to be used as the BUSY output for the input port. After the input port data has been read and stored in the RAM buffer, BFLAG is de-asserted by performing a dummy read of port 6, which clears IBF. To complete the input port logic, one of the port 3 pins, P3.4 is used as the acknowledge signal, and is asserted/ de-asserted by software. The /ODS pin is tied to ground to permanently enable the port 6 output drivers. This does not cause difficulty as no data is being input into the port.

Note that programming port 6 to operate in the bidirectional mode as described above means the loss of /ODS as an acknowledge input. The acknowledge input is normally used to clear the OBF

(Output Buffer Full) flag, indicating that the printer is ready for another character. On the other hand, operating port 6 in the "output only" mode causes the loss of BFLAG as BUSY output. Because the input port requires an instant BUSY indication while the output port only needs to remember the occurrence of an acknowledge pulse, it makes sense to program port 6 to operate in the bidirectional mode, with /ODS grounded to enable the output drivers. The /INT1 pin can be used instead of /ODS to record the occurrence of an acknowledge pulse with the interrupt system.

Priority and Execution of Tasks

There are three tasks that must be performed in this system: Receive – servicing the input port and storing the input character, Transmit – sending stored characters to the output port as required, and Refresh – performing DRAM refresh. The timers and interrupt system are used to manage the execution and priority of these tasks. Figure 2 illustrates the flow charts for each task. Figure 3 is a listing of the firmware code, which breaks up into sections performing these three functions, as well as an initialization section.

The 51C256 DRAMs require a 256 row refresh every 4 milliseconds. Rather than do an entire refresh cycle every 4 milliseconds, it is done as 64 rows every millisecond. This leaves time for other tasks to get service "slices" more frequently. As DRAM refresh is obviously the highest priority, timer 0 is used as the refresh interval timer, and is programmed to the 16-bit mode, and set to the higher priority level in the Interrupt Priority (IP) register. The refresh code is written in-line rather than in a loop to maximize speed.

An interesting point to note is that when there are no characters stored, the DRAM does not need to be refreshed. If power consumption is of concern, the 87C451 could be programmed to go into idle mode whenever the buffer were empty. A character strobed into the input port would cause an interrupt, restarting the 87C451; DRAM refresh would be maintained until the buffer was once again empty.

The next highest priority should be input port service, as the reason for having a printer buffer is to get the data out of the computer as quickly as possible. Therefore, the input port /STB signal is connected to the /INT0 pin (as well as U4's clock pin and /IDS). Interrupt 0 is programmed in the Interrupt Priority register to be at the lower interrupt level so it cannot prevent refresh service. The interrupt 0 service routine stores the input character at the next location in the DRAM array, using the technique of a circular FIFO buffer. The routine also sends back an acknowledge pulse by clearing and setting the P3.4 pin, and then clears the BUSY (BFLAG) pin by performing a dummy read of port 6 (unless this character caused the buffer to be completely full).

During periods of access to the DRAM array by the input and output routines, the global interrupt enable bit (EA) is cleared so that the refresh interrupt does not disturb the contents of ports 1 and 4, or the /RAS, /CAS, and /WR signals.

The printer (output port) service routine runs all the time, except when the CPU is called to service the other conditions, therefore having the lowest priority. If there are characters in the buffer, polling is used to check for output port BUSY status. If the printer is not busy, then the character is sent, and the output port /STB pin (P4.3) is cleared and set. The output port /ACK line is connected to the /INT1 pin, so that the negative going edge of the /ACK signal is recorded as an interrupt pending. A very short INT1 service routine sets a software flag to indicate that the printer acknowledge the last character.

Possible Enhancements

There are a number of features that could be added to this design. As mentioned previously, the microcontroller could be put into the idle mode when the buffer is empty, conserving power.

The software could be enhanced to provide features such as multiple copies off a document, data compression, data conversion, automatic printer setup, etc. The PC operating system could be suitably modified to send a header for each file to be printed, containing these parameters. There is plenty of room for operating firmware expansion, and plenty of horsepower left in the 87C451 to handle these features.

The two serial port pins RxD and TxD were deliberately left unused so that input and/or output ports are easily implemented for serial interfaces or printers using the built-in UART. The pins used for parallel port handshaking could then be used as serial handshaking lines, providing the standard "modem" signals.

Combining the above two features, this circuit could act as a "splitter". By connecting a daisy-wheel printer to the serial port, a dot-matrix printer to the parallel port, and sending an "address" flag in the file header, simultaneous letter-quality and draft printing could be done.

The size of the DRAM array is easily expanded to one megabyte or larger devices by connecting the additional address pins to port 4 bits 1 and 2. Only slight modifications to the operating firmware would be required.

Conclusion

The Signetics SC8XC451 microcontrollers provide plenty of I/O pins that previously had to be implemented by clumsy I/O expansion methods. The flexibility of port 6 means that this device can be used in a wide variety of applications requiring special port functions, while still using the industry standard 8051 instruction set.

This App Note, describing a typical parallel printer buffer makes full use of the 8XC451 features, yet allows room for enhancement and expansion.

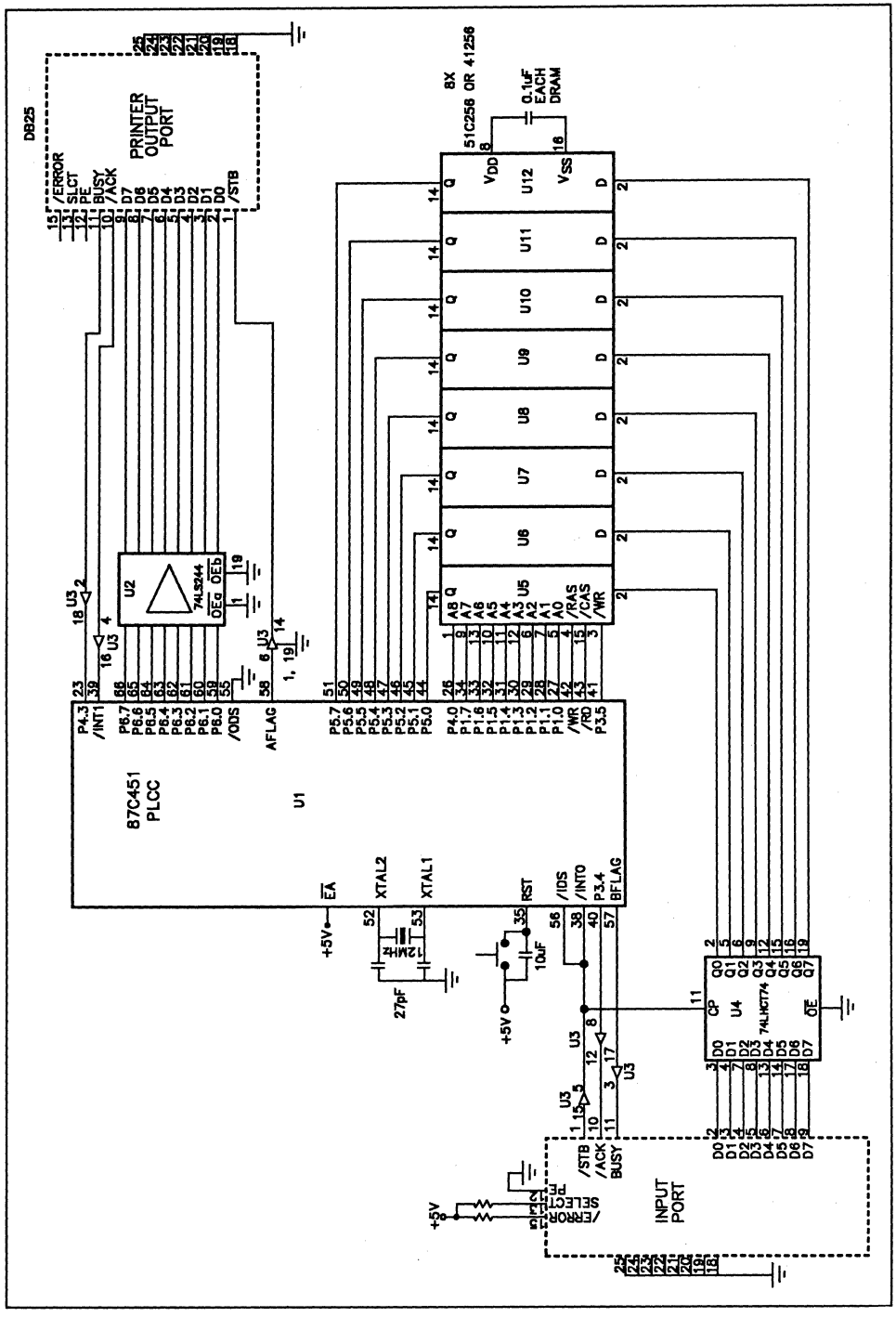


Figure 1. Schematic Diagram

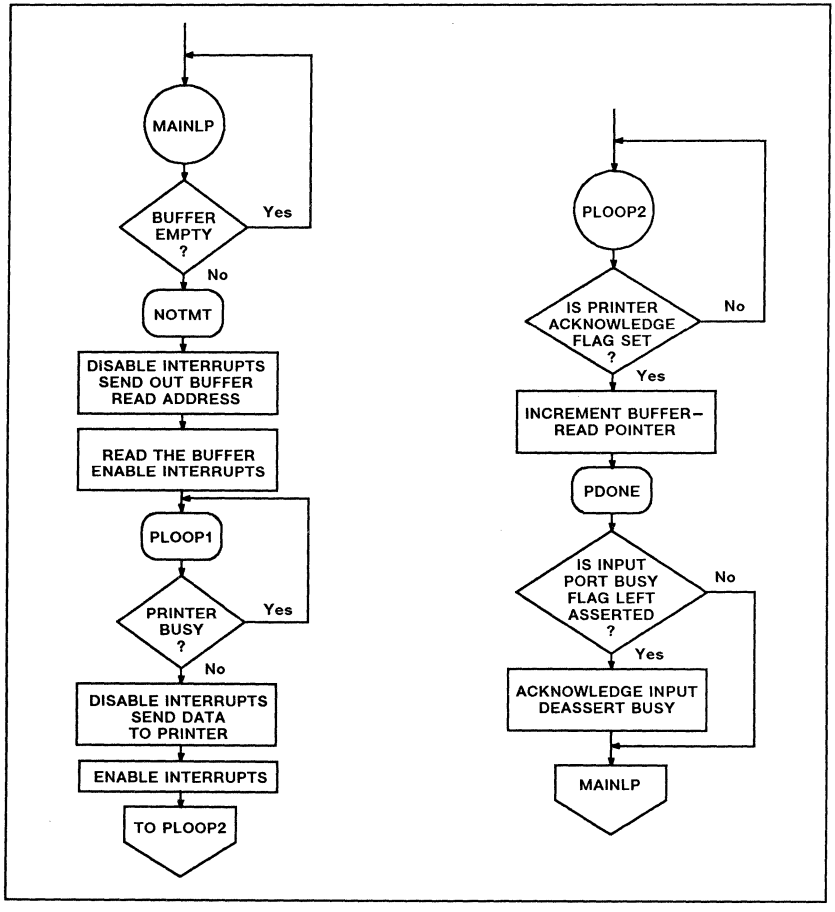


Figure 2. Flowchart of Transmit Operation

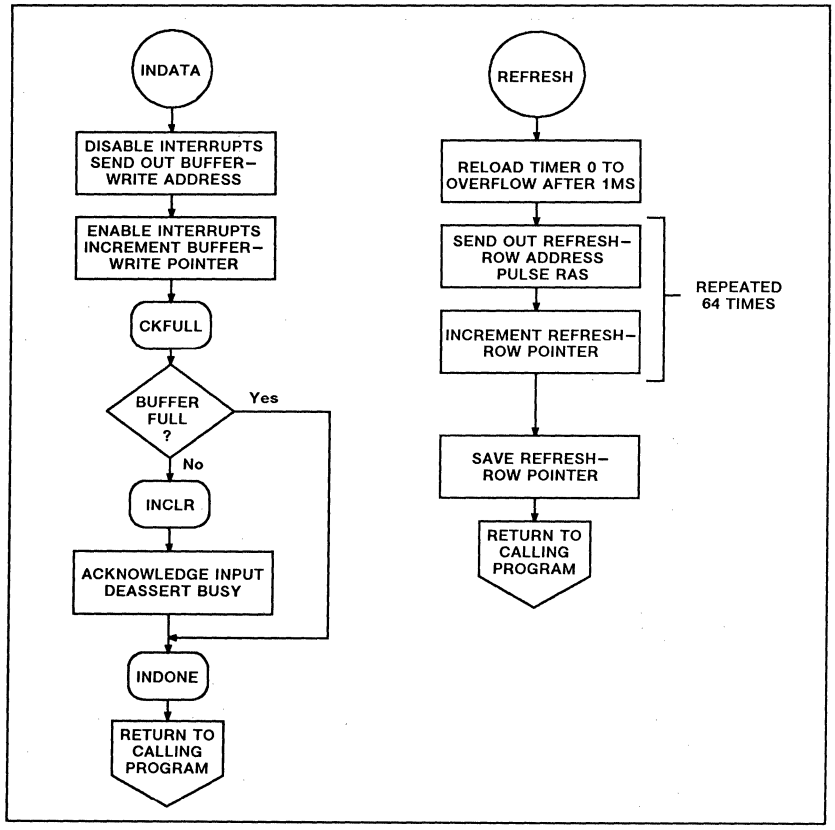


Figure 2. (Cont) Flowchart of Receive and Refresh Operation

```
;
;*******************************************************;
;                                                       ;
;     256K PRINTER BUFFER PROGRAM USING THE 8xC451      ;
;        FOR CENTRONICS PARALLEL PRINTER PORTS          ;
;                                                       ;
;                SIGNETICS  CORPORATION                 ;
;                    October, 1988                      ;
;                                                       ;
;*******************************************************;
;
$Mod451
$Title(8XC451 Printer Buffer)
$Date(10/28/88)
;
; PORT USAGE:
;
; P0            Not used (reserved for data/address bus when external
;                 program memory is used).
; P1            Lower 8 bits of DRAM address (A0 - A7).
; P2            Not used (reserved for high-order address bus when external
;                 program memory is used).
;
; P3.0          (Reserved for serial port.)
; P3.1          (Reserved for serial port.)
; P3.2 (/INT0)  Input port strobe input (interrupt).
; P3.3 (/INT1)  Output port acknowledge input (interrupt).
; P3.4          Input port acknowledge output.
; P3.5          DRAM write enable output.
; P3.6 (/WR)    DRAM row address select output.
; P3.7 (/RD)    DRAM column address select output.
;
; P4.0          Upper bit of DRAM address (A8).
; P4.1          Reserved as an extra address line for 1 Megabit DRAMS.
; P4.2          Not used.
; P4.3          Output port busy input (OBUSY).
; P4.4-P4.7     Unused (not available on 64-pin DIP package).
;
; P5            DRAM output data.
;
; P6            Parallel output port.
; /IDS          Input port strobe input (ISTB).
; BFLAG         Input port busy output (IBUSY).
;
; AFLAG         Output port strobe output (OSTB).
; /ODS          Port 6 output enable, tied low.
;
;
; Internal Register/RAM Usage:
;
REFCNT  EQU     020h          ; Low order refresh byte.
;
; The following refer to the circular FIFO buffer
; implemented in the DRAM array.
;
INLOW   EQU     22h           ; Incoming address low byte.
INMID   EQU     23h           ; Incoming address mid byte.
INHI    EQU     24h           ; Incoming address high byte.
OUTLOW  EQU     25h           ; Outgoing address low byte.
OUTMID  EQU     26h           ; Outgoing address mid byte.
OUTHI   EQU     27h           ; Outgoing address high byte.
```

```
;
OACK    EQU     28h             ; Holds flag for output port acknowledge.
FOACK   BIT     OACK.0          ; Bit-address of output port acknowledge flag.
;
; Miscellaneous Equates:
;
TIME    EQU     -1000           ; Value for 1000 timer clocks = 1 millisecond.
TIMEHI  EQU     HIGH TIME       ; High byte of timer value.
TIMELO  EQU     LOW TIME        ; Low byte of timer value.
RAS     BIT     P3.6            ; DRAM column address select.
CAS     BIT     P3.7            ; DRAM row address select.
DRAMWR  BIT     P3.5            ; DRAM write control line.
IACK    BIT     P3.4            ; Input port ACK output.
ISTB    BIT     P3.2            ; Input port strobe line (INT0).
OBUSY   BIT     P4.3            ; Output port BUSY input.
OSTB    BIT     MA0             ; Output port strobe (MA0 bit in port 6 CSR).
;
;*******************************************************************************
;
; Reset and Interrupt Jump Table
;
        ORG     00h             ; Power-on reset.
        AJMP    START
;
        ORG     03h             ; INT 0.
        AJMP    INDATA          ; Data at input port.
;
        ORG     0Bh             ; Timer 0.
        AJMP    REFRESH         ; Refresh DRAM array.
;
        ORG     13h             ; INT 1.
        AJMP    OPACK           ; Output port acknowledge.
;*******************************************************************************
;
; Power up reset routine:
; Set up refresh timer, enable timer interrupt and
; external interrupt, initialize circular buffer pointers.
;
        ORG     18h
START:  MOV     SP,#40h         ; Initialize stack pointer.
        MOV     A,#00
        MOV     REFCNT,A        ; Initialize refresh counter.
        MOV     INLOW,A         ; Initialize FIFO pointers.
        MOV     INMID,A
        MOV     INHI,A
        MOV     OUTLOW,A
        MOV     OUTMID,A
        MOV     OUTHI,A
```

```
;Initialize interrupt priority register so that DRAM refresh
; (TF0) gets high priority, input port service (IE0) and output
; port acknowledge service get lower priority.  All other
; interrupts set to lower priority level.
;
        MOV     IE,#00000111b   ; Timer0, INT0, and INT1 enabled.
        MOV     IP,#00000010b   ; Timer0 high priority.
        MOV     TL0,#TIMELO
        MOV     TH0,#TIMEHI
        MOV     TMOD,#00000001b ; Operate Timer0 in mode 1.
        MOV     TCON,#00010101b ; Timer0 run, I0 and I1 = edge.
;
;
; Initialize Port 6 Control and Status Register.
;   - 'BFLAG' mode set to output value of IBF
;      (input port BUSY signal : IBUSY)
;   - 'AFLAG' set as logic 1 output
;      (output port strobe signal : OSTB)
;   - 'IDS' active on negative level
;      (input port strobe signal : ISTB)
;
        MOV     CSR,#10011100b
        MOV     A,P6            ; Dummy read of P6 to clear IBF (IBUSY).
        SETB    EA              ; Enable interrupts.
;
;*****************************************************************************
;
; Main Routine:
; Executes while not performing DRAM refresh or servicing
; input port interrupt.
;
; Check if buffer is not empty by comparing input and output
; pointers. If not empty, go to NOTMT to output a byte.
;
MAINLP: MOV     A,INLOW         ; Compare pointers.
        CJNE    A,OUTLOW,NOTMT
        MOV     A,INMID
        CJNE    A,OUTMID,NOTMT
        MOV     A,INHI
        CJNE    A,OUTHI,NOTMT
        SJMP    MAINLP
;
; Buffer is not empty: compute row & column addresses for
; a read cycle from DRAM.
;
NOTMT:  MOV     R4,OUTLOW       ; Save low byte of row.
        MOV     R5,OUTMID       ; Save upper bit of row.
        MOV     A,OUTHI         ; Shift to align correctly.
        RRC     A
        MOV     R7,A            ; Save upper column bit.
        MOV     A,OUTMID        ; Get low byte of column.
        RRC     A               ; Shift in bit from OUTHI.
        MOV     R6,A            ; Save.
;
; Now do actual DRAM access to get the data byte at computed
; address. Disable interrupts so we don't lose what we put
; out on the ports.
;
```

```
          CLR     EA              ; Disable interrupts.
          MOV     P1,R4           ; Low byte row address.
          MOV     A,R5            ; Get high byte row address.
          ORL     A,#0FEh         ; Make sure OBUSY stays high.
          MOV     P4,A
          CLR     RAS             ; /RAS low.
          MOV     P1,R6           ; Low byte column address.
          MOV     A,R7            ; High byte column address.
          ORL     A,#0FEh         ; Make sure OBUSY stays high.
          MOV     P4,A
          CLR     CAS             ; /CAS low.
          MOV     R4,P5           ; Get the data byte
;
          SETB    CAS             ; /CAS high.
          SETB    RAS             ; /RAS high.
          CLR     FOACK           ; Clear acknowledge flag.
          SETB    EA              ; Re-enable interrupts.
;
PLOOP1:   JB      OBUSY,PLOOP1    ; Loop if printer busy.
;
          CLR     EA              ; Disable interrupts.
          MOV     P6,R4           ; Move byte to output port.
          CLR     MA0             ; Assert output port strobe.
          NOP                     ; Kill some time.
          NOP
          NOP
          SETB    MA0             ; De-assert output port strobe.
          SETB    EA              ; Re-enable interrupts.
;
; Following waits for /ACK to occur on output port. Loops on
; acknowledge flag which is set by INT1 service routine when
; /ACK occurs.
;
PLOOP2: JNB       FOACK,PLOOP2    ; Wait till /ACK occurs.
;
          INC     OUTLOW          ; Increment output buffer pointer.
          MOV     A,OUTLOW
          CJNE    A,#00,PDONE
          INC     OUTMID
          MOV     A,OUTMID
          CJNE    A,#00,PDONE
          MOV     A,OUTHI
          INC     A
          ANL     A,#03h          ; Eliminate unused address bits
          MOV     OUTHI,A         ; and save.
;
; Check if input port busy flag was left asserted, indicating that
; the buffer was full after last input. If so, acknowledge input
; port and de-assert input busy signal.
;
PDONE:    JNB     IBF,MAINLP      ; Not busy, return to main loop.
          CLR     EA              ; Disable interrupts.
          CLR     IACK            ; Assert /IACK.
          NOP                     ; Wait 7 microseconds.
          NOP
          NOP
          NOP
          NOP
```

```
        NOP
        NOP
        MOV     A,P6            ; Dummy read of P6 clears IBF (IBUSY).
        NOP                     ; Wait 5 microseconds.
        NOP
        NOP
        NOP
        NOP
        SETB    IACK            ; De-assert /IACK.
        SETB    EA              ; Re-enable interrupts.
        AJMP    MAINLP          ; Return to main loop.
;
;*****************************************************************************
;
; Interrupt 1 Service Routine:
;
;   - Called when output port asserts /ACK.
;   - Sets FOACK flag and returns.
;
OPACK:  SETB    FOACK
        RETI
;
;*****************************************************************************
;
; DRAM Refresh (Timer0) Interrupt Service:
;
;   - Called once every millisecond by timer interrupt.
;   - Refreshes 64 rows and then returns.
;   - Therefore refreshes all rows every 4 milliseconds.
;     (Note that 41256/51C256 DRAM only requires a 256 row refresh.)
;
REFRESH: PUSH   PSW
         MOV    TH0,#TIMEHI     ; Reload timer registers.
         MOV    TL0,#TIMELO

         MOV    P1,REFCNT       ; Get next row to refresh.
         MOVX   @R0,A           ; Pulse /RAS (/WR).
         INC    P1
         MOVX   @R0,A           ; 1
         INC    P1
         MOVX   @R0,A           ; 2
         INC    P1
         MOVX   @R0,A           ; 3
         INC    P1
         MOVX   @R0,A           ; 4
         INC    P1
         MOVX   @R0,A           ; 5
         INC    P1
         MOVX   @R0,A           ; 6
         INC    P1
         MOVX   @R0,A           ; 7
         INC    P1
         MOVX   @R0,A           ; 8
         INC    P1
         MOVX   @R0,A           ; 9
         INC    P1
         MOVX   @R0,A           ; 10
         INC    P1
         MOVX   @R0,A           ; 11
         INC    P1
```

```
MOVX    @R0,A          ; 12
INC     P1
MOVX    @R0,A          ; 13
INC     P1
MOVX    @R0,A          ; 14
INC     P1
MOVX    @R0,A          ; 15
INC     P1
MOVX    @R0,A          ; 16
INC     P1
MOVX    @R0,A          ; 17
INC     P1
MOVX    @R0,A          ; 18
INC     P1
MOVX    @R0,A          ; 19
INC     P1
MOVX    @R0,A          ; 20
INC     P1
MOVX    @R0,A          ; 21
INC     P1
MOVX    @R0,A          ; 22
INC     P1
MOVX    @R0,A          ; 23
INC     P1
MOVX    @R0,A          ; 24
INC     P1
MOVX    @R0,A          ; 25
INC     P1
MOVX    @R0,A          ; 26
INC     P1
MOVX    @R0,A          ; 27
INC     P1
MOVX    @R0,A          ; 28
INC     P1
MOVX    @R0,A          ; 29
INC     P1
MOVX    @R0,A          ; 30
INC     P1
MOVX    @R0,A          ; 31
INC     P1
MOVX    @R0,A          ; 32
INC     P1
MOVX    @R0,A          ; 33
INC     P1
MOVX    @R0,A          ; 34
INC     P1
MOVX    @R0,A          ; 35
INC     P1
MOVX    @R0,A          ; 36
INC     P1
MOVX    @R0,A          ; 37
INC     P1
MOVX    @R0,A          ; 38
INC     P1
MOVX    @R0,A          ; 39
INC     P1
MOVX    @R0,A          ; 40
INC     P1
MOVX    @R0,A          ; 41
INC     P1
```

```
        MOVX    @R0,A           ; 42
        INC     P1
        MOVX    @R0,A           ; 43
        INC     P1
        MOVX    @R0,A           ; 44
        INC     P1
        MOVX    @R0,A           ; 45
        INC     P1
        MOVX    @R0,A           ; 46
        INC     P1
        MOVX    @R0,A           ; 47
        INC     P1
        MOVX    @R0,A           ; 48
        INC     P1
        MOVX    @R0,A           ; 49
        INC     P1
        MOVX    @R0,A           ; 50
        INC     P1
        MOVX    @R0,A           ; 51
        INC     P1
        MOVX    @R0,A           ; 52
        INC     P1
        MOVX    @R0,A           ; 53
        INC     P1
        MOVX    @R0,A           ; 54
        INC     P1
        MOVX    @R0,A           ; 55
        INC     P1
        MOVX    @R0,A           ; 56
        INC     P1
        MOVX    @R0,A           ; 57
        INC     P1
        MOVX    @R0,A           ; 58
        INC     P1
        MOVX    @R0,A           ; 59
        INC     P1
        MOVX    @R0,A           ; 60
        INC     P1
        MOVX    @R0,A           ; 61
        INC     P1
        MOVX    @R0,A           ; 62
        INC     P1
        MOVX    @R0,A           ; 63

        INC     P1              ; Adjust for next time
        MOV     REFCNT,P1       ; and save.
        POP     PSW
        RETI
;
;*******************************************************************************
;
; Data at Input Port:
;
; This routine is called via interrupt INT0 whenever data
; is strobed into the input port. It saves the data into the
; DRAM array and increments the input pointer. If the output
; pointer is now equal to the input pointer, then the buffer
; is full, and we leave the busy flag set so that no more
; data can be input until some is output and the buffer is
; no longer full.
```

```
;
;
INDATA: PUSH    PSW
        PUSH    ACC
        MOV     R1,INLOW        ; Lower 8 bits of row to R1.
        MOV     R2,INMID        ; Upper bit of row to R2.
        MOV     A,INHI          ; Get upper 2 bits.
        RRC     A               ; LSB to carry.
        MOV     R0,A
        MOV     A,INMID
        RRC     A               ; Shift bit into MSB.
        MOV     R3,A            ; Save.
;
        CLR     EA              ; Disable interrupts.
        MOV     P1,R1           ; LSB row address.
        MOV     A,R2            ; MSB row address.
        ORL     A,#0FEh         ; Make sure OBUSY stays high.
        MOV     P4,A            ; MSB row address.
STBLP:  JNB     ISTB,STBLP      ; Check for end of strobe before DRAM write.
        CLR     RAS             ; /RAS low.
        CLR     DRAMWR          ; /WR low.
        MOV     P1,R3           ; LSB column address.
        MOV     A,R0            ; MSB column address.
        ORL     A,#0FEh         ; Make sure OBUSY stays high.
        MOV     P4,A            ; MSB column address.
        MOVX    A,@R0           ; Pulse /CAS low.
        SETB    RAS             ; /RAS high.
        SETB    DRAMWR          ; /WR high.
        SETB    EA              ; Re-enable interrupts.
;
        INC     INLOW           ; Increment input buffer pointer.
        MOV     A,INLOW
        CJNE    A,#00,CKFULL
        INC     INMID
        MOV     A,INMID
        CJNE    A,#00,CKFULL
        MOV     A,INHI
        INC     A
        ANL     A,#03h          ; Eliminate unused address bits.
        MOV     INHI,A
;
; Compare input pointer to output pointer to see if the buffer is full.
;
CKFULL: MOV     A,INLOW
        CJNE    A,OUTLOW,INCLR
        MOV     A,INMID
        CJNE    A,OUTMID,INCLR
        MOV     A,INHI
        CJNE    A,OUTHI,INCLR
;
; If we get here, the buffer is full, so skip the acknowledge pulse.
;
        SJMP    INDONE
;
;
; Send acknowledge pulse on /IACK line for 7 microseconds,
; de-assert input BUSY signal halfway through.
;
INCLR:  CLR     EA              ; Disable interrupts.
        CLR     IACK            ; Assert /IACK.
```

```
        NOP                 ; Wait 7 microseconds.
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOV     A,P6        ; Dummy read of P6 clears IBF (IBUSY).
        NOP                 ; Wait 5 microseconds before clearing /IACK.
INDONE: POP     ACC
        POP     PSW
        SETB    IACK        ; De-assert /IACK.
        SETB    EA          ; Re-enable interrupts.
        RETI
;
        END
```

Signetics

AN418 Counter/Timer 2 of the 83C552

Application Note

Microprocessor Division

Introduction to the 83C552

The 83C552 is an 80C51 derivative with several extended features: 8k ROM, 256 bytes RAM, 10-bit A/D converter, two PWM channels, two serial I/O channels, six 8-bit I/O ports, and four counter timers. The architecture of the 83C552 is identical to that of the 80C51 making the two devices fully code compatible. The additional peripheral functions are added to the 80C51 Special Function Register space and the interrupt structure is modified accordingly. This information is detailed in other references on the 83C552. The focus of this application note is on one of the timers of the 83C552, Counter Timer 2.

This counter timer includes capture, compare and high speed output capabilities which facilitate many control oriented tasks. The objective of this note is to make users of the 83C552 aware of this counter timer subsystem and assist the use of this subsystem by a detailed explanation of its operation supported by actual application examples.

Timer 2 of the 83C552

Timer 2 of the 83C552 is in fact a timing controller and has an associated programmable array. The Timer 2 subsystem consists of three parts:

1. the time base consists of a 16-bit timer with a 3-bit prescalar. The master clock for the subsystem can be derived from the on-chip oscillator (fosc) or an external input, T2. It has an external reset, RT2, by which a signal applied to this input can reset the timer if the external reset is enabled.
2. a capture system consisting of four capture registers and four capture inputs which can be used for a wide variety of time measurements on external signals.
3. a compare system consisting of three compare registers and eight associated high-speed outputs which can be activated upon a match between the 16-bit timer and one of the compare registers.

For reference a complete block diagram of the 83C552 Counter Timer 2 subsystem is shown in Figure 1.

16-Bit Counter Timer

The description of Counter Timer 2 in the following paragraphs is intended to be a general overview. Details on architecture, address locations, interrupt structure and timer operation are given in the 83C552 Users Manual. This users manual may be useful to complement the material presented in this application note. References to registers, bits, I/O ports and on-chip hardware will relate directly to 83C552 Users Manual nomenclature. This application note will focus on the use of Counter Timer 2 as a powerful input capture and high speed output facilitator through some specific examples and not on the detailed coding.

The counter timer consists of a 16-bit counter which is readable by software through special function registers TM2L and TM2H. The timer itself has two overflow flags, one after the entire 16-bit counter and one attached to the eighth stage. This latter flag reflects an overflow from the first byte of the counter. These two flags are present in register TM2IR and are labeled T2BO for the overflow from the first byte and T2OV for the overflow from the entire 16-bits. These flags may be used to generate an interrupt.

The counter timer is controlled directly through the special function register TM2CON, the timer 2 control register. This register also contains certain status flags.

The prescaler divides the input clock by a progammable ratio. The prescaler divide value is progammable to divide by 1, 2, 4, or 8 as controlled by T2P0 and T2P1 in TM2CON.

The input clock to the prescalar is either fosc / 12 or the external input, T2. The clock input to the prescaler may also be shut off. This clock input selection is controlled by bits T2MS0 and T2MS1 in TM2CON.

If T2 is used as the input clock to the timer 2 subsystem, the hardware logic samples this input and looks for a low to high transition. If the logic detects a logic 0 at the T2 input in state S2P1 of the microcontroller and a logic 1 in state S5P1, then this is recognized as a low to high transition and the prescaler is incremented. The prescaler is incremented in the second cycle after the cycle in which the transition was detected. If the transition is detected before S2P1 is finished, the prescalar is incremented in the next cycle. This timing is shown in Figure 2. Note that this sampling rate is twice that of the normal 80C51 timers, T0 and T1, therefore T2 has twice the maximum external counting rate as compared to the standard timers.

Any programming of the clock source or the prescalar divide ratio results in a reset of the prescaler. This allows the state of the timer subsystem to be in a known state upon programming. The main 16-bit timer can not be reset by software but it is reset by activating the reset pin or using the external reset, RT2. The external reset, RT2 can be enabled or disabled by bit T2ER in TM2CON. These resets reset the prescalar as well as the 16-bit counter.

Only one interrupt is available from the 16-bit counter timer. Two bits in TM2CON control whether TM2L, TM2H, or both flags will be used to generate the interrupt. A selection for no interrupt is also possible.

Capture System

The capture system is a powerful tool to measure the width of pulses or repetition rates. There are four independent inputs for the signals to be analyzed, CTI0 through CTI3. These inputs are alternate functions to Port 1. Each input is connected to a dedicated capture register. A transition at any of these inputs will cause the content of the 16-bit counter timer to be loaded into the respective capture register. The capture can occur upon various conditions of the input signal as specified by certain bits in the capture control register, CTCON. Each input can be set to cause a capture on a low to high transition, a high to low transition, or on both transitions. Upon a capture taking place, each input causes an interrupt flag to be set in the Timer 2 Interrupt Flag Register, TM2IR. If enabled an interrupt will be generated.

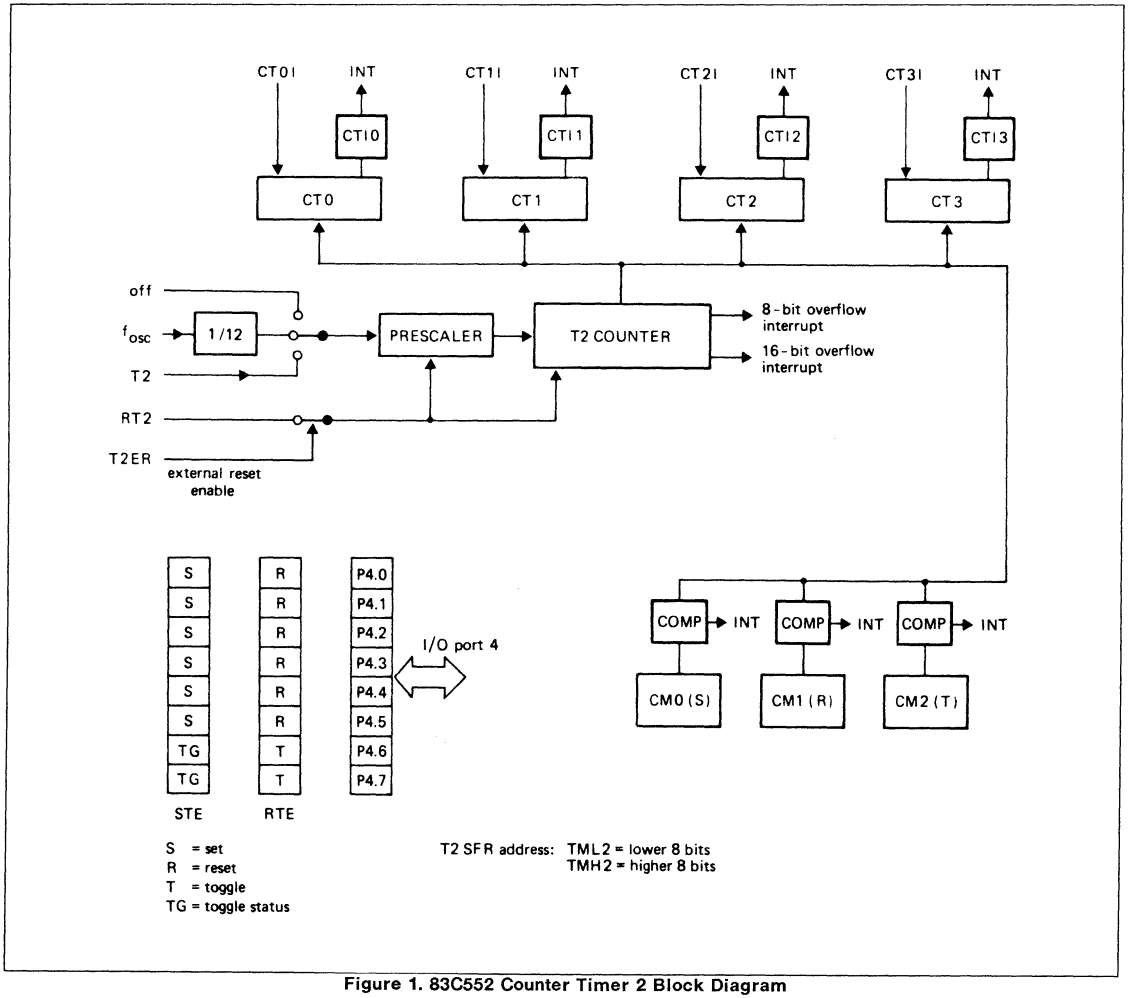


Figure 1. 83C552 Counter Timer 2 Block Diagram

One of the capture inputs is shown in more detail in Figure 3. All of the other capture inputs are similar to this one. The capture input is gated with the capture enable bits CTN0 and CTP0 which are located in CTCON. According to the status of these bits, the desired edges are selected to generate the capture enable pulse. The input pulse transient detection is at the input of the enable pulse generator. The input signal is sampled at S1P1 of the machine cycle. If a logic 1 is detected when a logic 0 was detected at the same time in the previous cycle, then the event is taken as a transition. An enable pulse is sent to the capture register and the contents of timer 2 is copied into the capture register at the end of this machine cycle. The interrupt flag CTI0 is also set.

Compare System

The compare system of Timer 2 can be used to generate a set of outputs whose transitions are controlled directly by the time defined by the 16-bit counter timer. There are eight of these high speed outputs which are directly controlled from Counter Timer 2. These outputs are alternate functions to Port 4. Six of these outputs are set – reset controlled (CMSR0 through CMSR5) and two are toggle controlled (CMT0 and CMT1). To clarify the operation, these two types will be discussed separately. In the following discussions, refer to Figure 4 which shows the compare system for P4.5 (a set – reset high speed output) and P4.6 (a toggle high speed output).

There are two compare registers associated with the set – reset outputs. These registers are CM0 and CM1. In addition there are two enable registers: one to enable setting of an output and the other to enable resetting of an output. These registers are STE and RTE respectively. The content of CM0 and CM1 are continuously compared to the content of the 16-bit counter. Whenever there is a match between the 16-bit counter and the contents of CM0, a SET pulse is generated. Similarly, whenever there is a match between the timer and

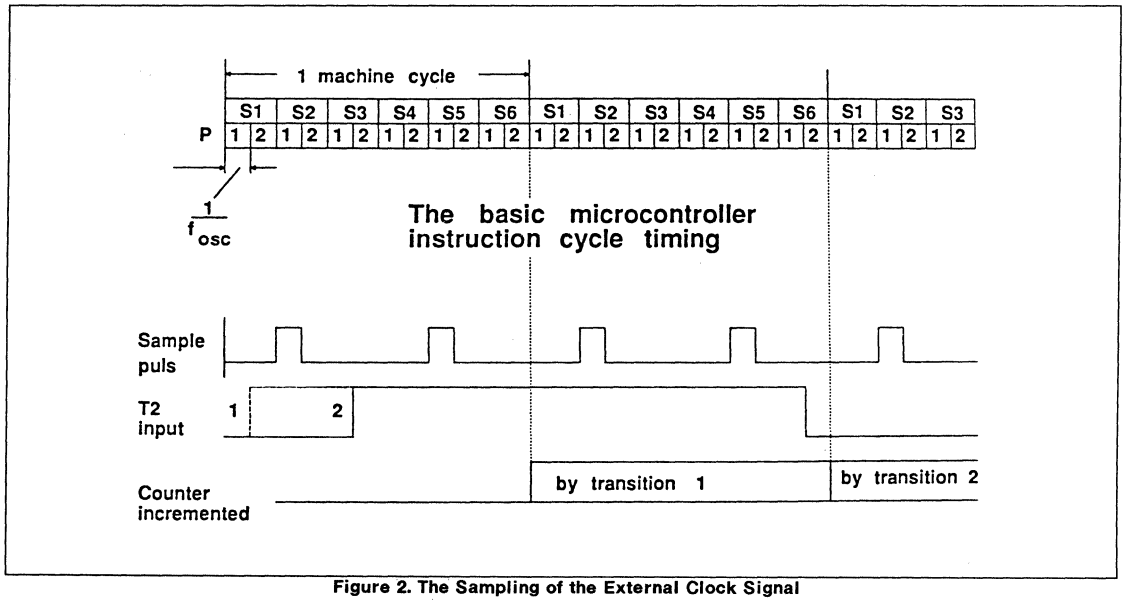


Figure 2. The Sampling of the External Clock Signal

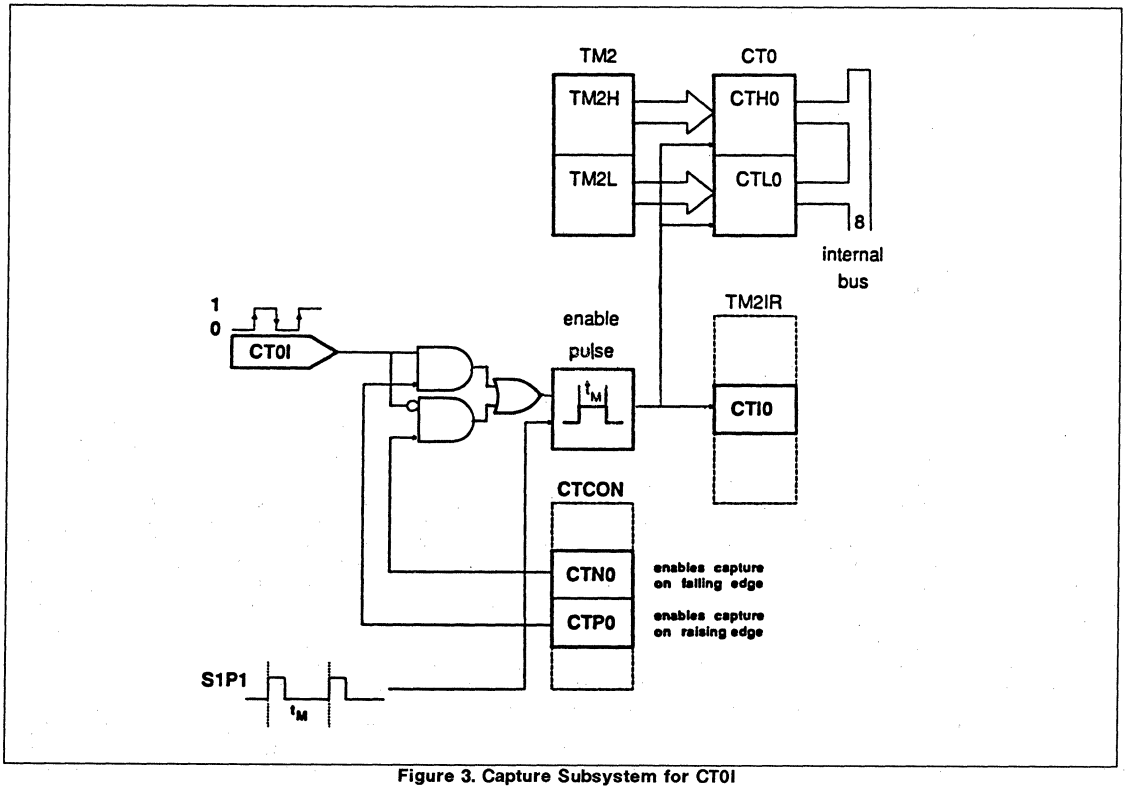


Figure 3. Capture Subsystem for CT0I

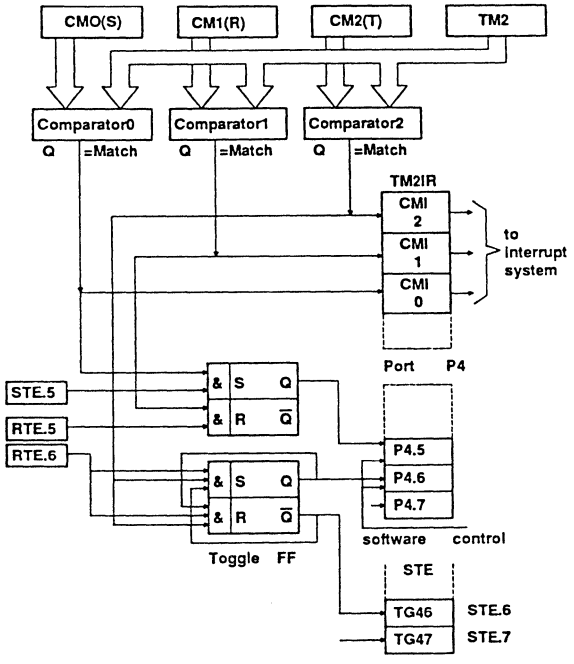


Figure 4. Compare System for P4.5 and P4.6

the contents of CM1, a RESET pulse is generated. The set pulse is applied to the set – reset outputs, CMSR0 through CMSR5, through gates controlled by bits in STE. Bits 0 through 5 in STE control the application of the SET pulse to one of the high speed outputs. For example STE.0 controls the gating of the SET pulse to CMSR0, STE.1 controls the gating of the SET pulse to CMSR1 and so forth. Thus if the corresponding SET enable bit in STE is a 1 and a compare occurs with CM0, then that high speed output will become set. Similarly, the reset pulse from CM1 is applied to the high speed outputs CMSR0 through CMSR5 through gates controlled by bits in RTE. As with STE, bits 0 through 5 in RTE control the application of the reset pulse to one of the high speed outputs. If a compare occurs between the timer and CM1, a high speed output will be reset if its corresponding enable bit in RTE is a 1. Compares with CM0 and CM1 set interrupt flags which, if enabled, can be used to generate an interrupt.

The two toggle controlled outputs are CMT0 and CMT1 and these are associated with compare register CM2. These outputs are also alternate functions on Port 4. Upon a compare between the counter and the contents of CM2, CM2 generates a toggle pulse which is applied to the high speed outputs CMT0 and CMT1 through a set of gates. The gates control the application of the toggle pulse to the toggle outputs as specified by the high order bits of register RTE. RTE.6 controls CMT0 and RTE.7 controls CMT1. Should the corresponding bit of RTE be set, then the toggle pulse is enabled to the associated high speed output and that output will toggle upon generation of the toggle pulse from CM2. The structure of these toggled outputs is different from the other high speed outputs in that the toggling is actually accomplished in a separate toggled flip flop and not directly in the port latch. This toggle flip flop can not be controlled directly by software and powers up in an indeterminate state. The state of the toggle flip flops is readable in STE bits 6 and 7.

Application Example – Timed Fuel Injection

In modern automobiles, optimal combustion is necessary to meet emission standards and improve fuel consumption. Optimal combustion depends on several factors and is enhanced by proper fuel injection based upon these factors which vary according to engine speed and other factors. Thus the task is to control the opening and closing of the engine fuel injectors of each cylinder relative to the crankshaft reference point.

For the application example here, we will not consider the factors which determine the timing relationships. These are assumed to be given quantities. The example here will focus upon the implementation of the injector timing control signals and how they are generated using the Counter Timer 2 system. The illustration considers a four cylinder engine. While this is an automotive application which serves to clearly illustrate Counter Timer 2 Subsystem operation, it is clear that many systems share similar

timing requirements and the techniques employed here are applicable to a wide class of timing tasks. The 83C552 will also support six cylinder engine control.

