

DESIGNING AND FABRICATION OF AT89S52 & AT89S8252 BASED KIT

A Project Report submitted by in the partial fulfillment of the requirement
for award of Degree of

**Master of Science
in
Applied Physics**

by

**SANJEEV KUMAR
(ROLL NO. – 6481)**

Under the Guidance of

MR. D.V. Chaure



**DEPARTMENT OF APPLIED PHYSICS
DELHI COLLEGE OF ENGINEERING
FACULTY OF TECHNOLOGY
UNIVERSITY OF DELHI**

DELHI – 110007

JUNE - 2005

CERTIFICATE

This is to certify that the thesis entitled **“Deigning and Fabrication of AT89S52 & AT89S8252 based Kit”** submitted by **Mr. SANJEEV KUMAR** in partial fulfillment of requirements for the award of the degree of Master of Science in Applied Physics at Delhi College of Engineering, Faculty of Technology, University of Delhi, Delhi, India.