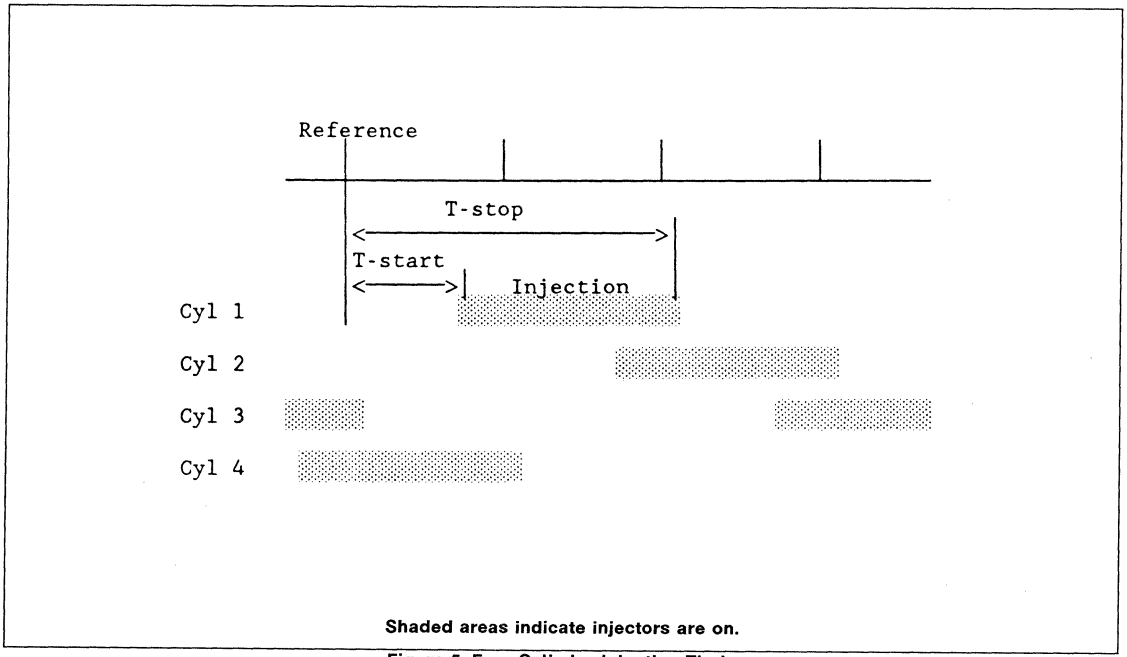
Figure 5 shows the injection timing required for two consecutive revolutions of the engine crankshaft. Start and stop of the injection are given relative to a reference point on the crankshaft. The cylinders are numbered in the order of the injection sequence (not with reference to their physical location). Start of the injection is usually given in angular measure with respect to top dead center and the injection duration is assumed to be a time value calculated from engine environmental factors and operating parameters. The angle for the start of the injection must be converted into time with respect to the reference point.

The injector drivers are assumed to be connected to the Port 4 high speed outputs CMSR0 through CMSR3. To obtain the top dead center reference point, the signal from the appropriate sensor is connected to the capture input CT0I. The interrupt for this capture input is enabled so that software can synchronize its operation to this time reference and make use of the top dead center time in the injector timing calculations. The software synchronization takes two forms. First the captured time is an absolute reference for all real time output operations. This time is available in capture register CT0. Note that at 12 MHz operation, Timer 2 can have a resolution as fine as 1 microsecond with a total time before overflow of over 65 milliseconds and these times are adjustable by increasing the prescalar divide ratio. A proper selection can make the timing calculations relatively simple. Second, at the time the input is captured, flags which keep track of the phases of the crankshaft cycle are reset when cylinder 1 is at top dead center. These flags are used in the interrupt service routines to tell which action is required for that phase of the crankshaft.

Consider now the sequence of events in one rotation of the engine crankshaft and refer to Figure 5 during the discussion. Assume that the engine is running, that all relevant parameters are available and that it has been determined that the processor is responding to the interrupt associated with the top dead center capture, CT0I. Interrupts for CT0I, CM0 (compare register 0), and CM1 (compare register 1) are enabled. Upon entering the interrupt service routine for CT0I, the previous value of the captured top dead center time is subtracted from the present value and the crankshaft rotation time is determined. This is used to compute the time to open the first injector from the required angle at which the injector is to open. This time is made available for the interrupt service routine which responds to a compare from CM0. The interrupt service routine is exited.

The next interrupt to occur per the Figure for this example is a result of a compare with CM1 which will be a result of the injector stop time for cylinder 4 having been reached. The flags in an internal status register are employed to keep track of the cylinder number that is presently active for both injection stop and injection start times. After identifying this interrupt from the flags, the processor uses the injector start time for cylinder 1 (previously loaded into CM0) and the predetermined duration to calculate the injector stop time for cylinder 1. This value is loaded into compare register CM1 and the reset enable bit for high speed output CMSR0 is programmed to a 1. This is bit RTE.0 the reset enable bit for the cylinder 4 injector is set to 0 (bit RTE.3). The interrupt routine is now exited.

The next interrupt to occur will be for the start time for the injector for cylinder 2. This and all subsequent cases follow the same sequence of events as for the cylinder 1 CM0 interrupt described above. In this case calculations are made for cylinder 2 and loaded into CM0,



Shaded areas indicate injectors are on.

Figure 5. Four Cylinder Injection Timing

STE.1 is programmed to a 1 and STE. 0 is programmed to a 0. Similarly, the next interrupt for CM1 is treated in the same way and the sequence of events rotates around through all cylinders in turn. The flag bits associated with this operation keep track of the injector sequencing.

While this example shows the injection stop time of one cylinder overlapping into the injector on time of the subsequent cylinder, close examination of the operations described above reveal that the start and stop events are independent and can overlap or not as required. In this way all injectors may be driven independently and have overlapping on times.

Given that this is an example applicable to general usage, it is possible that interrupt service routine could be relatively long as it would be in an actual injector application. Since the service routine has other interrupts disabled, the length may cause real time conflicts. To eliminate this potential problem, the interrupt service routines are divided into two parts. In the first part, all other interrupts are disabled and the essential register loading is done to prepare for the next interrupt. After this is completed, all interrupts are enabled and the ancillary service routine functions are performed prior to a return to the main routine.

As an example consider the interrupt service routine for CM0. Upon entering the routine, all interrupts are disabled. Then the following actions are performed:

- set bit in STE to start next injector
- clear bit in STE for injector just started
- load CM0 with start time for next injector
- clear CMI0 interrupt flag in TM2IR

Now that the essential set up is made for the next interrupt, all interrupts are now enabled. However the return to the main program is not invoked until the following ancillary processing is completed:

- calculate the next absolute start time for the next injector (the next load value for CM0)
- increment the flag so that the next entry to this interrupt service routine will be able to identify the next injector to start.

The process performing these calculations can be interrupted to service real time functions.

Application Example – Timed Ignition

In electronic ignition systems, multiple ignition coils may be used and each coil is fired by electronic means rather than with the old stile mechanical breakers. In a four cylinder engine, there may be two ingnition coils, one coil providing spark for a pair of cylinders. Both plugs fire at the same time. For one cylinder, the spark occurs at the appropriate time while for the other cylinder, the spark occurs at the end of the exhaust stroke and has no effect. With timing references to crankshaft top dead center provided by an external sensor, the ignition timing for the engine may be generated in the 83C552 and applied to the electronic drivers for the ignition coils.

To illustrate the toggle high speed outputs of the 83C552 Counter Timer 2 subsystem, the following example will discuss the ignition timing in a four cylinder engine employing the two coil approach with one coil for a pair of cylinders. The coil timing is illustrated in Figure 6. A reference time is used which is a given interval prior to top dead center so that the times used in the illustration can be always after the reference. There are two times of interest for each coil: the load time and the ignition point.

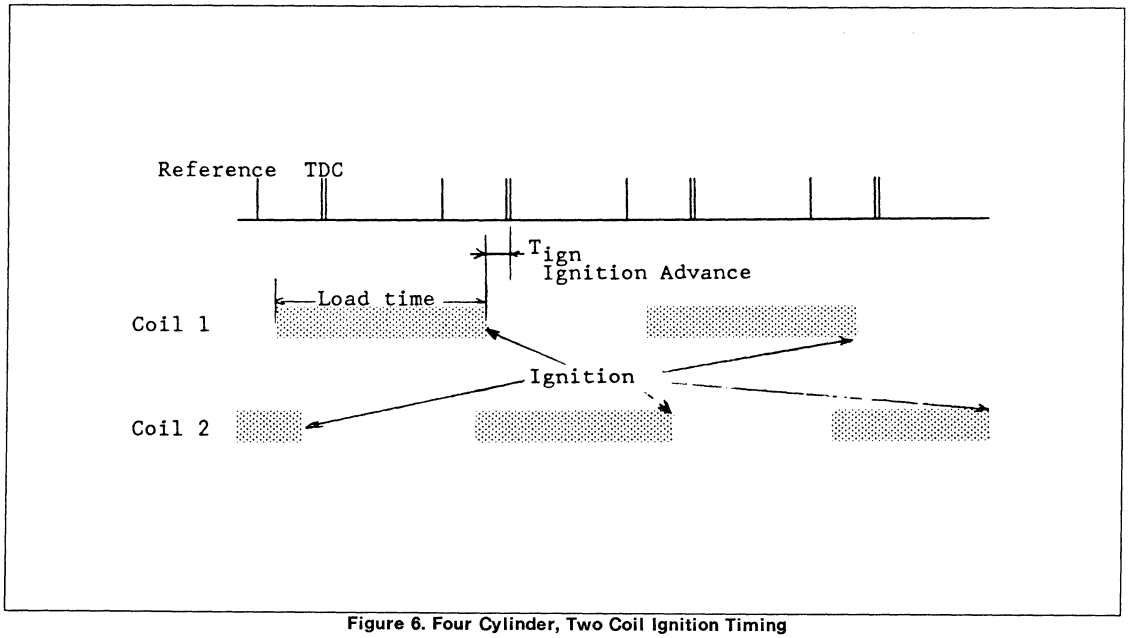


Figure 6. Four Cylinder, Two Coil Ignition Timing

Ignition advance is usually given in degrees crankshaft angle prior to top dead center. As with injection, this angle is assumed to be derived from other calculations and is a given value for this illustration. This angle must be converted into a time with respect to the reference point. The load time (the time at which the coil has to be switched on to reach the current that will give sufficient energy for an adequate spark) must be subtracted from the desired ignition point. At the ignition time, the coil will be switched off and the spark will be generated.

The coil driver electronics are connected to port bits P4.6 and P4.7. Ignition Coil 1 is connected to P4.6 and ignition Coil 2 is connected to P4.7. These outputs are the toggle high speed outputs controlled by the 16-bit compare register, CM2. The program simply needs to set up the compare and control registers to turn the coils on and off at the appropriate times. It is assumed in this example that the ignition and load times are given quantities and have been determined previously.

Consider now the sequence of events in two rotations of the engine crankshaft and refer to Figure 6. Assume that the engine is running, that all relevant parameters are available and that it has been determined that the processor is responding to the interrupt associated with a compare to CM2. The top dead center time and crankshaft rotation speed have been already determined through the top dead center capture, CT0I. This is the same as in the injector example. The interrupt for CT0I is enabled. From the top dead center time, the times to turn on and turn off the coil drivers are computed and made available in data storage locations in the microcontroller. It is also convenient to have flags to identify the step in the complete ignition cycle. The flags are cleared in the interrupt service routine for top dead center of cylinder 1.

Upon entering the interrupt service routine, other interrupts are disabled. Examination of the flags reveals that the state of the ignition sequence is that Coil 1 has been turned on to begin the current build up (load time). The next event will therefore be turning off Coil 2 to cause ignition. The interrupt service routine then performs the following actions: The time to turn off Coil 2 is moved into compare register 2, CM2. Bit 6 of RTE is cleared; this disconnects the output of CM2 from the toggle flip flop of P4.6 (Coil 1). Bit 7 of RTE is set; this connects the output of CM2 to the toggle flip flop of P4.7 (Coil 2). The flags are incremented to indicate that the next interrupt will be a result of Coil 2 turning off and causing ignition. The other interrupts can be enabled and a return to the main program can be executed. After the other interrupts are enabled and before a return is made to the main program, it may be convenient to do any necessary calculations to determine the time value to be loaded into CM2 in the next CM2 interrupt.

Since the flip flops are toggled, it is likely that upon power up of the microcontroller, the toggle flip flops will not be in the desired state. To get the toggle flip flops in the correct state in the ignition cycle, the flip flops must be toggled if they are in the wrong state. To determine if this is necessary, the state of the toggle flip flops can be read from the STE register. The state of the P4.6 flip flop is present in STE bit 6 and the state of the P4.7 flip flop is present in STE bit 7. Comparing the actual state to the required state determines which if any or both of the flip flops must be toggled. If a toggle is necessary to put one or both of the flip flops in the correct state, the corresponding bits in RTE would be set for those flip flops requiring the toggle and CM2 would be loaded with a value that is slightly larger than the present contents of Timer 2. If desired for reliability purposes, the state of the flip flops could be checked periodically against the ignition cycle flags to determine if a correction is necessary.

Conclusion

This application note has examined one aspect of the 83C552 CMOS 80C51 derivative microcontroller. The Counter Timer 2 Subsystem has been applied to a complex timing task of gasoline engine injector valve and ignition coil timing control. While this is a specific application to the automotive interests, the result are applicable to a wide variety of time measurement and control applications. The 83C552 would be ideal for many electromechanical systems such as copy machines, fax machines, industrial process control equipment, automatic transmission control, and anti-skid and anti-lock braking control.

These application areas are those which can successfully employ the 83C552 Counter Timer 2, however the other features should not be overlooked. When combined with the 10-bit A to D Converter, the Pulse Width Modulator, the I2C serial bus and peripheral device family, the 83C552 provides minimum component count solutions for cellular radio systems, professional audio systems, and medical instrumentation products such as bed side patient monitors and analyzers for home care and sports use.

Signetics

AN420
Using up to 5 External Interrupts on 8051 Family Microcontrollers

Application Note

Microprocessor Division

8051 family microcontrollers are equipped with up to two inputs which may be used as general-purpose interrupts. A typical device provides a total of 5 interrupt sources. Timer 0 and Timer 1 generate vectored interrupts, as does the Serial Port. Applications that require more than two externally signaled vectored interrupts, and do not use one or more of the counters or the serial port, can be configured to use these facilities for additional external interrupt inputs.

This note describes a method to configure the timer/counters and the serial port for use as interrupt inputs. Minimum response time is a goal for this configuration.

Another popular method to implement extra interrupt inputs is to poll under software control a port pin configured as an input. This method is necessary when the on-chip peripherals are in use. Applications where this approach is recommended are ones in which the processor spends more than half of the time executing a "wait loop", or a short code sequence which jumps or branches back on itself without performing any functions. In this case, the instructions that will check the state of input used as an interrupt source are inserted into this sequence. Consequently, this input is ignored when other routines are being executed. This input may have to be latched externally, or the processor may miss the signal while executing other routines.

Dedicated interrupt inputs that vector the processor to individual service routines (as the two general purpose interrupt inputs work) do not have the drawbacks of the method described above.

COUNTER/TIMER CONFIGURATION

Timers 0 and 1 are placed in Mode 2, which configures the timer register as an 8-bit counter with automatic reload. The counter and reload register are loaded with FF hexadecimal which is stored in TH1 and TL1 or TH0 and TL0.

To prepare one of the timers for this kind of operation, a number of control bits have to be set up. The following is a list of these bits and their values:

In TMOD:	In TCON:	In IE:
GATE = 0	TRi = 1	ETi = 1
C/T = 1		EA = 1
M1 = 1		
M0 = 0		

Where "i" is the timer number being used as the external interrupt. The TMOD value would be 66 hexadecimal if both timer are being used as external interrupt sources, x6 hex for timer 0, and 6x hex for timer 1. The interrupt priority may also be set in the IP register.

A falling edge on the corresponding Timer0 or Timer1 input (T0 or T1) will cause the counter to overflow and generate a timer interrupt. The counter will be automatically loaded with another FF from the reload register, so the interrupt can occur again as soon as the interrupt service routine completes. Counter/Timer operation is described in detail in elsewhere in this manual.

SERIAL PORT CONFIGURATION

The serial port can be placed in mode 2, which is a 9-bit UART with the baud rate derived from the oscillator. The external interrupt is signaled through this port on the RXD receive data pin. Reception is initiated by a detected 1-to-0 transition at RxD. The signal must stay at 0 for at least five-eighths of a bit period for this level to be recognized. Refer to the description of baud rates to determine the length of a bit period at the oscillator frequency selected for the application. The input signal should remain low for at least one bit period and for not more than 9 bit periods.

To prepare the serial port for use as an external interrupt, the following bits must be set up:

In SCON:
SM0 = 1
SM1 = 0
SM2 = 0
REN = 1

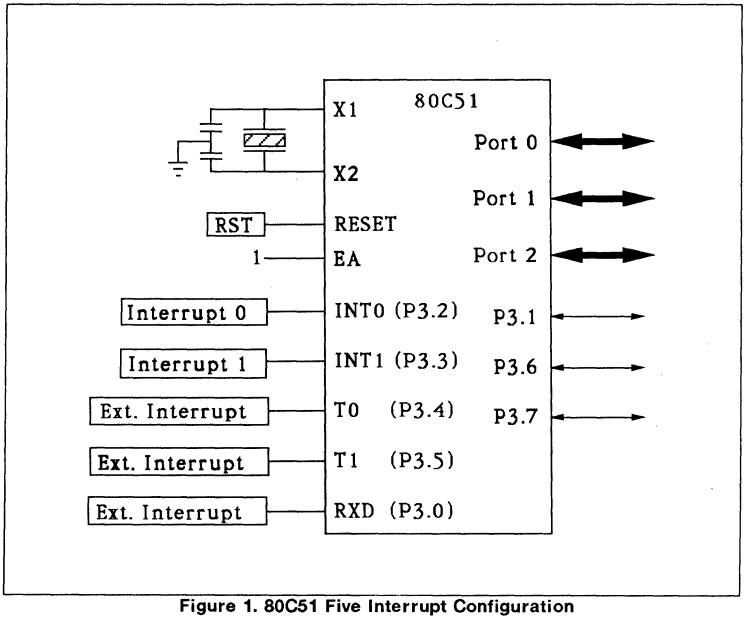


Figure 1. 80C51 Five Interrupt Configuration

The Serial Port Interrupt is then used as a general purpose interrupt. The contents of receive buffer should be ignored, and will subsequently be overwritten during the next interrupt.

Note that the response time for this input will be slower than for the Counter/Timer inputs. This is due to the fact that the RI is generated after the eighth serial data bit time after the falling edge on RxD.

```
; Demonstration program for five external interrupts.

$MOD51
$TITLE (Five Vectored External Interrupts)

;  Interrupt Jump Table
       ORG     0H              ;Reset
       AJMP    Setup

       ORG     3H              ;External interrupt 0.
       RETI                    ;(not implemented in this demo)

       ORG     0BH             ;Timer 0 interrupt.
       AJMP    Tim0

       ORG     13H             ;External interrupt 1.
       RETI                    ;(not implemented in this demo)

       ORG     1BH             ;Timer 1 interrupt.
       AJMP    Tim1

       ORG     23H             ;Serial port interrupt.
       AJMP    Serial

; Begin setup code
Setup  MOV     SP,#7FH         ;Initialize the stack pointer.

; Configure both timers
       MOV     TMOD,#66H       ;Put both counters into mode 2.
       MOV     A,#0FFH
       MOV     TL0,A           ;Load FF hex into both counters
       MOV     TH0,A
       MOV     TL1,A
       MOV     TH1,A
       SETB    ET0             ;Enable Timer 0 interrupt.
       SETB    ET1             ;Enable Timer 1 interrupt.
       SETB    TR0             ;Enable Timer 0 to run.
       SETB    TR1             ;Enable Timer 1 to run.

; Configure the serial port
       SETB    ES              ;Enable serial port interrupt.
       MOV     SCON,#90H       ;Put the serial port in mode 2.
       SETB    EA              ;Enable interrupt system.

Wait:  NOP                     ;Wait for an interrupt.
       JMP     Wait

Serial:NOP                     ;Serial interrupt service routine.
       CLR     RI              ;Clear receiver interrupt flag.
       RETI

Tim0:  NOP                     ;Timer 0 interrupt service routine.
       RETI

Tim1:  NOP                     ;Timer 0 interrupt service routine.
       RETI

       END
```

Signetics

Microprocessor Products

Section 4
Inter–Integrated (I^2C) Circuit Bus

INDEX

I²C Bus Specification

INTRODUCTION

For 8-bit applications, such as those requiring single-chip microcomputers, certain design criteria can be established:

- **A complete system usually consists of at least one microcomputer and other peripheral devices, such as memories and I/O expanders.**
- **The cost of connecting the various devices within the system must be kept to a minimum.**
- **Such a system usually performs a control function and does not require high-speed data transfer.**
- **Overall efficiency depends on the devices chosen and the interconnecting bus structure.**

In order to produce a system to satisfy these criteria, a serial bus structure is needed. Although serial buses don't have the throughput capability of parallel buses, they do require less wiring and fewer connecting pins. However, a bus is not merely an interconnecting wire, it embodies all the formats and procedures for communication within the system.

Devices communicating with each other on a serial bus must have some form of protocol which avoids all possibilities of confusion, data loss and blockage of information. Fast devices must be able to communicate with slow devices. The system must not be dependent on the devices connected to it, otherwise modifications or improvements would be impossible. A procedure has also to be resolved to decide which device will be in control of the bus and when. And if different devices with different clock speeds are connected to the bus, the bus clock source must be defined.

All these criteria are involved in the specification of the I²C bus.

THE I²C BUS CONCEPT

Any manufacturing process (NMOS, CMOS, I²L) can be supported by the I²C bus. Two wires (SDA – serial data, SCL – serial clock) carry information between the devices connected to the bus. Each device is recognized by a unique address – whether it is a microcomputer, LCD driver, memory or keyboard interface – and can operate as either a transmitter or receiver, depending on the function of the device. Obviously an LCD driver is only a receiver, while a memory can both receive and transmit data. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers (see Table 1). A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.

The I²C bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. As masters are usually microcomputers, let's consider the case of a data transfer between two microcomputers connected to the I²C bus (Figure 1). This highlights the master-slave and receiver-transmitter relationships to be found on the I²C bus. It should be noted that these relationships are not permanent, but only depend on the direction of data transfer at that time. The transfer of data would follow in this way:

1) Suppose microcomputer A wants to send information to microcomputer B
   - microcomputer A (master) addresses microcomputer B (slave)
   - microcomputer A (master transmitter) sends data to microcomputer B (slave receiver)
   - microcomputer A terminates the transfer.
2) If microcomputer A wants to receive information from microcomputer B
   - microcomputer A (master) addresses microcomputer B (slave)
   - microcomputer A (master receiver) receives data from microcomputer B (slave transmitter)
   - microcomputer A terminates the transfer.

Even in this case, the master (microcomputer A) generates the timing and terminates the transfer.

The possibility of more than one microcomputer being connected to the I²C bus means that more than one master could try to initiate a data transfer at the same time. To avoid the chaos that might ensue from such an event, an arbitration procedure has been developed. This procedure relies on the wired-AND connection of all devices to the I²C bus.

If two or more masters try to put information on to the bus, the first to produce a one when the other produces a zero will lose the arbitration. The clock signals during arbitration are a synchronized combination of the clocks generated by the masters using the wired-AND connection to the SCL line (for more detailed information concerning arbitration see Arbitration and Clock Generation).

Generation of clock signals on the I²C bus is always the responsibility of master devices; each master generates its own clock signals when transferring data on the bus. Bus clock signals from a master can only be altered when they are stretched by a slow slave

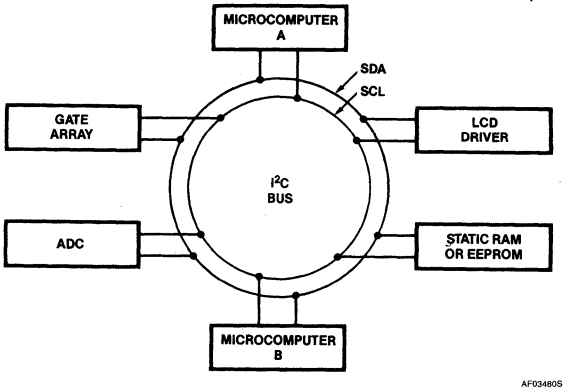


Figure 1. Typical I²C Bus Configuration

I²C Bus Specification

Table 1. Definition of I²C Bus Terminology

TERM	DESCRIPTION
Transmitter	The device which sends data to the bus
Receiver	The device which receives data from the bus
Master	The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by a master
Multi-master	More than one master can attempt to control the bus at the same time without corrupting the message
Arbitration	Procedure to ensure that if more than one master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted
Synchronization	Procedure to synchronize the clock signals of two or more devices

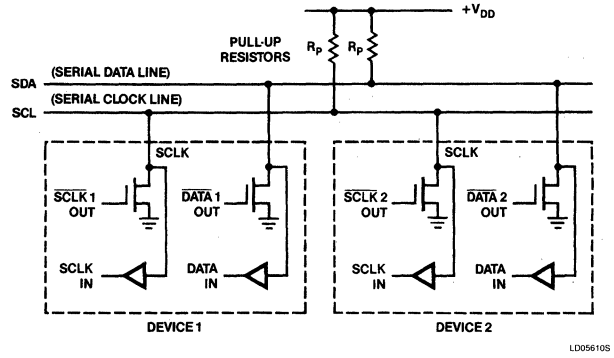


Figure 2. Connection of Devices to the I²C Bus

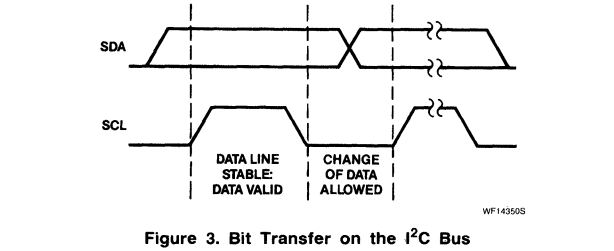


Figure 3. Bit Transfer on the I²C Bus

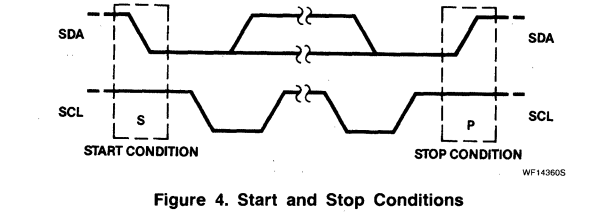


Figure 4. Start and Stop Conditions

device holding down the clock line or by another master when arbitration takes place.

GENERAL CHARACTERISTICS

Both SDA and SCL are bidirectional lines, connected to a positive supply voltage via a pull-up resistor (see Figure 2). When the bus is free, both lines are High. The output stages of devices connected to the bus must have an open-drain or open-collector in order to perform the wired-AND function. Data on the I²C bus can be transferred at a rate up to 100kbit/s. The number of devices connected to the bus is solely dependent on the limiting bus capacitance of 400pF.

BIT TRANSFER

Due to the variety of different technology devices (CMOS, NMOS, I²L) which can be connected to the I²C bus, the levels of the logical 0 (Low) and 1 (High) are not fixed and depend on the appropriate level of V_{DD} (see Electrical Specifications). One clock pulse is generated for each data bit transferred.

Data Validity

The data on the SDA line must be stable during the High period of the clock. The High or Low state of the data line can only change when the clock signal on the SCL line is Low (Figure 3).

Start and Stop Conditions

Within the procedure of the I²C bus, unique situations arise which are defined as start and stop conditions (see Figure 4).

A High-to-Low transition of the SDA line while SCL is High is one such unique case. This situation indicates a start condition.

A Low-to-High transition of the SDA line while SCL is High defines a stop condition.

Start and stop conditions are always generated by the master. The bus is considered to be busy after the start condition. The bus is considered to be free again a certain time after the stop condition. This bus free situation will be described later in detail.

Detection of start and stop conditions by devices connected to the bus is easy if they possess the necessary interfacing hardware. However, microcomputers with no such interface have to sample the SDA line at least twice per clock period in order to sense the transition.

TRANSFERRING DATA

Byte Format

Every byte put on the SDA line must be 8 bits long. The number of bytes that can be transmitted per transfer is unrestricted. Each byte must be followed by an acknowledge bit.

I²C Bus Specification

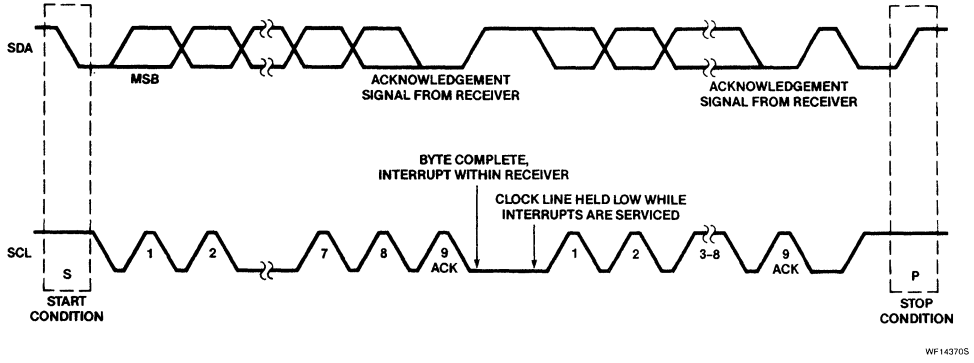


Figure 5. Data Transfer on the I²C Bus

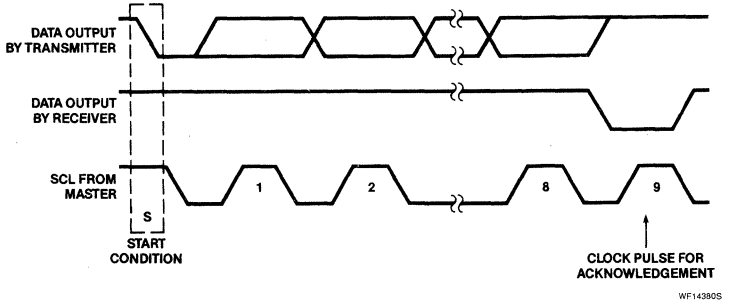


Figure 6. Acknowledge on the I²C Bus

Data is transferred with the most significant bit (MSB) first (Figure 5). If a receiving device cannot receive another complete byte of data until it has performed some other function, for example, to service an internal interrupt, it can hold the clock line SCL Low to force the transmitter into a wait state. Data transfer then continues when the receiver is ready for another byte of data and releases the clock line SCL.

In some cases, it is permitted to use a different format from the I²C bus format, such as CBUS compatible devices. A message which starts with such an address can be terminated by the generation of a stop condition, even during the transmission of a byte. In this case, no acknowledge is generated.

Acknowledge

Data transfer with acknowledge is obligatory. The acknowledge-related clock pulse is generated by the master. The transmitting device releases the SDA line (High) during the acknowledge clock pulse.

The receiving device has to pull down the SDA line during the acknowledge clock pulse so that the SDA line is stable Low during the high period of this clock pulse (Figure 6). Of course, setup and hold times must also be taken into account and these will be described in the Timing section.

Usually, a receiver which has been addressed is obliged to generate an acknowledge after each byte has been received (except when the message starts with a CBUS address.

When a slave receiver does not acknowledge on the slave address, for example, because it is unable to receive while it is performing some real-time function, the data line must be left High by the slave. The master can then generate a STOP condition to abort the transfer.

If a slave receiver does acknowledge the slave address, but some time later in the transfer cannot receive any more data bytes, the master must again abort the transfer. This is indicated by the slave not generating the acknowledge on the first byte following. The slave leaves the data line High and the master generates the STOP condition.

In the case of a master receiver involved in a transfer, it must signal an end of data to the slave transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave transmitter must release the data line to allow the master to generate the STOP condition.

ARBITRATION AND CLOCK GENERATION

Synchronization

All masters generate their own clock on the SCL line to transfer messages on the I²C bus. Data is only valid during the clock High period on the SCL line; therefore, a defined clock is needed if the bit-by-bit arbitration procedure is to take place.

Clock synchronization is performed using the wired-AND connection of devices to the SCL LINE. This means that a High-to-Low transi-

I²C Bus Specification

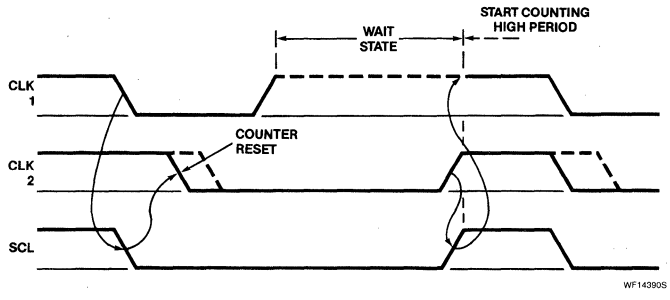


Figure 7. Clock Synchronization During the Arbitration Procedure

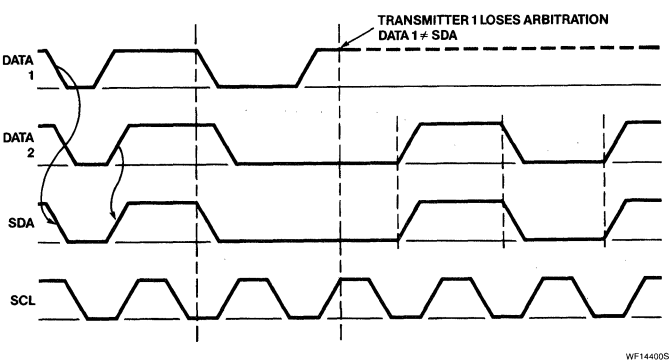


Figure 8. Arbitration Procedure of Two Masters

tion on the SCL line will affect the devices concerned, causing them to start counting off their Low period. Once a device clock has gone Low it will hold the SCL line in that state until the clock High state is reached (Figure 7). However, the Low-to-High change in this device clock may not change the state of the SCL line if another device
clock is still within its Low period. Therefore, SCL will be held Low by the device with the longest Low period. Devices with shorter Low periods enter a High wait state during this time.

When all devices concerned have counted off their Low period, the clock line will be released and go High. There will then be no difference between the device clocks and the state of the SCL line and all of them will start counting their High periods. The first device to complete its High period will again pull the SCL line Low.

In this way, a synchronized SCL clock is generated for which the Low period is determined by the device with the longest clock Low period while the High period on SCL is determined by the device with the shortest clock High period.

Arbitration

Arbitration takes place on the SDA line in such a way that the master which transmits a High level, while another master transmits a Low level, will switch off its DATA output stage since the level on the bus does not correspond to its own level.

Arbitration can carry on through many bits. The first stage of arbitration is the comparison of the address bits. If the masters are each trying to address the same device, arbitration continues into a comparison of the data. Because address and data information is used on the I²C bus for the arbitration, no information is lost during this process.

A master which loses the arbitration can generate clock pulses until the end of the byte in which it loses the arbitration.

If a master does lose arbitration during the addressing stage, it is possible that the winning master is trying to address it. Therefore, the losing master must switch over immediately to its slave receiver mode.

Figure 8 shows the arbitration procedure for two masters. Of course more may be involved, depending on how many masters are connected to the bus. The moment there is a difference between the internal data level of the master generating DATA 1 and the actual level on the SDA line, its data output is switched off, which means that a High output level is then connected to the bus. This will not affect the data transfer initiated by the winning master. As control of the I²C bus is decided solely on the address and data sent by competing masters, there is no central master, nor any order of priority on the bus.

Use of the Clock Synchronizing Mechanism as a Handshake

In addition to being used during the arbitration procedure, the clock synchronization mechanism can be used to enable receiving devices to cope with fast data transfers, either on a byte or bit level.

On the byte level, a device may be able to receive bytes of data at a fast rate, but needs more time to store a received byte or prepare another byte to be transmitted. Slave devices can then hold the SCL line Low, after reception and acknowledge of a byte, to force the master into a wait state until the slave is ready for the next byte transfer in a type of handshake procedure.

On the bit level, a device such as a microcomputer without a hardware I²C interface on-chip can slow down the bus clock by extending each clock Low period. In this way, the speed of any master is adapted to the internal operating rate of this device.

I²C Bus Specification

FORMATS

Data transfers follow the format shown in Figure 9. After the start condition, a slave address is sent. This address is 7 bits long; the eighth bit is a data direction bit (R/$\overline{W}$). A zero indicates a transmission (WRITE); a one indicates a request for data (READ). A data transfer is always terminated by a stop condition generated by the master. However, if a master still wishes to communicate on the bus, it can generate another start condition, and address another slave without first generating a stop condition. Various combinations of read/write formats are then possible within such a transfer.

At the moment of the first acknowledge, the master transmitter becomes a master receiver and the slave receiver becomes a slave transmitter. This acknowledge is still generated by the slave.

The stop condition is generated by the master.

During a change of direction within a transfer, the start condition and the slave address are both repeated, but with the R/$\overline{W}$ bit reversed.

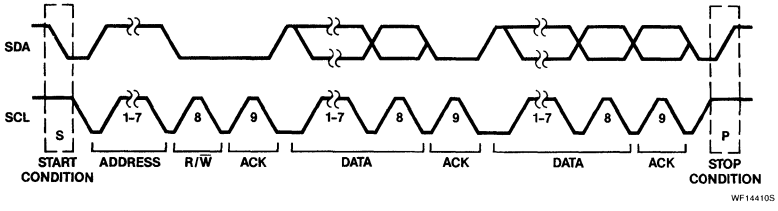
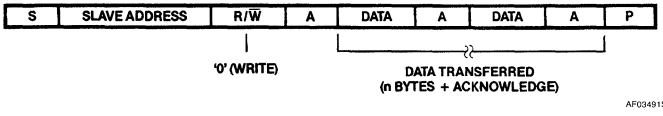


Figure 9. A Complete Data Transfer

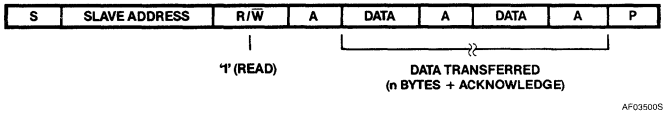
Possible Data Transfer Formats are:

a) Master transmitter transmits to slave receiver. Direction is not changed.

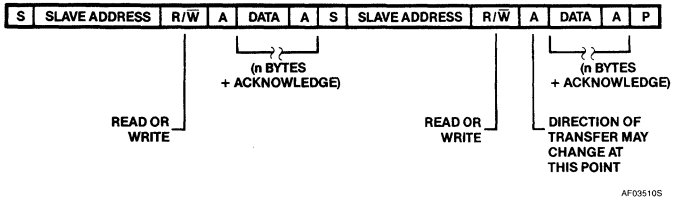
A = ACKNOWLEDGE
S = START
P = STOP



b) Master reads slave immediately after first byte.



c) Combined formats.



NOTES:
1. Combined formats can be used, for example, to control a serial memory. During the first data byte, the internal memory location has to be written. After the start condition is repeated, data can then be transferred.
2. All decisions on auto-increment or decrement of previously accessed memory locations, etc., are taken by the designer of the device.
3. Each byte is followed by an acknowledge as indicated by the A blocks in the sequence.
4. I²C devices have to reset their bus logic on receipt of a start condition so that they all anticipate the sending of a slave address.

I²C Bus Specification

ADDRESSING

The first byte after the start condition determines which slave will be selected by the master. Usually, this first byte follows that start procedure. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an acknowledge, although devices can be made to ignore this address. The second byte of the general call address then defines the action to be taken.

Definition of Bits in the First Byte

The first seven bits of this byte make up the slave address (Figure 10). The eighth bit (LSB – least significant bit) determines the direction of the message. A zero on the least significant position of the first byte means that the master will write information to a selected slave; a one in this position means that the master will read information from the slave.

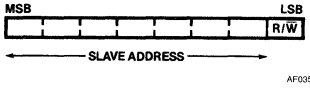


Figure 10. The First Byte After the Start Procedure

When an address is sent, each device in a system compares the first 7 bits after the start condition with its own address. If there is a match, the device will consider itself addressed by the master as a slave receiver or slave transmitter, depending on the R/$\overline{W}$ bit.

The slave address can be made up of a fixed and a programmable part. Since it is expected that identical ICs will be used more than once in a system, the programmable part of the slave address enables the maximum possible number of such devices to be connected to the I²C bus. The number of programmable address bits of a device depends on the number of pins available. For example, if a device has 4 fixed and 3 programmable address bits, a total of eight identical devices can be connected to the same bus.

The I²C bus committee is available to coordinate allocation of I²C addresses.

The bit combination 1111XXX of the slave address is reserved for future extension purposes.

The address 1111111 is reserved as the extension address. This means that the addressing procedure will be continued in the next byte(s). Devices that do not use the extended addressing do not react at the reception of this byte. The seven other possibilities in group 1111 will also only be used for extension purposes but are not yet allocated.

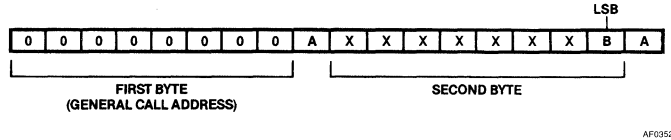


Figure 11. General Call Address Format

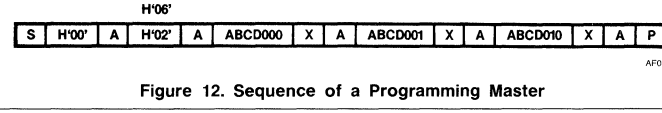


Figure 12. Sequence of a Programming Master

The combination 0000XXX has been defined as a special group. The following addresses have been allocated:

FIRST BYTE			
Slave Address		R/$\overline{W}$	
0000	000	0	General call address
0000	000	1	Start byte
0000	001	X	CBUS address
0000	010	X	Address reserved for different bus format
0000	011	X	To be defined
0000	100	X	
0000	101	X	
0000	110	X	
0000	111	X	

No device is allowed to acknowledge at the reception of the start byte.

The CBUS address has been reserved to enable the intermixing of CBUS and I²C devices in one system. I²C bus devices are not allowed to respond at the reception of this address.

The address reserved for a different bus format is included to enable the mixing of I²C and other protocols. Only I²C devices that are able to work with such formats and protocols are allowed to respond to this address.

General Call Address

The general call address should be used to address every device connected to the I²C bus. However, if a device does not need any of the data supplied within the general call structure, it can ignore this address by not acknowledging. If a device does require data from a general call address, it will acknowledge this address and behave as a slave receiver. The second and following bytes will be acknowledged by every slave receiver capable of handling this data. A slave which cannot process one of these bytes must ignore it by not acknowledging.

The meaning of the general call address is always specified in the second byte (Figure 11).

There are two cases to consider:
1. When the least significant bit B is a zero.
2. When the least significant bit B is a one.

When B is a zero, the second byte has the following definition:

00000110 (H'06') Reset and write the programmable part of slave address by software and hardware. On receiving this two-byte sequence, all devices (designed to respond to the general call address) will reset and take in the programmable part of their address.
Precautions must be taken to ensure that a device is not pulling down the SDA or SCL line after applying the supply voltage, since these low levels would block the bus.

00000010 (H'02') Write slave address by software only. All devices which obtain the programmable part of their address by software (and which have been designed to respond to the general call address) will enter a mode in which they can be programmed. The device will not reset.

I²C Bus Specification

An example of a data transfer of a programming master is shown in Figure 12 (ABCD represents the fixed part of the address).

00000100 (H'04') Write slave address by hardware only. All devices which define the programmable part of their address by hardware (and which respond to the general call address) will latch this programmable part at the reception of this two-byte sequence. The device will not reset.

00000000 (H'00') This code is not allowed to be used as the second byte.

Sequences of programming procedure are published in the appropriate device data sheets.

The remaining codes have not been fixed and devices must ignore these codes.

When B is a one, the two-byte sequence is a hardware general call. This means that the sequence is transmitted by a hardware master device, such as a keyboard scanner, which cannot be programmed to transmit a desired slave address. Since a hardware master does not know in advance to which device the message must be transferred, it can only generate this hardware general call and its own address, thereby identifying itself to the system (Figure 13).

The seven bits remaining in the second byte contain the device address of the hardware master. This address is recognized by an intelligent device, such as a microcomputer, connected to the bus which will then direct the information coming from the hardware master. If the hardware master can also act as a slave, the slave address is identical to the master address.

In some systems an alternative could be that the hardware master transmitter is brought in the slave receiver mode after the system reset. In this way, a system configuring master can tell the hardware master transmitter (which is now in slave receiver mode) to which address data must be sent (Figure 14). After this programming procedure, the hardware master remains in the master transmitter mode.

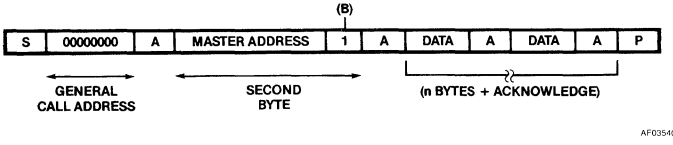
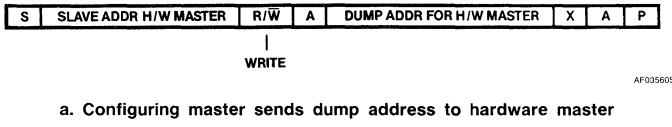
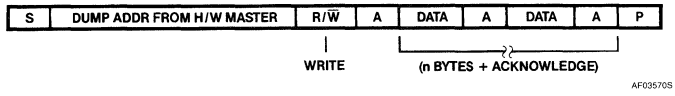


Figure 13. Data Transfer From Hardware Master Transmitter



a. Configuring master sends dump address to hardware master



b. Hardware master dumps data to selected slave device

Figure 14. Data Transfer of Hardware Master Transmitter Capable of Dumping Data Directly to Slave Devices

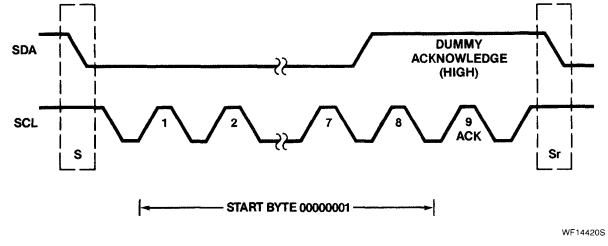


Figure 15. Start Byte Procedure

Start Byte

Microcomputers can be connected to the I²C bus in two ways. If an on-chip hardware I²C bus interface is present, the microcomputer can be programmed to be interrupted only by requests from the bus. When the device possesses no such interface, it must constantly monitor the bus via software. Obviously, the more times the microcomputer monitors, or polls, the bus, the less time it can spend carrying out its intended function.

Therefore, there is a difference in speed between fast hardware devices and the relatively slow microcomputer which relies on software polling.

In this case, data transfer can be preceded by a start procedure which is much longer than normal (Figure 15). The start procedure consists of:

a) A start condition, (S)
b) A start byte 00000001
c) An acknowledge clock pulse
d) A repeated start condition, (Sr)

After the start condition (S) has been transmitted by a master requiring bus access, the start byte (00000001) is transmitted. Another microcomputer can therefore sample the SDA line on a low sampling rate until one of the seven zeros in the start byte is detected. After detection of this Low level on the SDA line, the microcomputer is then able to switch to a higher sampling rate in order to find the second start condition (Sr) which is then used for synchronization.

A hardware receiver will reset at the reception of the second start condition (Sr) and will therefore ignore the start byte.

After the start byte, an acknowledge-related clock pulse is generated. This is present only to conform with the byte handling format used on the bus. No device is allowed to acknowledge the start byte.

I²C Bus Specification

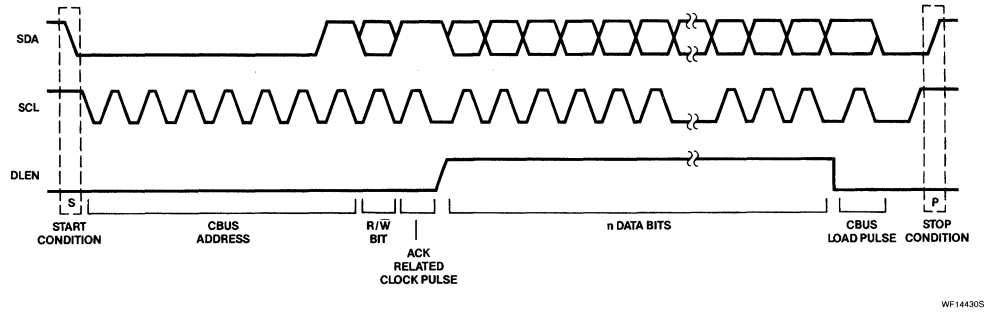


Figure 16. Data Format of Transmissions With CBUS Receiver/Transmitter

CBUS Compatibility

Existing CBUS receivers can be connected to the I²C bus. In this case, a third line called DLEN has to be connected and the acknowledge bit omitted. Normally, I²C transmissions are multiples of 8-bit bytes; however, CBUS devices have different formats.

In a mixed bus structure, I²C devices are not allowed to respond on the CBUS message. For this reason, a special CBUS address (0000001X) has been reserved. No I²C device will respond to this address. After the transmission of the CBUS address, the DLEN line can be made active and transmission, according to the CBUS format, can be performed (Figure 16).

After the stop condition, all devices are again ready to accept data.

Master transmitters are allowed to generate CBUS formats after having sent the CBUS address. Such a transmission is terminated by a stop condition, recognized by all devices. In the low speed mode, full 8-bit bytes must always be transmitted and the timing of the DLEN signal adapted.

If the CBUS configuration is known and no expansion with CBUS devices is foreseen, the user is allowed to adapt the hold time to the specific requirements of device(s) used.

ELECTRICAL SPECIFICATIONS OF INPUTS AND OUTPUTS OF I²C DEVICES

The I²C bus allows communication between devices made in different technologies which might also use different supply voltages.

For devices with fixed input levels, operating on a supply voltage of +5V ± 10%, the following levels have been defined:

V_{ILmax} = 1.5V (maximum input Low voltage)

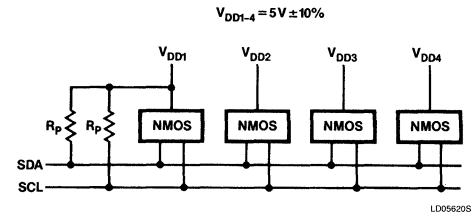


Figure 17. Fixed Input Level Devices Connected to the I²C Bus

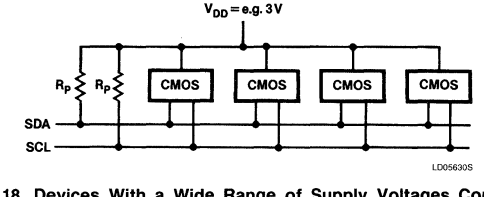


Figure 18. Devices With a Wide Range of Supply Voltages Connected to the I²C Bus

V_{IHmin} = 3V (minimum input High voltage)

Devices operating on a fixed supply voltage different from +5V (e.g. I²L), must also have these input levels of 1.5V and 3V for V_{IL} and V_{IH}, respectively.

For devices operating over a wide range of supply voltages (e.g. CMOS), the following levels have been defined:

V_{ILmax} = 0.3V_{DD} (maximum input Low voltage)

V_{IHmin} = 0.7V_{DD} (minimum input High voltage)

For both groups of devices, the maximum output Low value has been defined:

V_{OLmax} = 0.4V (max. output voltage Low) at 3mA sink current

The maximum low-level input current at V_{OLmax} of both the SDA pin and the SCL pin of an I²C device is −10μA, including the leakage current of a possible output stage.

The maximum high-level input current at 0.9V_{DD} of both the SDA pin and SCL pin of an I²C device is 10μA, including the leakage current of a possible output stage.

The maximum capacitance of both the SDA pin and the SCL pin of an I²C device is 10pF.

Devices with fixed input levels can each have their own power supply of +5V ± 10%. Pull-up resistors can be connected to any supply (see Figure 17).

However, the devices with input levels related to V_{DD} must have one common supply line to which the pull-up resistor is also connected (see Figure 18).

I²C Bus Specification

When devices with fixed input levels are mixed with devices with V_{DD}-related levels, the latter devices have to be connected to one common supply line of +5V ± 10% along with the pull-up resistors (Figure 19).

Input levels are defined in such a way that:

1. The noise margin on the Low level is 0.1 V_{DD}.
2. The noise margin on the High level is 0.2 V_{DD}.
3. Series resistors (R_S) up to 300Ω can be used for flash-over protection against high voltage spikes on the SDA and SCL line (due to flash-over of a TV picture tube, for example) (Figure 20).

The maximum bus capacitance per wire is 400pF. This includes the capacitance of the wire itself and the capacitance of the pins connected to it.

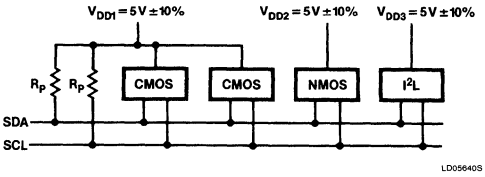


Figure 19. Devices With V_{DD} Related Levels Mixed With Fixed Input Level Devices on the I²C Bus

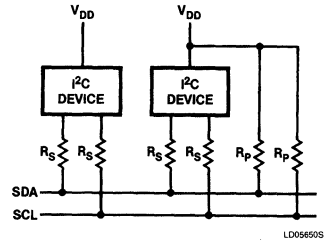


Figure 20. Serial Resistors (R_S) for Protection Against High Voltage

TIMING

The clock on the I²C bus has a minimum Low period of 4.7μs and a minimum High period of 4μs. Masters in this mode can generate a bus clock with a frequency from 0 to 100kHz.

All devices connected to the bus must be able to follow transfers with frequencies up to 100kHz, either by being able to transmit or receive at that speed or by applying the clock synchronization procedure which will force the master into a wait state and stretch the Low periods. In the latter case the frequency is reduced.

Figure 21 shows the timing requirements in detail. A description of the abbreviations used is shown in Table 2. All timing references are at V_{ILmax} and V_{ILmin}.

LOW-SPEED MODE

As explained previously, there is a difference in speed on the I²C bus between fast hardware devices and the relatively slow microcomputer which relies on software polling. For this reason a low speed mode is available on the I²C bus to allow these microcomputers to poll the bus less often.

Start and Stop Conditions

In the low-speed mode, data transfer is preceded by the start procedure.

Data Format and Timing

The bus clock in this mode has a Low period of 130μs ± 25μs and a High period of 390μs ± 25μs, resulting in a clock frequency of approx. 2kHz. The duty cycle of the clock has this Low-to-High ratio to allow for more efficient use of microcomputers without an on-chip hardware I²C bus interface. In this mode also, data transfer with acknowledge is obligatory. The maximum number of bytes transferred is not limited (Figure 22).

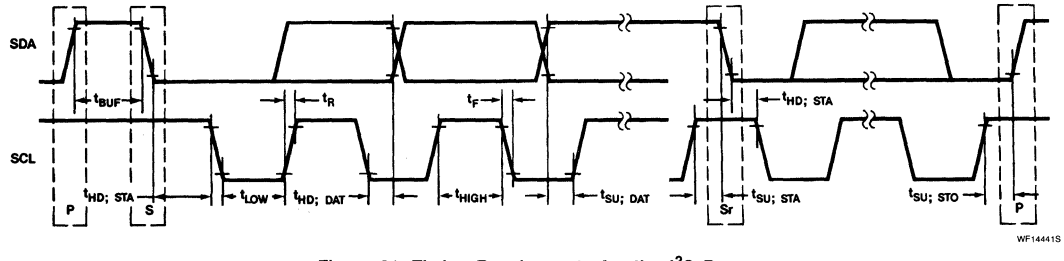


Figure 21. Timing Requirements for the I²C Bus

I²C Bus Specification

Table 2. Timing Requirement for the I²C Bus

SYMBOL	PARAMETER	LIMITS		UNIT
		Min	Max	
f_{SCL}	SCL clock frequency	0	100	kHz
t_{BUF}	Time the bus must be free before a new transmission can start	4.7		μs
$t_{HD;\ STA}$	Hold time start condition. After this period the first clock pulse is generated	4		μs
t_{LOW}	The Low period of the clock	4.7		μs
t_{HIGH}	The High period of the clock	4		μs
$t_{SU;\ STA}$	Setup time for start condition (Only relevant for a repeated start condition)	4.7		μs
$t_{HD;\ DAT}$	Hold time DATA for CBUS compatible masters for I²C devices	 5 0*		 μs μs
$t_{SU;\ DAT}$	Setup time DATA	250		ns
t_R	Rise time of both SDA and SCL lines		1	μs
t_F	Fall time of both SDA and SCL lines		300	ns
$t_{SU;\ STO}$	Setup time for stop condition	4.7		μs

NOTES:
All values referenced to V_{IH} and V_{IL} levels.
* Note that a transmitter must internally provide a hold time to bridge the undefined region (300ns max.) of the falling edge of SCL.

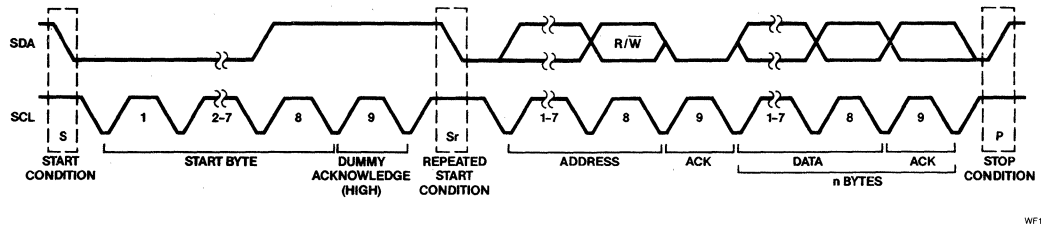


Figure 22. Data Transfer Low-Speed Mode

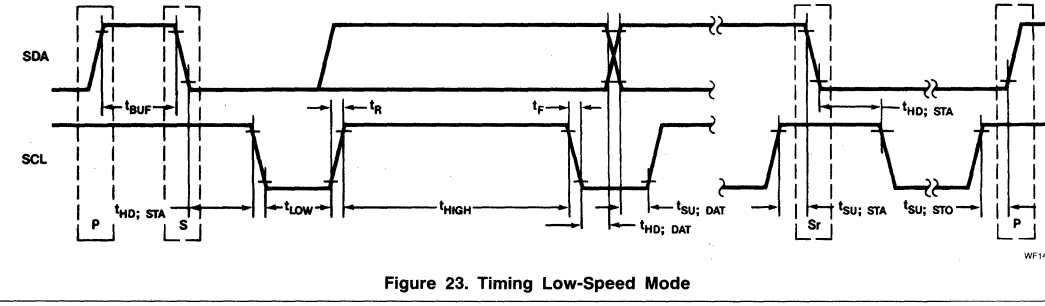


Figure 23. Timing Low-Speed Mode

I²C Bus Specification

LOW SPEED MODE

CLOCK	: $t_{LOW} = 130\mu s \pm 25\mu s$
DUTY CYCLE	: $t_{HIGH} = 390\mu s \pm 25\mu s$
	: 1:3 Low-to-High (Duty cycle of clock generator)
START BYTE	: 0000 0001
MAX. NO. OF BYTES	: UNRESTRICTED
PREMATURE TERMINATION OF TRANSFER	: NOT ALLOWED
ACKNOWLEDGE CLOCK BIT	: ALWAYS PROVIDED
ACKNOWLEDGEMENT OF SLAVES	: OBLIGATORY

In this mode, a transfer cannot be terminated during the transmission of a byte.

The bus is considered busy after the first start condition. It is considered free again one minimum clock Low period, 105μs, after the detection of the stop condition. Figure 23 shows the timing requirements in detail, Table 3 explains the abbreviations.

Table 3. Timing Low Speed Mode

SYMBOL	PARAMETER	LIMITS		UNIT
		Min	Max	
t_{BUF}	Time the bus must be free before a new transmission can start	105		μs
$t_{HD;\ STA}$	Hold time start condition. After this period the first clock pulse is generated	365		μs
$t_{HD;\ STA}$	Hold time (repeated start condition only)	210		μs
t_{LOW}	The Low period of the clock	105	155	μs
t_{HIGH}	The High period of the clock	365	415	μs
$t_{SU;\ STA}$	Setup time for start condition (Only relevant for a repeated start condition)	105	155	μs
$t_{HD;\ DAT}$	Hold time DATA for CBUS compatible masters for I²C devices	 5 0*		 μs μs
$t_{SU;\ DAT}$	Setup time DATA	250		ns
t_R	Rise time of both SDA and SCL lines		1	μs
t_F	Fall time of both SDA and SCL lines		300	ns
$t_{SU;\ STO}$	Setup time for stop condition	105	155	μs

NOTES:
All values referenced to V_{IH} and V_{IL} levels.
* Note that a transmitter must internally provide a hold time to bridge the undefined region (300ns max.) of the falling edge of SCL.

I²C Bus Specification

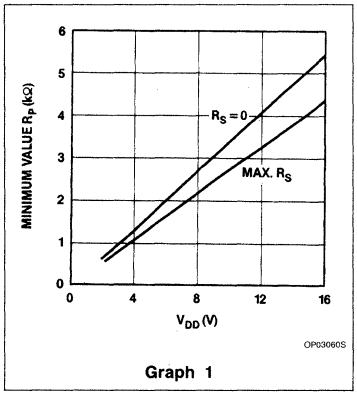
APPENDIX A

Maximum and minimum values of the pull-up resistors R_P and series resistors R_S (See Figure 20).

In a I²C bus system these values depend on the following parameters:

- Supply voltage
- Bus capacitance
- Number of devices (input current + leakage current)

1) The supply voltage limits the minimum value of the R_P resistor due to the specified 3mA as minimum sink current of the output stages, at 0.4V as maximum low voltage. In Graph 1, V_{DD} against R_{Pmin} is shown.

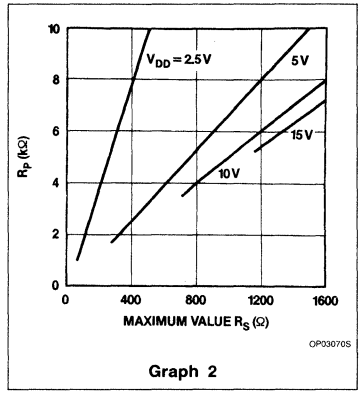


Graph 1

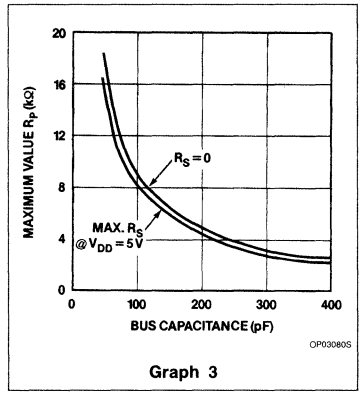
The desired noise margin of 0.1 V_{DD} for the low level limits the maximum value of R_S.

In Graph 2, R_{Smax} against R_P is shown.

2) The bus capacitance is the total capacitance of wire, connections, and pins. This capacitance limits the maximum value of R_P because of the specified rise time of 1μs.



Graph 2

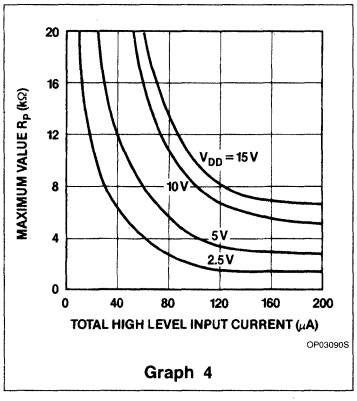


Graph 3

In Graph 3, the bus capacitance – R_{Pmax} relationship is shown.

3) The maximum high-level input current of each input/output connection has a specified value of 10μA max. Due to the desired noise margin of 0.2 V_{DD} for the high level, this input current limits the maximum value of R_P. This limit is dependent on V_{DD}.

In Graph 4 the total high-level input current – R_{Pmax} relationship is shown.



Graph 4

I²C LICENSE

Purchase of Signetics or Philips I²C components conveys a license under the Philips I²C patent rights to use these components in an I²C system, provided that the system conforms to the I²C standard specification as defined by Philips.

Signetics

I²C PERIPHERAL SELECTION GUIDE

General Purpose ICs

LCD Drivers

PCF8566:	96-segment LCD driver 1:1 - 1:4 Mux
PCF8576:	160-segment LCD driver 1:1 - 1:4 Mux
PCF8577/A:	64-segment LCD driver 1:1 - 1:2 Mux
PCF8578/79:	Row/column LCD dot-matrix driver; 1:8 - 1:32 Mux

I/O Expandors

PCF8574/A:	8-bit remote I/O port (I²C bus to parallel converter)
PCF8584:	8-bit parallel to I²C converter*
SAA1064:	4-digit LED driver
SAA1300:	5-bit high-current driver

Data Converters

PCF8591:	4-channel, 8-bit Mux ADC + one DAC
TDA8442:	Quad 6-bit DAC
TDA8444:	Octal 6-bit DAC

Memory

PCF8570/C:	256-byte static RAM
PCF8571:	128-byte static RAM
PCF8581:	128-byte EEPROM*
PCF8582A:	256-byte EEPROM
PCF8583:	256-byte RAM/clock/calendar

Clock/Calendars

PCF8573:	Clock/calendar
PCF8583:	Clock/calendar/256-byte RAM

Application-Oriented ICs

Video/Radio/Audio

PCF8200:	Voice synthesizer (male/female speech)
SAA3028:	Transcoder (RC-5) for IR remote control
SAB3035/36/37:	Digital tuning circuits for computer-controlled TV
TDA8440:	Video/audio switch
TDA8443/A:	YUV/RGB matrix switch
TEA6000/6100:	FM/IF and digital tuning IC for computer-controlled radio
TEA6300:	Sound fader control and preamplifier/source selector for car radio
TEA6310T:	Sound fader control with tone and volume control for car radio
TSA5510:	PLL frequency synthesizer for radio
TSA6057:	PLL frequency synthesizer for radio

Telecom

NE5750/51:	Audio processor pair*
PCD3311/12:	Tone generator (DTMF/modem/musical)
PCD3341:	Advanced 10 to 110-number repertory dialer with LCD control
PCD3343:	Microcontroller with 224-byte RAM/3k ROM
PCD3348:	Microcontroller with 256-byte RAM/8k ROM
UMA1000T:	Data processor for mobile telephones*
UMA1010T:	1GHz frequency synthesizer for mobile telephones*

68000-Based CMOS µProcessor

SCC68070:	68000 CPU/MMU/UART/DMA/timer

80C51-Based CMOS µControllers+

S80C552:	ROM-less version of S83C552
S80C652:	ROM-less version of S83C652
S83C552:	256-byte RAM/8k ROM/ADC/UART
S83C652:	256-byte RAM/8k/ ROM
S83C751:	64-byte RAM/2k ROM
S87C552:	EPROM version of S83C552*
S87C652:	EPROM version of S83C652*
S87C751:	EPROM version of S83C751

+also available with extended temperature ranges

8048 Instruction-Set Based CMOS µControllers

PCF84C00:	256-byte RAM/bond-out version for prototype development
PCF84C21:	64-byte RAM/2k ROM
PCF84C41:	128-byte RAM/4k ROM
PCF84C81:	256-byte RAM/8k ROM
PCF84C85:	256-byte RAM/8k ROM/Extended I/O
PCF84C430:	128-byte RAM/4k ROM/96-segment LCD driver*

*** Future Device**

Signetics

Microprocessor Products

Section 5
Development Support Tools

INDEX

DEVELOPMENT SUPPORT TOOLS

Signetics stocks development support tools to help simplify your design activities. These include:

- In-circuit emulation development systems
- Cross-assemblers
- Low-cost EPROM programming equipment

In addition, Signetics works closely with many 'third-party' vendors who provide support tools for our wide variety of 80C51-based microcontroller derivatives.

DEVELOPMENT SYSTEMS
In most cases, our development systems are available in two versions for ROM and ROMless applications. The ROM emulation products are capable of supporting all versions of a given device type including EPROM, ROM, and ROMless devices. For example, the SMI-83C451 emulator can support designs based on either the 87C451, 83C451, or 80C451. In contrast, a ROMless emulator can only support applications designed for a ROMless microcontroller. For example, the SMI-80C451 emulator can only support applications using the 80C451.

These development systems are designed to connect to the serial port of an IBM-PC or compatible personal computer. The development system package includes the emulator hardware, host emulation software, cross-assembler, user's manuals, cables, and power supply. Note that these emulators do not include provisions for programming ('burning') the EPROM in EPROM-based microcontrollers.

CROSS-ASSEMBLER
The cross-assembler provided with the development system package is also available separately. This assembler supports macros and conditional assembly operations and is designed to run on an IBM-PC or compatible processor. This assembler uses an external text file to define the architecture of specific microcontrollers. Support for new microcontroller types can be added by creating a simple text file to define the new micro-controller, thereby allowing the assembler to support a variety of product derivatives now and in future applications.

EPROM PROGRAMMING SUPPORT
Signetics works closely with major suppliers of EPROM programming equipment to support our family of EPROM micro-controllers. As a result, EPROM programming support is available within the programming facilities of many major distributors and customers.

Your local Signetics sales office or representative can provide a list of manufacturers of EPROM programming equipment that offer programming support products for Signetics EPROM microcontrollers.

SIGNETICS MICROCONTROLLER DEVELOPMENT SYSTEMS

Devices Supported / Development System	SCN8031	SC80C31	SCN8051	SC80C51	SC87C51	SCN8032	SCN8052	SC80C451	SC83C451	SC87C451	S80C652	S80C552	S83C751	S87C751
SUI-8051SD	X	X	X	X	X	X	X							
SMI-80C451SD								X						
SMI-83C451SD								X	X	X				
SMI-80C652SD											X			
SMI-80C552SD												X		
SMI-83C751SD													X	X
SM8051ASMSD	Macro Cross-Assembler													
SMI-CNV451SD	PLCC to DIP Adapter for SMI-80C451SD													

The emulator package includes emulator hardware, power supplies, interface cables, host software and cross-assembler. This package requires the use of an IBM-PC or 100% compatibles with 640k RAM, PC DOS version 2.0 or later and an RS232 port.

EPROM MICROCONTROLLER PROGRAMMING SYSTEM

SPGM-100SD	EPROM PROGRAMMING SYSTEM		
	Adapter Sockets for SPGM-100SD		
Part No.	**Microcontroller**	**DIP**	**LCC**
SAM-751SD	87C751	24-Pin	
SAM-751ASD	87C751		28-Pin
SAM-51SD	87C51	40-Pin	
SAM-51ASD	87C51		44-Pin
SAM-451SD	87C451	64-Pin	
SAM-451ASD	87C451		68-Pin
SAM-752SD	87C752	28-Pin	
SAM-752ASD	87C752		28-Pin

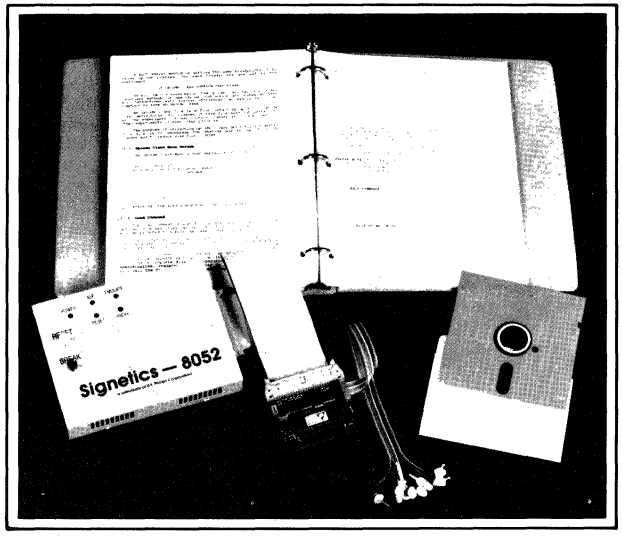
SHIPMENTS THROUGH SIGNETICS DISTRIBUTORS ONLY

Signetics

Microcontroller Support Tools

System Overview

Signetics-8052 In-Circuit Emulator for 8051 or 8052 Microcontroller



- Serially linked to IBM PC or 100% compatible hosts
- Advanced menu driven human interface
- Real time and transparent emulation up to 16 MHz
- Disassembler and Single Line Assembler
- Examine/Modify Memory capabilities
- 16 Break and Trace Trigger Conditions
- Supports both modes:
  Microcontroller
  Microprocessor
- 9 Probe Clips
  7 External Events
  1 External Trigger Input
  1 External Trigger Output
- Over 128,000 Break Triggers and 64,000 Trace Triggers
- Emulation Memory:

	Standard	Optional
— Program	16K	64K
— External Data	16K	64K

- Full Symbolic Debug Capability
- High Level Language Support
- Up to 64K Pass Counts
- Separate Program and Data Memory Mapping in 16 byte blocks
- Experiment Editor/Compiler
- Trace with 2K frames
- Opcode Class Editor

PHILIPS

PHILIPS

98-8270-010
1029-000/688/IM

Functional Description

The Signetics-8052 is an in-circuit emulator which is designed for use in developing, testing and debugging designs based on 8051 or 8052 single chip microcontrollers. The Signetics-8052 allows the development of hardware and software designs to occur simultaneously.
The Signetics-8052 emulator assists in the following design phases:

- Software Development
- Manufacturing
- Integration of target software and system hardware
- Field Service

Support

The Signetics-8052 emulator not only assists the designer in developing, testing and debugging 8051 or 8052 microcontroller designs, but is also backed by Signetics with an extensive service and support policy that includes:

- Updates to the Signetics User's Manual and Signetics host Operating System software that are provided at no charge for a ninety (90) day period.
- Toll free phone number (1-800-A-HLP-4-51) to resolve system hardware and/or software concerns.
- A 72-hour repair on disabled systems under warranty.

Features Description

Emulator Functions

- Microcontrollers supported:
  8031 80C31 8032
  8051 80C51 8052
  87C51
- Performance
  Real-time 3.5 to 16 MHz
- Transparency:
  Operational and electrical
- Examine and modify:
  Program
  Internal data
  External data

User Interface

- Advanced menu driven operating system
- 11 functional capabilities:
  Load OS-Escape
  Upload Help
  Download Configure
  Store Restore
  Interrogate Exit
  Macro

Mode

- Allows user to emulate all the operating modes of:
  Full operation with external address/data bus
  Single-chip operation with NO external address/data bus

Interrogation

- Allows the user to:
  Run experiments
  Examine the system status
  Set break and trace triggers
  Examine/modify data
- 16 functional capabilities:
  Run
  Single-step
  Reset
  OS-Escape
  Set pass count
  Set simple break and trace triggers
  Set repetition counter
  Set phantom break and trace triggers
  Set trace triggers (start, end & center)
  Turn trace trigger ON/OFF
  View up to 2K of trace buffer
  Examine/modify:
  Special Function Registers
  Internal data memory
  External data memory
  Program memory
  Emulator experiments

Experiment

- Used to specify complex break and trace triggers
- Uses the if-then construct
- Allowable trigger conditions are:
  PC address
  PC address range
  Opcode value
  Opcode class
  Special function registers
  Direct byte address
  Direct byte address range
  Direct bit address
  Direct bit address range
  Immediate operand value
  Read/write to bit or direct addresses
  External data address
  External data address range
  Logical AND or OR of any of the above
  Pass count overflow
  External input
- Over: 128,000 break triggers
  64,000 trace triggers
- Experiment Editor allows user to create or modify experiments by:
  Edit Delete
  Compile Store
  Load Specify Opcode Class
- Opcode class is collection of various 8051 instructions that make up a set.
  Set is user defined
  Opcode class editor allows user to create, delete or edit Opcode classes

Examine/Modify Memory

- Program memory operations:
  Disassemble
  Single line assemble
  Examine/modify raw data
  Mapping
- Microcontroller internal and external data memory operations:
  Dump
  Scan and modify
  Fill
  Move
  Search
  Compare
  Mapping
  Examine/modify addressable bits

Macro

- Repetitive routine
- User created, edited and callable at any time

High Level Language

- 'C' and PLM support
- Variables and Line number break/trace triggers
- Variables Accessible
- Step - Proceed and Step - Trace of Line numbers

Symbolic Debug

- User and Pre-Defined Symbols
- Supports MetaLink, Enertec, IAR, Archimedes, Signetics, Microtec Research or Intel OMF files
- Use a name not address to alter content of:
  bits, bytes, code

Electrical Specification

Input Power (typical):
1.5 amps @ + 5 volts DC + / − 5%

Mechanical Specification

Emulator dimensions:
1.0″ × 7.0″ × 5.5″
2.5cm × 17.8cm × 14.0cm
Target system cable length:
14.0″
35.6cm
Emulator and target system cable weight:
2.0 lbs.
0.9 kg
Emulator probe head:
Compatible with 600 mil wide 40 pin DIP on 100 mil centers

Host Specification

An IBM PC, PC XT, PC AT or 100% compatible system with 640K bytes of RAM.
PC-DOS 2.0 or later.
Two (2) floppy disk drives.
One (1) RS232C interface card for the PC and cable.

Warranty

Ninety (90) days limited warranty, parts and labor.

Signetics
a division of North American Philips Corporation

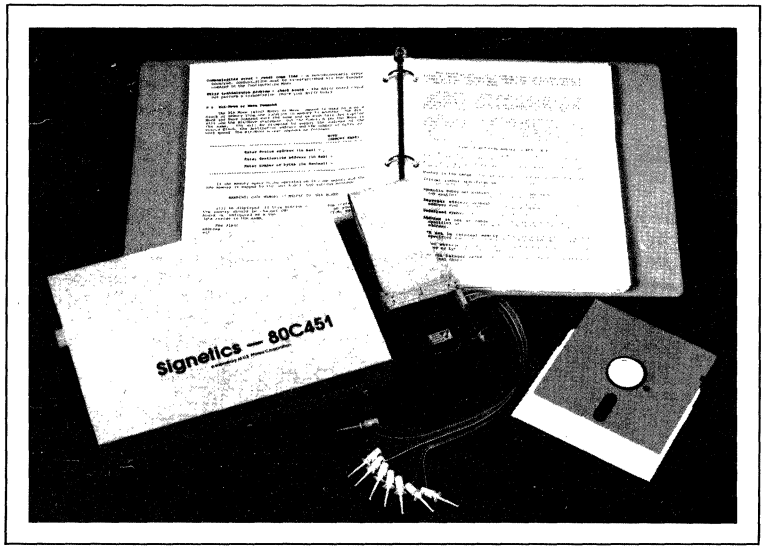
Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

Signetics

Microcontroller Support Tools

System Overview

Signetics-80C451 In-Circuit Emulator for 80C451 Microcontroller



- Serially linked to IBM PC or 100% compatible hosts
- Advanced menu driven human interface
- Real time and transparent emulation up to 12 MHz
- Disassembler and Single Line Assembler
- Examine/Modify Memory capabilities
- 16 Break and Trace Trigger Conditions
- Supports 80C451
- 9 Probe Clips
  7 External Events
  1 External Trigger Input
  1 External Trigger Output
- Opcode Class Editor
- Over 128,000 Break and Trace Triggers
- Emulation Memory:
  — 64K Program
  — 64K External Data
- Full Symbolic Debug Capability
- Up to 64K Pass Counts
- Separate Program and Data Memory Mapping in 16 byte blocks
- Experiment Editor/Compiler
- Trace with 4K frames

PHILIPS

PHILIPS

98-8270-070
1034-000/988/IM

Functional Description

The Signetics-80C451 is an in-circuit emulator which is designed for use in developing, testing and debugging designs based on an 80C451 single chip microcontroller. The Signetics-80C451 allows the development of hardware and software designs to occur simultaneously.

The Signetics-80C451 emulator assists in the following design phases:

- Software Development
- Manufacturing
- Integration of target software and system hardware
- Field Service

Support

The Signetics-80C451 emulator not only assists the designer in developing, testing and debugging 80C451 microcontroller designs, but is also backed by Signetics with an extensive service and support policy that includes:

- Updates to the Signetics User's Manual and Signetics host Operating System software that are provided at no charge for a ninety (90) day period.
- Toll free phone number (1-800-A-HLP-4-51) to resolve system hardware and software concerns.
- A 72-hour repair on disabled systems under warranty.

Features Description

Emulator Functions

- Microcontrollers supported:
  80C451
- Performance
  Real-time .5 MHz to 12 MHz
- Transparency:
  Operational and electrical
- Examine and modify:
  Program
  Internal data
  External data

User Interface

- Advanced menu driven operating system
- 11 functional capabilities:

Load	OS-Escape
Upload	Help
Download	Configure
Store	Restore
Interrogate	Exit
Macro	

Interrogation

- Allows the user to:
  Run experiments
  Examine the system status
  Set break and trace triggers
  Examine/modify data
- 15 functional capabilities:
  Run
  Single-step
  Reset
  OS-Escape
  Set pass count
  Set simple break and trace triggers
  Set repetition counter
  Set phantom break and trace triggers
  Set trace triggers (start, end & center)
  View up to 4K of trace buffer
  Examine/modify:
    Special Function Registers
    Internal data memory
    External data memory
    Program memory
    Emulator experiments

Experiment

- Used to specify break and trace triggers
- Uses the if-then construct
- Allowable trigger conditions are:
  PC address
  PC address range
  Opcode value
  Opcode class
  Special function registers
  Direct byte address
  Direct byte address range
  Direct bit address
  Direct bit address range
  Immediate operand value
  Read/write to bit or direct addresses
  External data address
  External data address range
  Logical AND or OR of any of the above
  Pass count overflow
  External input
- Over 128,000 break and trace triggers
- Experiment Editor allows user to create or modify experiments by:

Edit	Delete
Compile	Store
Load	Specify Opcode Class

- Opcode class is collection of 80C451 instructions that make up a set.
  Set is user defined
  Opcode class editor allows user to create, delete or edit Opcode classes

Examine/Modify Memory

- Program memory operations:
  Disassemble
  Single line assemble
  Examine/modify raw data
  Mapping
- Microcontroller internal and external data memory operations:
  Dump
  Scan and modify
  Fill
  Move
  Search
  Compare
  Mapping
  Examine/modify addressable bits

Macro

- Repetitive routine
- User created, edited and callable at any time

Symbolic Debug

- User and Pre-Defined Symbols
- Supports MetaLink, Enertec, IAR, Archimedes, Signetics, Microtec Research or Intel OMF files
- Use a name not address to alter content of:
  bits, bytes, code

Optional Products

- MetaWARE™ Converter compatible with 900 mil wide 64 pin DIP on 100 mil centers.

Electrical Specification

Input Power (typical):
1 amp @ +25 volts DC +/−5%

Mechanical Specification

Emulator dimensions:
2.0″ × 11.0″ × 7.62″
5.1 cm × 27.9 cm × 19.3 cm
Target system cable length:
14.0″
35.6 cm
Emulator and target system cable weight:
5.0 lbs.
2.2 kg
Emulator probe head:
Compatible with 68 lead PLCC (J-Bend) on 50 mil centers

Host Specification

An IBM PC, PC XT, PC AT or 100% compatible system with 640K bytes of RAM.
PC-DOS 2.0 or later.
Two (2) floppy disk drives.
One (1) RS232C interface card for the PC and cable

Warranty

Ninety (90) days limited warranty, parts and labor.

Ordering Information

Order Code: SMI-80C451SD

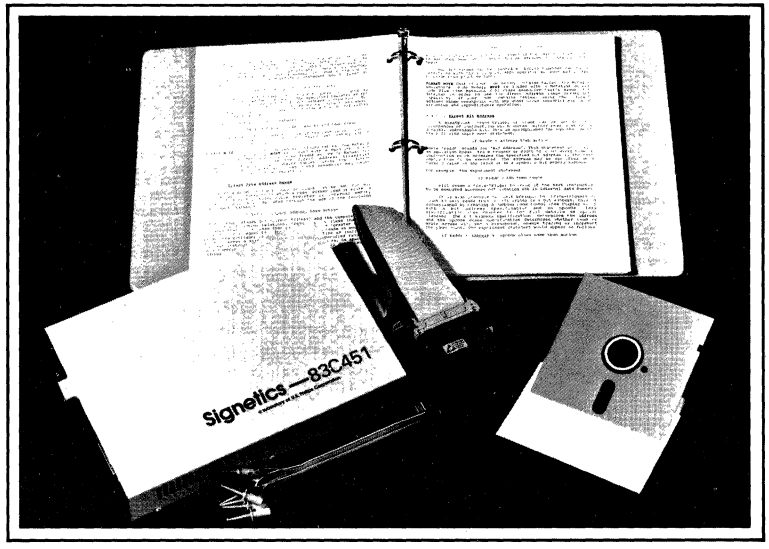
Signetics
a division of North American Philips Corporation
Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

Signetics

Microcontroller Support Tools

System Overview

Signetics-83C451 In-Circuit Emulator for 83C451 Microcontroller



- Serially linked to IBM PC or 100% compatible hosts
- Advanced menu driven human interface
- Real time and transparent emulation up to 12 MHz
- Disassembler and Single Line Assembler
- Examine/Modify Memory capabilities
- 16 Break and Trace Trigger Conditions
- Supports both modes:
  80C451 with External Addresses
  83C451 Single Chip
- 9 Probe Clips
  7 External Events
  1 External Trigger Input
  1 External Trigger Output
- Opcode Class Editor
- Over 128,000 Break and Trace Triggers
- Emulation Memory:
  — 64K Program
  — 64K External Data
- Full symbolic Debug Capability
- Up to 64K Pass Counts
- Separate Program and Data Memory Mapping in 16 byte blocks
- Experiment Editor/Compiler
- Trace with 4K frames

Functional Description

The Signetics-83C451 is an in-circuit emulator which is designed for use in developing, testing and debugging designs based on an 83C451 single chip microcontroller. The Signetics-83C451 allows the development of hardware and software designs to occur simultaneously. The Signetics-83C451 emulator assists in the following design phases:

- Software Development
- Manufacturing
- Integration of target software and system hardware
- Field Service

Support

The Signetics-83C451 emulator not only assists the designer in developing, testing and debugging 83C451 microcontroller designs, but is also backed by Signetics with an extensive service and support policy that includes:

- Updates to the Signetics User's Manual and Signetics host Operating System software that are provided at no charge for a ninety (90) day period.
- Toll free phone number (1-800-A-HLP-4-51) to resolve system hardware and software concerns.
- A 72-hour repair on disabled systems under warranty.

Features Description

Emulator Functions

- Microcontrollers supported:
  83C451 80C451 87C451
- Performance
  Real-time .5 MHz to 12 MHz
- Transparency:
  Operational and electrical
- Examine and modify:
  Program
  Internal data
  External data

User Interface

- Advanced menu driven operating system
- 11 functional capabilities:

Load	OS-Escape
Upload	Help
Download	Configure
Store	Restore
Interrogate	Exit
Macro	

Mode

- Allows user to emulate all the operating modes of the 83C451:
  Full 80C451 operation with external address bus
  Single-chip 83C451 with NO external address bus and 4K of on-chip code memory

Interrogation

- Allows the user to:
  Run experiments
  Examine the system status
  Set break and trace triggers
  Examine/modify data
- 15 functional capabilities:
  Run
  Single-step
  Reset
  OS-Escape
  Set pass count
  Set simple break and trace triggers
  Set repetition counter
  Set phantom break and trace triggers
  Set trace triggers (start, end & center)
  View up to 4K of trace buffer
  Examine/modify:
    Special Function Registers
    Internal data memory
    External data memory
    Program memory
    Emulator experiments

Experiment

- Used to specify break and trace triggers
- Uses the if-then construct
- Allowable trigger conditions are:
  PC address
  PC address range
  Opcode value
  Opcode class
  Special function registers
  Direct byte address
  Direct byte address range
  Direct bit address
  Direct bit address range
  Immediate operand value
  Read/write to bit or direct addresses
  External data address
  External data address range
  Logical AND or OR of any of the above
  Pass count overflow
  External input
- Over 128,000 break and trace triggers
- Experiment Editor allows user to create or modify experiments by:

Edit	Delete
Compile	Store
Load	Specify Opcode Class

- Opcode class is collection of 83C451 instructions that make up a set.
  Set is user defined
  Opcode class editor allows user to create, delete or edit Opcode classes

Examine/Modify Memory

- Program memory operations:
  Disassemble
  Single line assemble
  Examine/modify raw data
  Mapping
- Microcontroller internal and external data memory operations:
  Dump
  Scan and modify
  Fill
  Move
  Search
  Compare
  Mapping
  Examine/modify addressable bits

Macro

- Repetitive routine
- User created, edited and callable at any time

Symbolic Debug

- User and Pre-Defined Symbols
- Supports MetaLink, Enertec, IAR, Signetics, Archimedes, Microtec Research or Intel OMF files
- Use a name not address to alter content of:
  bits, bytes, code

Optional Products

- Converter compatible with 900 mil wide 64 pin DIP on 100 mil centers.

Electrical Specification

Input Power (typical):
1 amp @ + 23 volts DC + / − 5%

Mechanical Specification

Emulator dimensions:
2.0″ × 11.0″ × 7.62″
5.1cm × 27.9cm × 19.3cm
Target system cable length:
14.0″
35.6cm
Emulator and target system cable weight:
5.0 lbs.
2.2 kg
Emulator probe head:
Compatible with 68 lead plastic leaded chip carrier (J-bend) on 50 mil centers

Host Specification

An IBM PC, PC XT, PC AT or 100% compatible system with 640K bytes of RAM.
PC-DOS 2.0 or later.
Two (2) floppy disk drives.
One (1) RS232C interface card for the PC and cable.

Warranty

Ninety (90) days limited warranty, parts and labor.

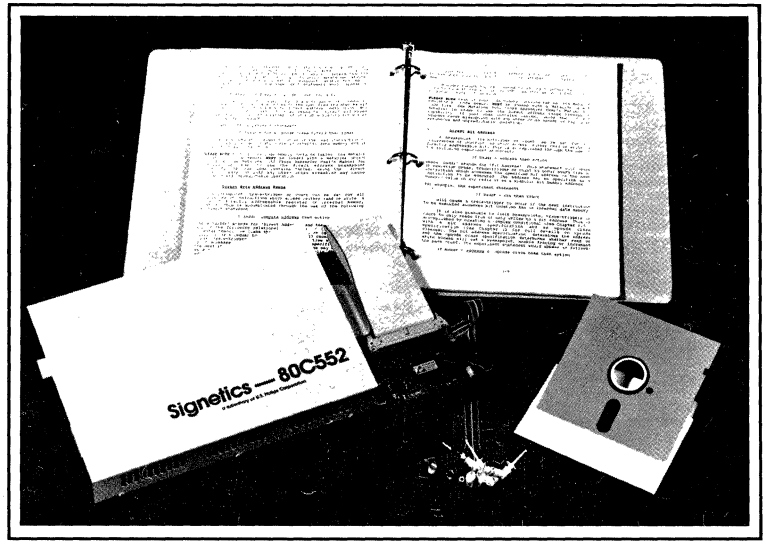
Signetics
a division of North American Philips Corporation
Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

Signetics

Microcontroller Support Tools

System Overview

Signetics-80C552 In-Circuit Emulator for 80C552 Microcontroller



- Serially linked to IBM PC or 100% compatible hosts
- Advanced menu driven human interface
- Real time and transparent emulation up to 12 MHz
- Disassembler and Single Line Assembler
- Examine/Modify Memory capabilities
- 16 Break and Trace Trigger Conditions
- Supports 80C552
- 9 Probe Clips
  7 External Events
  1 External Trigger Input
  1 External Trigger Output
- Opcode Class Editor
- Over 128,000 Break and Trace Triggers
- Emulation Memory:
  — 64K Program
  — 64K External Data
- Full symbolic Debug Capability
- Up to 64K Pass Counts
- Separate Program and Data Memory Mapping in 16 byte blocks
- Experiment Editor/Compiler
- Trace with 4K frames

PHILIPS

PHILIPS

98-8270-060
1030-000/688/IM

Functional Description

The Signetics-80C552 is an in-circuit emulator which is designed for use in developing, testing and debugging designs based on an 80C552 single chip microcontroller. The Signetics-80C552 allows the development of hardware and software designs to occur simultaneously. The Signetics-80C552 emulator assists in the following design phases:

- Software Development
- Manufacturing
- Integration of target software and system hardware
- Field Service

Support

The Signetics-80C552 emulator not only assists the designer in developing, testing and debugging 80C552 microcontroller designs, but is also backed by Signetics with an extensive service and support policy that includes:

- Updates to the Signetics User's Manual and Signetics host Operating System software that are provided at no charge for a ninety (90) day period.
- Toll free phone number (1-800-A-HLP-4-51) to resolve system hardware and software concerns.
- A 72-hour repair on disabled systems under warranty.

Features Description

Emulator Functions

- Microcontrollers supported:
  80C552
- Performance
  Real-time 3.5 MHz to 12 MHz
- Transparency:
  Operational and electrical
- Examine and modify:
  Program
  Internal data
  External data

User Interface

- Advanced menu driven operating system
- 11 functional capabilities:

Load	OS-Escape
Upload	Help
Download	Configure
Store	Restore
Interrogate	Exit
Macro	

Interrogation

- Allows the user to:
  Run experiments
  Examine the system status
  Set break and trace triggers
  Examine/modify data
- 16 functional capabilities:
  Run
  Single-step
  Reset
  OS-Escape
  Set pass count
  Set simple break and trace triggers
  Set repetition counter
  Set phantom break and trace triggers
  Set trace triggers (start, end & center)
  View up to 4K of trace buffer
  Examine A/D data
  Examine/modify:
    Special Function Registers
    Internal data memory
    External data memory
    Program memory
    Emulator experiments

Experiment

- Used to specify break and trace triggers
- Uses the if-then construct
- Allowable trigger conditions are:
  PC address
  PC address range
  Opcode value
  Opcode class
  Special function registers
  Direct byte address
  Direct byte address range
  Direct bit address
  Direct bit address range
  Immediate operand value
  Read/write to bit or direct addresses
  External data address
  External data address range
  Logical AND or OR of any of the above
  Pass count overflow
  External input
- Over 128,000 break and trace triggers
- Experiment Editor allows user to create or modify experiments by:

Edit	Delete
Compile	Store
Load	Specify Opcode Class

- Opcode class is collection of 80C552 instructions that make up a set.
  Set is user defined
  Opcode class editor allows user to create, delete or edit Opcode classes

Examine/Modify Memory

- Program memory operations:
  Disassemble
  Single line assemble
  Examine/modify raw data
  Mapping
- Microcontroller internal and external data memory operations:
  Dump
  Scan and modify
  Fill
  Move
  Search
  Compare
  Mapping
  Examine/modify addressable bits

Macro

- Repetitive routine
- User created, edited and callable at any time

Symbolic Debug

- User and Pre-Defined Symbols
- Supports MetaLink, Enertec, IAR, Signetics, Archimedes, Microtec Research or Intel OMF files
- Use a name not address to alter content of:
  bits, bytes, code

Optional Products

- Converter

Electrical Specification

Input Power (typical):
1 amp @ + 23 volts DC + / − 5%

Mechanical Specification

Emulator dimensions:
2.0″ × 11.0″ × 7.62″
5.1cm × 27.9cm × 19.3cm
Target system cable length:
14.0″
35.6cm
Emulator and target system cable weight:
5.0 lbs.
2.2 kg
Emulator probe head:
Compatible with 68 lead plastic leaded chip carrier (J-bend) on 50 mil centers

Host Specification

An IBM PC, PC XT, PC AT or 100% compatible system with 640K bytes of RAM.
PC-DOS 2.0 or later.
Two (2) floppy disk drives.
One (1) RS232C interface card for the PC and cable.

Warranty

Ninety (90) days limited warranty, parts and labor.

Signetics
a division of North American Philips Corporation
Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

Signetics

Microcontroller Support Tools

System Overview

Signetics-80C652 In-Circuit Emulator for 80C652 Microcontroller

- Serially linked to IBM PC or 100% compatible hosts
- Advanced menu driven human interface
- Real time and transparent emulation up to 12 MHz
- Disassembler and Single Line Assembler
- Examine/Modify Memory capabilities
- 16 Break and Trace Trigger Conditions
- Supports 80C652
- 9 Probe Clips
  7 External Events
  1 External Trigger Input
  1 External Trigger Output
- Opcode Class Editor
- Over 128,000 Break and Trace Triggers
- Emulation Memory:
  — 64K Program
  — 64K External Data
- Full symbolic Debug Capability
- Up to 64K Pass Counts
- Separate Program and Data Memory Mapping in 16 byte blocks
- Experiment Editor/Compiler
- Trace with 4K frames

PHILIPS

PHILIPS

98-8270-020
1031-000/688/IM

Functional Description

The Signetics-80C652 is an in-circuit emulator which is designed for use in developing, testing and debugging designs based on an 80C652 single chip microcontroller. The Signetics-80C652 allows the development of hardware and software designs to occur simultaneously. The Signetics-80C652 emulator assists in the following design phases:

- Software Development
- Manufacturing
- Integration of target software and system hardware
- Field Service

Support

The Signetics-80C652 emulator not only assists the designer in developing, testing and debugging 80C652 microcontroller designs, but is also backed by Signetics with an extensive service and support policy that includes:

- Updates to the Signetics User's Manual and Signetics host Operating System software that are provided at no charge for a ninety (90) day period.
- Toll free phone number (1-800-A-HLP-4-51) to resolve system hardware and software concerns.
- A 72-hour repair on disabled systems under warranty.

Features Description

Emulator Functions

- Microcontrollers supported:
  80C652
- Performance
  Real-time 3.5 MHz to 12 MHz
- Transparency:
  Operational and electrical
- Examine and modify:
  Program
  Internal data
  External data

User Interface

- Advanced menu driven operating system
- 11 functional capabilities:

Load	OS-Escape
Upload	Help
Download	Configure
Store	Restore
Interrogate	Exit
Macro	

Interrogation

- Allows the user to:
  Run experiments
  Examine the system status
  Set break and trace triggers
  Examine/modify data
- 15 functional capabilities:
  Run
  Single-step
  Reset
  OS-Escape
  Set pass count
  Set simple break and trace triggers
  Set repetition counter
  Set phantom break and trace triggers
  Set trace triggers (start, end & center)
  View up to 4K of trace buffer
  Examine/modify:
  Special Function Registers
  Internal data memory
  External data memory
  Program memory
  Emulator experiments

Experiment

- Used to specify break and trace triggers
- Uses the if-then construct
- Allowable trigger conditions are:
  PC address
  PC address range
  Opcode value
  Opcode class
  Special function registers
  Direct byte address
  Direct byte address range
  Direct bit address
  Direct bit address range
  Immediate operand value
  Read/write to bit or direct addresses
  External data address
  External data address range
  Logical AND or OR of any of the above
  Pass count overflow
  External input
- Over 128,000 break and trace triggers
- Experiment Editor allows user to create or modify experiments by:

Edit	Delete
Compile	Store
Load	Specify Opcode Class

- Opcode class is collection of 80C652 instructions that make up a set.
  Set is user defined
  Opcode class editor allows user to create, delete or edit Opcode classes

Examine/Modify Memory

- Program memory operations:
  Disassemble
  Single line assemble
  Examine/modify raw data
  Mapping
- Microcontroller internal and external data memory operations:
  Dump
  Scan and modify
  Fill
  Move
  Search
  Compare
  Mapping
  Examine/modify addressable bits

Macro

- Repetitive routine
- User created, edited and callable at any time

Symbolic Debug

- User and Pre-Defined Symbols
- Supports MetaLink, Enertec, IAR, Signetics, Archimedes, Microtec Research or Intel OMF files
- Use a name not address to alter content of:
  bits, bytes, code

Electrical Specification

Input Power (typical):
1 amp @ + 23 volts DC + / − 5%

Mechanical Specification

Emulator dimensions:
2.0″ × 11.0″ × 7.62″
5.1cm × 27.9cm × 19.3cm
Target system cable length:
14.0″
35.6cm
Emulator and target system cable weight:
5.0 lbs.
2.2 kg
Emulator probe head:
Compatible with 40 lead DIP on 100 mil centers

Host Specification

An IBM PC, PC XT, PC AT or 100% compatible system with 640K bytes of RAM.
PC-DOS 2.0 or later.
Two (2) floppy disk drives.
One (1) RS232C interface card for the PC and cable.

Warranty

Ninety (90) days limited warranty, parts and labor.

Signetics

a division of North American Philips Corporation

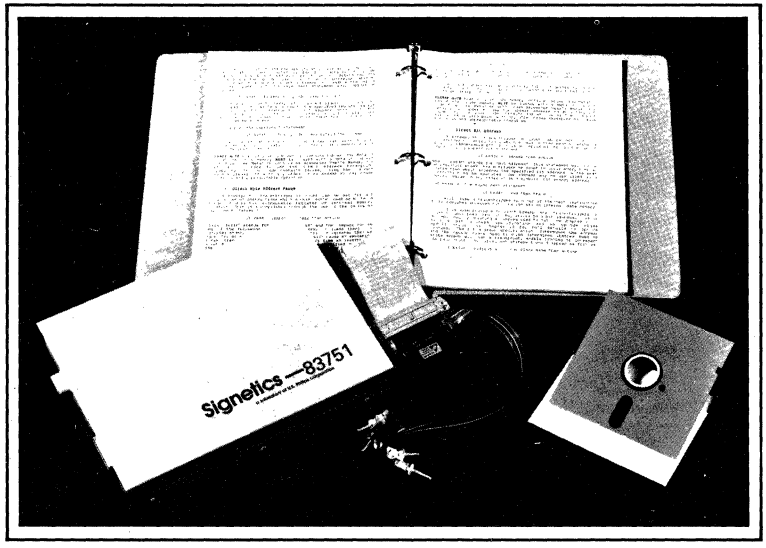
Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

Signetics

Microcontroller Support Tools

System Overview

Signetics-83C751 In-Circuit Emulator for 83C751 Microcontroller



- Serially linked to IBM PC or 100% compatible hosts
- Advanced menu driven human interface
- Real time and transparent emulation up to 16 MHz
- Disassembler and Single Line Assembler
- Examine/Modify Memory capabilities
- 14 Break and Trace Trigger Conditions
- Supports 83C751
- 9 Probe Clips
  7 External Events
  1 External Trigger Input
  1 External Trigger Output
- Opcode Class Editor
- Break and Trace Triggers
- Emulation Memory:
  — 2K Program
- Full symbolic Debug Capability
- Up to 64K Pass Counts
- Program Mapping in 16 byte blocks
- Experiment Editor/Compiler
- Trace with 4K frames

PHILIPS

PHILIPS

98-8270-040
1028-000/688/IM

Functional Description

The Signetics-83C751 is an in-circuit emulator which is designed for use in developing, testing and debugging designs based on an 83C751 single chip microcontroller. The Signetics-83C751 allows the development of hardware and software designs to occur simultaneously. The Signetics-83C751 emulator assists in the following design phases:

- Software Development
- Manufacturing
- Integration of target software and system hardware
- Field Service

Support

The Signetics-83C751 emulator not only assists the designer in developing, testing and debugging 83C751 microcontroller designs, but is also backed by Signetics with an extensive service and support policy that includes:

- Updates to the Signetics User's Manual and Signetics host Operating System software that are provided at no charge for a ninety (90) day period.
- Toll free phone number (1-800-A-HLP-4-51) to resolve system hardware and software concerns.
- A 72-hour repair on disabled systems under warranty.

Features Description

Emulator Functions

- Microcontrollers supported:
  83C751, 87C751
- Performance
  Real-time 3.5 MHz to 16 MHz
- Transparency:
  Operational and electrical
- Examine and modify:
  Program
  Internal data

User Interface

- Advanced menu driven operating system
- 11 functional capabilities:

Load	OS-Escape
Upload	Help
Download	Configure
Store	Restore
Interrogate	Exit
Macro	

Interrogation

- Allows the user to:
  Run experiments
  Examine the system status
  Set break and trace triggers
  Examine/modify data
- 14 functional capabilities:
  Run
  Single-step
  Reset
  OS-Escape
  Set pass count
  Set simple break and trace triggers
  Set repetition counter
  Set phantom break and trace triggers
  Set trace triggers (start, end & center)
  View up to 4K of trace buffer
  Examine/modify:
    Special Function Registers
    Internal data memory
    Program memory
    Emulator experiments

Experiment

- Used to specify break and trace triggers
- Uses the if-then construct
- Allowable trigger conditions are:
  PC address
  PC address range
  Opcode value
  Opcode class
  Special function registers
  Direct byte address
  Direct byte address range
  Direct bit address
  Direct bit address range
  Immediate operand value
  Read/write to bit or direct addresses
  Logical AND or OR of any of the above
  Pass count overflow
  External input
- Break and trace triggers
- Experiment Editor allows user to create or modify experiments by:

Edit	Delete
Compile	Store
Load	Specify Opcode Class

- Opcode class is collection of 83C751 instructions that make up a set.
  Set is user defined
  Opcode class editor allows user to create, delete or edit Opcode classes

Examine/Modify Memory

- Program memory operations:
  Disassemble
  Single line assemble
  Examine/modify raw data
  Mapping
- Microcontroller internal memory operations:
  Dump
  Scan and modify
  Fill
  Move
  Search
  Compare
  Examine/modify addressable bits

Macro

- Repetitive routine
- User created, edited and callable at any time

Symbolic Debug

- User and Pre-Defined Symbols
- Supports MetaLink, Enertec, IAR, Signetics, Archimedes, Microtec Research or Intel OMF files
- Use a name not address to alter content of:
  bits, bytes, code

Electrical Specification

Input Power (typical):
1 amp @ + 23 volts DC + / − 5%

Mechanical Specification

Emulator dimensions:
2.0″ × 11.0″ × 7.62″
5.1cm × 27.9cm × 19.3cm
Target system cable length:
14.0″
35.6cm
Emulator and target system cable weight:
5.0 lbs.
2.2 kg
Emulator probe head:
Compatible with 24 lead 300 mil wide DIP on 100 mil centers

Host Specification

An IBM PC, PC XT, PC AT or 100% compatible system with 640K bytes of RAM.
PC-DOS 2.0 or later.
Two (2) floppy disk drives.
One (1) RS232C interface card for the PC and cable.

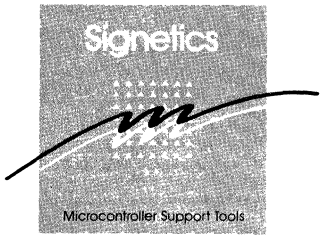
Warranty

Ninety (90) days limited warranty, parts and labor.

Signetics

a division of North American Philips Corporation

Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000



System Overview

Signetics-ASM51 8051 Macro Cross Assembler

- IBM PC or 100% Compatible Host
- Supports All Members of the MCS-51 Family
- Supports All 5 Memory Spaces
- Uses Standard Mnemonics and Syntax
- Generates Intel HEX and MetaLink or Signetics Debug Format for use with Signetics emulators
- Conditional Assembly and Full Macro Capability including Nesting
- Symbolic Access to Predefined Hardware Registers
- Fast Assembler Execution Time
- Full Range of Assembly Time Operators, Complete Listing and Output Controls and choice of Radix
- INCLUDE Statement Allows Development of Code In Modules

Functional Description

The Signetics-ASM51 Macro Cross Assembler takes an assembly language source file created with a text editor and translates it into a machine language object file. This translation process is done in two passes over the source file. The Signetics-ASM51 Macro Cross Assembler is supported on IBM PCs and as such has faster assembly times than traditional methods. The Signetics-ASM51 Macro Cross Assembler supports modular code development or will assemble previously developed code modules through the use of the INCLUDE capability that brings together these code modules at assembly time.

Features Description

Products Supported

8052	8051	80C51
8032	8031	80C31
		87C51
83C751	80C451	
87C751	83C451	
	87C451	
80C552	80C652	
83C552	83C652	
87C552	87C652	

Symbols

- Symbols can be up to 255 characters long, with the first 32 being significant
- Symbols character set include:
  ? and ___ (underline)
  A . . . Z
  a . . . z
  0 . . . 9

Numbers

- Numbers can be entered in decimal (default), binary, hexadecimal or octal.

Predefined Addresses

- All MCS-51 architecturally defined Special Function Registers (SFR) are symbolically defined in the Signetics-ASM51 Macro Cross Assembler.

Memory Spaces Supported

- Code, Data, Bit, External and Indirect.

Instructions Supported

- Standard mnemonics plus generic CALL/JMP.

Assembly Time Operators

- Operations supported are: +, −, HIGH, LOW, MOD, /, *, SHR, SHL, NOT, AND, OR, XOR, =, <, >, <>, <=, >=.
- Operations are done in 16-bit 2's complement arithmetic.

Listing Controls

- Controls supported: Title, Date, List, Nolist, Paging, Nopaging, Eject, Pagelength and Pagewidth.

Output Controls

- Controls supported: Object, Noobject, Print, Noprint, Symbols, Nosymbols and Debug.

Object File Format

- Standard Intel Hexadecimal Object Code Format.
- MetaLink or Signetics Debug Format for use with Signetics emulators.

Include Capability

- Any number of files can be included in the source file, nested up to eight (8) levels deep.

Conditional Assembly

- IF-THEN-ELSE conditional capability, nested up to 255 levels.

Macro Capability

- Full macro capability exists with up to nine (9) levels of nesting.
- Up to 16 parameters can be specified in a macro.

Assembler Type

- Two (2) pass, absolute assembler.

Minimum System Requirements

An IBM PC, PC XT, PC AT or 100% compatible system with 96K bytes of RAM.
PC-DOS 2.0 or later.
One (1) floppy disk drive.

Warranty

Ninety (90) days free update service.

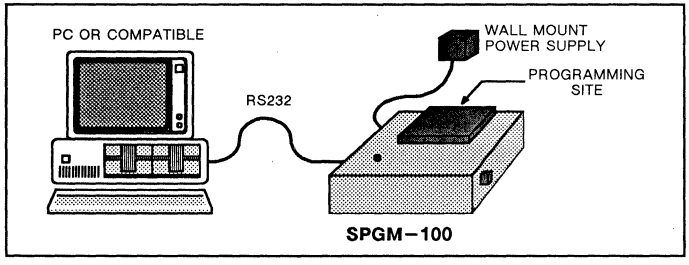
Signetics
a division of North American Philips Corporation
Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

Universal EPROM Programmer

SPGM-100 EPROM Microcontroller and Standard EPROM Programmer

FEATURES

- Supports standard EPROMs
- Modules Available for Microcontroller Support
  - 87C751
  - 87C752
  - Others to follow
- Connects to IBM PC or 100% Compatibles
- Uses standard, intelligent and quick pulse programming algorithms
- 8K x 8 Data Buffer Standard (32K, 64K Optional)
- Supports Intel Hex Format
- System Includes Power Supply, RS-232 Cable, System Software, Documentation and Programmer



SPGM–100 System Block Diagram

OPERATIONS

- Select EPROM Type
- Blank Check
- Program EPROM array
- Display/Alter Data Buffer
- Fill Buffer with Constant
- Copy EPROM to Buffer
- Verify EPROM Versus Buffer
- Error Display (Program/Verify)
- Help

Signetics

Signetics

Microprocessor Products

Section 6
Additional Microcontroller Data Sheets

INDEX

Signetics

8X305
Microcontroller

Product Specification

Microprocessor Products

FEATURES

- **Fetch, Decode, and Execute a 16-bit instruction in a minimum of 200 ns (one machine cycle)**
- **Bit-oriented instruction set (addressable single-or-multiple bit subfields)**
- **Separate buses for Instruction, Instruction Address and Three-State I/O**
- **Thirteen 8-bit general-purpose working registers**
- **Source/destination architecture**
- **Bipolar low-power Schottky technology/TTL inputs and outputs**
- **On-chip oscillator and timing generation**
- **Single +5V supply**
- **0.9-in. 50-pin DIP**
- **68-Pin PLCC**

PRODUCT DESCRIPTION

The Signetics 8X305 Microcontroller (Figure 1) is a high-speed bipolar microprocessor implemented with low-power Schottky technology. In a single chip, the 8X305 combines speed, flexibility, and a bit-oriented instruction set. These features and other basic characteristics of the chip combine to provide cost-effective solutions for a broad range of applications. The 8X305 is particularly useful in systems that require high-speed bit manipulations — sophisticated controllers, data communications, very fast interface control, and other applications of a similar nature.

The 8X305 can fetch, decode, and execute a 16-bit instruction word in a minimum of 200ns. Within one instruction cycle, the 8-bit data-processing path can be programmed to rotate, mask, shift, and/or merge single or multiple bit subfields and, in addition, perform an ALU operation. In the same instruction, an external data field can be input, processed, and output to a specified destination — likewise, single or multiple bit data fields can be internally moved from a given source to a given destination. To summarize, fixed or variable-length data fields can be fetched, processed, operated on by the ALU, and moved to a different location — all in a timeframe of 200ns. To interface with I/O and program memory, the 8X305 uses a 13-bit instruction address bus, a 16-bit instruction bus, an 8-bit bidirectional multiplexed I/O data/address bus and a 5-bit I/O control bus.

A wide selection of I/O devices, interface chips, and special-purpose parts are available for systems use. In most applications, the more powerful 8X305 is functionally interchangeable with its predecessor — the 8X300.

ASSOCIATED DOCUMENTATION

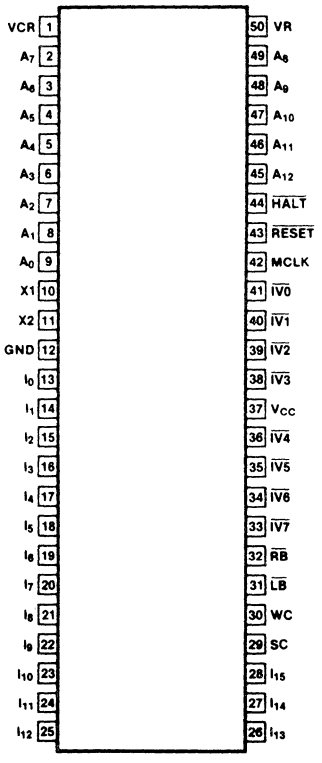
Other documents directly relating to *design* and *applications use* of the 8X305 Microcontroller are:

- Product Capabilities Manual
- 8X305 Users Manual

These documents and other current literature (Data Sheets, Product Bulletins, Applications Notes, etc.) are available at all Signetics Sales and Service Offices — see rear cover of this data sheet for the office in your locality.

PIN CONFIGURATION

N, I PACKAGES

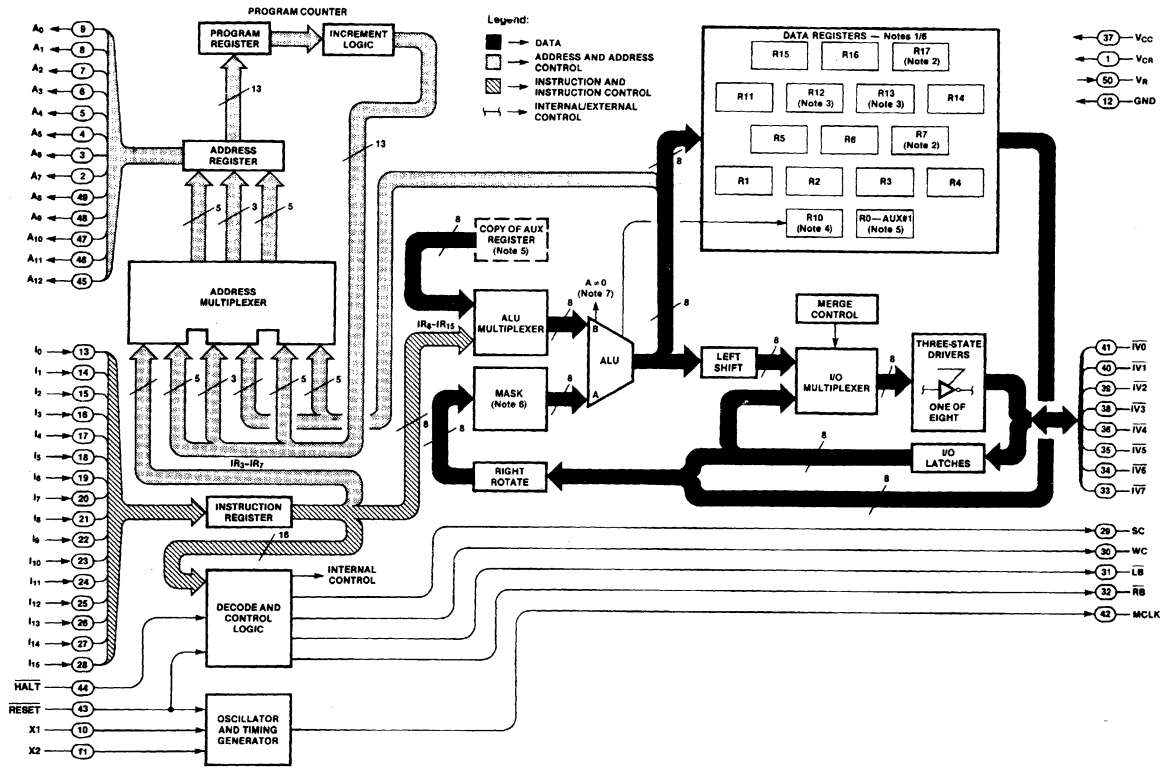


TOP VIEW

CD09451S

ORDERING INFORMATION

DESCRIPTION	ORDER CODE
50-Pin plastic DIP	N8X305N
50-Pin ceramic DIP	N8X305I
68-Pin PLCC	N8X305A

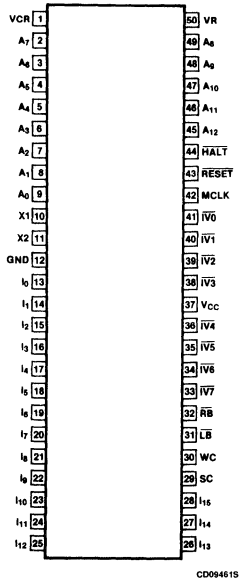


NOTES:

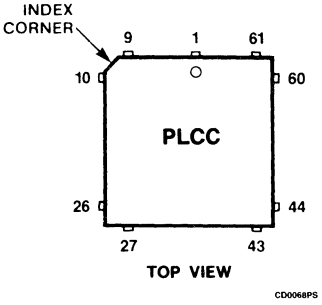
1. Registers R1 – R6, R11 and R14 – R16 are general-purpose working registers.
2. In any instruction where R7 (IVL) or R17 (IVR) is specified as the destination, the 8-bit value is output on the IV bus as an IV device enable address (SC = High) — R7 = left bank and R17 = right bank; the results are also stored into the specified internal register and may later be accessed as source data.
3. R12 and R13 are general-purpose working registers for all operations except transmit (XMIT).
4. The least significant bit of register R10 (OVF) is used to reflect the carryout status resulting from the most recent ADD operation.
5. Auxiliary register R0 #1 is a general-purpose working register that holds the implied operand for Arithmetic and Logical operations; the content of this register is repeated in AUX #2 (shown dotted). The duplicate register is physically part of the ALU and is shown separate only for layout convenience.
6. Internal working registers cannot be operated on by the MASK logic.
7. During NZT instructions the ALU tests for all bits equal to "0" (Transfer if A ≠ 0) — refer to BASIC OPERATIONS that follow.

Figure 1. Architecture and Pin Designations for 8X305 Microcontroller
Pin Numbers Shown are for DIP Packaging

Microcontroller 8X305



CD09461S



CD0068PS

PLCC PIN NO.	DIP PIN NO.	IDENTIFIER	FUNCTION
1, 68	1	VCR	Regulated voltage input from series-pass transistor (2N5320 or equivalent).
4-11, 62-66	2-9, 45-49	$A_0 - A_{12}$	**Program Address Lines:** These active-high outputs permit direct addressing of up to 8192 words of program storage; A_{12} is least significant bit.
12, 13	10, 11	X1, X2	Timing generator connections for a capacitor, a series resonant crystal, or an external clock source with complementary outputs.
2,3, 14-16	12	GND	Ground.
17-23, 28-36	13-28	$I_0 - I_{15}$	**Instruction Lines:** These active-high input lines receive 16-bit instructions from program storage; I_{15} is least significant bit.
37	29	SC	**Select Command:** When high (binary 1), an address is being output on pins $\overline{IV0}$ through $\overline{IV7}$.
38	30	WC	**Write Command:** When high (binary 1), data is being output on pins $\overline{IV0}$ through $\overline{IV7}$.
39	31	$\overline{LB}$	**Left Bank Control:** When low (binary 0), devices connected to the Left Bank are accessed. (Note: *Typically, the $\overline{LB}$ signal is tied to the $\overline{ME}$ input pin of I/O peripherals*).
45	32	RB	**Right Bank Control:** When low (binary 0), devices connected to the Right Bank are accessed (Note. *Typically, the $\overline{RB}$ signal is tied to the $\overline{ME}$ input pin of I/O peripherals*).

PLCC PIN NO.	DIP PIN NO.	IDENTIFIER	FUNCTION
46-49, 55-58	33-36, 38-41	$\overline{IV0}$ – $\overline{IV7}$	**Interface Vector** (Input/Output Bus) — these bidirectional active-low three-state lines communicate data and/or addresses to I/O devices and memory locations. A low voltage level equals a binary "1"; $\overline{IV7}$ is Least Significant Bit.
50-52	37	V_{CC}	+5V power supply.
59	42	MCLK	**Master Clock:** This active-high output signal is used for clocking I/O devices and/or synchronization of external logic.
60	43	$\overline{RESET}$	When **$\overline{RESET}$** input is low (binary 0), the 8X305 is initialized — sets Program Counter/Address Register to zero and inhibits MCLK. For the period of time $\overline{RESET}$ is low, the Left Bank/Right Bank ($\overline{LB}/\overline{RB}$) signals are forced high asynchronously.
61	44	$\overline{HALT}$	When **$\overline{HALT}$** input is low (binary 0), internal operation of the 8X305 stops at the start of next instruction; MCLK is not inhibited nor is any internal register affected. However, both the Left Bank/Right Bank ($\overline{LB}/\overline{RB}$) signals are synchronously driven high during the first quarter of the instruction cycle time and remain high during the time HALT is low.
67	50	VR	Internally-generated reference output voltage for external series-pass regulator transistor.
24-27, 40-44, 53, 54	–	No Connect	

NOTE:
Multiple V_{CC}, GND, and V_{CR} pins must be externally connected.

Figure 2. Designations and Descriptions for Pins of 8X305 Microcontroller.

FUNCTIONAL OPERATION

Typical System Configuration

Although the system hookup shown in Figure 3 is of the simplest form, it provides a fundamental look at the 8X305 Microcontroller and peripheral relationships. As indicated, the 8X305 can directly address up to 8K words of program storage — either ROM or PROM. The user interface ($\overline{IV0}$ through $\overline{IV7}$) is capable of uniquely addressing 256 Input/Output locations and, with additional bank bits ($\overline{LB}$, $\overline{RB}$), this number is expanded to 512 — each bank comprising 256 addressable locations. The addressable locations of each bank can be used in a variety of ways; a simple method of implementation is shown in Figure 3. When $\overline{LB}$ is active low, the left bank is enabled and any one of 256 locations within the RAM memory can be accessed for input/output operations. A similar set of "enable/access" conditions are applicable to the right bank when $\overline{RB}$ is active low.

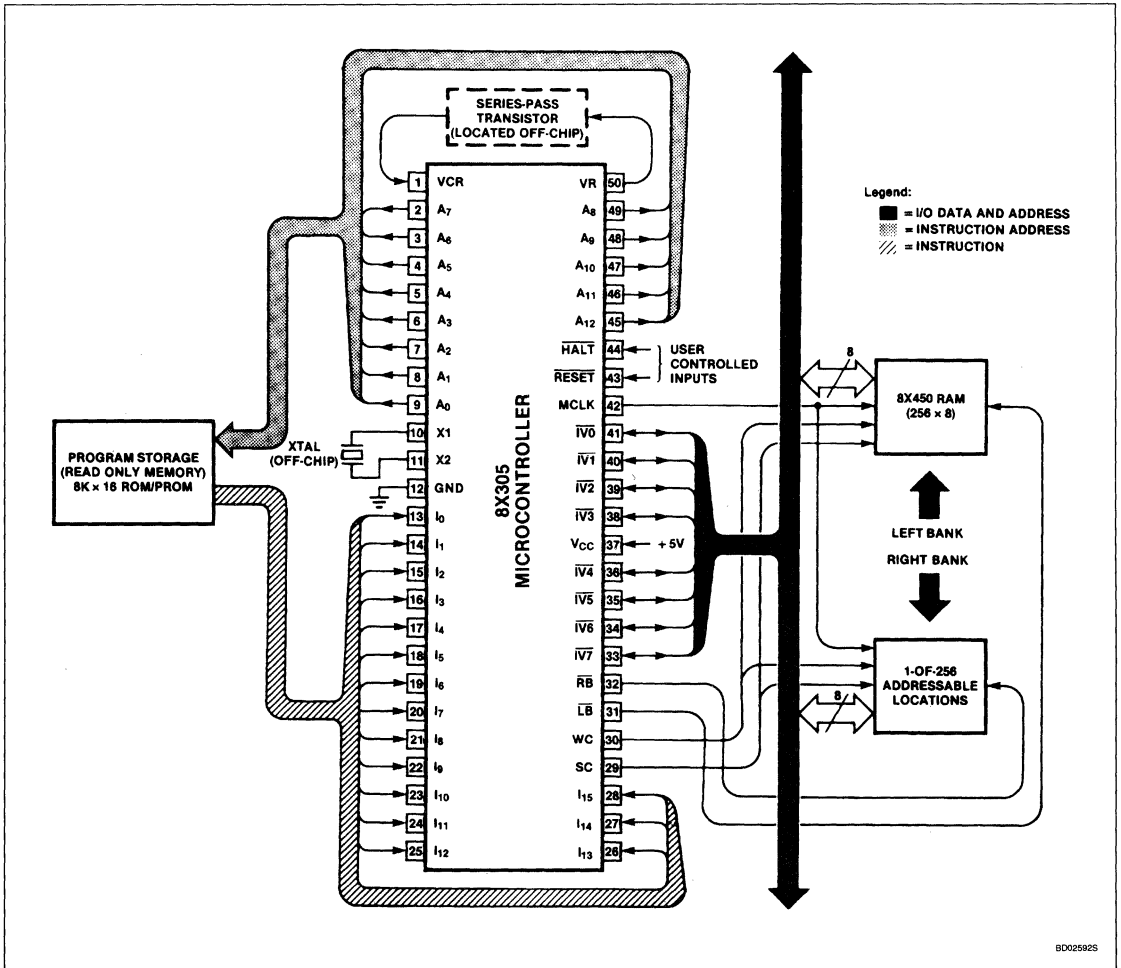


Figure 3. Typical 8X305 System Hookup

BASIC OPERATIONS OF 8X305

Refer to a later discussion of "Instruction Fields" for a detailed examination of all operand fields and subdivisions thereof — "S" (S_0, S_1), "D" (D_0, D_1), "R", "L", "J", and "A".

MOVE OPERATIONS

REGISTER-TO-REGISTER

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
R0-R17 as specified by "S" field of instruction.	Right rotate as specified by "R" field of instruction.	No operation.	No operation.	R0-R7, R11-R17 as specified by "D" field of instruction.

REGISTER-TO-$\overline{IV}$ BUS

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
R0-R17 as specified by "S" field of instruction.	No operation.	No operation.	Shift and merge as specified by "D_0" and "L" fields of instruction.	Variable length field of $\overline{IV}$ bus — Left Bank ($\overline{LB}$) or Right Bank ($\overline{RB}$) as specified by "D_0" and "D_1" fields of instruction.

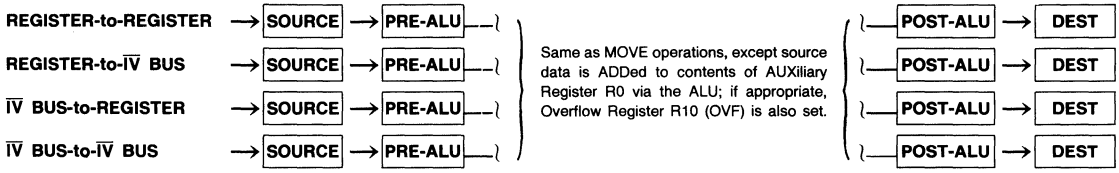
$\overline{IV}$ BUS-TO-REGISTER

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
Variable length field of $\overline{IV}$ bus — Left Bank ($\overline{LB}$) or Right Bank ($\overline{RB}$) as specified by "S_1" and "S_0" fields of instruction.	Right rotate and mask as specified by "S_0" and "L" fields of instruction.	No operation.	No operation.	R0-R7, R11-R17 as specified by "D" field of instruction.

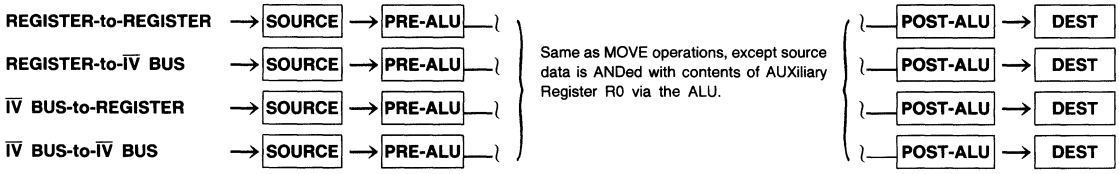
$\overline{IV}$ BUS-TO-$\overline{IV}$ BUS

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
Variable length field of $\overline{IV}$ bus — Left Bank ($\overline{LB}$) or Right Bank ($\overline{RB}$) as specified by "S_1" and "S_0" fields of instruction.	Right rotate and mask as specified by "S_0" and "L" fields of instruction.	No operation.	Shift and merge as specified by "D_0" and "L" fields of instruction.	Variable length field of $\overline{IV}$ bus — Left Bank ($\overline{LB}$) or Right Bank ($\overline{RB}$) as specified by "D_0" and "D_1" fields of instruction.

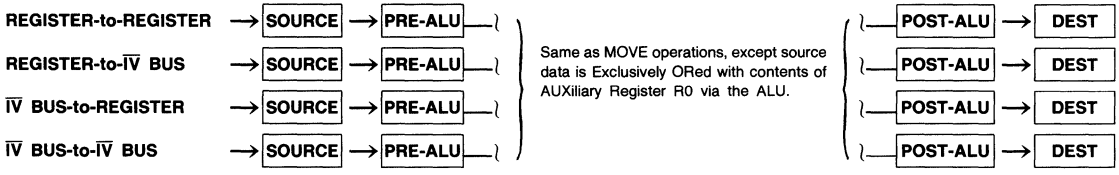
ADD OPERATIONS



AND OPERATIONS



EXCLUSIVE OR (XOR) OPERATIONS



EXECUTE (XEC) OPERATIONS

XEC, REGISTER

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
R0 – R17 as specified by "S" field of instruction.	No operation.	Add source data to 8-bit field specified by instruction literal ($0 \leqslant J \leqslant 377_8$).	No operation.	Replace 8 LSB of Address Register* with 8-bit sum from ALU. *PGM CTR unchanged.

XEC, $\overline{IV}$ BUS

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
Left or Right Bank of $\overline{IV}$ bus as specified by "S" field of instruction.	Rotate and mask as specified by "S_0" and "L" fields of instruction.	Add masked field of source data to 5-bit literal specified by "J" field of instruction ($0 \leqslant J \leqslant 37_8$).	No operation.	Replace 5 LSB of Address Register* with 5-bit sum from ALU. *PGM CTR unchanged.

NON-ZERO TRANSFER (NZT) OPERATIONS

NZT, REGISTER

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DESTINATION
R0-R17 as specified by "S" field of instruction.	No operation.	Test contents of source register for all zeroes.	If S = 0, increment PC by 1; if S ≠ 0, replace 8 LSB of AR and PC with literal specified by "J" field of instruction.

NZT, $\overline{IV}$ BUS

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DESTINATION
Left or Right Bank of $\overline{IV}$ bus as specified by "S" field of instruction.	Rotate and mask as specified by "S_0" and "L" fields of instruction.	Test contents of masked field for all zeroes.	If S = 0, increment PC by 1; if S ≠ 0, replace 5 LSB of AR and PC with literal specified by "J" field of instruction.

TRANSMIT (XMIT) OPERATIONS

XMIT, REGISTER

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
$0 \leq J \leq 377_8$ – value specified by "J" field of instruction.	No operation.	No operation.	No operation.	R0-R7, R11, R14-R17. Load 8-bit integer specified by "J" field into register specified by "D" field.

XMIT 8-BIT IMMEDIATE, $\overline{IV}$ BUS

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
$0 \leq J \leq 377_8$ – value specified by "J" field of instruction.	No operation.	No operation.	No operation.	Left (R12) or Right (R13) Bank of $\overline{IV}$ bus as specified by "D" field of instruction.

XMIT VARIABLE-BIT FIELD IMMEDIATE $\overline{IV}$ BUS

SOURCE	DATA PROCESSING (PRE-ALU)	ALU	DATA PROCESSING (POST-ALU)	DESTINATION
$0 \leq J \leq 37_8$ – value specified by "J" field of instruction.	No operation.	No operation.	Shift and merge source data as specified by "D_0" and "L" fields of instruction.	Left or Right Bank of $\overline{IV}$ bus as specified by "D" field of instruction.

Table 1.

$\overline{LB}$	$\overline{RB}$	FUNCTION
Low	Low	This state is not generated by the 8X305.
Low	High	Enable left bank devices.
High	Low	Enable right bank devices.
High	High	Disable all devices; $\overline{IV}$ bus is 3-State.

Table 2.

$\overline{LB}/\overline{RB}$	SC	WC	FUNCTION
High	Low	Low	The $\overline{IV}$ bus is 3-State and not looking for input data.
Low	Low	Low	The $\overline{IV}$ bus is reading input data.
Low	Low	High	Data is being output.
Low	High	Low	Address is being output.
X	High	High	This condition is never generated.

Program Storage Interface

As shown in Figure 3, program storage is connected to output address lines A_0 through A_{12} (A_{12} = LSB) and input instruction lines I_0 through I_{15}. An address output on A_0/A_{12} identifies one 16-bit instruction word in program storage. The instruction word is subsequently input on I_0/I_{15} and defines the Micro-Controller operation which is to follow — one instruction word equals one completed operation. Any TTL-compatible memory can be used for program storage provided the worst-case access time is compatible with the instruction cycle time used for the application — see timing section for appropriate calculations.

I/O Interface and Control

An 8-bit bidirectional I/O bus, referred to as the Interface Vector ($\overline{IV}$) bus, provides a communication link between the Microcontroller and the two banks of I/O devices. The $\overline{LB}$ (Left Bank) and $\overline{RB}$ (Right Bank) control signals identify which bank is enabled; when both $\overline{LB}$ and $\overline{RB}$ are high (inactive), neither bank is enabled and the $\overline{IV}$ bus is inactive (three-state). A functional analysis of the Left and Right Bank signals is shown in Table 1.

Both data and I/O address information are multiplexed on the $\overline{IV}$ bus. The SC (Select Command) and WC (Write Command) signals distinguish between data and I/O address information as shown in Table 2.

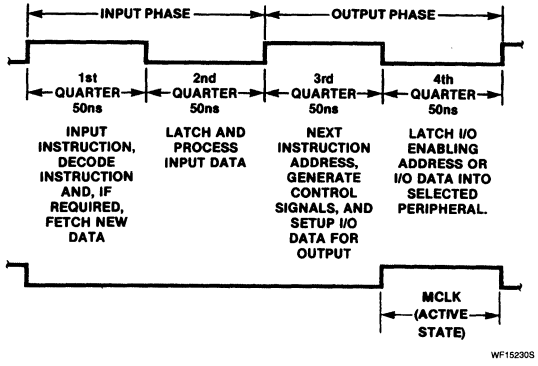
Data Processing

Basically, the data processing path of the 8X305 consists of the Rotate/Mask logic, the Arithmetic Logic Unit (ALU), the Shift/Merge functions, on-chip memory (sixteen 8-bit registers), and the bidirectional $\overline{IV}$ bus interface with its associated driver circuits and internal latches. The on-board memory and the $\overline{IV}$ bus are connected to both inputs and outputs of the ALU via internal 8-bit data paths — see Figure 1. Inputs to the ALU are preceded by right-rotate and data-mask functions; the ALU output is followed by the left-shift and merge operations. Depending on the desired operation, any one or all of the functions (Rotate/Mask/Shift/Merge) can operate on 8 bits of data in a single instruction cycle. For a summary of all data-processing capabilities, refer to BASIC OPERATIONS OF THE 8X305 described earlier in this data sheet.

Instruction Cycle

Each operation of the 8X305 is executed in a single instruction cycle. The instruction cycle is internally divided into four equal parts — each part being as short as 50ns. Figure 4 shows the general functions that occur during each quarter cycle; specifics regarding minimum/maximum timing and other critical values are described later in this data sheet. During the first quarter cycle, a new instruction from program storage is input via $I_0 - I_{15}$ and decoded. If an I/O operation is indicated, new data is fetched from a specified internal register or via the $\overline{IV}$ bus. At the end of the first quarter cycle, the new instruction is latched into the instruction register.

In the second quarter cycle, the I/O input data stabilizes and preliminary processing is completed. At the end of this quarter, the $\overline{IV}$ latches close and final processing can be accomplished, thus completing the input phase of the instruction cycle. During the third quarter cycle, the address for the next instruction is output to the instruction address bus, $\overline{IV}$ control signals are generated, and both data and destination are setup for the remainder of the output phase. During the fourth quarter cycle, a master clock signal (MCLK) generated by the 8X305 is used to latch either the I/O-enabling address or the I/O data into peripheral devices connected to the $\overline{IV}$ bus. MCLK can also be used to synchronize any external logic with timing circuits of the 8X305. To summarize the action, the first half of the instruction cycle deals primarily with input functions and the second half is mostly concerned with output functions.



NOTES:
1. New instruction must be accepted and latched at end of first quarter cycle.
2. The I/O data latches are open for the first two quarter cycles, that is, for 100ns.
3. The address changes during third quarter cycle.
4. $\overline{IV}$ bus drivers are active (turned on) during third and fourth quarter cycles.

Figure 4. Instruction Cycle and MCLK with: Crystal = 10MHz and Cycle Time = 200ns.

Table 3.

	↓ MSB														LSB ↓
BIT POSITIONS →	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	OPCODE			OPERAND(S)												

Table 4.

- **Data Control —**
  - **ADD, AND, XOR** — **Arithmetic and Logic Operations**
  - **MOVE, XMIT** — **Movement of Data and Constants**
- **Program Control —**
  - **XEC, NZT, JMP** — **Branch or Test**

INSTRUCTION SET

General Format and Operating Principles

The 16-bit instruction word (I_0 through I_{15}) from program storage is input to the instruction register (Figure 1) and is subsequently decoded to implement the events to occur during the current instruction cycle.

The general format for each instruction word is shown in Table 3.

The 3-bit operation code (OPCODE) define any one of eight classes of instructions; variations within each class are specified by the remaining thirteen operand bits. The eight instruction classes can be separated into two control areas — *data* and *program*; general functions within these areas are as shown in Table 4.

Instruction Fields

As shown in Table 5, each instruction word consists of an operation code (OPCODE) field and from one to three operand fields. The possible operand fields are: Source (S), Destination (D), Rotate/Length (R/L), Literal (J), and Address (A). The OPCODE and operand fields are described in the paragraphs that follow the table.

Table 5. Functional Description of Instruction Set

INSTRUCTION WORD	DESCRIPTION	STATE OF CONTROL SIGNAL DURING INSTRUCTION CYCLE — SEE FIGURE 4		
		CONTROL SIGNAL	INPUT PHASE	OUTPUT PHASE
CLASS = **MOVE** OPCODE = **0** OPERATION = **(S) → D**				
Register-to-Register 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | R | D $S = 00_8 - 17_8$ $D = 00_8 - 07_8, 11_8 - 17_8$	Move content of internal register specified by S-field to internal register specified by D-field. Prior to the "MOVE" operation, right-rotate contents of internal source register by octal value (0 through 7) defined by the R-field.	SC	L	H if $D = 07_8, 17_8$
		WC	L	L
		$\overline{LB}$	H	L if $D = 07_8$
		$\overline{RB}$	H	L if $D = 17_8$
Register-to-$\overline{IV}$ Bus (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | L | D $D_1 \vdots D_0$ $S = 00_8 - 17_8$ $D = 20_8 - 37_8$	Move contents of internal register specified by the S-field to the $\overline{IV}$ bus. Before outputting on $\overline{IV}$ bus, data is shifted as specified by the least significant octal digit of the D-field and the bits specified by the L-field are merged with the latched I/O data.	SC	L	L
		WC	L	H
		$\overline{LB}$	L if $D = 20_8 - 27_8$	L if $D = 20_8 - 27_8$
		$\overline{RB}$	L if $D = 30_8 - 37_8$	L if $D = 30_8 - 37_8$
$\overline{IV}$ Bus-to-Register (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | L | D $S_1 \vdots S_0$ $S = 20_8 - 37_8$ $D = 00_8 - 07_8, 11_8 - 17_8$	Move right-rotated $\overline{IV}$ bus (source) data specified by the S-field to internal register specified by the D-field. The L-field specifies the length of source data starting from the LSB-position and, if less than 8 bits, the remaining bits are filled with zeros.	SC	L	H if $D = 07_8, 17_8$
		WC	L	L
		$\overline{LB}$	L if $S = 20_8 - 27_8$	L if $D = 07_8$
		$\overline{RB}$	L if $S = 30_8 - 37_8$	L if $D = 17_8$
$\overline{IV}$ Bus-to-$\overline{IV}$ Bus (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | L | D $S_1 \vdots S_0$ | $D_1 \vdots D_0$ $S = 20_8 - 37_8$ $D = 20_8 - 37_8$	Move right-rotated $\overline{IV}$ bus (source) data specified by the S-field to the I/O latches. Before outputting on $\overline{IV}$ bus, shift data as specified by the D-field; then merge source and latched I/O data as specified by the L (length) field.	SC	L	L
		WC	L	H
		$\overline{LB}$	L if $S = 20_8 - 27_8$	L if $D = 20_8 - 27_8$
		$\overline{RB}$	L if $S = 30_8 - 37_8$	L if $D = 30_8 - 37_8$
CLASS = **ADD** OPCODE = **1** OPERATION = **(S) + (AUX) → D**				
Same as MOVE instruction class	Same as MOVE instruction class except that contents of AUX (R0) register are ADDed to the source data. If there is a "carry" from MSB, then R10 (OVF) = 1 (overflow), otherwise OVF = 0.	Same as MOVE instruction class		
CLASS = **AND** OPCODE = **2** OPERATION = **(S) ∧ (AUX) → D**				
Same as MOVE instruction class	Same as MOVE instruction class except that contents of AUX (R0) register are ANDed with source data.	Same as MOVE instruction class		
CLASS = **XOR** OPCODE = **3** OPERATION = **(S) ⊕ (AUX) → D**				
Same as MOVE instruction class	Same as MOVE instruction class except that contents of AUX (R0) register are Exclusively ORed with source data.	Same as MOVE instruction class		
CLASS = **XEC** OPCODE = **4** OPERATION = **Refer to Description**				
Register Immediate 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | J $S = 00_8 - 17_8$ $J = 000_8 - 377_8$	Execute instruction at current page address offset by J (literal) + (S). Return to normal instruction flow unless a branch is encountered. Execute instruction at an address determined by replacing the low-order 8 bits of the Address Register with the following derived sum: Value of literal (J-field) plus contents of internal register specified by S-field The PC is not incremented and the overflow status (OVF) is not changed.	SC	L	L
		WC	L	L
		$\overline{LB}$	H	H
		$\overline{RB}$	H	H
$\overline{IV}$ Bus Immediate (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | L | J $S_1 \vdots S_0$ $S = 20_8 - 37_8$ $J = 00_8 - 37_8$	Execute instruction at an address determined by replacing the low-order 5 bits of Address Register with the following derived sum: 5-bit value of literal (J-field) plus value of rotated source data specified by S-field. The L-field specifies the length of source data starting from the LSB position and, if less than 8 bits, the remaining bits are filled with zeros; the Program Counter is not incremented and the overflow status (OVF) is not changed.	SC	L	L
		WC	L	L
		$\overline{LB}$	L if $S = 20_8 - 27_8$	H
		$\overline{RB}$	L if $S = 30_8 - 37_8$	H

Table 5. Functional Description of Instruction Set (Continued)

INSTRUCTION WORD	DESCRIPTION	STATE OF CONTROL SIGNAL DURING INSTRUCTION CYCLE — SEE FIGURE 4		
		CONTROL SIGNAL	INPUT PHASE	OUTPUT PHASE
CLASS = **NZT** OPCODE = **5** OPERATION = **Refer to Description**				
Register Immediate 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | J $S = 00_8 - 17_8$ $J = 000_8 - 377_8$	If data specified by the S-field is not equal to zero, jump to current page address offset by value of J-field; otherwise, increment the Program Counter. If contents of internal register specified by S-field is non-zero, transfer to address determined by replacing the low-order 8 bits of Address Register and Program Counter with "J", otherwise, increment PC.	SC	L	L
		WC	L	L
		$\overline{LB}$	H	H
		$\overline{RB}$	H	H
$\overline{IV}$ Bus Immediate (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | S | L | J $S_1 \vdots S_0$ $S = 20_8 - 37_8$ $J = 00_8 - 37_8$	If right-rotated and masked IV bus is non-zero, transfer to address determined by replacing low-order 5 bits of Address Register and Program Counter with "J", otherwise, increment PC. (The L-field specifies the length of source I/O data starting from the LSB-position and, if less than 8 bits, the remaining bits are filled with zeros.)	SC	L	L
		WC	L	L
		$\overline{LB}$	L if $S = 20_8 - 27_8$	H
		$\overline{RB}$	L if $S = 30_8 - 37_8$	H
CLASS = **XMIT** OPCODE = **6** OPERATION = **J → D**				
XMIT, Register 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | D | J $D = 00_8 - 06_8, 11_8, 14_8 - 16_8$ $J = 000_8 - 377_8$	Store 8-bit value specified by "J" into register specified by "D".	SC	L	L
		WC	L	L
		$\overline{LB}$	H	H
		$\overline{RB}$	H	H
XMIT, $\overline{IV}$ Bus Address 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | D | J $D = 07_8, 17_8$ $J = 000_8 - 377_8$	Enable I/O device on the bank specified by "D", whose address is the 8-bit integer specified by "J". Address "J" is stored in register "D".	SC	L	H
		WC	L	L
		$\overline{LB}$	H	L if $D = 07_8$
		$\overline{RB}$	H	L if $D = 17_8$
XMIT 8 Bits Immediate, $\overline{IV}$ Bus (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | D | J $D = 12_8 - 13_8$ $J = 000_8 - 377_8$	Store value of 8-bit integer in the previously enabled I/O port, at the bank destination ($\overline{LB}$ or $\overline{RB}$) specified by "D". Contents of R12 or R13 remain unchanged.	SC	L	L
		WC	L	H
		$\overline{LB}$	H	L if $D = 12_8$
		$\overline{RB}$	H	L if $D = 13_8$
XMIT Variable Bit Field Immediate, $\overline{IV}$ Bus (Note) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | D | L | J $D_1 \vdots D_0$ $D = 20_8 - 37_8$ $J = 00_8 - 37_8$	Transmit Least Significant "L" bits of "J" field to "L-bit" field of $\overline{IV}$ bus specified by "D"; if "L" is greater than 5 bits, the MSB bits of destination field is filled with zeros.	SC	L	L
		WC	L	H
		$\overline{LB}$	L if $D = 20_8 - 27_8$	L if $D = 20_8 - 27_8$
		$\overline{RB}$	L if $D = 30_8 - 37_8$	L if $D = 30_8 - 37_8$
CLASS = **JMP** OPCODE = **7** OPERATION = **Refer to Description**				
Address Immediate 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 OPCODE | A $A = 00000_8 - 17777_8$	Jump to address in program storage specified by A-field; this address is loaded into the Address Register and the Program Counter.	SC	L	L
		WC	L	L
		$\overline{LB}$	H	H
		$\overline{RB}$	H	H

NOTE:
S_0 specifies the LSB of rotated input data field
S_1 specifies the bank of $\overline{IV}$ bus from which source data will be input
D_0 specifies bit position in I/O device with which LSB of processed data will be aligned, and
D_1 specifies the bank of $\overline{IV}$ bus which will be the destination.

Table 6. Octal Addresses and Source/Destination Fields for 8X305 Registers

ADDRESS	REGISTER DESIGNATION	SOURCE	DESTI-NATION	ADDRESS	REGISTER DESIGNATION	SOURCE	DESTI-NATION
00_8	R0 (AUX) — General purpose register	X	X	10_8	R10 (OVF — Overflow register)	X	
01_8	R1 — General purpose register	X	X	11_8	R11 — General purpose register	X	X
02_8	R2 — General purpose register	X	X	12_8	R12 — General purpose register (Note)	X	X
03_8	R3 — General purpose register	X	X	13_8	R13 — General purpose register (Note)	X	X
04_8	R4 — General purpose register	X	X	14_8	R14 — General purpose register	X	X
05_8	R5 — General purpose register	X	X	15_8	R15 — General purpose register	X	X
06_8	R6 — General purpose register	X	X	16_8	R16 — General purpose register	X	X
07_8	R7 — Special purpose register (refer to next paragraph)	X	X	17_8	R17 — Special purpose register (refer to next paragraph)	X	X

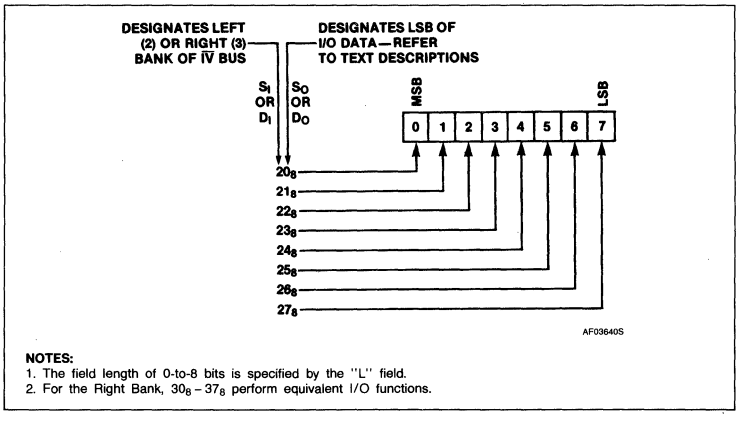
NOTE:
R12 and R13 function as general purpose working registers for all operations except transmit (XMIT). During a transmit instruction where R12 or R13 is the destination, the 8-bit "J" field is immediately transferred to the $\overline{IV}$ bus; for this operation, the contents of the designated register remain unchanged.

Operations Code Field. The 3-bit OPCODE field specifies one of eight classes of 8X305 instructions; octal designations for this field and operands for each instruction class are shown in Table 5.

Source (S) and Destination (D) Fields. The 5-bit "S" and "D" fields specify the source and destination, respectively, for whatever operation is defined by the OPeration CODE. The "S" and/or "D" fields can specify an internal 8X305 register or any one-to-eight bit field within an I/O device; octal values and source/destination field assignments for all internal registers are shown in Table 6.

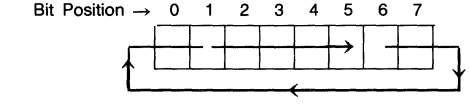
In instructions where $R7_8$ (IVL) or $R17_8$ (IVR) is specified as the destination, the 8-bit value is output on the $\overline{IV}$ bus as an I/O device address or memory location; register R7 selects the Left Bank and register R17 selects the Right Bank. The results are also stored into the specified internal register ($R7_8$ or $R17_8$) and may later be accessed as source data. When the $\overline{IV}$ bus is specified as a source and/or destination, the "S" and "D" fields are split into two parts, that is,

- Source (S) = S_1, S_0 and Destination (D) = D_1, D_0 where,
  S_0 specifies the LSB of rotated input data field
  S_1 specifies the bank of $\overline{IV}$ bus from which source data will be input
  D_0 specifies bit position in I/O device with which LSB of processed data will be aligned and
  D_1 specifies the bank of $\overline{IV}$ bus which will be the destination.



NOTES:
1. The field length of 0-to-8 bits is specified by the "L" field.
2. For the Right Bank, $30_8 - 37_8$ perform equivalent I/O functions.

RIGHT-ROTATE FUNCTION



Rotate (R) and Length (L) Field. The 3-bit R/L field performs one of two functions, specifying either the field length (L) for I/O operations or a right-rotate (R) for internal operations. For a given instruction, the specified function depends upon the contents of the Source (S) and Destination (D) fields.

When an internal register is specified by both the source and destination fields, the "R" field is invoked and it specifies a right-rotate of the data specified in the "S" field (see accompanying diagram.) The source-register data (up to 8 bits) is right-rotated during the "input phase" of the instruction cycle (Figure 4). This function is always performed prior to any ALU operation. (*Note: The right-rotate function is implemented on the bus and not in the source register.*)

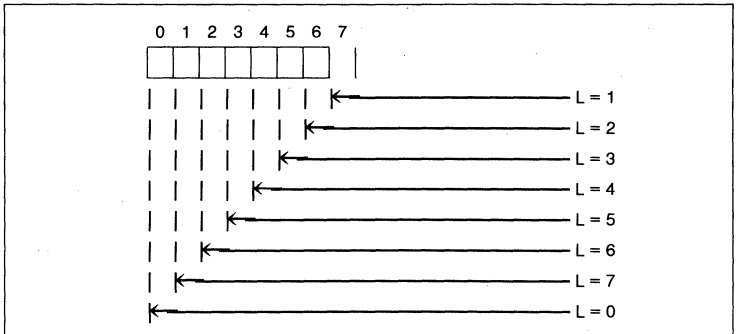
When either or both of the source and destination fields specify a variable-length I/O

data field, the "L" field specifies the length of the I/O data field (see following diagram). If the source field specifies an $\overline{IV}$ address ($20_8 - 37_8$) and the destination field specifies an internal register ($00_8 - 07_8$, $11_8 - 17_8$), the "L" field specifies the length of source data; the source data is formed by right-rotating the $\overline{IV}$ bus data according to the source address and then masking result as specified by the "L" field. If length is less than 8 bits, all remaining bits are set to zero prior to processing data in the ALU. If the source field specifies an internal register ($00_8 - 17_8$) and the destination field specifies $\overline{IV}$ bus data ($20_8 - 37_8$), the "L" field specifies the length of the destination data. To form the destination data, the ALU output is left-shifted according to the destination address and then masked to the required length (see $\overline{IV}$ DATA LENGTH SPECIFICATION). The destination data is merged with data in the I/O latches to finalize the $\overline{IV}$ bus data. Hence, a one-to-eight bit destination data field can be inserted into the existing 8-bit I/O port without modifying surrounding bits. If both the source and destination fields specify $\overline{IV}$ bus data ($20_8 - 37_8$), the "L" field specifies the length of both the source and destination data.

To form the source data, the $\overline{IV}$ bus input data is right-rotated according to the source address and then masked to the required length — see $\overline{IV}$ DATA LENGTH SPECIFICATION. If length is less than 8 bits, all remaining bits are set to zero before processing in the ALU. To form the destination data, the ALU output is left-shifted according to the destination address and masked to the required length specification. The destination data is then merged into the $\overline{IV}$ bus data that was used to obtain the source; thus, if the source and destination addresses are on the same bank, the $\overline{IV}$ bus data written to the destination I/O Port appears unmodified, except for bits changed during the shift-and-mask operations. If the source and destination addresses refer to different banks, the destination I/O Port is changed to contain the contents of the source I/O Port in those bit positions not affected by the destination data.

J Field. The 5-bit or 8-bit "J" field is used to load a literal value (contained in the instruction) into a register, into a variable I/O data field, or to modify the low-order bits of the Program Counter. The bit length of the "J" field is implied by the "S" and "L" fields in the XEC, NZT, and XMIT instructions, based on the following conditions:

- When the Source (S) field specifies an internal register, the literal value of the "J" field is an 8-bit binary number.
- When the Source (S) field specifies a variable I/O data field, the literal value of the "J" field is a 5-bit binary number.



$\overline{IV}$ DATA LENGTH SPECIFICATION (No Rotate Function Specified)

A Field. The 13-bit "A" field is an address field which allows the 8X305 to directly branch to any of the 8192 locations in Program Storage memory.

Formation of Instruction Address

The Address Register and Program Counter are used to generate addresses for accessing an instruction from program storage. The instruction address is formed in one of the following ways:

- For all except the JMP, XEC, and a "satisfied" NZT instruction, the Program Counter is incremented by one and placed in the Address Register.
- For the JMP instruction, the 13-bit "A" field contained in the JMP instruction word replaces the contents of both the Address Register and the Program Counter.
- For the XEC instruction, the Address Register is loaded with bits from the Program Counter modified as follows:
  XEC using $\overline{IV}$ Bus Data — low-order 5 bits of ALU output replaces counterpart bits in Address Register.

  XEC using Data from Internal Register — low-order 8 bits of ALU output replaces counterpart bits in Address Register.

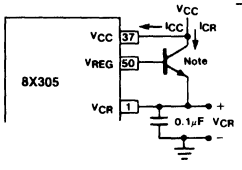
The Program Counter is not modified for either of the above conditions.

- For a "satisfied" NZT instruction, the low-order 5 bits (NZT source is $\overline{IV}$ bus data) or low-order 8 bits (NZT source is an internal register) of both the Address Register and Program Counter are loaded with the literal value specified by the "J" field of instruction word.

Data Addressing

The source and/or destination addresses of the data to be operated upon are specified as part of the instruction word. As shown earlier, source/destination addresses are specified using a 5-bit code ($00_8 - 37_8$). When the most significant octal digit is a "0" or "1", the source and/or destination address is an internal register; if the most significant digit is a 2 or 3, an $\overline{IV}$ bus operation is indicated — 2 specifying a Left-Bank ($\overline{LB}$) operation and 3 specifying a Right-Bank ($\overline{RB}$) operation. The least significant octal digit (0 through 7) indicates either a specific internal register address or positioning information for the least significant bit when specifying $\overline{IV}$ bus data. Referring to Table 5, AUXiliary register R0 (00_8) is the implied source of the second argument for the ADD, AND, and XOR operations. IVL register R7 and IVR register R17 (destination addresses 07_8 and 17_8, respectively) provide a means of routing enabling address information to I/O peripherals. With IVL or IVR specified as the destination address, data is placed on the $\overline{IV}$ bus during the output phase of the instruction cycle; simultaneously, a Select Command (SC) is generated to inform all I/O devices that information on the $\overline{IV}$ bus is to be considered as an I/O address. Since the contents of IVL and IVR are preserved, either register may later be accessed as a source of data.

Control outputs $\overline{LB}$ and $\overline{RB}$ are used to partition I/O bus devices into two fields of 256 addresses. With $\overline{LB}$ in the active-low state and a source address of $20_8 - 27_8$, the left bank of I/O devices are enabled during the input phase of the instruction cycle. With $\overline{RB}$ in the active-low state and a source address of $30_8 - 37_8$, the right bank of devices are enabled. During the output phase, $\overline{LB}$ is low if the destination address is 07_8 or $20_8 - 27_8$, whereas $\overline{RB}$ is low if the destination address is 17_8 or $30_8 - 37_8$. Each address field ($\overline{LB}$ and $\overline{RB}$) can have a different I/O device selected, that is, data can be transferred from a device in one bank to a device in the other in one instruction cycle.



PARAMETER	CONDITIONS	LIMITS
h_{fe}	V_{CE} = 2V; 100mA < I_C < 500mA	> 50
VBE_{ON}	V_{CE} = 5V; I_C = 500mA	< 1V
V_{CESAT}	I_C = 500mA; I_B = 50mA	< 0.5V
BV_{CEO}		> 15V
f_t		> 30MHz

LD05661S

NOTE:
Typical approved parts — 2N5320, 2N5337

Voltage Regulator

DESIGN PARAMETERS

Hardware design of an 8X305-based system largely consists of the following operations:

- Selecting and interfacing a Program Storage device — ROM, PROM, etc.
- Selecting and interfacing input/output devices — RAM, Ports, and other 8-bit addressable I/O devices.
- Choosing and implementing System Clock — Capacitor-Controlled, Crystal-Controlled, or Externally-Driven.
- Selection of an off-chip series-pass transistor.

VOLTAGE REGULATOR

All internal logic of the 8X305 is powered by an on-chip voltage regulator that requires an external series-pass transistor. Electrical specifications for the off-chip power transistor and a typical hook-up are shown in the accompanying diagram. To minimize lead inductance, the transistor should be as close as possible to the 8X305 package and the emitter should be AC-grounded via a 0.1μf ceramic capacitor.

All information required for easy implementation of these design requirements is provided under the following captions:

- Ordering Information
- Voltage Regulator
- DC Characteristics
- AC Characteristics
- Timing Considerations
- Clock Considerations
- $\overline{\text{HALT}}/\overline{\text{RESET}}$ Logic

ABSOLUTE MAXIMUM RATINGS Storage Temperature (T_{STG}) rating are from −65°C to +150°C

SYMBOL	PIN	DESCRIPTION	RATING	UNIT
V_{CC}	V_{CC}	Supply voltage	+7.0	V
	X1, X2	Crystal input voltage	2.0	V
	All other pins	Logic input voltage	5.5	V

DC ELECTRICAL CHARACTERISTICS (Commercial Part) $4.75V \leqslant V_{CC} \leqslant 5.25V$, $0°C \leqslant T_A \leqslant 70°C$

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT	COMMENTS
			Min	Typ	Max		
V_{CC}	Supply voltage		4.75	5	5.25	V	
V_{IH}	High-level input voltage		0.9 2		2 5.5	V	X1 and X2 All other pins
V_{IL}	Low-level input voltage				0.5 0.8	V	X1 and X2 All other pins
V_{OH}	High-level output voltage	V_{CC} = min; I_{OH} = −3mA	2.4			V	
V_{OL}	Low-level output voltage	V_{CC} = min; I_{OL} = 6mA V_{CC} = min; I_{OL} = 16mA			0.55 0.55	V	A_0 through A_{12} All other outputs
V_{CR}	Regulator voltage	V_{CC} = 5V		3.1 2.9		V	T_A = 0°C T_A = 70°C
V_{IC}	Input clamp voltage	V_{CC} = min; I_{IN} = −10mA			−1.5	V	Crystal inputs X1 and X2 do not have internal clamp diodes
I_{IH}	High-level input current	V_{CC} = max V_{IH} = 0.9V V_{IH} = 4.5V			4 50	mA μA	X1 and X2 All other pins
I_{IL}	Low-level input current	V_{CC} = max; V_{IL} = 0.4V			−3 −0.2 −1.6 −0.4	mA	X1 and X2 $\overline{IV0} - \overline{IV7}$ I0 − I15 $\overline{HALT}$ and $\overline{RESET}$
I_{OS}	Short circuit output current	V_{CC} = max; (Note: At any time, no more than one output should be connected to ground.)	−30		−140	mA	All output pins
I_{CC}	Supply current	V_{CC} = max			180 195	mA	T_A = 70°C T_A = 0°C
I_{REG}	Regulator control	V_{CC} = 5.0V	−10		−25	mA	Max available base drive for series-pass transistor
I_{CR}	Regulator current	V_{CC} = max			200 230	mA	T_A = 70°C T_A = 0°C

NOTES:
1. Operating temperature ranges are guaranteed after thermal equilibrium has been reached.
2. All voltages measured with respect to ground terminal.

AC ELECTRICAL CHARACTERISTICS (Commercial Part) Conditions: $4.75V \leq V_{CC} \leq 5.25V$; $0°C \leq T_A \leq 70°C$

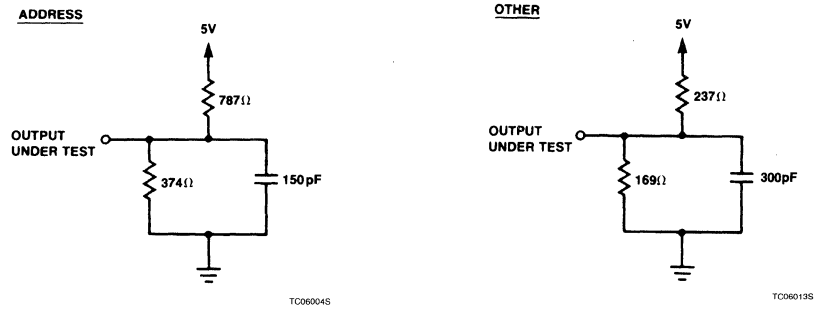
Loading: (See test circuits)

SYMBOL	PARAMETER (NOTE 1)	LIMITS (INSTRUCTION CYCLE TIME = 200ns)			LIMITS (INSTRUCTION CYCLE TIME > 200ns)			UNITS	COMMENTS
		Min	Typ	Max	Min	Typ	Max		
t_{PC}	Processor cycle time	200			200			ns	
t_{CP}	X1 clock period	100			100			ns	
t_{CH}	X1 clock high time	50			50			ns	
t_{CL}	X1 clock low time	50			50			ns	
t_{MCL}	MCLK low delay	15		40	15		40	ns	
t_W	MCLK pulse width	35		55	$T_{4Q} - 15$		$T_{4Q} + 5$	ns	Note 2
t_{DD}	Input data to output data	70		105	70		105	ns	
t_{MHS}	MCLK falling edge to $\overline{HALT}$ falling edge			30			$T_{1Q} - 20$	ns	Note 2
t_{MHH}	$\overline{HALT}$ hold time (MCLK falling edge)	65			$T_{1Q} + 15$			ns	Note 2
t_{ACC}	Program storage access time			60				ns	
t_{IO}	I/O port output enable time ($\overline{LR}/\overline{RB}$ to valide $\overline{IV}$ data input)			30				ns	
t_{MAS}	MCLK falling edge to address stable			140			$T_{1Q} + T_{2Q} + 40$	ns	Notes 2, 3, & 4
t_{IA}	Instruction to address			140			$T_{2Q} + 90$	ns	Notes 2, 3 & 5
t_{IVA}	Input data to address			85			85	ns	Notes 3 & 6
t_{MIS}	MCLK falling edge to instruction stable			25			$T_{1Q} - 25$	ns	Notes 2 & 7
t_{MIH}	Instruction hold time (MCLK falling edge)	55			$T_{1Q} + 5$			ns	Notes 2 & 8
t_{MWH}	MCLK falling edge to SC/WC rising edge	105		125	$T_{1Q} + T_{2Q} + 5$		$T_{1Q} + T_{2Q} + 25$	ns	Note 2
t_{MWL}	MCLK falling edge to SC/WC falling edge	2		15	2		15	ns	
t_{MIBS}	MCLK falling edge to $\overline{LB}/\overline{RB}$ (Input phase)	5		25	5		25	ns	
t_{IIBS}	Instruction to $\overline{LB}/\overline{RB}$ (Input phase)			25			25	ns	
t_{MOBS}	MCLK falling edge to $\overline{LB}/\overline{RB}$ (Output phase)	115		145	$T_{1Q} + T_{2Q} + 15$		$T_{1Q} + T_{2Q} + 45$	ns	Note 2
t_{MIDS}	MCLK falling edge to input data stable			55			$T_{1Q} + T_{2Q} - 45$	ns	Note 2
t_{MIDH}	Input data hold time (MCLK falling edge)	115			$T_{1Q} + T_{2Q} + 15$			ns	Note 2
t_{MODH}	Output data hold time (MCLK falling edge)	11			11			ns	
t_{MODS}	Output data stable (MCLK falling edge)	123		150	$T_{1Q} + T_{2Q} + 23$		$T_{1Q} + T_{2Q} + 50$	ns	Note 2
t_{ODSM}	Output data stable (MCLK rising edge)	10			$t_{3Q} - 40$			ns	Note 2

NOTES:

1. X1 and X2 inputs are driven by an external pulse generator with an amplitude of 1.5 volts; all timing parameters are measured at this voltage level.
2. Respectively, T_{1Q}, T_{2Q}, T_{3Q}, and T_{4Q} represent time intervals for the first, second, third, and fourth quarter cycles.
3. Capacitive loading for the address bus is 150 picofarads.
4. T_{MAS} is obtained by forcing a valid instruction and an I/O bus input to occur earlier than the specified minimum set up time.
5. T_{IA} is obtained by forcing a valid instruction input to occur earlier than the minimum set up time.
6. T_{IVA} is obtained by forcing a valid I/O bus input to meet the minimum set up time.
7. T_{MIS} represents the setup time required by internal latches of the 8X305. In system applications, the instruction input may have to be valid before the worst-case set up time in order for the system to respond with a valid I/O bus input that meets the I/O bus input set up time (T_{IDS} and T_{MIDS}).
8. T_{MIH} represents the hold time required by internal latches of the 8X305. To generate proper $\overline{LB}/\overline{RB}$ signals, the instruction must be held valid until the address bus changes.

AC TEST CIRCUITS



NOTE:
Load capacitance includes Test Jig and Probe Capacitance

TIMING CONSIDERATIONS (Commercial Part)

As shown in the AC CHARACTERISTICS table for the commercial part, the minimum instruction cycle time is 200ns; whereas, the maximum is determined by the on-chip oscillator frequency and can be any value the user chooses. With an instruction cycle time of 200ns, the part can be characterized in terms of absolute values; these are shown in the first "LIMITS" column of the table. When the instruction cycle time is greater than 200ns, certain parameters are cycle-time dependent; thus, these parameters are specified in terms of the four quarter cycles (T_{1Q}, T_{2Q}, T_{3Q}, and T_{4Q}) that make up one instruction cycle — see 8X305 TIMING DIAGRAM. As the time interval for each instruction cycle increases (becomes greater than 200ns), the delay for all parameters that are cycle-time dependent is likewise increased. In some cases, these delays have a significant impact on timing relationships and other areas of systems design; subsequent paragraphs describe these timing parameters and reliable methods of calculation.

Timing parameters for the 8X305 are normally measured with reference to MCLK.

System determinants for the instruction cycle time are:

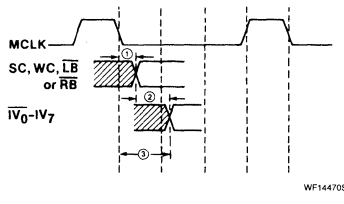
- Propagation delays within the 8X305
- Access time of Program Storage
- Enable time of the I/O port

Normally, the instruction cycle time is constrained by one or more of the following conditions:

Condition 1 — Instruction or MCLK to $\overline{LB}/\overline{RB}$ (input phase) plus I/O port access time (TIO) $\leq \overline{IV}$ data setup time (Figure 5a).

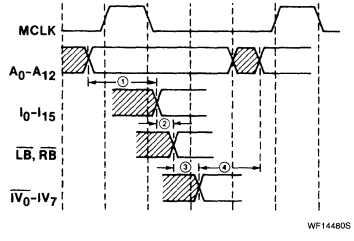
Condition 2 — Program storage access time (TACC) plus instruction to $\overline{LB}/\overline{RB}$ (input phase) plus I/O port access time (TIO) plus $\overline{IV}$ data (input phase) to address $\leq$ instruction cycle time (Figure 5b).

Condition 3 — Program storage access time plus instruction to address $\leq$ instruction cycle time (Figure 5c).



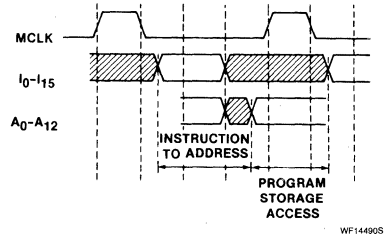
NOTES:
① MCLK to $\overline{LB}/\overline{RB}$ (input phase) or instruction to $\overline{LB}/\overline{RB}$ (input phase).
② I/O port access (TIO).
③ $\overline{IV}$ data setup time (referenced to MCLK).

a. Condition #1



NOTES:
① Program storage access time.
② MCLK to $\overline{LB}/\overline{RB}$ (input phase) or instruction to $\overline{LB}/\overline{RB}$ (input phase).
③ I/O port access (TIO).
④ $\overline{IV}$ data (input phase to address.)

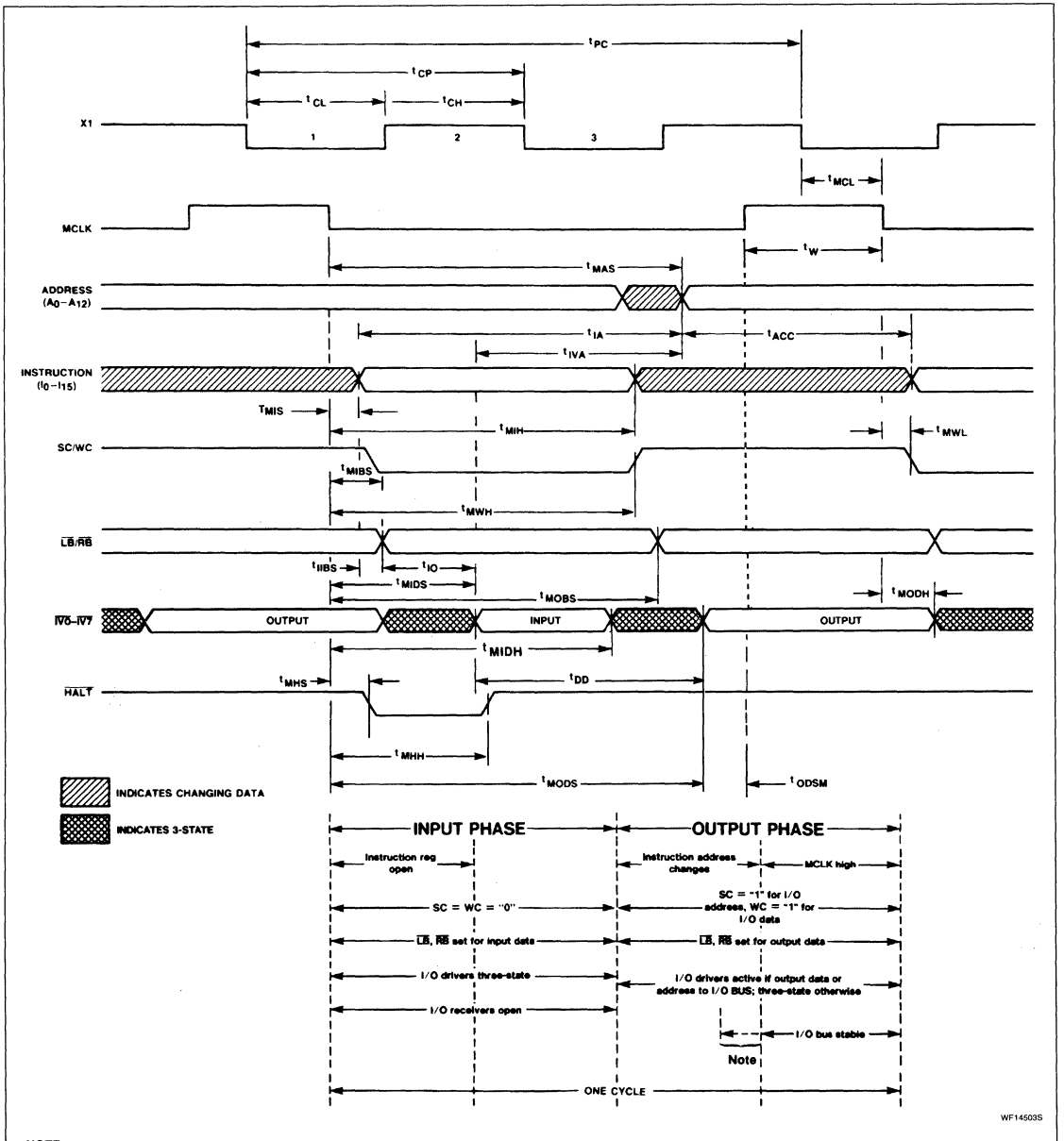
b. Condition #2



c. Condition #3

Figure 5. Constraints of 8X305 Instruction Cycle Time

8X305 TIMING DIAGRAM



NOTE:
For an instruction cycle time greater than 200ns, the I/O bus can be stable sometime within the third quarter (T_{3Q}) cycle.

From condition #1 and with an instruction cycle time of 200ns, the I/O port access time (TIO) can be calculated as follows:

$$\begin{aligned} & TMIBS + TIO \leqslant TMIDS \\ \text{transposing,}\ & TIO \leqslant TMIDS = TMIBS \\ \text{substituting,}\ & TIO \leqslant 55ns - 25ns \\ \text{result,}\ & TIO \leqslant 30ns \end{aligned}$$

Using 30ns for TIO, the constraint imposed by condition #1 can also be used to calculate the minimum cycle time:

$$\begin{aligned} & TMIBS + TIO \leqslant TMIDS \\ \text{thus,}\ & 25ns + 30ns \leqslant T_{1Q} + T_{2Q} - 45 \\ & 25ns + 30ns \leqslant 1/2\ \text{cycle} - 45 \end{aligned}$$

therefore, the worst-case instruction cycle time is 200ns. With subject parameters referenced to X1, the same calculations are valid:

$$\begin{aligned} & TIBS + TIO + TIDS \leqslant 1/2\ \text{cycle} \\ \text{thus,}\ & 45ns + 30ns + 25ns \leqslant 1/2\ \text{cycle} \end{aligned}$$

therefore, the worst-case instruction cycle time is again 200ns. From condition #2 and with an instruction cycle time of 200ns, the program storage access time can be calculated:

$$\begin{aligned} & TACC + TIIBS + TIO + TIVA \\ & \leqslant 200ns \\ \text{transposing,}\ & TACC \leqslant 200ns - TIIBS - TIO \\ & - TIVA \\ \text{substituting,}\ & TACC \leqslant 200ns - 25ns - 30ns \\ & - 85ns \\ \text{thus,}\ & TACC \leqslant 60ns \end{aligned}$$

hence, for instruction cycle time of 200ns, a program storage access time of 60ns is implied. The constraint imposed by condition #3 can be used to verify the maximum program storage access time:

$$\begin{aligned} & TIA + TACC \leqslant \text{Instruction Cycle} \\ \text{thus,}\ & TACC \leqslant 200ns - 140ns \\ \text{and,}\ & TACC \leqslant 60ns \end{aligned}$$

confirming that a program storage access time of 60ns is satisfactory.

For an instruction cycle time of 200ns and a program storage access time of 60ns (Condition #2/Figure 5b), the instruction should be valid at the falling edge of MCLK. This relationship can be derived by the following equation:

$$\begin{aligned} & 200ns - TMAS - TACC \\ & = 200ns - 140ns - 60ns \\ & = 0ns \end{aligned}$$

It is important to note that, during the input phase, the beginning of a valid $\overline{LB}/\overline{RB}$ signal is determined by either the instruction to $\overline{LB}/\overline{RB}$ delay (TIIBS) or the delay from the falling edge of MCLK to $\overline{LB}/\overline{RB}$ (TMIBS). Assuming the instruction is valid at the falling edge of MCLK and adding the instruction-to-$\overline{LB}/\overline{RB}$ delay (TIIBS = 25ns), the $\overline{LB}/\overline{RB}$ signal will be valid 25ns after the falling edge of MCLK. With a fast program storage memory and with a valid instruction before the falling edge of MCLK — the $\overline{LB}/\overline{RB}$ signal will, due to the TMIBS delay, still be valid 25ns after the falling edge of MCLK. Using a worst-case instruction cycle time of 200ns, the user cannot gain a speed advantage by selecting a memory with faster access time. Under the same conditions, a speed advantage cannot be obtained by using an I/O port with fast access time (TIO) because the address bus will be stable 55ns (TAS) after the beginning of the third quarter cycle — no matter how early the $\overline{IV}$ data input is valid.

CLOCK CONSIDERATIONS

The on-chip oscillator and timing-generation circuits of the 8X305 can be controlled by any one of the following methods:

Capacitor — if timing is not critical
Crystal — if precise timing is required
External Drive — if application requires that the 8X305 be driven from a system clock

Capacitor Timing. A non-polarized ceramic or mica capacitor with a working voltage equal to or greater than 25V is recommended. The lead lengths of capacitor should be approximately the same and as short as possible; also, the timing circuits should not be in close proximity to external sources of noise. For various capacitor (C_X) values, the cycle time can be approximated as:

C_X (in pF)	APPROXIMATE CYCLE TIME
100	300ns
200	500ns
500	1.1μs
1000	2.0μs

Crystal Timing. When a crystal is used, the on-chip oscillator operates at the resonant frequency (f_o) of the crystal. The series-resonant quartz crystal connects to the 8X305 via pins 10 (X1) and 11 (X2). The lead lengths of the crystal should be approximately equal and as short as possible. Also, the timing circuits should not be in close proximity to external sources of noise. The crystal should be hermetically sealed (HC type can) and have the following electrical characteristics:

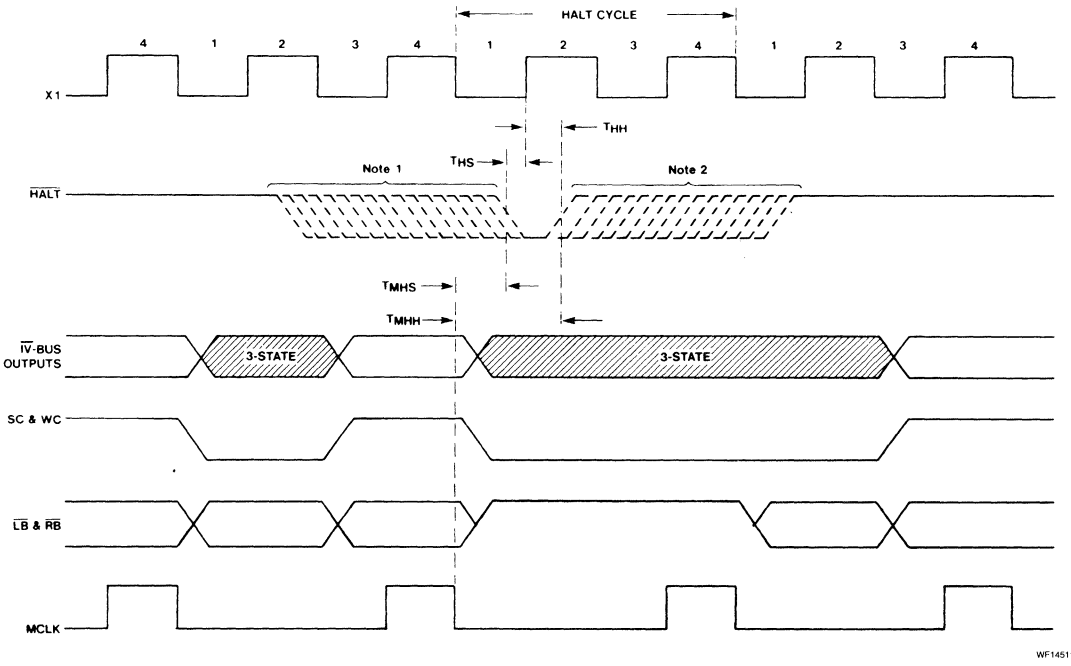
Type — Fundamental mode, series resonant
Impedance at Fundamental — 35Ω max.
Impedance at Harmonics and Spurs — 50Ω min.

The resonant frequency (f_o) of the crystal is related to the desired cycle time (T) by the equation: $f_o = 2/T$; thus, for a cycle time of 200ns, f_o = 10MHz.

$\overline{HALT}$ Logic

The $\overline{HALT}$ signal is sampled via internal chip logic at the end of the first internal quarter of each instruction cycle. If, when sampled, the $\overline{HALT}$ signal is active-low, a halt is immediately executed and the current instruction cycle is terminated. However, the halt cycle does not inhibit MCLK nor does it affect any internal registers of the 8X305. As long as the $\overline{HALT}$ line is active-low, the SC and WC lines are low (inactive), the Left Bank ($\overline{LB}$)/Right Bank ($\overline{RB}$) signals are high (inactive), and the $\overline{IV}$ bus remains in the 3-State mode of operation. Normal operation resumes at the next cycle in which $\overline{HALT}$ is high when sampled (see $\overline{HALT}$ TIMING DIAGRAM.)

HALT TIMING DIAGRAM



NOTES:
1. The $\overline{HALT}$ signal can switch from High to Low at any time during this interval.
2. The $\overline{HALT}$ signal can switch from Low to High at any time during this interval.

TIMING DESCRIPTIONS:
T_{HS} — setup time from $\overline{HALT}$ to X1 (independent of instruction cycle time)
T_{HH} — hold time from X1 to $\overline{HALT}$ (independent of instruction cycle time)
T_{MHS} — setup time from MCLK to $\overline{HALT}$ (dependent upon instruction cycle time)
T_{MHH} — hold time from MCLK to $\overline{HALT}$ (dependent upon instruction cycle time)

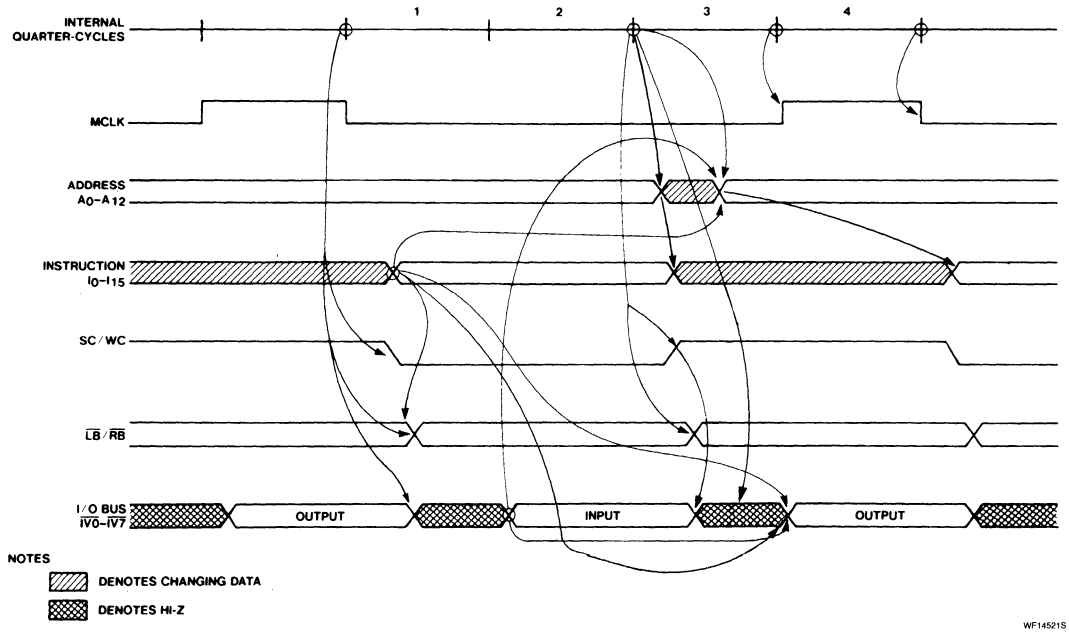


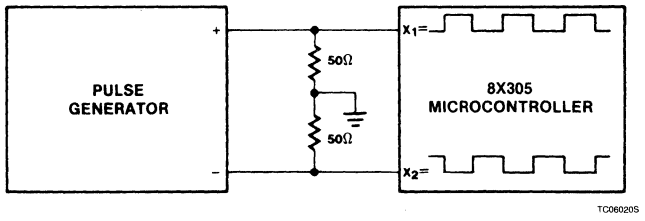
Figure 7. Timing Relationships of 8X305 I/O Signals

Using an External Clock. The 8X305 can be synchronized with an external clock by simply connecting appropriate drive circuits to the X1/X2 inputs. Figure 8 shows how the on-chip oscillator can be driven from the complementary outputs of a pulse generator. In applications where the Microcontroller must be driven from a master clock, the X1/X2 lines can be interfaced to TTL logic as shown in Figure 9.

$\overline{\text{RESET}}$ Logic

$\overline{\text{RESET}}$ (pin 43) can be driven from a high (inactive) state to a low (active) state at any time with respect to the system clock, that is, the reset function is asynchronous. To ensure proper operation, $\overline{\text{RESET}}$ must be held low (active) for one full instruction time. When the line is driven from a high state to an active-low state, several events occur — the precise instant of occurrence is basically a function of the propagation delay for that particular event. As shown in the $\overline{\text{RESET}}$ TIMING DIAGRAM, these events are:

- The Program Counter and Address Register are set to address zero and remain in that state as long as the $\overline{\text{RESET}}$ line is low. Other than PC and AR, $\overline{\text{RESET}}$ does not affect other internal registers.



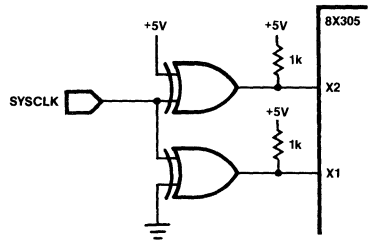
PULSE GENERATOR CHARACTERISTICS:
$Z_{OUT} = 50\Omega$
RISETIME $\leq$ 10ns
V_{OUT} = 0 to 1.5V
Skew $\leq$ 10ns

Figure 8. Clocking With a Pulse Generator

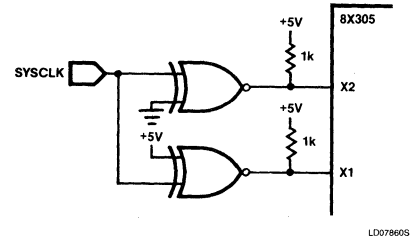
- The input/output ($\overline{\text{IV}}$) bus goes 3-State and remains in that condition as long as the $\overline{\text{RESET}}$ line is low.
- The Select Command and Write Command signals are driven low and remain low as long as the $\overline{\text{RESET}}$ line is low.
- The Left Bank/Right Bank ($\overline{\text{LB}}/\overline{\text{RB}}$) signals are forced high asynchronously for the period in which the $\overline{\text{RESET}}$ line is low.

During the time $\overline{\text{RESET}}$ is active-low, MCLK is inhibited. Moreover, if the $\overline{\text{RESET}}$ line is driven low during the last two quarter cycles, MCLK may be shortened for that particular machine cycle. When $\overline{\text{RESET}}$ line is driven high (inactive) — one quarter to one full instruction cycle later, MCLK appears just before normal operation is resumed. The $\overline{\text{RESET}}$/MCLK relationship is clearly shown by "B" in the timing diagram. As long as the $\overline{\text{RESET}}$ line is active-low, the $\overline{\text{HALT}}$ signal (described next) is not sampled by internal logic of the 8X305.

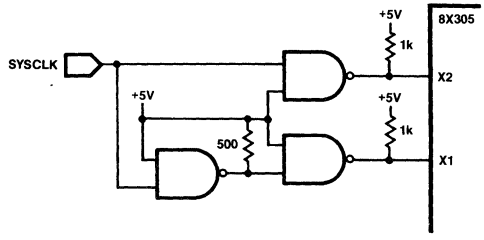
1. Using a 74LS136 Quad Execlusive-OR gate (open collector).



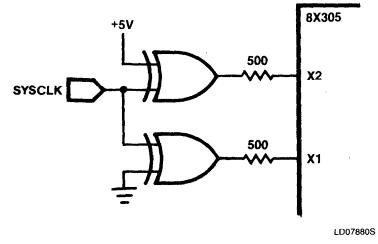
2. Using a 74LS266 Quad Exclusive-NOR gate (open collector).



3. Using a 74F38* Quad NAND gate (open collector).



4. Using a 74LS286 Quad Exclusive-OR gate.

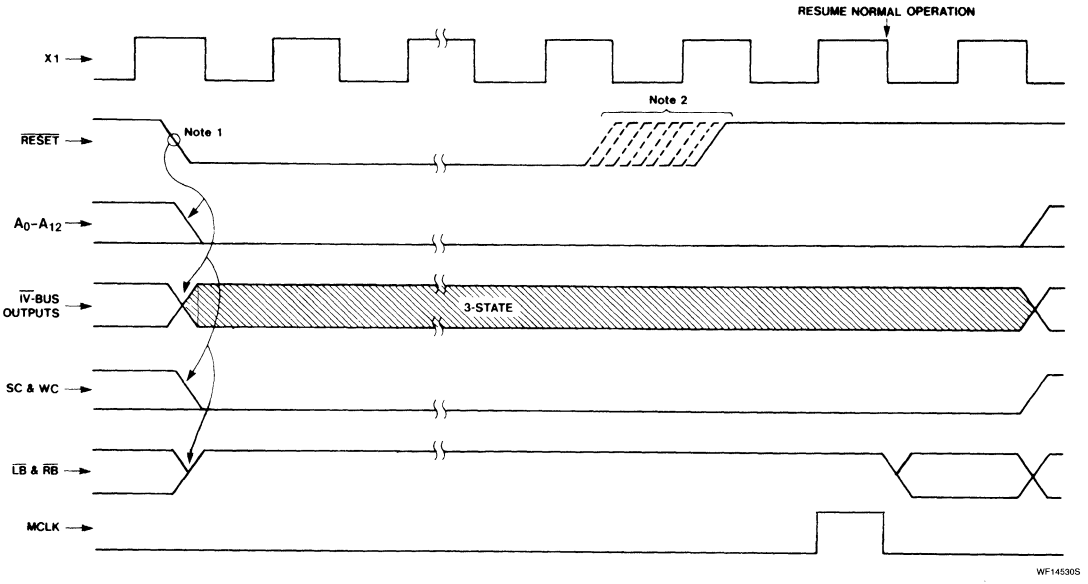


TTL DRIVER CHARACTERISTICS:
Fall Time ⩽ 10ns
Skew Between Complementary Outputs ⩽ 10ns

NOTES:
1. All circuits, as drawn, preserve the phase relationship of SYSCLK to X1.
2. The resistor values of 1k ohms for open-collector pull-ups and 500 ohms for active pull-up series resistors are calculated to be optimum for all opening conditions and silicon varations.
3. The 8X305 clock may be driven by circuits other than those shown here. The circuits shown however, have been tested under conditions in excess of those that will be found in a normal system. Exclusive-OR/NOR type gates were selected to minimize the skew between the X1 and X2 inputs.
4. 74LS38 and 74S38 gates were tested in addition to the 74F38 chip. These were found to be the least robust configurations of all those tested, although they did work over normal operating conditions abd beyond. When failure occured it was due to excessive skew between X1 and X2 caused by the inverting gate in the X1 leg.

Figure 9. Clocking With TTL

RESET TIMING DIAGRAM



NOTES:
1. A High to Low transition of the RESET signal will force the Address Bus to an all-zero configuration.
2. The RESET signal can switch from Low to High at any point within this time interval and, in all cases, MCLK will occur at least one-quarter cycle time later as shown.

Signetics

8X401
Microcontroller

Product Specification

Microprocessor Products

DESCRIPTION

The Signetics 8X401 Microcontroller is a very high-speed bipolar microprocessor implemented with internal ECL technology. The 8X401 Microcontroller combines speed, flexibility, and a bit-oriented instruction set to accommodate many sophisticated applications. It excels in systems that require high-speed bit or byte manipulations, such as high-speed controllers and data communications.

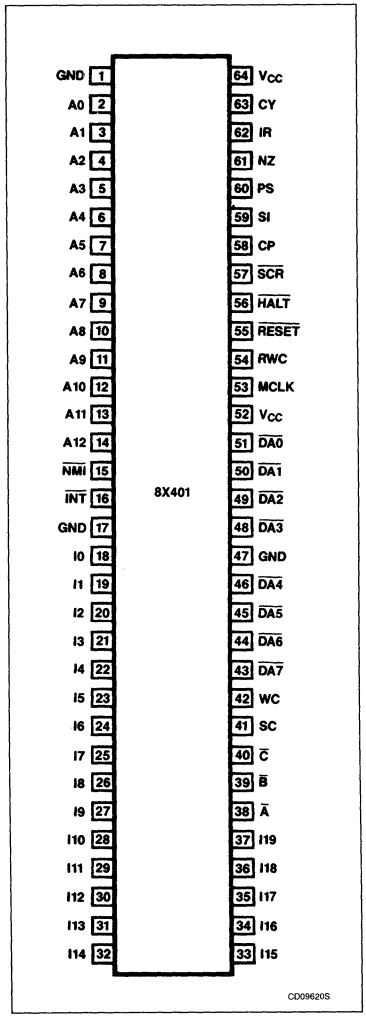
The 8X401 can fetch, execute, and generate the next instruction address for a 20-bit instruction in a minimum of 150ns. Within one instruction cycle, the 8X401 can be programmed to input, right-rotate and mask single or multiple bit subfields, perform an ALU operation, left-rotate, merge the subfield into the destination, and output.

To interface with program memory, the 8X401 uses a 13-bit address bus and a 20-bit instruction bus. An 8-bit bidirectional data/address bus, and an I/O control and timing bus is used to access external peripheral devices.

FEATURES

- **Fetches and executes all instructions in a minimum of 150ns**
- **Bit manipulation-oriented instruction set**
- **Separate buses for instruction, instruction address and I/O**
- **Sixteen 8-bit registers**
- **On-chip interrupt control**
- **TTL compatible I/O**
- **Single +5V supply**
- **0.9-inch 64-pin DIP, 68-pin plastic leaded chip carrier**
- **Two user-definable status flags**
- **Single TTL clock input**
- **Three independent I/O banks**
- **On-board control sequencer**
- **On-chip subroutine capabilities**
- **Fixed instruction set — 32 instructions**
- **Complete development support**

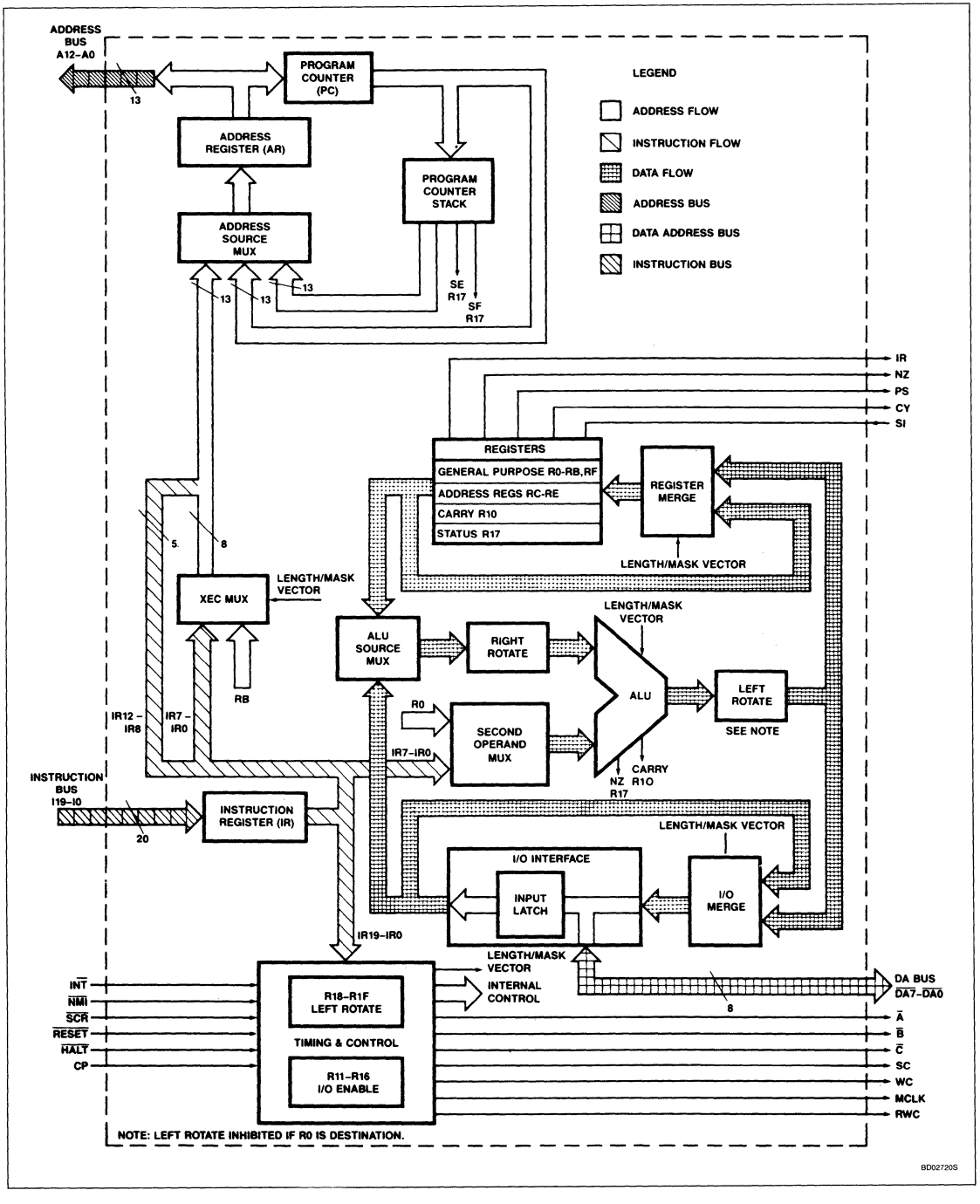
PIN CONFIGURATION



ORDERING INFORMATION

DESCRIPTION	ORDER CODE
64-pin Ceramic DIP – 900 Mil Wide	N8X401I
64-pin Plastic DIP – 900 Mil Wide	N8X401N

BLOCK DIAGRAM OF THE 8X401



PIN DESCRIPTION

PIN NO.	IDENTIFIER	FUNCTION
1, 17, 47	GND	Ground.
2 – 14	A12 – A0	**Program Address Lines:** These active-high outputs permit direct addressing of up to 8192 locations of program storage; A0 is LSB.
15	$\overline{NMI}$	**Non-Maskable Interrupt:** The falling edge of this active-low input pin generates a non-maskable interrupt.
16	$\overline{INT}$	**Interrupt:** This active-low input pin is tested during the fourth quarter of each instruction cycle. If an interrupt is indicated and if interrupts are enabled, the address of the next instruction that was to be executed is stored onto the program counter stack before the interrupt is serviced.
18 – 37	I19 – I0	**Instruction Lines:** These active-high input lines receive 20-bit instructions from program storage; I0 is LSB.
38	$\overline{A}$	**Bank A:** When low, devices connected to bank A are accessed. (Note: Typically, the A signal is tied to the $\overline{ME}$ input pin of I/O peripherals.)
39	$\overline{B}$	**Bank B:** When low, devices connected to bank B are accessed. (Note: Typically, the B signal is tied to the $\overline{ME}$ input pin of I/O peripherals.)
40	$\overline{C}$	**Bank C:** When low, devices connected to bank C are accessed. (Note: Typically, the C signal is tied to the $\overline{ME}$ input pin of I/O peripherals.)
41	SC	**Select Control:** When high, an address is being output on pins $\overline{DA7}$ through $\overline{DA0}$.
42	WC	**Write Control:** When high, data is being output on pins $\overline{DA7}$ through $\overline{DA0}$.
43 – 46 48 – 51	$\overline{DA7} - \overline{DA0}$	**Data Address Bus:** These active-low, bidirectional, three-state lines are used for I/O; $\overline{DA0}$ is LSB.
52, 64	V_{CC}	+5V power supply.
53	MCLK	**Master Clock:** This active-high output signal is used to strobe data into data peripherals for clocking I/O devices and/or synchronization of external logic. MCLK is active-high in the fourth quarter cycle.
54	RWC	**Read/Write Clock:** This active-high output signal is used for synchronization of external logic and is active-high during the third and fourth quarter cycles.
55	$\overline{RESET}$	**Reset:** The $\overline{RESET}$ input pin is used to initialize the 8X401.
56	$\overline{HALT}$	**Halt:** The $\overline{HALT}$ input is sampled during the first quarter cycle of each instruction cycle. When the $\overline{HALT}$ input is low, the instruction cycle is not executed.
57	$\overline{SCR}$	**Slow Clock Request:** This active-low control input is sampled during the first quarter cycle of each instruction. When $\overline{SCR}$ is asserted, it will cause the current instruction to be executed at half of the normal clock rate. This control input is necessary to accommodate I/O devices that cannot operate at the 8X401's full speed, without having to continuously run the 8X401 at half speed.
58	CP	**Clock Pulse:** Each 8X401 quarter cycle will correspond to one full cycle of the clock pulse.
59	SI	**Status Input:** The value of the SI pin during the fourth quarter cycle is transferred to SI bit in the status register.
60	PS	**Programmable Status:** The programmable status pin is controlled entirely by the user program.
61	NZ	**Non-Zero:** The NZ bit of the status register is reflected on this pin.
62	IR	**Interrupt Receivable:** The IR pin indicates whether an interrupt applied at any point in time will be serviced. Interrupts are receivable when the interrupt mask (status register, bit 0) is clear and the stack is not full (IM = 0 and SF = 0).
63	CY	**Carry:** Carry bit from R10 is output on this pin.

SMALL SYSTEM CONFIGURATION

The system hookup shown on the next page, although of the simplest form, provides a fundamental example of the 8X401 Microcontroller and compatible peripheral relationships. As shown, the 8X401 can directly address up to 8K locations of program storage.

Each of the three bank pins ($\overline{A}$, $\overline{B}$, or $\overline{C}$) are capable of uniquely addressing 256 input/output locations via the Data Address bus ($\overline{DA7}$ – $\overline{DA0}$).

The addressable locations for each bank can be used in a variety of ways. The hookup shown below is just one method of implementation.

When a particular bank signal is asserted, that bank is enabled and any one of 256 locations on that bank can be accessed for input/output operations.

PROGRAM STORAGE INTERFACE

As shown in the 8X401 small system hookup, program memory is connected to output address lines A12 through A0 (A0 = LSB) and input instruction lines I19 through I0 (I0 = LSB). An address output on A12 – A0 identifies one 20-bit instruction word in program memory. The program memory outputs an instruction word on I19 – I0 which defines the microcontroller operation which is to follow. One instruction word equals one completed operation. Any TTL-compatible memory can be used for program storage, provided the worst-case access time is compatible with the instruction cycle time used for the application. See timing section for appropriate calculations.

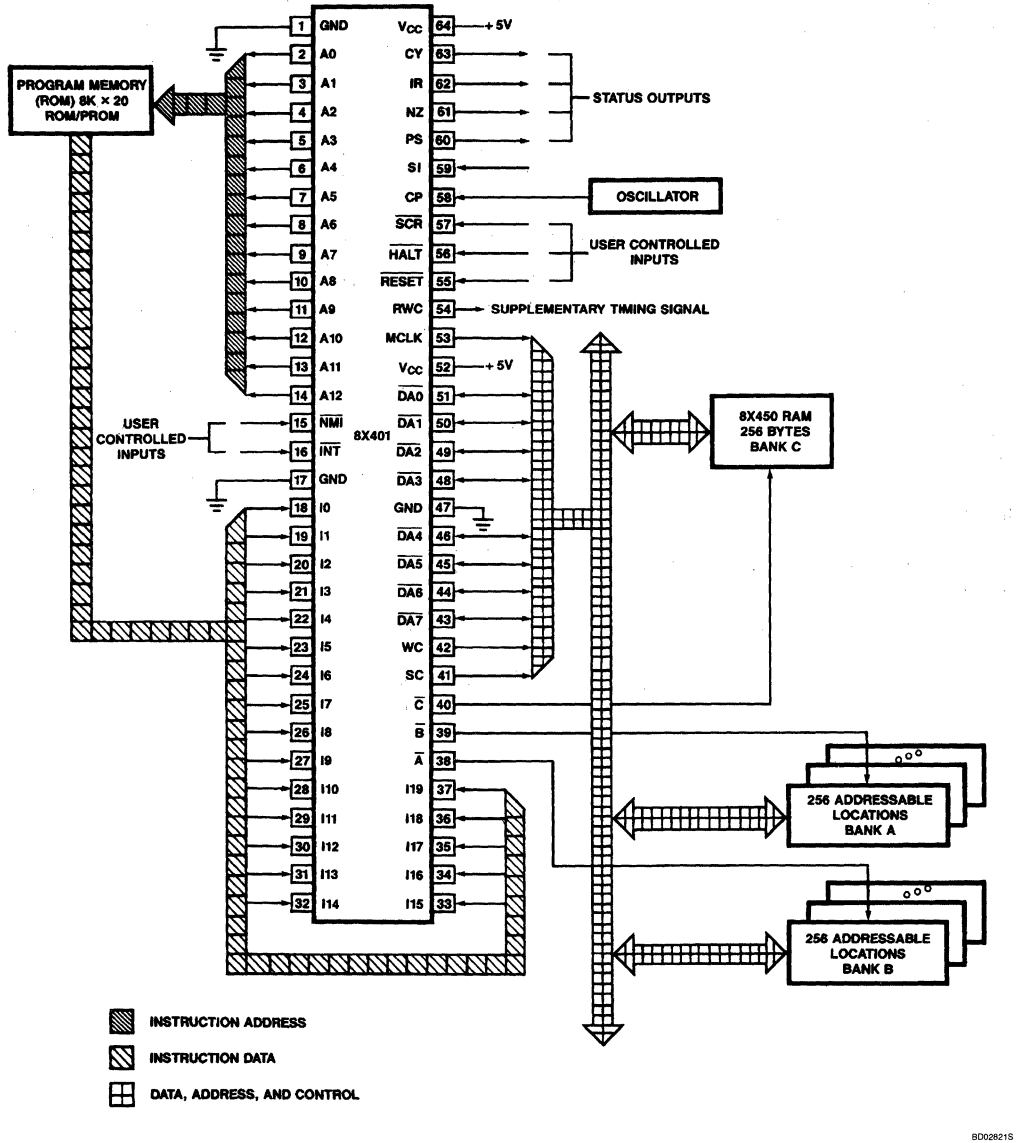
I/O INTERFACE AND CONTROL

The Data Address (DA) bus is an 8-bit bidirectional I/O bus which provides a communication link between the 8X401 and the three banks of the I/O devices. The $\overline{A}$ (A bank), $\overline{B}$ (B bank), and $\overline{C}$ (C bank) control signals identify which bank is enabled. When all three banks go high (inactive), neither bank is enabled and the DA bus is inactive (three-state). A functional analysis of the three bank signals is shown below:

A	B	C	FUNCTION
Low	Low	Low	This state is not generated by the 8X401.
Low	High	High	Enable A bank devices.
High	Low	High	Enable B bank devices.
High	High	Low	Enable C bank devices.
High	High	High	Disable all devices; DA bus is three-state

Both data and I/O address information are multiplexed on the DA bus. The SC (Select Command) and WC (Write Command) signals distinguish between data and I/O address information as shown in the table below. Although the table shows bank A only, the same conditions apply to banks B and C.

BANK A	SC	WC	FUNCTION
High	Low	Low	DA bus is three-state and not looking for input data.
Low	Low	Low	The DA bus is reading input data.
Low	Low	High	Data is being output.
Low	High	Low	Address is being output.
X	High	High	This condition is never generated.



8X401 Small System Hookup

DATA PROCESSING

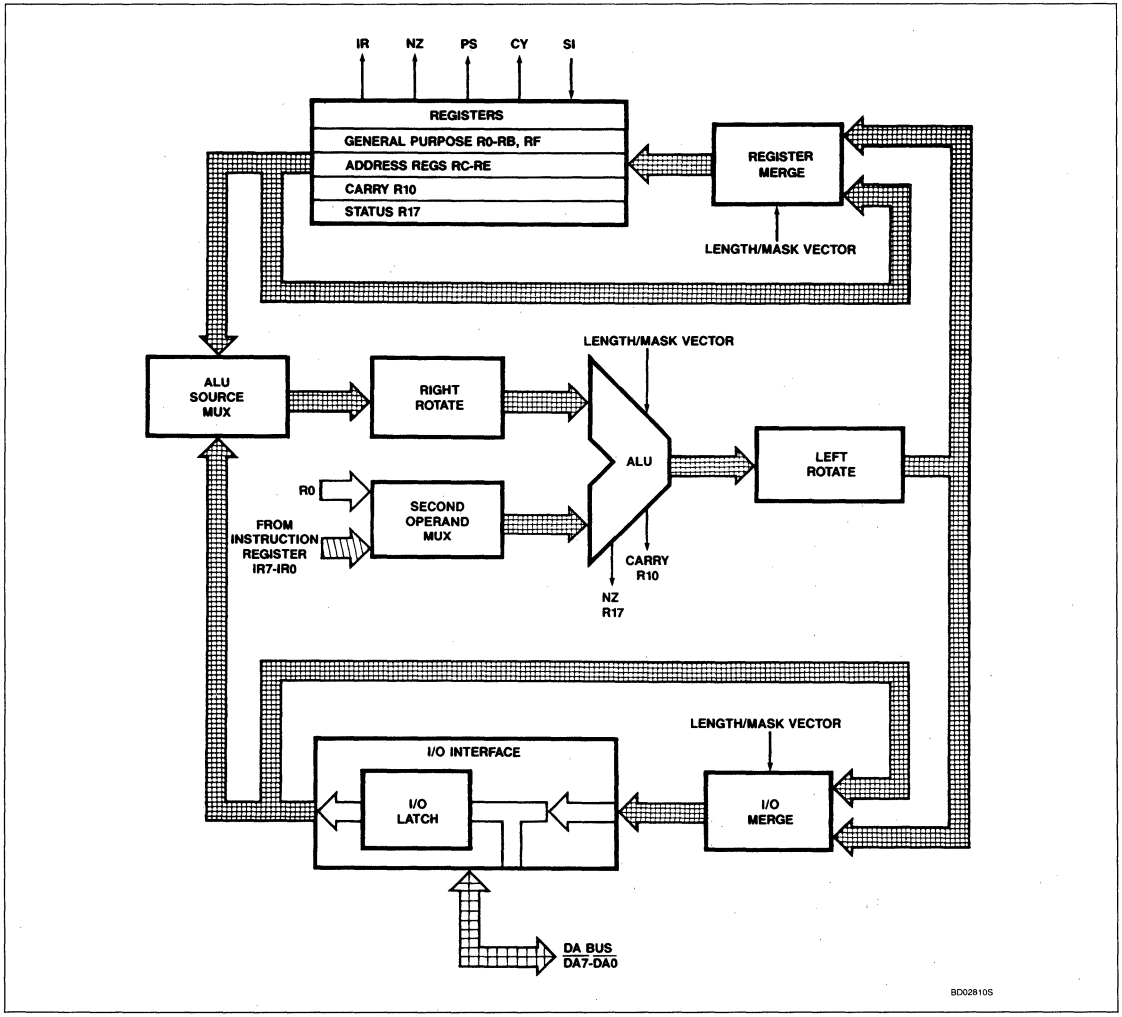
The data processing section of the 8X401 consists of a number of logical subsections. In order of processing, the data sees the right rotator, the ALU, the left rotator, and the merge circuits. Data sources and destinations can be various on-chip registers, the bidirectional DA bus, or immediate subfields. The data processing paths are shown below.

DATA REGISTERS

General-Purpose Storage

There are 13 source/destination general-purpose registers available on the 8X401. Three of these registers, specifically Registers 0, B, and F, have other special functions in addition to being general purpose. A summary of the registers is listed below:

- **R0 (Auxiliary Register)** — Register 0 is also used as the implied second operand for two operand instructions (ADD, ADD with CARRY, XOR, AND). The primary operand is specified in the source field of the instruction word and the AUX register is the implied second operand. Prior to performing arithmetic or logical operations (other than the IMMEDIATE operation), it is assumed that R0 contains the appropriate data. In order to reduce the possibility of erroneous results and to minimize the number of instructions required to transfer a right-justified second operand into the AUX, the left-rotate and merge functions are inhibited when specifying the AUX as a destination address. This allows subfields from any internal register or I/O bank to be transferred to the AUX with the subfield LSB right-justified and unspecified bits set to zero.
- **R1 through RA** — These 10 addresses specify general-purpose, on-chip storage registers.
- **RB** — Register B is also used as the implied source for the XEC instruction.
- **RF** — Register F is also used as the implied destination for the XOR IMMEDIATE and AND IMMEDIATE instruction classes.



Enabled I/O Addresses

These three registers (RC RD, and RE) always contain the address of the most recently enabled I/O device for each of the three I/O banks. When register C, D, or E is the specified destination address, the destination data is sent to both the on-chip register and the corresponding bank on the DA bus. The relationship between the register addresses and I/O banks is shown below:

REGISTER	BANK
C	A
D	B
E	C

When these registers are specified as a destination address, the L field must be set to 0 (full 8-bit operation). Also, note that registers C, D, and E may not be used with the XMIT 5 or ADD IMMEDIATE 5 instructions.

Carry

Register 10 contains the Carry bit. Bit position 0 (LSB) is the Carry bit, and positions one through seven are always zero. The Carry bit is updated each time an ADD, ADD IMMEDIATE, or ADD WITH CARRY instruction is performed. When specifying address 10 as a destination, only bit 0 (the Carry bit) will be written to. Data written to the Carry bit will be the LSB of the right-rotated data after any specified operation.

When the Carry register is the explicit destination of any ADD instruction, it will contain the carry resulting from the add operation rather than the LSB of the sum. Carry can also be affected via the Return and Set Carry or Return and Clear Carry instructions.

Status Register

This address specifies the current condition of the 8X401 system. The status register may be either a source or destination; however, certain bits in the status register are read-only. Four status outputs are available on 8X401 pins. They are NZ (Not Equal to Zero), PS (Programmable Status), IR (Interrupts Receivable), and Carry (R10, bit 0). The IR pin goes high when the interrupt mask is clear and the stack is not full. The IR output is updated during the 4th quarter cycle. The following descriptions define the bits within the status register.

Bit 0: (IM) — This bit represents the Interrupt Mask control. When IM is set, the interrupt is inhibited. This bit is set automatically by a response from a standard or non-maskable interrupt, or RESET. IM can also be set or cleared by a write to the status register.

Bit 1: (NZ) — This bit is set whenever the ALU output data is not equal to zero after any of the following instructions: MOVE, ADD, AND, XOR, ADD IMMEDIATE, AND IMMEDIATE, XOR IMMEDIATE, or ADD WITH CARRY. NZ can also be written to directly when specified in the destination field. This operation will negate and take priority over the normal setting by the ALU output. NZ is not affected after an XMIT instruction, or after a write to R17.

Bit 2: (PS) — This is the Programmable Status bit. The contents are reflected on the PS output pin. This status is controlled entirely by the user program.

Bits 3 and 4: (UF0, UF1) — These two bits represent user flags and have no assigned functions. They can be used as 1-bit internal flags and are entirely under control of the user program.

Bit 5: (SI) — This bit reflects the state of the status input pin. This read-only bit is updated during the 4th quarter cycle.

Bits 6 and 7: (SE, SF) — These read-only bits indicate Stack Empty and Stack Full, respectively. The bits are updated during the 3rd quarter cycle within the instruction that alters the stack status.

INSTRUCTION WORD (See Table 3)

Operations Code Field — The 4-bit opcode specifies one of 16 classes of instructions. Some instructions require two additional subopcode fields, X and XS. Variations and interpretations are displayed in Table 2.

Source (S) and Destination (D) Fields — The 5-bit "S" and "D" fields specify the source and destination, respectively, for the operation that is defined by the opcode. The "S" and/or "D" fields specify an internal 8X401 register or a variable length field from an I/O device. Hexadecimal values and source/destination field assignments for all internal registers are shown in Table 1.

When RC-RE (banks A, B, or C, respectively) are specified as the destination, the data is output onto the DA bus using the specified bank. The data is also stored in the specified register and may be later accessed as source data.

Rotate (R) and Length (L) Fields — The R field is used in conjunction with the L field to define the desired data within a register or I/O device. The source data is right-rotated prior to ALU operations, such that the bit specified by the R field is right-justified. The L field specifies the number of bits of data to be used for the operation. After the ALU operation, the data is left-rotated back to the original position prior to merging the data in the destination register.

When the L field specification is 0 (indicating a full 8-bit operation), the left-rotate is suppressed. This allows byte rotate operations to be performed. The left-rotate is also suppressed when the destination is register 0. This is the AUX register and is used as the implied second operand in certain instructions.

It should also be noted that subfields are defined at the ends of a register; for example, bit positions 1, 0, 7, and 6 constitute a contiguous 4-bit subfield.

Data Field — The data field holds data that can be processed directly from the instruction word.

LEFT-ROTATE OVERRIDE BLOCK

Register addresses 18-1F are destination only and are used to independently control left rotation of data prior to storage in the destination. Specifying 18-1F as a destination causes the data to be returned to the source address.

In order to move a processed subfield within the same register but in different bit positions (the LSB of the contiguous subfield can vary), it is necessary to independently specify the LSB for both the source and destination. The order of operation is as follows:

- Register or I/O source data is right-rotated as specified by the "R" field. Along with the "L" field, the subfield data is defined.
- Subfield data is processed via the ALU.
- Data is left-rotated 0 – 7 bits, depending on the corresponding register addresses 18-1F as specified in the destination field rather than using the "R" field.
- After left-rotation the specified subfield is merged into those bits of the original source data. The unspecified bits of the original source data remain unchanged.
- Result is stored in the register address specified by the source field.

Note that the left-rotate is always inhibited if the "L" field is zero. Also, addresses 18-1F may not be used in the destination field for the XMIT or ADD IMMEDIATE instructions. The destination addresses and corresponding left-rotate values are shown in Table 2.

DA BUS CONTROL BLOCK

Register addresses 11 – 16 are used by the 8X401 to access I/O devices for either a source or destination specified within the instruction. Register addresses 13, 15, and 16 specify banks A, B, and C, respectively, whereas addresses 11, 12, and 14 specify bank pairs AB, CA, and BC, respectively (Table 4). One bank of each pair is known as the preferred bank. The preferred banks for

pairs AB, CA, and BC are banks A, C, and B, respectively. The first letter from each bank pair can serve as a mnemonic aid as to which bank is the preferred bank. Having a preferred bank is simply a method of determining which bank to read when an instruction would otherwise indicate that two banks should be read at once.

When used as a source, the appropriate I/O bank is enabled and data is read from the activated I/O device on that bank. The I/O device may have been activated by a previous address select instruction where registers C-E were the specified destination. If a bank pair (addresses 11, 12, or 14) is specified as a source, only data from the preferred bank of that pair will be read in.

When addresses 11 – 16 are specified as the destination address, the destination data is sent to the DA bus. The Write Control (WC) signal goes high, indicating data (as opposed to an address) is on the DA bus and is to be written to the activated I/O device on the selected bank(s).

When addresses 11 – 16 are specified as the destination and the "L" field is not zero, the following statements apply: If the source is a register and the destination is a single bank, the bank will be read (or the preferred bank of a bank pair will be read) to obtain the data required to perform the merge operation. The result is that processed data from the specified subfield of the source register is returned to that selected field of the destination bank and any bits outside of the specified subfield will be loaded with unprocessed data from the I/O device just read. If the destination is a bank pair, the data from the procedure just described is sent to both banks. Below is an example of the outcome of an ADD instruction with Length = 4, Rotate = 2, Source = R1, and Destination = R11 (bank pair AB).

Example 1:

OPCODE	L	S	R	D
ADD	4	R1	2	R11

Bank Pair AB

	7	6	5	4	3	2	1	0
R1	a	b	c	d	e	f	g	h

Specified Subfield

Source register R1. Note specified subfield from L = 4 and R = 2.

R1 + R0	a'	b'	c'	d'	e'	f'	g'	h'

Add R1 with R0 (AUX). Result after left-rotate.

Bank A	i	j	k	l	m	n	o	p

Read preferred bank A of AB.

Bank A	i	j	c'	d'	e'	f'	o	p
Bank B	i	j	c'	d'	e'	f'	o	p

Result after merge. Data put out on both banks A and B.

If, however, the specified source is a bank or bank pair, any unspecified bits will contain unprocessed data from the source I/O device. Below is an example of the outcome with Source = R14 (bank pair BC) and Destination = R13 (bank A).

Example 2:

OPCODE	L	S	R	D
ADD	4	R14	2	R13

Bank Pair BC — Bank A

	7	6	5	4	3	2	1	0
Bank B	a	b	c	d	e	f	g	h

Specified Subfield

Read Preferred bank B of BC. Note specified subfield from L = 4 and R = 2.

Bank B + R0	a'	b'	c'	d'	e'	f'	g'	h'

Add bank B data with R0 (AUX). Result after left-rotate.

Bank A	a	b	c'	d'	e'	f'	g	h

Result after merge. Unspecified bits contain original source data.

PROGRAM COUNTER STACK

The 8X401 stack is capable of saving up to four return addresses for subroutines and interrupts. Addresses are pushed onto the stack as a result of a call or a maskable or non-maskable interrupt. Addresses are popped from the stack as a result of an unconditional RETURN, a satisfied conditional RETURN, or a Pop Stack and Jump instruction. The status of the stack (whether empty, full, or neither) is available from the SE and SF flags in the status register and the Interrupts Receivable (IR) output pin.

INTERRUPTS

Interrupt ($\overline{INT}$)

The interrupt input is tested once each instruction cycle, during the fourth quarter cycle (see Figure 1). When the interrupt input is taken low and is enabled, the address of the next instruction is pushed onto the program counter stack.

Program flow is transferred to address 2 for the start of the service routine (Figure 2). This is accomplished by inserting a dummy instruction cycle after the interrupt is accepted.

The interrupt mask bit (R17, bit 0) is set automatically as part of the interrupt response.

The Interrupts Receivable (IR) pin indicates whether an interrupt applied at any point in time will be serviced. Interrupts are receivable when the interrupt mask is clear and the stack is not full (IM = 0 and SF = 0).

Non-Maskable Interrupt ($\overline{NMI}$)

The function of the non-maskable interrupt is similar to the standard interrupt, except that the interrupt receivable status has no effect on its operation and the address jumped to is 1 rather than 2 (Figure 2). Address 1 should contain an unconditional JUMP to the start of the NMI service routine. An NMI is triggered by a falling edge on the $\overline{NMI}$ input. The interrupt mask is set to prevent normal interrupts from interfering with the NMI service routine. Note that it may not always be possible to recover from an NMI, since the condition of the interrupt mask prior to the NMI is not known, and the NMI response may overflow the stack.

SLOW CLOCK REQUEST ($\overline{SCR}$)

This control input is sampled during the first quarter cycle of each instruction along with the instruction data. If the input is low, it will cause the current instruction to be executed at half of the normal clock rate. The purpose of this function is to facilitate accesses to I/O devices that cannot operate at the 8X401's

full speed, without the need to run the 8X401 continuously at half speed.

HALT

The HALT input is sampled during the first quarter cycle of each instruction. If the HALT input is low, the instruction cycle is not executed. The MCLK continues to operate normally (high every fourth quarter cycle), even though program execution has ceased. When the HALT input goes high, program execution will resume at the next falling edge of MCLK. The DA bus is also inactive during a HALT operation. Like MCLK, RWC continues to operate during a HALT operation.

RESET LOGIC

The RESET pin is used to initialize the 8X401. When RESET is low, the address outputs (A12 – A0) are high impedance, the stack pointer is set to the top of the stack (empty), MCLK is inhibited, RWC is low, and the interrupt mask (bit 0, register 17) is set.

When RESET is released, the address outputs all go low (program address 0). A dummy instruction cycle occurs to allow time to fetch the first instruction from program storage at address 0. Only MCLK, RWC, and the address bus are in operation during the dummy cycle. The first active instruction cycle will begin following the first MCLK after RESET is released. The instruction at address 0 should be an unconditional jump to the beginning of the main program (which may be proceeded by a power-up sequence to initialize the system (Figure 2).

If RESET is applied during program execution, its effect is immediate. That is, if MCLK is high, it may be prematurely terminated by RESET.

Table 1. Hexadecimal Addresses and Source/Destination Specification

ADDRESS	DESIGNATION	S	D	ADDRESS	DESTINATION	S	D
00	R0 (AUX) — General Purpose[1]	X	X	10	R10 — Carry[6]	X	X
01	R1 — General Purpose	X	X	11	R11 — Bank Access Command (Bank Pair AB)	X	x
02	R2 — General Purpose	X	X	12	R12 — Bank Access Command (Bank Pair CA)	X	x
03	R3 — General Purpose	X	X	13	R13 — Bank Access Command (Bank A)	X	x
04	R4 — General Purpose	X	X	14	R14 — Bank Access Command (Bank Pair BC)	X	x
05	R5 — General Purpose	X	X	15	R15 — Bank Access Command (Bank B)	X	x
06	R6 — General Purpose	X	X	16	R16 — Bank Access Command (Bank C)	X	x
07	R7 — General Purpose	X	X	17	R17 — Status[4]	X	X
08	R8 — General Purpose	X	X	18	R18 — Suppress Left-Rotate[5]		X
09	R9 — General Purpose	X	X	19	R19 — Left-Rotate 1 Place[5]		X
0A	RA — General Purpose	X	X	1A	R1A — Left-Rotate 2 Places[5]		X
0B	RB — General Purpose[2]	X	X	1B	R1B — Left-Rotate 3 Places[5]		X
0C	RC — Address Reg (Bank A)	X	X	1C	R1C — Left-Rotate 4 Places[5]		X
0D	RD — Address Reg (Bank B)	X	X	1D	R1D — Left-Rotate 5 Places[5]		X
0E	RE — Address Reg (Bank C)	X	X	1E	R1E — Left-Rotate 6 Places[5]		X
0F	RF — General Purpose[3]	X	X	1F	R1F — Left-Rotate 7 Places[5]		X

NOTES:
1. Also used as implied second operand for two operand instructions.
2. Also used as implied source for XEC instructions.
3. Also used as implied destination for XOR IMMEDIATE and AND IMMEDIATE instructions.
4. Certain bits in the status register are read-only. (See Status Register within text.)
5. The result is returned to the register address specified by the source field.
6. Carry register, bit 0 is the carry bit. Bits 1 – 7 are always set to zero.

Table 2. Various of Instruction Types

INSTRUCTION TYPE	VARIATION	OPCODE	X	XS	DESCRIPTION
MOVE	MOV	0000	–	–	MOVE
ADD	ADD	0001	–	–	ADD
	ADC	0101	–	–	ADD with CARRY
	AD8	1000	–	–	ADD IMMEDIATE 8
	AD5	1001	–	–	ADD IMMEDIATE 5
AND	AND	0010	–	–	AND
	AN8	1010	–	–	AND IMMEDIATE 8
	AN5	1011	–	–	AND IMMEDIATE 5
XOR	XOR	0011	–	–	Exclusive-OR
	XR8	1100	–	–	Exclusive-OR IMMEDIATE 8
	XR5	1101	–	–	Exclusive-OR IMMEDIATE 5
XEC	XEC	0100	–	–	EXECUTE
XMIT	XT8	0110	–	–	Transmit IMMEDIATE 8
	XT5	0111	–	–	Transmit IMMEDIATE 5
RETURN	RIF NS	1110	000	–	RETURN IF SI = 0
	RIF S	1110	001	–	RETURN IF SI = 1
	RIF NC	1110	010	–	RETURN IF CARRY = 0
	RIF C	1110	011	–	RETURN IF CARRY = 1
	RIF Z	1110	100	–	RETURN IF ALU = 0
	RIF NZ	1110	101	–	RETURN IF ALU ≠ 0
	PSJ	1110	110	–	POP STACK and JUMP
	RTN	1110	111	00	RETURN
	RCC	1110	111	10	RETURN and CLEAR CARRY
	RSC	1110	111	11	RETURN and SET CARRY
JUMP	JIF NS	1111	000	–	JUMP IF SI = 0
	JIF S	1111	001	–	JUMP IF SI = 1
	JIF NC	1111	010	–	JUMP IF CARRY = 0
	JIF C	1111	011	–	JUMP IF CARRY = 1
	JIF Z	1111	100	–	JUMP IF ALU = 0
	JIF NZ	1111	101	–	JUMP IF ALU ≠ 0
	JSR	1111	110	–	JUMP to SUBROUTINE
	JMP	1111	111	–	JUMP

Instruction Set Overview: The 8X401 instruction set is summarized in Table 2. Subsets of each instruction type are grouped together showing the variations of each instruction type. The hardware and software descriptions can be found in the data operations section.

Table 3. 8X401 Instruction Formats

1. Format for the MOVE, ADD, XOR, and ADD with Carry Instructions:

OPCODE	L	SOURCE	R	DESTINATION

2. Format for the XMIT 8 and ADD Immediate 8 Instruction:

OPCODE	L	DESTINATION	DATA8

3. Format for the XMIT 5 and ADD Immediate 5 Instruction:

OPCODE	L	DESTINATION	R	DATA5

4. Format for the AND Immediate 8, and XOR Immediate 8 Instruction:

OPCODE	L	SOURCE	DATA8

5. Format for the AND Immediate 5 and XOR Immediate 5 Instruction:

OPCODE	L	SOURCE	R	DATA5

6. Format for the JUMP, Subroutine Jump, Conditional Jump, and Conditional Return Instruction:

OPCODE	X	ADDRESS

7. Format for the Unconditional Return, Return and Set Carry, and Return and Clear Carry Instruction:

OPCODE	X	XS	(UNUSED

8. Format for the XEC Instruction:

OPCODE	L	ADDRESS

Table 4. I/O Acess Register

REGISTER	INPUT BANK	OUTPUT BANK
11	A	A & B
12	C	C & A
13	A	A
14	B	B & C
15	B	B
16	C	C

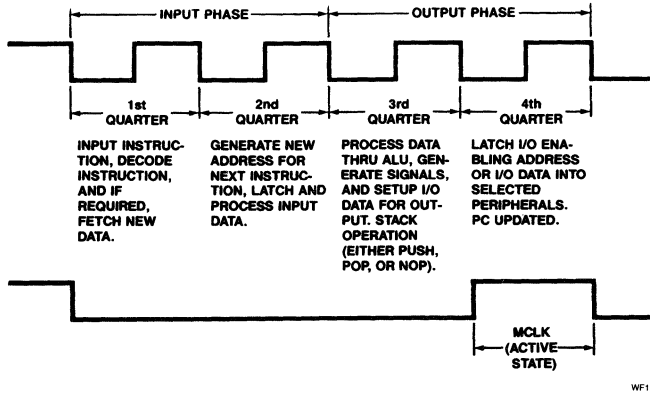


Figure 1. Instruction Cycle and MCLK with: Clock Input = 26.67MHz and Cycle Time = 150ns

DATA OPERATIONS OF THE 8X401 (See Tables 2 and 3)

MOVE, ADD, AND, XOR, ADD with CARRY

SOURCE	PRE-ALU	ALU	POST-ALU	DESTINATION
R0 – R17 as specified by "S" field of instruction.	Right rotate as specified by "R" field of instruction.	Perform appropriate ALU operation. (Note 1)	Left-rotate and mask as specified by the "R" and "L" fields of instruction.	R0 – R17 as specified by the "Destination" field of instruction. (Notes 2 & 3)

MOVE, ADD, AND, XOR, ADD with CARRY (Using left-rotate override R18 – R1F)

SOURCE	PRE-ALU	ALU	POST-ALU	DESTINATION
R0 – R17 as specified by "S" field of instruction.	Right-rotate as specified by "R" field of instruction.	Perform appropriate ALU operation. (Note 1)	Left-rotate and mask as specified by the "D" and "L" fields of instruction. (Note 4)	When R18 – R1F are specified in the "D" field, data is returned to the address specified in the "Source" field.

XMIT 8 (XT8), ADD IMMMEDIATE 8 (AD8), AND IMMEDIATE 8 (AN8), XOR IMMEDIATE 8 (XR8)

SOURCE	PRE-ALU	ALU	POST-ALU	DESTINATION
One to eight bit constants from data field of instruction and register/ address specified by either the source or destination field.	NOP.	Perform appropriate ALU operation. (Note 5)	Mask as specified by the "L" field.	**XT8, AD8:** Results are sent to register/address, starting from LSB, specified by the destination field of the instruction word. **AN8, XR8:** Results are sent to RF, starting from LSB.

XMIT 5 (XT5), ADD IMMMEDIATE 5 (AD5), AND IMMEDIATE 5 (AN5), XOR IMMEDIATE 5 (XR5)

SOURCE	PRE-ALU	ALU	POST-ALU	DESTINATION
One to five bit constants from data field of instruction and register/ address specified by either the source or destination.	Right-rotate as specified by "R" field, the data defined in either the source or destination field.	Perform appropriate ALU operation. (Note 6)	Left-rotate and mask as specified by the "R" and "L" fields.	**XT5, AD5:** Subfield is merged into register/ address specified by destination field. LSB of data field is sent to bit position of destination defined by "R" field. **AN5, XR5:** Subfield is merged into register/ address specified by source field. The "R" field specifies the LSB position of source and RF. Results are returned to RF.

ALL CONDITIONAL AND UNCONDITIONAL RETURNS

ADDRESS REGISTER (AR)	PROGRAM COUNTER (PC)	STACK
Conditional Returns: If condition is true. AR = address from top of stack, else load address from instruction word. **Unconditional Returns:** AR = address from top of stack.	PC = AR.	**Conditional Returns:** If condition is true. POP stack, else NOP. **Unconditional Returns:** POP stack.

ALL CONDITIONAL JUMPS, POP STACK AND JUMP (PSJ), and JUMP (JMP)

ADDRESS REGISTER	PROGRAM COUNTER	STACK
Conditional Jumps: If condition is true. AR = instruction word address, else AR = PC + 1. **PSJ and JMP:** AR = instruction word address.	PC = AR.	**Conditional Jumps:** NOP. **PSJ:** POP stack. **JMP:** NOP.

DATA OPERATIONS OF THE 8X401 (Continued)

CALL (JSR), INTERRUPT ($\overline{INT}$), NON-MASKABLE INTERRUPT ($\overline{NMI}$)

ADDRESS REGISTER	STACK	PROGRAM COUNTER
JSR: AR = instruction address. **$\overline{INT}$:** AR = address 2. **$\overline{NMI}$:** AR = address 1.	PUSH PC + 1. (Note 6)	PC = AR.

XEC

ADDRESS REGISTER	PROGRAM COUNTER	STACK
Right-most L bits of RB merged into corresponding bits in instruction address	PC not updated.	NOP.

NOTES:

1. **ALU Descriptions:**
   - MOVE: • No operation
   - ADD: • Source data ADDed to contents of auxiliary register (R0 – AUX). Carry bit set if carry is generated at MSB of selected data field. NZ status bit set if specified bits are not zero after ALU add.
   - AND: • Source data ANDed to contents of AUX register. NZ status bit updated accordingly.
   - XOR: • Source data Exclusive-ORed with contents of AUX register. NZ status bit set accordingly.
   - ADD with CARRY: • Sum is formed from source data. AUX register, and carry bit (register 10, bit 0). Carry and NZ status bits are set when appropriate.
2. Left-rotate is suppressed when destination is R0 (AUX).
3. When address registers RC, RD and RE are specified in the destination, source data will also go out on banks A, B, C, respectively. The L-field should be zero (a full 8-bit operation) to ensure duplication of the two outputs.
4. A left-rotate of 0 – 7 bits will correspond to R18 – R1F as specified in the "Destination" field of instruction word.
5. **ALU Descriptions:**
   - XMIT: • Input constants from the instruction word to specified destination. NZ flag is not updated when an XMIT is performed: however, NZ can be written to by an XMIT if R17 bit 1 is within the destination field.
   - ADD IMMEDIATE: • Instruction word data is ADDed to data specified by destination fied. The carry bit is set if a carry is generated at the MSB of the selected data field. NZ status bit is updated to reflect the value of "L" bits of data after the addition.
   - AND IMMEDIATE: • Instruction word data is ANDed to data specified by source field. Returning the destination data to RF allows the operation to be performed without destroying the original data field. This will facilitate testing of data for certain pre-defined values while still preserving the original data for other uses. NZ status bit updated accordingly. Unspecified bits in RF remain unchanged.
   - XOR IMMEDIATE: • Same as AND IMMEDIATE, except the logical operation performed is Exclusive-OR.
6. Note that the stack operation is show before the PC in the CALL and INTERRUPT formats. This is because the stack is actually in operation in cycle 3, and the PC is updated in cycle 4 (see Figure 1). In fact, for the Call (JSR) instruction and interrupt servicing, cycle order is important for the user to understand the current status of the PC. The other instructions are in reverse order for visual simplicity in keeping with block diagram flow, and cycle order is irrelevant.

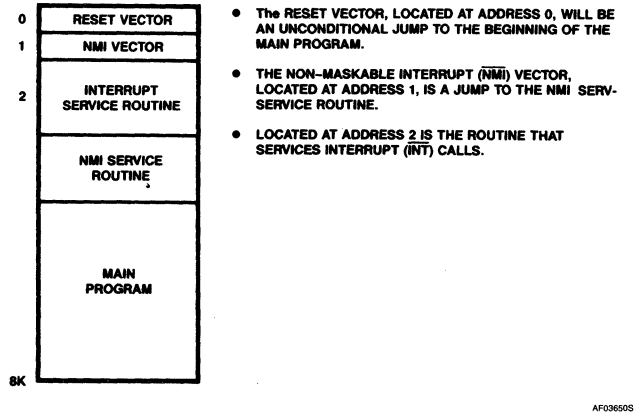
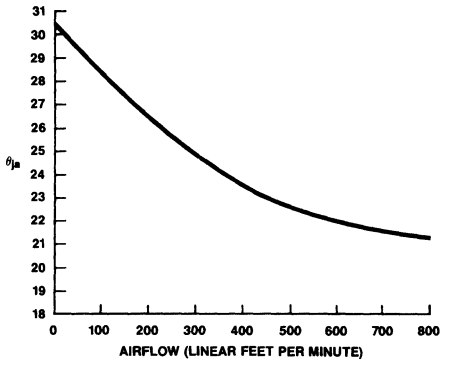


Figure 2. Typical 8X401 System Memory Map

Thermal Junction Temperature vs Airflow

The ceramic package used for the 8X401 has no heat sink and a θ_{ja} rating of 30.5°C/W in still air. Currently, to ensure operation at 150ns, the junction temperature of the devices must be kept below 115°C. The maximum power dissipation at that junction temperature will be 2.4W so that airflow will be required for full commercial range operation. The θ_{ja} versus Airflow curve is drawn here:



DC ELECTRICAL CHARACTERISTICS Commercial Part $4.75V \leq V_{CC} \leq 5.25V$, $0°C \leq T_A \leq 70°C$[1]

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT	COMMENTS
			Min	Typ	Max		
V_{CC}	Supply voltage		4.75	5	5.25	V	
V_{IH}	High-level input voltage		2			V	
V_{IL}	Low-level input voltage				0.8	V	
V_{OH}	High-level output voltage	V_{CC} = Min, I_{OH} = −3mA	2.4			V	$\overline{DA0}$ through $\overline{DA7}$, MCLK, SC, WC, AB, BB, CB
		V_{CC} = Min, I_{OH} = −400μA	2.4			V	All others
V_{OL}	Low-level output voltage	V_{CC} = Min, I_{OL} = 16mA			0.5	V	$\overline{DA0}$ through $\overline{DA7}$, MCLK, RWC, SC, WC, $\overline{A}$, $\overline{B}$, $\overline{C}$
		V_{CC} = Min, I_{OL} = 8mA			0.5	V	A0 through A12, PS, NZ, CY, IR
V_{IC}	Input clamp voltage	V_{CC} = Min, I_{IN} = −10mA			−1.5	V	
I_{IH}	High-level input current	V_{CC} = Max, V_I = 2.7			20	μA	
I_{IL}	Low-level input current	V_{CC} = Max, V_I = 0.4V			−400	μA	
I_{OZH}	Off-state output current, high-level voltage applied	V_{CC} = Max, V_O = 2.7V			50	μA	$\overline{DA0}$ through $\overline{DA7}$
I_{OZL}	Off-state output current low-level voltage applied	V_{CC} = Max, V_O = 0.4V			−400	μA	$\overline{DA0}$ through $\overline{DA7}$
I_{OS}	Short circuit output current[2]	V_{CC} = Max, V_O = 0V	−30		−140	mA	
I_{CC}	Supply Current	V_{CC} = Max			500	mA	T_A = 0°C; Cold start[3]
					430	mA	T_J = 115°C

NOTES:
1. 64-pin CDIP, airflow required for commercial operation. (The plastic 64-pin DIP with internal heatsink does not have this requirement.) See above for thermal characteristics.
2. Not more than one output should be tested at a time.
3. Guaranteed by operation to Icc measured at 25°C.

AC ELECTRICAL CHARACTERISTICS (V_{CC} = 5V +5%, o°C ≤ T_A ≤ 70°C)[1]

SYMBOL	PARAMETER	150ns CYCLE			> 150ns CYCLE			UNIT	COMMENTS
		Min	Typ	Max	Min	Typ	Max		
t_{PC}	Processor cycle time	150						ns	
t_{CP}	Clock pulse period	37.5						ns	
t_{CH}	Clock pulse high time	15						ns	See Note 2
t_{CL}	Clock pulse low time	15						ns	
t_{MCL}	CP1 to MCLK low			30				ns	See Note 2
t_{MCH}	CP4 to MCLK high			40				ns	
t_W	MCLK pulse width	25	31	38	T4Q – 12		T4Q	ns	
t_{RWCL}	CP1 to RWC low		34	40				ns	
t_{RWCH}	CP3 to RWC high			40				ns	
t_{RWCW}	RWC pulse width	65	75	80	T3Q + T4Q – 10		T3Q + T4Q + 5	ns	
t_{AS}	CP2 to address stable			52				ns	
t_{MAS}	MCLK to address stable			62			T1Q + 24	ns	
t_{IS}	Instruction setup to CP1	0						ns	See Note 3
t_{MIS}	Instruction setup to MCLK	25						ns	
t_{IH}	Instruction hold from CP2	20						ns	
t_{MIH}	Instruction hold from MLCK	25			T1Q – 12			ns	
t_{SCH}	CP3 to SC rising edge			45				ns	
t_{MSCH}	MCLK to SC rising edge			95			T1Q + T2Q + 20	ns	
t_{WCH}	CP3 to WC rising edge			55				ns	
t_{MWCH}	MCLK to WC rising edge			105			T1Q + T2Q + 20	ns	
t_{WL}	CP1 to SC/WC falling edge			35				ns	See Note 4
t_{MWL}	MCLK to SC/WC falling edge	0						ns	
t_{IBSL}	CP1 to input phase bank signal falling edge			60				ns	
t_{MIBSL}	MCLK to input phase bank signal falling edge			33				ns	
t_{IBSH}	CP3 to input phase bank signal rising edge			45				ns	
t_{MIBSH}	MCLK to input phase bank signal rising edge			95			T1Q + T2Q + 20	ns	
t_{OBSL}	CP3 to output phase bank signal falling edge			53				ns	
t_{MOBSL}	MCLK to output phase bank signal falling edge			105			T1Q + T2Q + 30	ns	
t_{OBSH}	CP1 to output phase bank signal rising edge			46				ns	
t_{MOBSH}	MCLK to output phase bank signal rising edge	0		20				ns	
t_{IDS}	Input data setup to CP3	–3						ns	
t_{MIDS}	Input data setup to MCLK	–50			25 – T1Q – T2Q			ns	
t_{IDH}	Input data hold from CP3	28						ns	
t_{MIDH}	Input data hold from MCLK	78			T1Q + T2Q + 3			ns	
t_{ODH}	Output data hold from CP1	35		55				ns	
t_{MODH}	Output data hold from MCLK	10		25				ns	

AC ELECTRICAL CHARACTERISTICS (Continued)

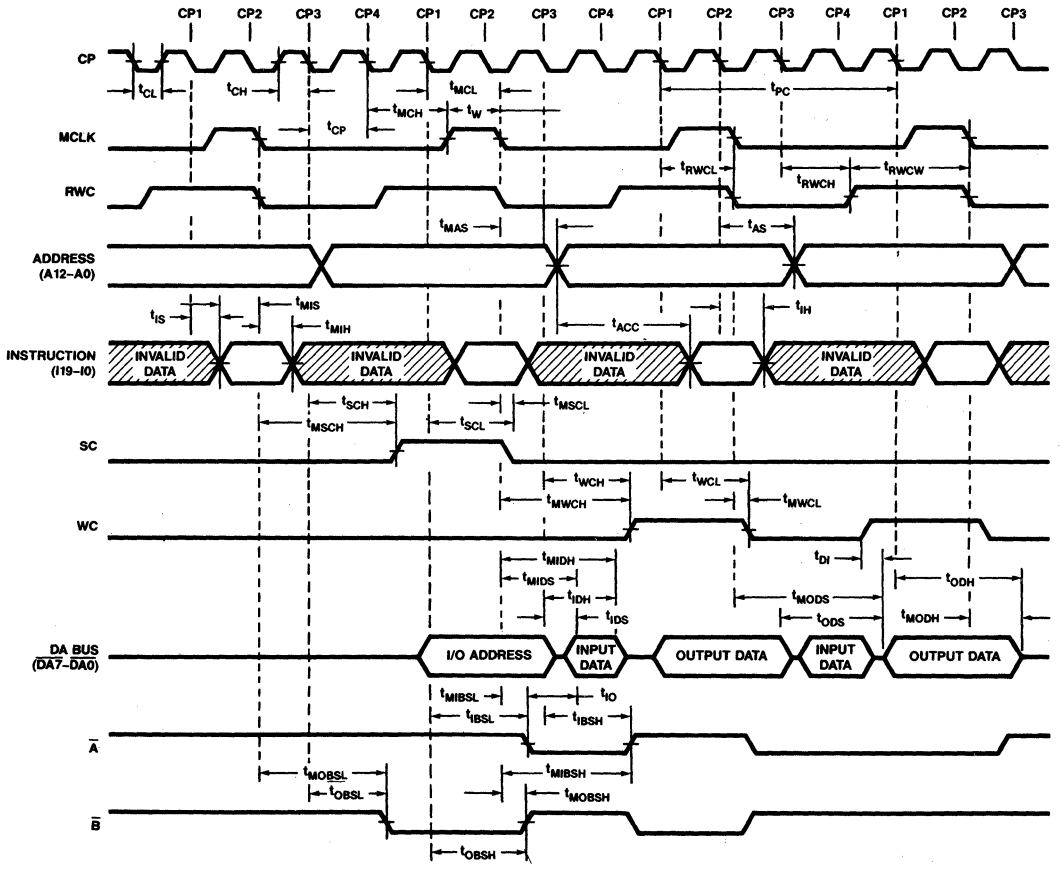
SYMBOL	PARAMETER	150ns CYCLE			> 150ns CYCLE			UNIT	COMMENTS
		Min	Typ	Max	Min	Typ	Max		
t_{ODS}	CP3 to output data stable			70				ns	
t_{MODS}	MCLK to output data stable			120			T1Q + T2Q + 45	ns	
t_{DI}	SC/WC rising edge to output driver turn on	18						ns	
t_{HS}	Halt setup to CP2	0						ns	
t_{MHS}	Halt setup to MCLK	-10						ns	
t_{HH}	Halt hold from CP2	50						ns	
t_{MHH}	Halt hold from MCLK	60			T1Q + 22			ns	
t_{SIS}	Status input setup to CP1	10						ns	
t_{MSIS}	Status input setup to MCLK	40						ns	
t_{SIH}	Status input hold from CP1	20						ns	
t_{MSIH}	Status input hold from MCLK	0						ns	
t_{SCRS}	SCR setup to CP1	0						ns	
t_{MSCRS}	SCR setup to MCLK	25						ns	
t_{SCRH}	SCR hold from CP2 (Slow CP2)	20						ns	See Diagram for CP2
t_{MSCRH}	SCR hold from MCLK	63			ST1Q - 12			ns	Slow T1Q
t_{INTS}	INT setup to CP1	10						ns	
t_{MINTS}	INT setup to MCLK	40						ns	
t_{INTH}	INT hold from CP1	10						ns	
t_{MINTH}	INT hold from MCLK	-10						ns	
t_{CYU}	CP4 to CY update			60				ns	
t_{MCYU}	MCLK to CY update			-10			28 - T4Q	ns	
t_{NZU}	CP4 to NZ update			60				ns	
t_{MNZU}	MCLK to NZ update			-5			33 - T4Q	ns	
t_{IRU}	CP4 to IR update			75				ns	
t_{MIRU}	MCLK to IR update			20			58 - T4Q	ns	
t_{PSU}	CP4 to PS update			60				ns	
t_{MPSU}	MCLK to PS update			-10			28 - T4Q	ns	
t_{ACC}	Program memory access time (address stable to valid instruction)			60		T2Q + T3Q + T4Q - 52			ns
t_{IO}	I/O port output enable time (bank signal to valid data on bus)			24			T1Q + T2Q - 51	ns	
t_{RW}	Reset pulse width	150			t_{PC}			ns	
t_{NMIW}	NMI pulse width	50						ns	
t_{NMIS}	NMI setup to CP2	15						ns	See Note 5
t_{MNMIS}	NMI setup to MCLK	10						ns	See Note 5

NOTES:

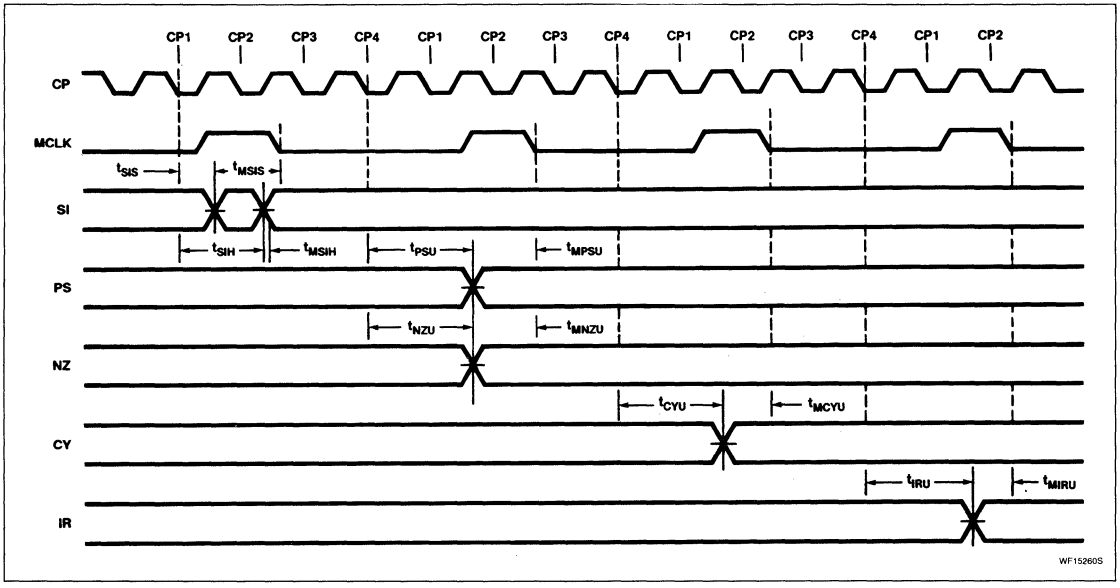
1. Inputs swing between 0V and 3V. All outputs are measured at 1.5V with loading as specified in the test circuits.
2. CP1, CP2, CP3, and CP4 refer to the clock pulse that causes the first, second, third, and fourth 8X401 quarter cycles, respectively. Parameters referenced to MCLK, CP1, CP2, CP3, and CP4 are measured to the falling edge of those signals. T1Q, T2Q, T3Q, and T4Q represent time intervals for the first, second, third, and fourth 8X401 quarter cycles, respectively. Duty cycle can be from 40% to 60%.
3. Instructions must be setup before CP1.
4. t_{WL} represents t_{SCL} and t_{WCL}. t_{MWL} represents t_{MSCL} and t_{MWCL}.
5. This guarantees NMI is serviced in the current cycle.

TEST CIRCUIT

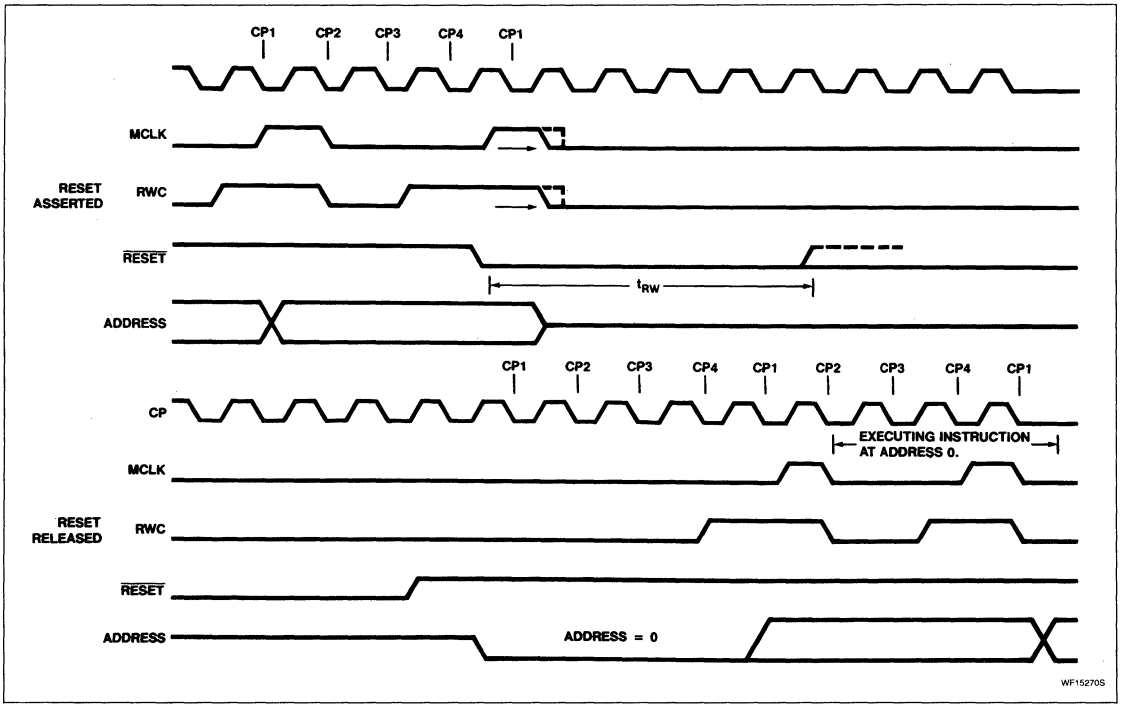
MAIN TIMING DIAGRAM



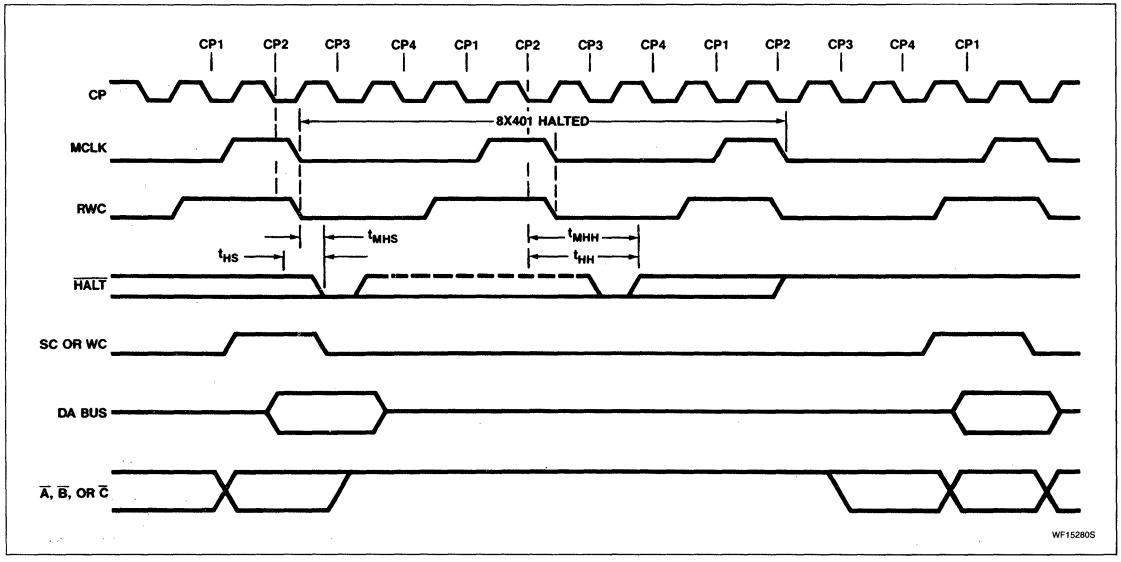
TIMING SUPPLEMENT: STATUS



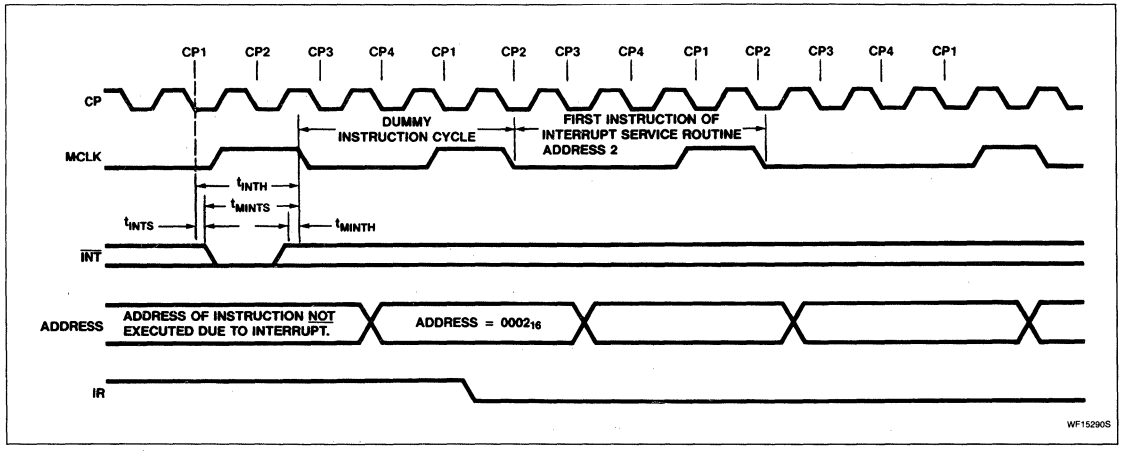
TIMING SUPPLEMENT: $\overline{\text{RESET}}$



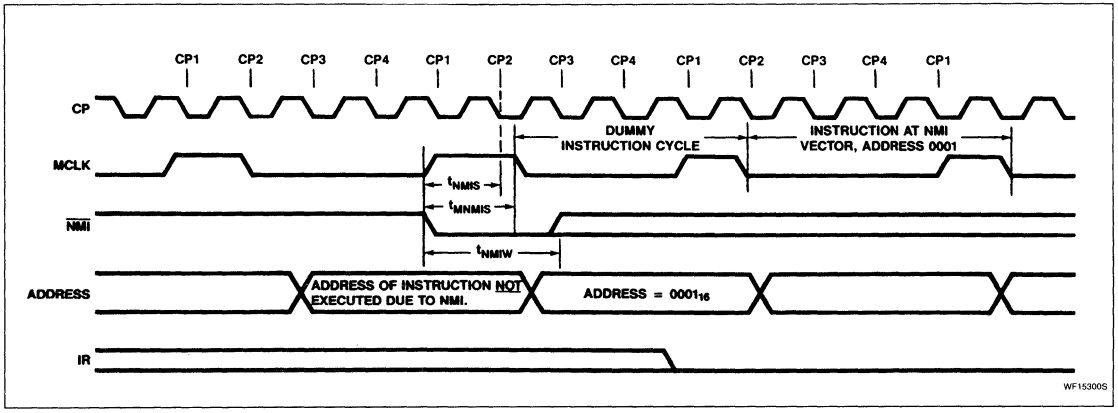
TIMING SUPPLEMENT: $\overline{\text{HALT}}$



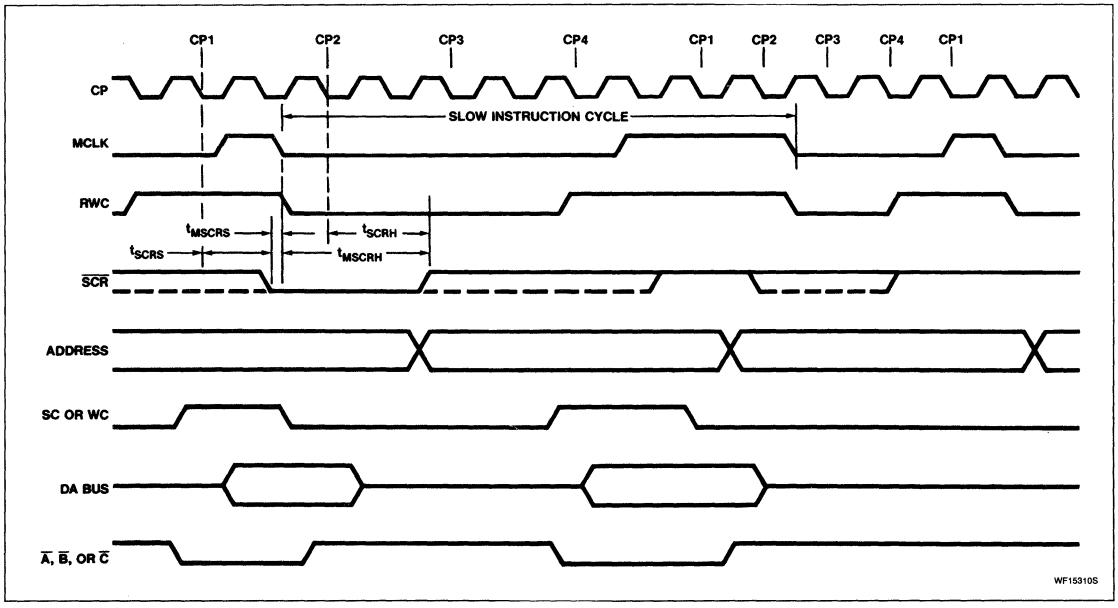
TIMING SUPPLEMENT: $\overline{\text{INT}}$



TIMING SUPPLEMENT: $\overline{NMI}$



TIMING SUPPLEMENT: $\overline{SCR}$



Signetics

SCN8049 Series

SCN8049, SCN8050, SCN8039, SCN8040 Single-Chip 8-Bit Microcontroller

Microprocessor Products

Product Specification

DESCRIPTION

The Signetics SCN8049 Series microcontrollers are self-contained, 8-bit processors which contain the system timing, control logic, RAM data memory, ROM program memory (8048/49/50 only), and I/O lines necessary to implement dedicated control functions. All SCN8049 Series devices are pin and program compatible, differing only in the size of the on-board program ROM and data RAM, as follows:

TYPE	RAM SIZE	ROM SIZE
SCN8049	128 × 8	2K × 8
SCN8050	256 × 8	4K × 8
SCN8039	128 × 8	—
SCN8040	256 × 8	—

Program memory can be expanded externally up to a maximum total of 4K bytes without paging. Data memory can also be expanded externally. I/O capabilities can be expanded using standard devices or the 8243 I/O expander.

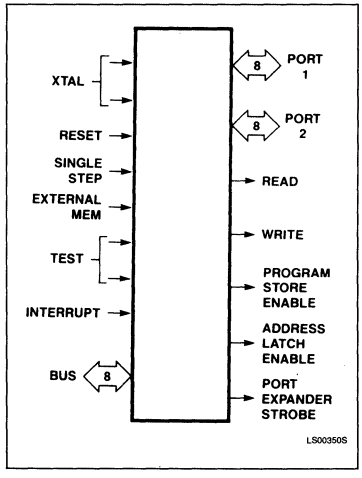
The SCN8049 Series processors are designed to be efficient control processors as well as arithmetic processors. They provide an instruction set which allows the user to directly set and reset individual lines within its I/O ports as well as test individual bits within the accumulator. A large variety of branch and table look-up instructions make these processors very efficient in implementing standard logic functions. Also, special attention has been given to code efficiency. Over 70% of the instructions are a single byte long and all others are only 2 bytes long.

An on-chip 8-bit counter is provided which can count, under program control, either internal clock pulses (with a divide by 32 prescaler) or external events. The counter can be programmed to cause an interrupt on terminal count.

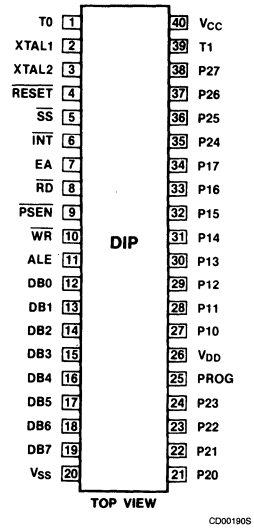
FEATURES

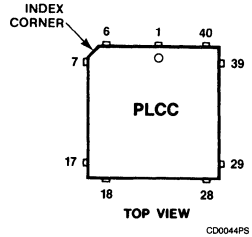
- **8-bit CPU, ROM, RAM, I/O in a 40-pin package**
- **24 quasi bidirectional I/O lines**
- **Two test inputs**
- **Internal counter/timer**
- **Single-level vectored interrupts: external, counter/timer**
- **Over 90 instructions, 70% single byte**
- **1.36μs or 2.5μs instruction cycle, all instructions one or two cycles**
- **Expandable memory and I/O**
- **Low voltage standby**
- **TTL compatible inputs and outputs**
- **Single +5V power supply**

LOGIC SYMBOL



PIN CONFIGURATIONS

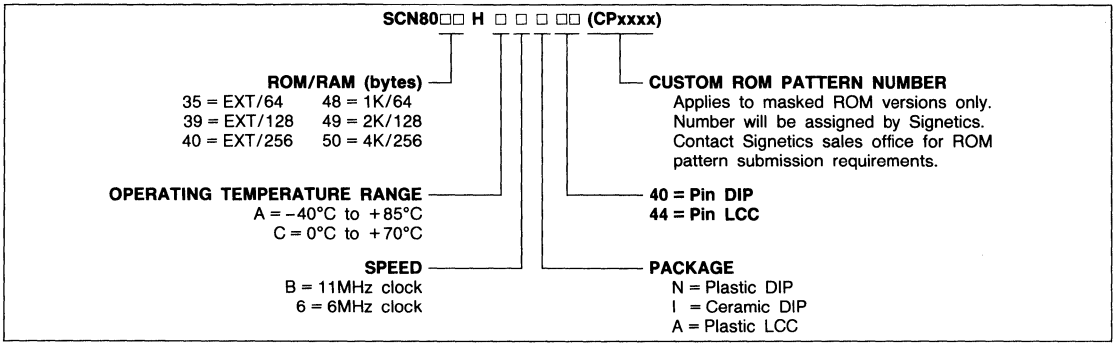




Pin	Function	Pin	Function
1	NC	23	NC
2	T0	24	P20
3	XTAL1	25	P21
4	XTAL2	26	P22
5	$\overline{\text{RESET}}$	27	P23
6	$\overline{\text{SS}}$	28	PROG
7	$\overline{\text{INT}}$	29	V_{DD}
8	EA	30	P10
9	$\overline{\text{RD}}$	31	P11
10	$\overline{\text{PSEN}}$	32	P12
11	$\overline{\text{WR}}$	33	P13
12	NC	34	NC
13	ALE	35	P14
14	DB0	36	P15
15	DB1	37	P16
16	DB2	38	P17
17	DB3	39	P24
18	DB4	40	P25
19	DB5	41	P26
20	DB6	42	P27
21	DB7	43	T1
22	V_{SS}	44	V_{CC}

SCN8049, SCN8050, SCN8039, SCN8040 Single-Chip 8-Bit Microcontroller

ORDERING INFORMATION



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. PLCC	TYPE	NAME AND FUNCTION
V_{SS}	20	22		Circuit ground potential.
V_{DD}	26	29		Low power standby.
V_{CC}	40	44		**Main Power Supply:** +5V during operation.
PROG	25	28	O	Output strobe for 8243 I/O expander.
P10 – P17	27 – 34	30 – 33, 35 – 38	I/O	**Port 1:** 8-bit quasi-bidirectional port.
P20 – P27	21 – 24, 35 – 38	24 – 27, 39 – 42	I/O	**Port 2:** 8-bit quasi-bidirectional port. P20-23 contain the four high-order program counter bits during an external program memory fetch and serve as a 4-bit I/O expander bus for 8243.
DB0 – DB7	12 – 19	14 – 21	I/O	**Data Bus:** True bidirectional port which can be written or read synchronously using the $\overline{RD}$, $\overline{WR}$ strobes. The port can also be statically latched. Contains the eight low-order program counter bits during an external program memory fetch and receives the addressed instruction under the control of $\overline{PSEN}$. Also contains the address and data during an external RAM data store instruction, under control of ALE, $\overline{RD}$ and $\overline{WR}$.
T0	1	2	I	Input pin testable using the conditional transfer instructions JT0 and JNT0. T0 and be designated as a clock output using the ENT0 CLK instruction.
T1	39	43	I	Input pin testable using the JT1 and JNT1 instructions. Can be designated the timer/counter input using the STRT CNT instruction.
XTAL1	2	3	I	**Crystal 1:** One side of the crystal input for internal oscillator. Also input for external source (non-TTL V_{IH}).
XTAL2	3	4	I	**Crystal 2:** Other side of crystal input.
$\overline{INT}$	6	7	I	**Interrupt:** Initiates an interrupt if interrupt is enabled. Interrupt is disabled after a reset. Also testable with conditional jump instruction. Interrupt must remain low for at least three machine cycles for proper operation.
$\overline{RESET}$	4	5	I	**Reset:** Used to initialize the microcomputer. Active low. Internal pullup ~75KΩ. During program verification the address is latched by a "0" to "1" transition on $\overline{RESET}$ and the data at the addressed location is output on BUS.
$\overline{RD}$	8	9	O	**Read:** Output strobe activated during a bus read. Can be used to enable data onto the bus from an external device. Used as a read strobe to external data memory.
$\overline{WR}$	10	11	O	**Write:** Output strobe during a bus write. Used as write strobe to external data memory.
ALE	11	13	O	**Address Latch Enable:** Occurs once during each cycle and is useful as a clock output. The negative edge of ALE strobes address into external data and program memory.
$\overline{PSEN}$	9	10	O	**Program Store Enable:** Output occurs only during a fetch to external program memory.
$\overline{SS}$	5	6	I	**Single Step:** Can be used in conjunction with ALE to "single step" the processor through each instruction.
EA	7	8	I	**External Access:** Forces all program memory fetches to reference external memory. Useful for emulation and debug, and essential for testing and program verification.

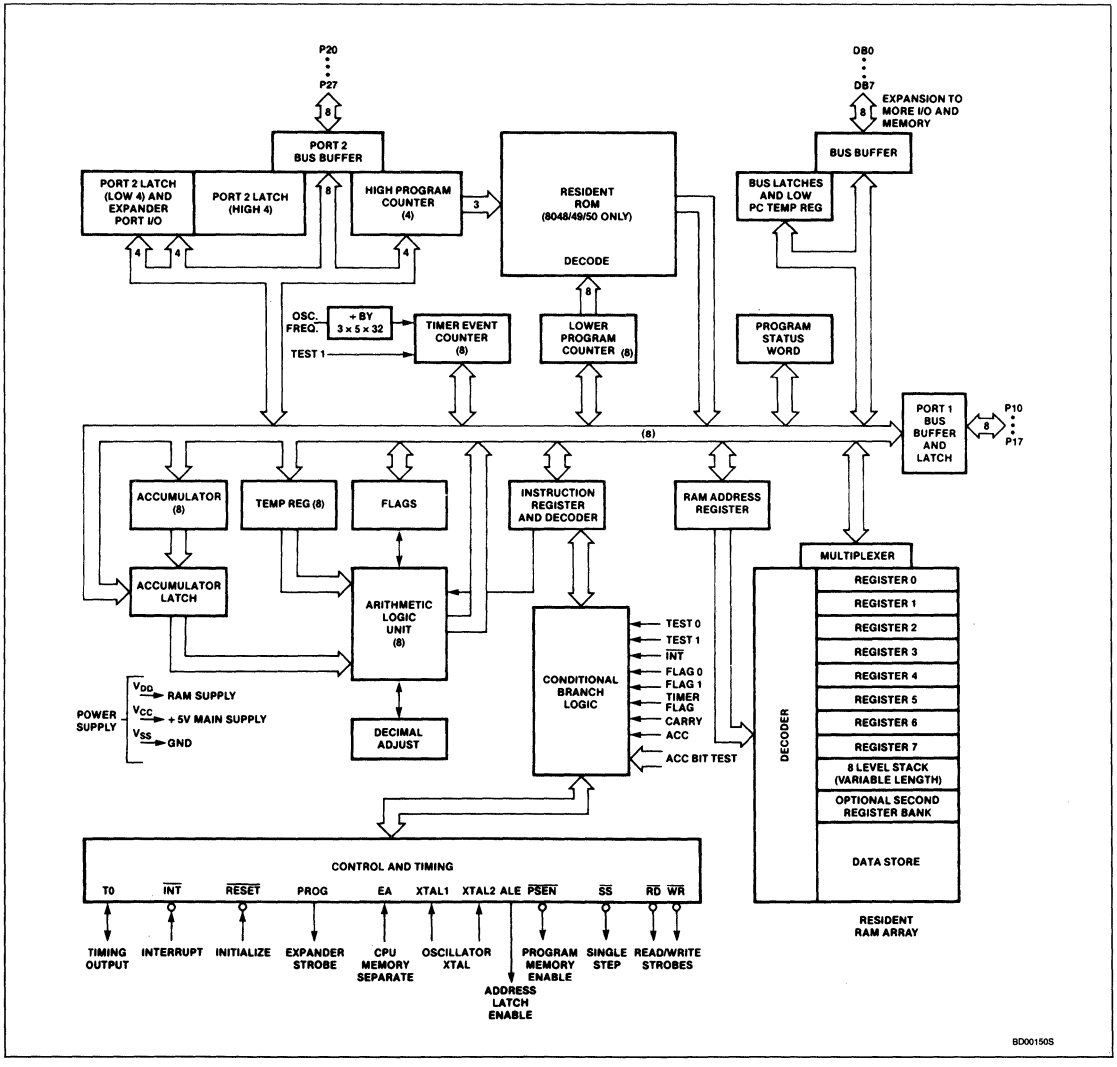
NOTE:
Each pin on these ports can be assigned, under program control, to be an input or an output. A pin is designated as an input by writing a logic "1" to the pin. $\overline{RESET}$ sets all pins to the input mode. Each pin has an internal pullup of approximately 50kΩ.

SCN8049, SCN8050, SCN8039, SCN8040 Single-Chip 8-Bit Microcontroller

FUNCTIONAL DESCRIPTION

The following is a general functional description of the SCN8049 Series microcomputers. Refer to the block diagram below.

BLOCK DIAGRAM



PROGRAM MEMORY

Resident program memory consists of up to 4K bytes of ROM. The program memory is divided into pages of 256 bytes each. As shown in the memory map, Figure 1, program memory is also divided into two 2048-byte banks, MB0 and MB1. A total of 4096 bytes can be addressed directly. If more memory is required, an I/O port can be used to address locations over 4095.

There are three locations in program memory of special importance. These locations contain the first instruction to be executed upon the occurrence of one of three events.

LOCATION	EVENT
0	Activation then deactivation of the $\overline{\text{RESET}}$ line.
3	Activation of the $\overline{\text{INT}}$ line when the external interrupt is enabled.
7	An overflow of the timer/counter if the T/C interrupt is enabled.

DATA MEMORY

Resident data memory, as shown in Figure 2, consists of up to 256 bytes of RAM. All locations are indirectly addressable by either of two RAM pointer registers at locations 0 and 1. The first eight locations of RAM (0–7) are designated as working registers and are directly addressable by several instructions.

By selecting register bank 1, RAM locations 24–31 become the working registers, replacing those in register bank 0 (0–7).

RAM locations 8–23 are designated as the stack. Two locations (bytes) are used per CALL, allowing nesting of up to eight subroutines.

If additional RAM is required, up to 256 bytes may be added and addressed directly using the MOVX instructions. If more RAM is required an I/O port can be used to select one (256-byte) bank of external memory at a time.

PROGRAM COUNTER AND STACK

The Program Counter (PC) is a 12-bit counter/register that points to the location from which the next instruction is to be fetched. The 8048 and 8049 will automatically address external memory when the boundary of their internal memory is exceeded. All processors access external memory if EA is high.

An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8 register pairs of the program counter stack. The pair to be used is determined by a 3-bit stack pointer which is part of the Program Status Word (PSW). Data RAM locations 8 through 23 are available as stack registers and are used to store the program counter and 4 bits of PSW. The stack pointer, when initialized to 000, points to RAM locations 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 and 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11 in anticipation of another CALL. Nesting of subroutines within subroutines can continue up to eight times without overflowing the stack. If overflow does occur the deepest address stored (location 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The end of a subroutine, which is signalled by a return instruction (RET or RETR), causes the stack pointer to be decremented and the contents of the resulting register pair to be transferred to the program counter.

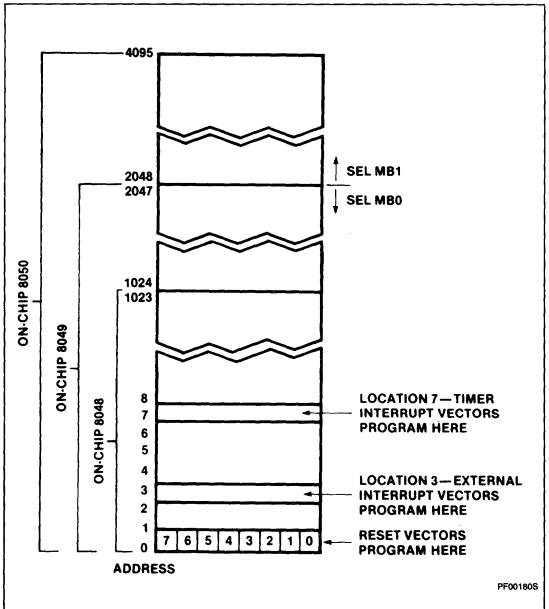


Figure 1. Program Memory Map

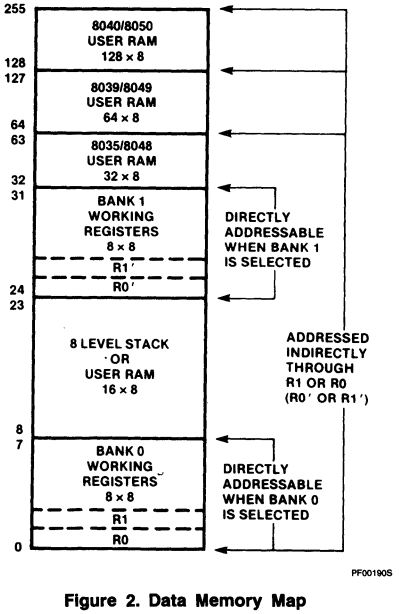
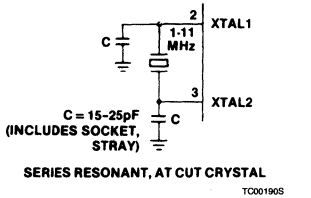


Figure 2. Data Memory Map

OSCILLATOR AND CLOCK

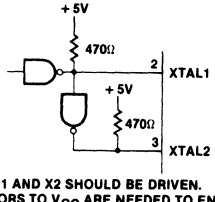
The processor contains its own internal oscillator and clock driver. A crystal, inductor, or external pulse generator may be used to determine the oscillator frequency (see Figure 3). The output of the oscillator is divided by three and can be output on the T0 pin by executing the ENT0 CLK instruction. This CLK signal is divided by 5 to define a machine (instruction) cycle. It is available on Pin 11 as ALE.



SERIES RESONANT, AT CUT CRYSTAL

TC00190S

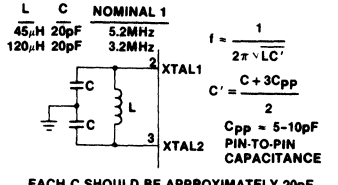
DRIVING FROM EXTERNAL SOURCE



BOTH X1 AND X2 SHOULD BE DRIVEN. RESISTORS TO V_{CC} ARE NEEDED TO ENSURE V_{IH} = 3.8V IF TTL CIRCUITRY IS USED. THE MINIMUM HIGH AND THE MINIMUM LOW TIMES ARE 45%.

TC00200S

LC OSCILLATOR MODE



EACH C SHOULD BE APPROXIMATELY 20pF, INCLUDING STRAY CAPACITANCE.

TC00210S

Figure 3. Crystal Oscillator Mode

TIMER/EVENT COUNTER

An internal counter is available which can count either external events or machine cycles (÷ 32). The machine cycles are divided by 32 before they are input to the 8-bit counter. External events are input directly to the counter. The maximum frequency that can be counted is one third of the frequency of the cycle counter. The minimum positive duty cycle that can be detected is 0.2 t_{CY}. The counter is under program control and can be made to generate an interrupt to the processor when it overflows.

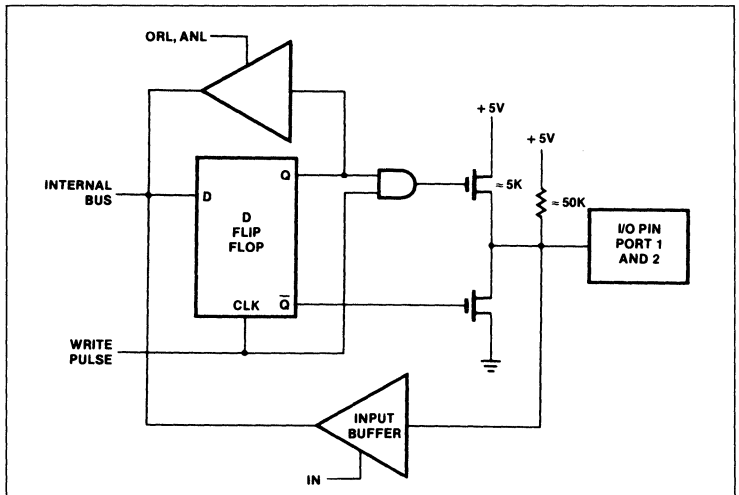


Figure 4. "Quasi Bidirectional" Port Structure

INTERRUPT

An interrupt may be generated by either an external input ($\overline{INT}$, Pin 6) or the overflow of the internal counter, when enabled. In either case, the processor completes execution of the present instruction and then does a CALL to the interrupt service routine. After service, a RETR instruction restores the machine to the state it was prior to the interrupt. The external interrupt has priority over the internal interrupt.

INPUT/OUTPUT

The processor has 27 lines which can be used for input or output functions. These lines are grouped as 3 ports of 8 lines each which serve as either inputs, outputs or bidirectional ports and 3 "test" inputs which can alter program sequences when tested by conditional jump instructions.

Ports 1 and 2

Ports 1 and 2 are each 8 bits wide and have identical characteristics. Data written to these ports is statically latched and remains unchanged until rewritten. As input ports these lines are non-latching; i.e., inputs must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

The lines of ports 1 and 2 are called quasi-bidirectional because of a special output circuit structure which allows each line to serve as an input, an output, or both even though outputs are statically latched. Figure 4 shows the circuit configuration. Each line is continuously pulled up to +5V through a resistive device of relatively high impedance (~50K). This pullup is sufficient to provide the source current for a TTL high level yet can be pulled low by a standard TTL gate thus allowing the same pin to be used for both input and output. To provide fast switching times in a "0" to "1" transition a relatively low impedance device (~50KΩ) is switched in momentarily (~500ns) whenever a "1" is written to the line. When a "0" is written to the line, a low impedance (~3000Ω) device overcomes the light pullup and provides TTL current sinking capability.

Since the pulldown transistor is a low impedance device a "1" must first be written to any line which is to be used as an input. Reset initializes all lines to the high impedance "1" state. This structure allows input and output on the same pin and also allows a mix of input lines and output lines on the same port. The quasi-bidirectional port in combination with the ANL and ORL logical instructions provide an efficient means for handling single line inputs and outputs within an 8-bit processor.

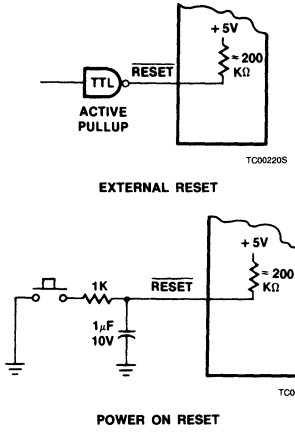


Figure 5. Reset Circuits

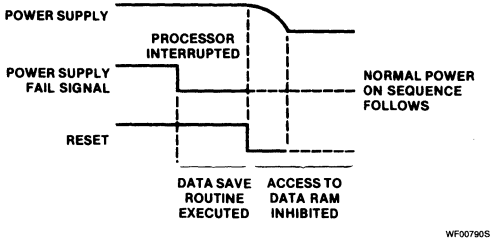


Figure 6. Power Down Sequence

BUS

BUS is also an 8-bit port which is a true bidirectional port with associated input and output strobes. If the bidirectional feature is not needed, BUS can serve as either a statically latched output port or non-latching input port. Input and output lines on this port cannot be mixed.

As a static port, data is written and latched using the OUTL instruction and input using the INS instruction. The INS and OUTL instructions generate pulses on the corresponding $\overline{RD}$ and $\overline{WR}$ output strobe lines; however, in the static port mode they are generally not used. As a bidirectional port, the MOVX instructions are used to read and write to the port. A write to the port generates a pulse on the $\overline{WR}$ output line and output data is valid at the trailing edge of $\overline{WR}$. A read of the port generates a pulse on the $\overline{RD}$ output line and input data must be valid at the trailing edge of $\overline{RD}$. When not being written or read, the BUS lines are in a high impedance state.

Test and INT inputs

Three pins serve as inputs and are testable with the conditional jump instruction. These are T0, T1, and $\overline{INT}$. These pins allow inputs to cause program branches without the necessity to load an input port into the accumulator. The T0, T1, and $\overline{INT}$ pins have other possible functions as well.

RESET INPUT

The reset input provides a means for initialization for the processor. This Schmitt-trigger input has an internal pullup resistor which in combination with an external 1μF capacitor provides an internal reset pulse of sufficient length to guarantee all circuitry is reset. If the reset pulse is generated externally, the reset pin must be held at ground (0.5V) for at least 10 milliseconds after the power supply is within tolerance. Only five machine cycles (12.5μs @ 6MHz) are required if power is already on and the oscillator has stabilized. Typical circuitry is shown in Figure 5.

SINGLE STEP

By proper control of the $\overline{SS}$ line, the mircrocomputer can be made to execute one instruction and then pause or wait until the single step switch is activated again.

POWER DOWN MODE

The SCN8049 Series devices permit power to be removed from all but the data RAM array for low power standby operation. In the power down mode the contents of data RAM can be maintained while drawing typically 5% of normal operating power.

V_{CC} serves as the 5V supply pin for the bulk of the circuitry while the V_{DD} pin supplies only the RAM array. In normal operation both pins are at + 5V. In standby, V_{CC} is at ground and only V_{DD} is maintained at its specified voltage. Applying $\overline{RESET}$ to the processor through the $\overline{RESET}$ pin inhibits any access to the RAM by the processor and guarantees that RAM cannot be inadvertently altered as power is removed from V_{CC}.

A typical power down sequence occurs as shown in Figure 6.

INSTRUCTION SET

The SCN8049 Series instruction set consists of over 90 one and two byte instructions (see Table 1). Program code efficiency is high because: (1) working registers and program variables are stored in RAM, which require only one byte to address and (2) program memory is divided into pages of 256 bytes each, which means that branch destination addresses require one byte.

The instruction set efficiently manipulates and tests bits in addition to performing logical and arithmetic operations upon and the testing of bytes. A set of move instructions operates indirectly upon either RAM or ROM, which permits efficient access of pointers and data tables. The indirect jump instruction performs a multi (up to 256) way branch upon the content of the accumulator to addresses stored in a lookup table. The "decrement register and jump if not zero" instruction saves a byte every time it is used versus using separate increment and test instructions.

The on-chip counter enables either external events or time to be counted off-line from the main program. The processor can either test the counter (under program control) or cause its overflow to generate an interrupt. These features are highly desirable for real time applications. See Table 2 for instruction timing.

ABSOLUTE MAXIMUM RATINGS [1]

SYMBOL	PARAMETER	RATING	UNIT
T_A	Operating ambient temperature[2] range SCN80xxHC SCN80xxHA	 0 to +70 −40 to +85	 °C °C
T_{STG}	Storage temperature range	−65 to +150	°C
V_{IN}	Input voltages with respect to V_{SS}[3]	−0.5 to +7	V
P_D	Power dissipation	1.5	W

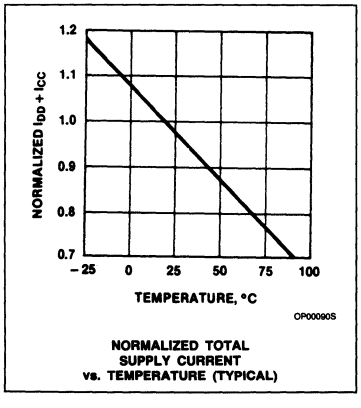
DC ELECTRICAL CHARACTERISTICS $T_A = 0°C$ to $70°C$, $V_{CC} = V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$[4, 5, 6]

SYMBOL	PARAMETER	TEST CONDITIONS[7]	LIMITS			UNIT
			Min	Typ	Max	
V_{IL}	Input low-voltage All except XTAL1, XTAL2		−0.5		0.8	V
V_{IL1}	XTAL1, XTAL2		−0.5		0.6	V
V_{IH}	Input high voltage All except $\overline{RESET}$, XTAL1, XTAL2		2.0		V_{CC}	V
V_{IH1}	$\overline{RESET}$, XTAL1, XTAL2		3.8		V_{CC}	V
V_{OL}	Output low-voltage	$I_{OL} = 2.0mA$			0.4	V
V_{OH}	Output high-voltage All except BUS BUS	 $I_{OH} = -125\mu A$ $I_{OH} = -400\mu A$	 2.4 2.4			 V V
I_{LI1}	Port1, Port2, EA, $\overline{SS}$	$V_{SS} + 0.45 \leq V_{IN} \leq V_{CC}$			−500	μA
I_{L1}	T1, $\overline{Int}$	$V_{SS} + 0.45 \leq V_{IN} \leq V_{CC}$			±10	μA
I_{LI2}	$\overline{RESET}$	$V_{SS} + 0.45 \leq V_{IN} \leq V_{CC}$	−10		−300	μA
I_{OL}	Output leakage current BUS, T0 (high inpedance state)	 $V_{SS} + 0.45 \leq V_{IN} \leq V_{CC}$			±10	μA
I_{DD}	Standby supply current 8035/8048 8039/8049 8040/8050	$\overline{RESET} \leq V_{IL}$ All inputs = 0V $V_{CC} = 0V$			 2.5 4.5 8.5	 mA mA mA
$I_{DD} + I_{CC}$	Total supply current 8035/8048 8039/8049 8040/8050	$\overline{RESET} \leq V_{IL}$		 45 50 60	 80 95 110	 mA mA mA
V_{DD}	Standby power supply		2.5			V
$T_A = -40$ to $85°C$, Automotive temperature range[8]						
V_{IH}	Input high voltage All except XTAL1 and XTAL2		2.2			V
V_{IH1}	$\overline{RESET}$, XTAL1, XTAL2		4.0			V
I_{IL1}	Input leakage current Port1, Port2, EA, $\overline{SS}$	$V_{SS} + .45 \leq V_{IN} \leq V_{CC}$			−750	μA
I_{LI2}	$\overline{RESET}$	$V_{SS} + .45 \leq V_{IN} \leq V_{CC}$	−5		−300	μA
I_{DD}	Standby supply current 8035/8048 8039/8049 8040/8050	$\overline{RESET} \leq V_{IL}$ All inputs = 0V $V_{CC} = 0V$			 3.75 6.75 12.75	 mA mA mA
$I_{CC} + I_{DD}$	Total supply current 8035/8048 8039/8049 8040/8050	$\overline{RESET} \leq V_{IL}$			 90 105 120	 mA mA mA

AC ELECTRICAL CHARACTERISTICS $T_A = 0°C$ to $70°C$, $V_{CC} = V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$[4, 5, 6]

SYMBOL	PARAMETER	TEST CONDITIONS[7]	11 MHz VERSIONS		6 MHz VERSIONS		UNIT
			Min	Max	Min	Max	
(Refer to Figures 7, 8 and 9)							
t_{LL}	ALE pulse width		150		400		ns
t_{AL}	Address setup to ALE		70		150		ns
t_{LA}	Address hold from ALE		50		80		ns
t_{CC}	Control pulse width ($\overline{PSEN}$, $\overline{RD}$, $\overline{WR}$)		300		700		ns
t_{DW}	Data setup before $\overline{WR}$		250		500		ns
t_{WD}	Data hold after $\overline{WR}$		40		120		ns
t_{CY}	Cycle time		1.36	3.75	2.5	15.0	μs
t_{DR}	Data hold		0	100	0	200	ns
t_{RD}	$\overline{PSEN}$, $\overline{RD}$ to data in			200		500	ns
t_{AW}	Address setup to $\overline{WR}$		200		230		ns
t_{AD}	Address setup to data in			400		950	ns
t_{AFC}	Address float to $\overline{RD}$, $\overline{PSEN}$		−10		0		ns
t_{CA}	Control pulse to ALE		10		10		ns
(Refer to Figure 10)							
t_{CP}	Port control setup before falling edge of PROG		100		110		ns
t_{PC}	Port control hold after falling edge of PROG		60		130		ns
t_{PR}	PROG to time P2 input must be valid			650		810	ns
t_{DP}	Output data setup time		200		250		ns
t_{PD}	Output data hold time		20		65		ns
t_{PF}	Input data hold time		0	150	0	150	ns
t_{PP}	PROG pulse width		700		1200		ns
t_{PL}	Port 2 I/O data setup		250		350		ns
t_{LP}	Port 2 I/O data hold		20		150		ns

NOTES:
1. Stresses above those listed under absolute maximum ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operation section of this specification is not implied.
2. For operating at elevated temperatures, the device must be derated based on + 150°C maximum junction temperature.
3. This product includes circuitry specifically designed for the protection of its internal devices from damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to aviod applying any voltages larger than the rated maximum.
4. Parameters are valid over operating temperature range unless otherwise specified.
5. All voltage measurements are referenced to ground (V_{SS}). For testing, all input signals swing between 0.4V and 2.4V with a transition time of 20ns maximum. All time measurements are referenced at input voltages of 0.8V and 2.0V and output voltages 0.8V and 2.0V as appropriate.
6. Typical values are at +25°C, typical ssupply voltages and typical processing parameters.
7. Control outputs: $C_L = 80pF$
   Bus outputs: $C_L = 150pF$
   $t_{CY} = 1.36\mu s$ for 11 MHz versions
   $t_{CY} = 2.5\mu s$ for 6 MHz versions
8. Where no specification is shown, the commercial temperature range specification applies.



NORMALIZED TOTAL SUPPLY CURRENT vs. TEMPERATURE (TYPICAL)

TIMING DIAGRAMS

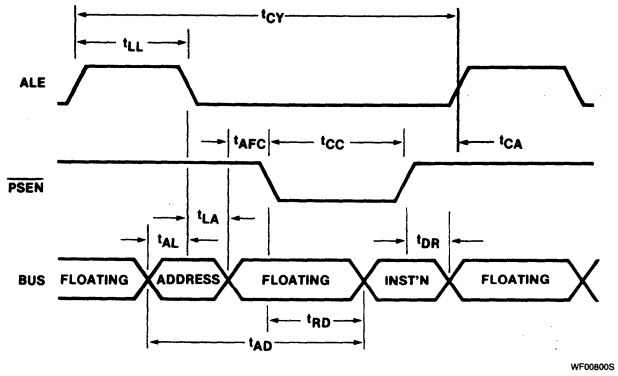


Figure 7. Instruction Fetch From External Program Memory

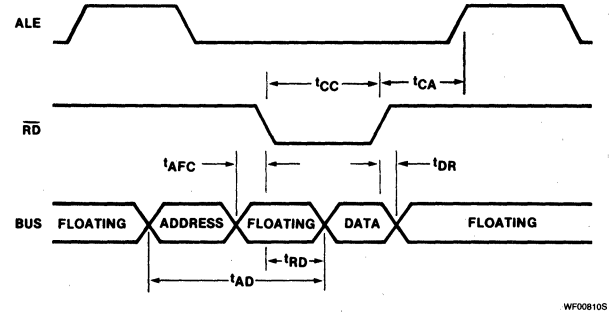


Figure 8. Read From External Data Memory

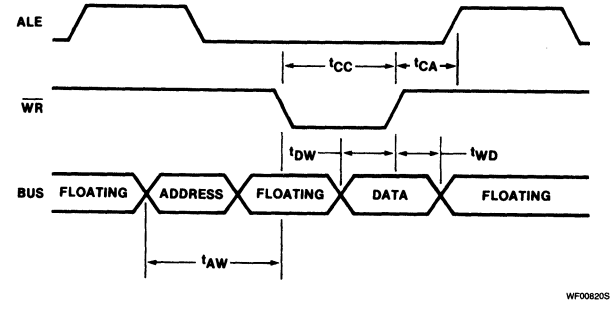


Figure 9. Write to External Data Memory

TIMING DIAGRAMS (Continued)

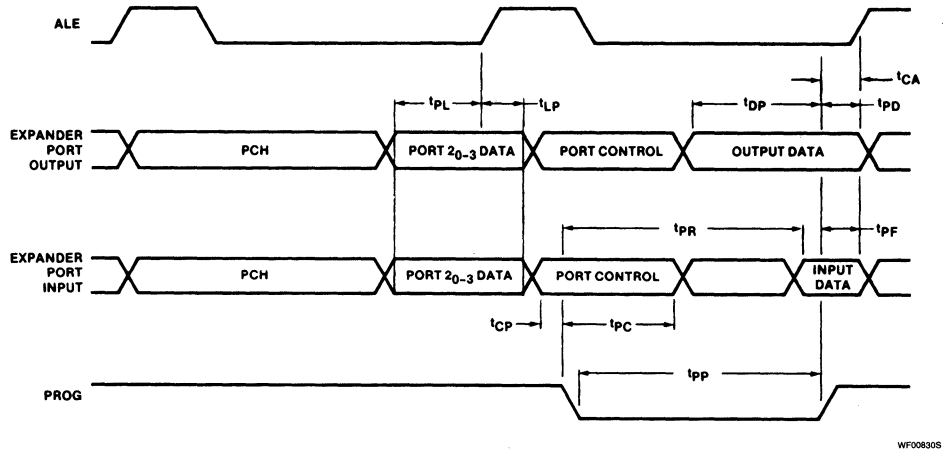


Figure 10. Port 2 Timing

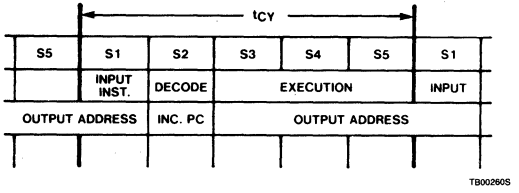


Figure 11. Instruction Cycle

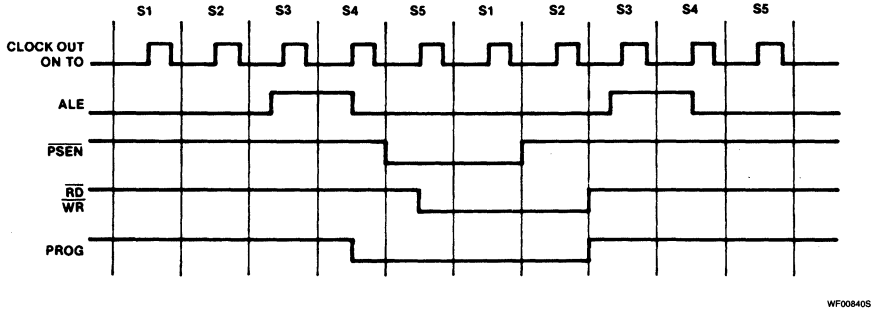


Figure 12. Instruction Cycle Timing

Table 1. Instruction Set

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE	CYCLES	BYTES	FLAGS				
			D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0			C	AC	F0	F1	F2
Accumulator										
ADD A, # data	(A) ← (A) + data	Add immediate the specified data to the accumulator.	0 0 0 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2	•	•			
ADD A, Rr	(A) ← (A) + (Rr) for r = 0 – 7	Add contents of designated register to the accumulator.	0 1 1 0 1 r r r	1	1	•	•			
ADD A, @ Rr	(A) ← (A) + ((Rr)) for r = 0 – 1	Add indirect the contents the data memory location to the accumulator.	0 1 1 0 0 0 0 r	1	1	•	•			
ADDC A, # data	(A) ← (A) + (C) + data	Add immediate with carry the specified data to the accumulator.	0 0 0 1 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2	•	•			
ADDC A, Rr	(A) ← (A) + (C) + (Rr) for r = 0 – 7	Add with carry the contents of the designated register to the accumulator.	0 1 1 1 1 r r r	1	1	•	•			
ADDC A, @ Rr	(A) ← (A) + (C) + ((Rr)) for r = 0 – 1	Add indirect with carry the contents of data memory location to the accumulator.	0 1 1 1 0 0 0 r	1	1	•	•			
ANL A, # data	(A) ← (A) AND data	Logical AND specified immediate data with accumulator.	0 1 0 1 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ANL A, Rr	(A) ← (A) AND (Rr) for r = 0 – 7	Logical AND contents of designated register with accumulator.	0 1 0 1 1 r r r	1	1					
ANL A, @ Rr	(A) ← (A) AND ((Rr)) for r = 0 – 1	Logical AND indirect the contents of data memory with accumulator.	0 1 0 1 0 0 0 r	1	1					
CPL A	(A) ← NOT (A)	Complement the contents of the accumulator.	0 0 1 1 0 1 1 1	1	1					
CLR A	(A) ← 0	Clear the contents of the accumulator.	0 0 1 0 0 1 1 1	1	1					
DA A		Decimal adjust the contents of the accumulator.	0 1 0 1 0 1 1 1	1	1	•	•			
DEC A	(A) ← (A) – 1	Decrement the accumulator's contents by 1.	0 0 0 0 0 1 1 1	1	1					
INC A	(A) ← (A) + 1	Increment the accumulator's contents by 1.	0 0 0 1 0 1 1 1	1	1					
ORL A, # data	(A) ← (A) OR data	Logical OR specified immediate data with accumulator.	0 1 0 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ORL A, Rr	(A) ← (A) OR (Rr) for r = 0 – 7	Logical OR contents of designated register with accumulator.	0 1 0 0 1 r r r	1	1					
ORL A, @ Rr	(A) ← (A) OR ((Rr)) for r = 0 – 1	Logical OR indirect the contents of data memory location with accumulator.	0 1 0 0 0 0 0 r	1	1					
RL A	(An + 1) ← (An) (A_0) ← (A_7) for N = 0 ← 6	Rotate accumulator left by 1-bit without carry.	1 1 1 0 0 1 1 1	1	1					
RLC A	(An + 1) ← (An); n = 0 – 6 (A_0) ← (C) (C) ← (A_7)	Rotate accumulator left by 1-bit through carry.	1 1 1 1 0 1 1 1	1	1	•				
RR A	(An) ← (An + 1); n = 0 – 6 (A_7) ← (A_0)	Rotate accumulator right by 1-bit without carry.	0 1 1 1 0 1 1 1	1	1					
RRC A	(An) ← (An + 1); n = 0 – 6 (A_7) ← (C) (C) ← (A_0)	Rotate accumulator right by 1-bit through carry.	0 1 1 0 0 1 1 1	1	1	•				
SWAP A	(A_{4-7}) ← (A_0 – 3)	Swap the 2 4-bit nibbles in the accumulator.	0 1 0 0 0 1 1 1	1	1					
XRL A, # data	(A) ← (A) XOR data	Logical XOR specified immediate data with accumulator.	1 1 0 1 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
XRL A, Rr	(A) ← (A) XOR (Rr) for r = 0 – 7	Logical XOR contents of designated register with accumulator.	1 1 0 1 1 r r r	1	1					
XRL A, @ Rr	(A) ← (A) XOR ((Rr)) for r = 0 – 1	Logical XOR indirect the contents of data memory location with accumulator.	1 1 0 1 0 0 0 r	1	1					

Table 1. Instruction Set (Continued)

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	CYCLES	BYTES	FLAGS C	AC	F0	F1	F2
Branch										
DJNZ Rr, addr	(Rr) ← (Rr) – 1; r = 0 – 7 if (Rr) ≠ 0: (PC 0 – 7) ← addr	Decrement the specified register and test contents.	1 1 1 0 1 r r r a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JBb addr	(PC 0 – 7) ← addr if Bb = 1 (PC) ← (PC) + 2 if Bb = 0	Jump to specified address if accumulator bit is set.	b_2 b_1 b_0 1 0 0 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JC addr	(PC 0 – 7) ← addr if C = 1 (PC) ← (PC) + 2 if C = 0	Jump to specified address if carry flag is set.	1 1 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JF0 addr	(PC 0 – 7) ← addr if F0 = 1 (PC) ← (PC) + 2 if F0 = 0	Jump to specified address if flag F0 is set.	1 0 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JF1 addr	(PC 0 – 7) ← addr if F1 = 1 (PC) ← (PC) + 2 if F1 = 0	Jump to specified address if flag F1 is set.	0 1 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JMP addr	(PC 8 – 10) ← addr 8 – 10 (PC 0 – 7) ← addr 0 – 7 (PC 11) ← (DBF)	Direct jump to specified address within the 2K address block.	a_{10} a_9 a_8 0 0 1 0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JMPP @ A	(PC 0 – 7) ← ((A))	Jump indirect to specified address within address page.	1 0 1 1 0 0 1 1	2	1					
JNC addr	(PC 0 – 7) ← addr if C = 0 (PC) ← (PC) + 2 if C = 1	Jump to specified address if carry flag is low.	1 1 1 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNI	(PC 0 – 7) ← addr if $\overline{INT}$ = 0 (PC) ← (PC) + 2 if $\overline{INT}$ = 1	Jump to specified address if $\overline{INT}$ input is low.	1 0 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNT0 addr	(PC 0 – 7) ← addr if T0 = 0 (PC) ← (PC) + 2 if T0 = 1	Jump to specified address if test 0 is low.	0 0 1 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNT1 addr	(PC 0 – 7) ← addr if T1 = 0 (PC) ← (PC) + 2 if T1 = 1	Jump to specified address if test 1 is low.	0 1 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNZ addr	(PC 0 – 7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 0	Jump to specified address if accumulator is non-zero.	1 0 0 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JTF addr	(PC 0 – 7) ← addr if TF = 1 (PC) ← (PC) + 2 if TF = 0	Jump to specified address if timer flag is set to 1.	0 0 0 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JT0 addr	(PC 0 – 7) ← addr if T0 = 1 (PC) ← (PC) + 2 if T0 = 0	Jump to specified address if test 0 is a 1.	0 0 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JT1 addr	(PC 0 – 7) ← addr if T1 = 1 (PC) ← (PC) + 2 if T1 = 0	Jump to specified address if test 1 is a 1.	0 1 0 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JZ addr	(PC 0 – 7) ← addr if A = 0 (PC) ← (PC) + 2 if A ≠ 0	Jump to specified address if accumulator is 0.	1 1 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
Control										
EN I		Enable the external ($\overline{INT}$) interrupt.	0 0 0 0 0 1 0 1	1	1					
DIS I		Disable the external ($\overline{INT}$) interrupt.	0 0 0 1 0 1 0 1	1	1					
SEL RB0	(BS) ← 0	Select bank 0 (locations 0 – 7) of data memory.	1 1 0 0 0 1 0 1	1	1					•

Table 1. Instruction Set (Continued)

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	CYCLES	BYTES	FLAGS C	AC	F0	F1	F2
Control (Cont.)										
SEL RB1	(BS) ← 1	Select bank 1 (locations 24 – 31) of data memory.	1 1 0 1 0 1 0 1	1	1					•
SEL MB0	(DBF) ← 0	Select program memory bank 0, addresses 0 – 2047.	1 1 1 0 0 1 0 1	1	1					
SEL MB1	(DBF) ← 1	Select program memory bank 1, addresses 2048 – 4095	1 1 1 1 0 1 0 1	1	1					
ENT0 CLK		Enable clock output on T0 pin.	0 1 1 1 0 1 0 1	1	1					
Data moves										
MOV A, # data	(A) ← data	Move immediate the specified data into the accumulator.	0 0 1 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
MOV A, Rr	(A) ← (Rr); r = 0 – 7	Move the contents of the designated register into the accumulator.	1 1 1 1 1 r r r	1	1					
MOV A, @ Rr	(A) ← ((Rr)); r = 0 – 1	Move indirect the contents of data memory location into the accumulator.	1 1 1 1 0 0 0 r	1	1					
MOV A, PSW	(A) ← (PSW)	Move contents of the program status word into the accumulator.	1 1 0 0 0 1 1 1	1	1					
MOV Rr, # data	(Rr) ← data; r = 0 – 7	Move immediate the specified data into the designated register.	1 0 1 1 1 r r r d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
MOV Rr, A	(Rr) ← (A); r = 0 – 7	Move accumulator contents into the designated register.	1 0 1 0 1 r r r	1	1					
MOV @ Rr, A	((Rr)) ← (A); r = 0 – 1	Move indirect accumulator contents into data memory location.	1 0 1 0 0 0 0 r	1	1					
MOV @ Rr, # data	((Rr)) ← data; r = 0 – 1	Move indirect the specified data into data memory.	1 0 1 1 0 0 0 r d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
MOV PSW, A	(PSW) ← (A)	Move contents of accumulator into the program status word.	1 1 0 1 0 1 1 1	1	1	•	•	•		•
MOVP A, @ A	(A) ← ((A))	Move data in the current page into the accumulator.	1 0 1 0 0 0 1 1	2	1					
MOVP3 A, @ A	(A) ← ((A)) in page 3	Move data in page 3 into the accumulator.	1 1 1 0 0 0 1 1	2	1					
MOVX A, @ Rr	(A) ← ((Rr)); r = 0 – 1	Move indirect the contents of external memory location into the accumulator.	1 0 0 0 0 0 0 r	2	1					
MOVX @ Rr, A	((Rr)) ← (A); r = 0 – 1	Move indirect the contents of the accumulator into external memory.	1 0 0 1 0 0 0 r	2	1					
XCH A, Rr	(A)↔(Rr); r = 0 – 7	Exchange the accumulator and designated register's contents.	0 0 1 0 1 r r r	1	1					
XCH A, @ Rr	(A)↔((Rr)); r = 0 – 1	Exchange indirect contents of accumulator and location in data memory.	0 0 1 0 0 0 0 r	1	1					
XCHD A, @ Rr	(A 0 – 3)↔(Rr)(0 – 3) r = 0 – 1	Exchange indirect 4-bit contents of accumulator and data memory.	0 0 1 1 0 0 0 r	1	1					
Flags										
CPL C	(C) ← NOT (C)	Complement content of carry bit.	1 0 1 0 0 1 1 1	1	1	•				
CPL F0	(F0) ← NOT (F0)	Complement content of flag F0.	1 0 0 1 0 1 0 1	1	1			•		
CPL F1	(F1) ← NOT (F1)	Complement content of flag F1.	1 0 1 1 0 1 0 1	1	1				•	
CLR C	(C) ← 0	Clear content of carry bit to 0.	1 0 0 1 0 1 1 1	1	1	•				
CLR F0	(F0) ← 0	Clear content of flag 0 to 0.	1 0 0 0 0 1 0 1	1	1			•		
CLR F1	(F1) ← 0	Clear content of flag 1 to 0.	1 0 1 0 0 1 0 1	1	1					•
Input/output										
ANL BUS, # data	(BUS) ← (BUS) AND data	Logical AND immediate specified data with BUS.	1 0 0 1 1 0 0 0 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ANL Pp, # data	(Pp) ← (Pp) AND data p = 1 – 2	Logical AND immediate specified data with designated port (1 or 2).	1 0 0 1 1 0 p p d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ANLD Pp, A	(Pp) ← (Pp) AND (A 0 – 3) p = 4 – 7	Logical AND contents of accumulator with designated port (4 – 7).	1 0 0 1 1 1 p p	2	1					

Table 1. Instruction Set (Continued)

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE	CYCLES	BYTES	FLAGS				
			D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0			C	AC	F0	F1	F2
Input/output (Cont.)										
IN A, Pp	(A) ← (Pp); p = 1 – 2	Input data from designated port (1 – 2) into accumulator.	0 0 0 0 1 0 p p	2	1					
INS A, BUS	(A) ← (BUS)	Input strobed BUS data into accumulator.	0 0 0 0 1 0 0 0	1	2					
MOVD A, Pp	(A 0 – 3) ← (Pp); p = 4 – 7 (A 4 – 7) ← 0	Move contents of designated port (4 – 7) into accumulator.	0 0 0 0 1 1 p p	2	1					
MOVD Pp, A	(Pp) ← A 0 – 3; p = 4 – 7	Move contents of accumulator to designated port (4 – 7).	0 0 1 1 1 1 p p	1	1					
ORLD Pp, A	(Pp) ← (Pp) OR (A 0 – 3) p = 4 – 7	Logical OR contents of accumulator with designated port (4 – 7).	1 0 0 0 1 1 p p	1	1					
ORL BUS, # data	(BUS) ← (BUS) OR data	Logical OR immediate specified data with BUS.	1 0 0 0 1 0 0 0 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ORL Pp, # data	(Pp) ← (Pp) OR data p = 1 – 2	Logical OR immediate specified data with designated port (1 – 2).	1 0 0 0 1 0 p p d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
OUTL BUS, A	(BUS) ← (A)	Output contents of accumulator onto BUS.	0 0 0 0 0 0 1 0	1	2					
OUTL Pp, A	(Pp) ← (A); p = 1 – 2	Output contents of accumulator to designated port (1 – 2).	0 0 1 1 1 0 p p	1	1					
Registers										
DEC Rr	(Rr) ← (Rr) – 1; r = 0 – 7	Decrement contents of designated register by 1.	1 1 0 0 1 r r r	1	1					
INC Rr	(Rr) ← (Rr) + 1; r = 0 – 7	Increment contents of designated register by 1.	0 0 0 1 1 r r r	1	1					
INC @ Rr	((Rr)) ← ((Rr)) + 1; r = 0 – 1	Increment indirect the contents of data memory location by 1.	0 0 0 1 0 0 0 r	1	1					
Subroutine										
CALL addr	((SP)) ← (PC), (PSW 4 – 7) (SP) ← (SP) + 1 (PC 8 – 10) ← addr 8 – 10 (PC 0 – 7) ← addr 0 – 7 (PC 11) ← DBF	Call designated subroutine.	a_{10} a_9 a_8 1 0 1 0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
RET	(SP) ← (SP) – 1 (PC) ← ((SP))	Return from subroutine without restoring program status word.	1 0 0 0 0 0 1 1	2	1					
RETR	(SP) ← (SP) – 1 (PC) ← ((SP)) (PSW 4 – 7) ← ((SP))	Return from subroutine restoring program status word.	1 0 0 1 0 0 1 1	2	1					
Timer/counter										
EN TCNTI		Enable timer/counter interrupt.	0 0 1 0 0 1 0 1	1	1					
DIS TCNTI		Disable timer/counter interrupt.	0 0 1 1 0 1 0 1	1	1					
MOV A, T	(A) ← (T)	Move contents of timer/counter into accumulator.	0 1 0 0 0 0 1 0	1	1					
MOV T, A	(T) ← (A)	Move contents of accumulator into timer/counter.	0 1 1 0 0 0 1 0	1	1					
STOP TCNT		Stop count for event counter or timer.	0 1 1 0 0 1 0 1	1	1					
STRT CNT		Start count for event counter.	0 1 0 0 0 1 0 1	1	1					
STRT T		Start count for timer.	0 1 0 1 0 1 0 1	1	1					
Miscellaneous										
NOP		No operation performed	0 0 0 0 0 0 0 0	1	1					

NOTES:
1. Instruction code designations r and p form the binary representation of the registers and ports involved.
2. The dot under the appropriate flag bit indicates that its content is subject to change by the instruction in which it appears.
3. Numerical subscripts appearing in the FUNCTION column reference the specific bits affected.

SYMBOL DEFINITIONS

SYMBOL	DESCRIPTION
A	The accumulator
AC	The auxiliary carry flag
addr	Program memory address (11 bits)
Bb	Bit designator (b = 0 – 7)
BS	The bank switch
C	Carry flag
CLK	Clock signal
CNT	Event counter
D	Nibble designator (4 bits)
DBF	Program memory bank flip-flop
data	Number or expression (8 bits)
F_0, F_1	Flags 0, 1
I	Interrupt
$\overline{INT}$	External interrupt

P	"In-Page" operation designator
P_p	Port designator (p = 1, 2 or 4 – 7)
PSW	Program status word
Rr	Register designator (r = 0, 1 or 0 – 7)
SP	Stack pointer
T	Timer
TF	Timer flag
T0, T1	Testable inputs 0, 1
#	Prefix for immediate data
@	Prefix for indirect address
$	Program counter's current value
←	Replaced by
↔	Exchanged with

Table 2. Instruction Timing**

	CYCLE 1					CYCLE 2				
INSTRUCTION	**S1**	**S2**	**S3**	**S4**	**S5**	**S1**	**S2**	**S3**	**S4**	**S5**
IN A,P	Fetch Instruction	Increment Program Counter	—	Increment Timer	—	—	Read Port	* —	—	—
OUTL P,A	Fetch Instruction	Increment Program Counter	—	Increment Timer	Output To Port	—	—	* —	—	—
ANL P, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
ORL P, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
INS A, BUS	Fetch Instruction	Increment Program Counter	—	Increment Timer	—	—	Read Port	* —	—	—
OUTL BUS, A	Fetch Instruction	Increment Program Counter	—	Increment Timer	Output To Port	—	—	* —	—	—
ANL BUS, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
ORL BUS, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
MOVX @R,A	Fetch Instruction	Increment Program Counter	Output RAM Address	Increment Timer	Output Data to RAM	—	—	* —	—	—
MOVX A,@R	Fetch Instruction	Increment Program Counter	Output RAM Address	Increment Timer	—	—	Read Data	* —	—	—
MOVD A, P_i	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	—	—	Read P2 Lower	* —	—	—
MOVD P_i, A	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	Output Data TO P2 Lower	—	—	* —	—	—
ANLD P, A	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	Output Data	—	—	* —	—	—
ORLD P, A	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	Output Data	—	—	* —	—	—
J (CONDITIONAL)	Fetch Instruction	* Increment Program Counter	Sample Condition	Increment Timer	—	Fetch Immediate Data	—	* Update Program Counter	—	—
START CNT/STRT T	Fetch Instruction	* Increment Program Counter	—	—	Start Counter					
STOP TCNT	Fetch Instruction	* Increment Program Counter	—	—	Stop Counter					
EN I	Fetch Instruction	* Increment Program Counter	—	Enable Interrupt	—					
DIS I	Fetch Instruction	* Increment Program Counter	—	Disable Interrupt	—					
ENT0 CLK	Fetch Instruction	* Increment Program Counter	—	Enable Clock	—					

NOTES:
*Valid instruction address are output at this time if external program memory is being accessed.
**See figures 11 and 12 for instruction cycle and cycle timing.

Signetics

Microprocessor Products

Section 7
Sales Offices, Representatives & Distributors

Signetics

Sales Offices, Representatives & Distributors

SIGNETICS HEADQUARTERS
811 East Arques Avenue
P.O. Box 3409
Sunnyvale, CA 94088–3409
Phone: (408) 991–2000

ALABAMA
Huntsville
Phone: (205) 830–4001

ARIZONA
Phoenix
Phone: (602) 968–5777

CALIFORNIA
Calabasas
Phone: (818) 880–6304

Irvine
Phone: (714) 833–8980
(714) 752–2780

Los Angeles
Phone: (213) 670–1101

San Diego
Phone: (619) 560–0242

Sunnyvale
Phone: (408) 991–3737

COLORADO
Aurora
Phone: (303) 751–5011

FLORIDA
Ft. Lauderdale
Phone: (305) 486–6300

GEORGIA
Atlanta
Phone: (404) 594–1392

ILLINOIS
Itasca
Phone: (312) 250–0050

INDIANA
Kokomo
Phone: (317) 459–5355

KANSAS
Overland Park
Phone: (913) 469–4005

MASSACHUSETTS
Westford
Phone: (508) 692–6211

MICHIGAN
Farmington Hills
Phone: (313) 553–6070

MINNESOTA
Edina
Phone: (612) 835–7455

NEW JERSEY
Parsippany
Phone: (201) 334–4405

NEW YORK
Hauppauge
Phone: (516) 348–7877

Wappingers Falls
Phone: (914) 297–4074

NORTH CAROLINA
Raleigh
Phone: (919) 781–1900

OHIO
Columbus
Phone: (614) 888–7143

Dayton
Phone: (513) 294–7340

OREGON
Beaverton
Phone: (503) 627–0110

PENNSYLVANIA
Plymouth Meeting
Phone: (215) 825–4404

TENNESSEE
Greeneville
Phone: (615) 639–0251

TEXAS
Austin
Phone: (512) 339–9944

Houston
Phone: (713) 668–1989

Richardson
Phone: (214) 644–3500

CANADA
SIGNETICS CANADA, LTD.
Etobicoke, Ontario
Phone: (416) 626–6676

Nepean, Ontario
Phone: (613) 225–5467

REPRESENTATIVES

ARIZONA
Scottsdale
Thom Luke Sales, Inc.
Phone: (602) 941–1901

CALIFORNIA
Orangevale
Webster Associates
Phone: (916) 989–0843

CONNECTICUT
Fairfield
NRG, Limited
Phone: (203) 384–1112

FLORIDA
Clearwater
Sigma Technical Assoc.
Phone: (813) 791–0271

Ft. Lauderdale
Sigma Technical Assoc.
Phone: (305) 731–5995

ILLINOIS
Hoffman Estates
Micro–Tex, Inc.
Phone: (312) 382–3001

INDIANA
Indianapolis
Mohrfield Marketing, Inc.
Phone: (317) 546–6969

IOWA
Cedar Rapids
J.R. Sales
Phone: (319) 393–2232

MARYLAND
Columbia
Third Wave Solutions, Inc.
Phone: (301) 290–5990

MASSACHUSETTS
Needham Heights
Kanan Associates
Phone: (617) 449–7400

MICHIGAN
Bloomfield Hills
Enco Marketing
Phone: (313) 338–8600

MINNESOTA
Eden Prairie
High Technology Sales
Phone: (612) 944–7274

MISSOURI
Bridgeton
Centech, Inc.
Phone: (314) 291–4230

Raytown
Centech, Inc.
Phone: (816) 358–8100

NEW HAMPSHIRE
Hookset
Kanan Associates
Phone: (603) 645–0209

NEW JERSEY
East Hanover
Emtec Sales, Inc.
Phone: (201) 428–0600

NEW MEXICO
Albuquerque
F.P. Sales
Phone: (505) 345–5553

NEW YORK
Ithaca
Bob Dean, Inc.
Phone: (607) 257–1111

OHIO
Centerville
Bear Marketing, Inc.
Phone: (513) 436–2061

Richfield
Bear Marketing, Inc.
Phone: (216) 659–3131

OKLAHOMA
Tulsa
Jerry Robison
and Associates
Phone: (918) 665–3562

OREGON
Beaverton
Western Technical Sales
Phone: (503) 644–8860

PENNSYLVANIA
Pittsburgh
Bear Marketing, Inc.
Phone: (412) 531–2002

Hatboro
Delta Technical
Sales, Inc.
Phone: (215) 975–0600

UTAH
Salt Lake City
Electrodyne
Phone: (801) 264–8050

WASHINGTON
Bellevue
Western Technical Sales
Phone: (206) 641–3900

Spokane
Western Technical Sales
Phone: (509) 922–7600

WISCONSIN
Waukesha
Micro–Tex, Inc.
Phone: (414) 542–5352

CANADA
Burnaby, B.C.
Tech–Trek, Ltd.
Phone: (604) 439–1373

Mississauga, Ontario
Tech–Trek, Ltd.
Phone: (416) 238–0366

Nepean, Ontario
Tech–Trek, Ltd.
Phone: (613) 225–5161

Ville St. Laurent, Quebec
Tech–Trek, Ltd.
Phone: (514) 337–7540

Sales Offices, Representatives & Distributors

DISTRIBUTORS

Contact one of our local distributors:
Anthem Electronics
Avnet Electronics
Aztech Electronics
Hamilton/Avnet Electronics
Marshall Industries
Schweber Electronics
Wyle/LEMG
Zentronics, Ltd.

FOR SIGNETICS PRODUCTS WORLDWIDE:

ARGENTINA
Philips Argentina S.A.
Buenos Aires
Phone: 54–1–541–7141

AUSTRALIA
Philips Electronic Components & Mat'l Ltd.
Artarmon, N.S.W.
Phone: 61–2–439–3322

AUSTRIA
Osterrichische Philips
Wien
Phone: 43–222–60–101 –820

BELGIUM
S.A. MBLE Components
Brussels
Phone: 32–2–525–61–11

BRAZIL
Philips Do Brasil, Ltda.
Sao Paulo
Phone: 55–11–211–2600

CHILE
Philips Chilena S.A.
Santiago
Phone: 56–02–077–3816

CHINA, PEOPLES REPUBLIC OF
Philips Hong Kong, Ltd.
Kwai Chung, Kowloon
Phone: 852–0–424–5121

COLUMBIA
Iprelenso, Ltda.
Bogota
Phone: 57–1–2497624

DENMARK
Philips Components A/S
Copenhagen S
Phone: 45–1–54–11–33

FINLAND
Oy Philips Ab
Espoo
Phone: 358–0–502–61

FRANCE
R.T.C. Compelec
Issy–les–Moulineaux
Cedex
Phone: 33–1–40–93–80–00

GERMANY
Valvo
Hamburg
Phone: 49–40–3–296–0

GREECE
Philips S.A. Hellenique
Athens
Phone: 30–1–4894–339

HONG KONG
Philips Hong Kong, Ltd.
Kwai Chung, Kowloon
Phone: 852–0–245–121

INDIA
Peico Electronics & Elect. Ltd.
Bombay
Phone: 91–22–493–0311

INDONESIA
P.T. Philips–Ralin Electronics
Jakarta Selatan
Phone: 62–21–512–572

IRELAND
Philips Electrical Ltd.
Dublin
Phone: 353–1–69–33–55

ISRAEL
Rapac Electronics, Ltd.
Tel Aviv
Phone: 972–3–477115

ITALY
Philips S.p.A.
Milano
Phone: 38–2–67–52–1

JAPAN
Philips Components Japan
Osaka–Shi
Phone: 81–6–389–7722

Philips Components Japan
Tokyo
Phone: 81–3–280–2620

KOREA
Philips Industries, Ltd.
Seoul
Phone: 82–2–794–5011 /2/3/4/5

MALAYSIA
Philips Malaysia SDN Bernhad
Pulau Penang
Phone: 60–4–870–055

MEXICO
Panamtek
Guadalajara, Jal
Phone: 52–36–30–30–29

Mexico, D.F.
Phone: 52–5–586–84–43

NETHERLANDS
Philips Nederland
Eindhoven
Phone: 31–40–444–755

NEW ZEALAND
Philips New Zealand Ltd.
Auckland
Phone: 64–9–605–914

NORWAY
Norsk A/S Philips
Oslo
Phone: 47–2–68–02–00

PERU
Cadesa
San Isidro
Phone: 51–14–707–080

PHILIPPINES
Philips Industrial Dev., Inc.
Makati Metro Manila
Phone: 63–2–810–01–61

PORTUGAL
Philips Portuguesa SA
Lisbon
Phone: 351–1–68–31–21

SINGAPORE
Philips Project Dev. Pte., Ltd.
Singapore
Phone: 65–350–2000

SOUTH AFRICA
SA Philips (PTY), Ltd.
Randburg
Phone: 27–11–889–3911

SPAIN
Copresa S.A.
Barcelona
Phone: 34–3–301–63–12

SWEDEN
Philips Components A.B.
Stockholm
Phone: 46–8–782–10–00

SWITZERLAND
Philips Components A.G.
Zuerich
Phone: 41–1–488–2211

TAIWAN
Philips Taiwan, Ltd.
Taipei
Phone: 886–2–712–0500

THAILAND
Philips Electrical Co. of Thailand Ltd.
Bangkok
Phone: 66–2–223–6330/9

TURKEY
Turk Philips Ticaret A.S.
Istanbul
Phone: 90–1–179–27–70

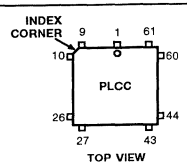
UNITED KINGDOM
Philips Components
London
Phone: 44–1–580–6633

UNITED STATES
Signetics International Corp.
Sunnyvale, California
Phone: (408) 991–2000

URUGUAY
Luzilectron S.A.
Montevideo
Phone: 598–91–56–41/ 42/43/44

VENEZUELA
Magnetica S.A.
Caracas
Phone: 58–2–241–7509

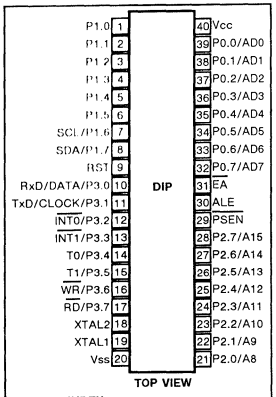
S83C552/S80C552



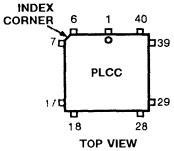
TOP VIEW

Pin	Function	Pin	Function
1	P5.0/ADC0	35	XTAL1
2	VDD	36	Vss
3	STADC	37	Vss
4	PWM0	38	NC
5	PWM1	39	P2.0/A08
6	EW	40	P2.1/A09
7	P4.0/CMSR0	41	P2.2/A10
8	P4.1/CMSR1	42	P2.3/A11
9	P4.2/CMSR2	43	P2.4/A12
10	P4.3/CMSR3	44	P2.5/A13
11	P4.4/CMSR4	45	P2.6/A14
12	P4.5/CMSR5	46	P2.7/A15
13	P4.6/CMT0	47	PSEN
14	P4.7/CMT1	48	ALE
15	RST	49	EA
16	P1.0/CT0I	50	P0.7/AD7
17	P1.1/CT1I	51	P0.6/AD6
18	P1.2/CT2I	52	P0.5/AD5
19	P1.3/CT3I	53	P0.4/AD4
20	P1.4/T2	54	P0.3/AD3
21	P1.5/RT2	55	P0.2/AD2
22	P1.6/SCL	56	P0.1/AD1
23	P1.7/SDA	57	P0.0/AD0
24	P3.0/RXD	58	AVref-
25	P3.1/TXD	59	AVref+
26	P3.2/INT0	60	AVSS
27	P3.3/INT1	61	AVDD
28	P3.4/T0	62	P5.7/ADC7
29	P3.5/T1	63	P5.6/ADC6
30	P3.6/WR	64	P5.5/ADC5
31	P3.7/RD	65	P5.4/ADC4
32	NC	66	P5.3/ADC3
33	NC	67	P5.2/ADC2
34	XTAL2	68	P5.1/ADC1

S83C652/S80C652



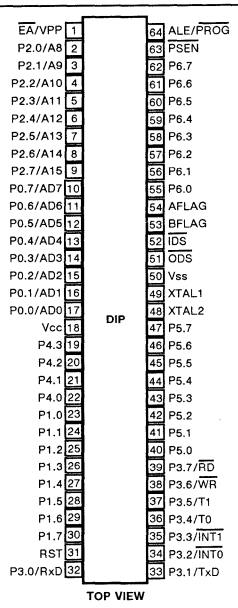
TOP VIEW



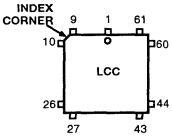
TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	P1.0	24	P2.0/A8
3	P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	SCL/P1.6	30	P2.6/A14
9	SDA/P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/DATA/P3.0	33	ALE
12	NC	34	NC
13	TxD/CLOCK/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

SC87C451



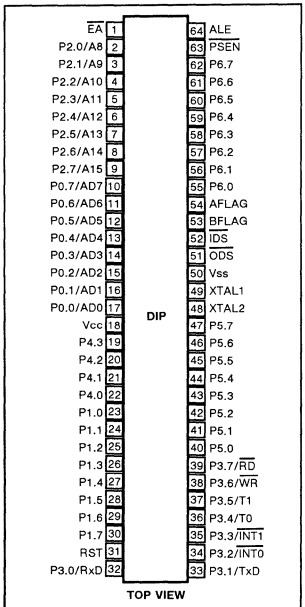
TOP VIEW



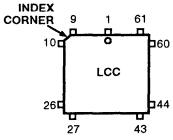
TOP VIEW

Pin	Function	Pin	Function	Pin	Function
1	EA/VPP	24	P4.2	47	P5.3
2	P2.0/A8	25	P4.1	48	P5.4
3	P2.1/A9	26	P4.0	49	P5.5
4	P2.2/A10	27	P1.0	50	P5.6
5	P2.3/A11	28	P1.1	51	P5.7
6	P2.4/A12	29	P1.2	52	XTAL2
7	P2.5/A13	30	P1.3	53	XTAL1
8	P2.6/A14	31	P1.4	54	Vss
9	P2.7/A15	32	P1.5	55	ODS
10	P0.7/AD7	33	P1.6	56	IDS
11	P0.6/AD6	34	P1.7	57	BFLAG
12	P0.5/AD5	35	RST	58	AFLAG
13	P0.4/AD4	36	P3.0/RxD	59	P6.0
14	P0.3/AD3	37	P3.1/TxD	60	P6.1
15	P0.2/AD2	38	P3.2/INT0	61	P6.2
16	P0.1/AD1	39	P3.3/INT1	62	P6.3
17	P0.0/AD0	40	P3.4/T0	63	P6.4
18	Vcc	41	P3.5/T1	64	P6.5
19	P4.7	42	P3.6/WR	65	P6.6
20	P4.6	43	P3.7/RD	66	P6.7
21	P4.5	44	P5.0	67	PSEN
22	P4.4	45	P5.1	68	ALE/PROG
23	P4.3	46	P5.2		

SC80C451/SC83C451



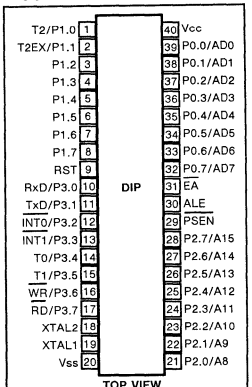
TOP VIEW



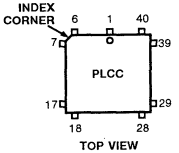
TOP VIEW

Pin	Function	Pin	Function	Pin	Function
1	EA	24	P4.2	47	P5.3
2	P2.0/A8	25	P4.1	48	P5.4
3	P2.1/A9	26	P4.0	49	P5.5
4	P2.2/A10	27	P1.0	50	P5.6
5	P2.3/A11	28	P1.1	51	P5.7
6	P2.4/A12	29	P1.2	52	XTAL2
7	P2.5/A13	30	P1.3	53	XTAL1
8	P2.6/A14	31	P1.4	54	Vss
9	P2.7/A15	32	P1.5	55	ODS
10	P0.7/AD7	33	P1.6	56	IDS
11	P0.6/AD6	34	P1.7	57	BFLAG
12	P0.5/AD5	35	RST	58	AFLAG
13	P0.4/AD4	36	P3.0/RxD	59	P6.0
14	P0.3/AD3	37	P3.1/TxD	60	P6.1
15	P0.2/AD2	38	P3.2/INT0	61	P6.2
16	P0.1/AD1	39	P3.3/INT1	62	P6.3
17	P0.0/AD0	40	P3.4/T0	63	P6.4
18	Vcc	41	P3.5/T1	64	P6.5
19	P4.7	42	P3.6/WR	65	P6.6
20	P4.6	43	P3.7/RD	66	P6.7
21	P4.5	44	P5.0	67	PSEN
22	P4.4	45	P5.1	68	ALE
23	P4.3	46	P5.2		

SCN8032AH/SCN8052AH



TOP VIEW



TOP VIEW

Pin	Function	Pin	Function
1	NC	23	NC
2	T2/P1.0	24	P2.0/A8
3	T2EX/P1.1	25	P2.1/A9
4	P1.2	26	P2.2/A10
5	P1.3	27	P2.3/A11
6	P1.4	28	P2.4/A12
7	P1.5	29	P2.5/A13
8	P1.6	30	P2.6/A14
9	P1.7	31	P2.7/A15
10	RST	32	PSEN
11	RxD/P3.0	33	ALE
12	NC	34	NC
13	TxD/P3.1	35	EA
14	INT0/P3.2	36	P0.7/AD7
15	INT1/P3.3	37	P0.6/AD6
16	T0/P3.4	38	P0.5/AD5
17	T1/P3.5	39	P0.4/AD4
18	WR/P3.6	40	P0.3/AD3
19	RD/P3.7	41	P0.2/AD2
20	XTAL2	42	P0.1/AD1
21	XTAL1	43	P0.0/AD0
22	Vss	44	Vcc

Signetics

a division of North American Philips Corporation

Signetics Company
811 E. Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088-3409
Telephone 408/991-2000

98-8000-000

Signetics

Philips Components

PHILIPS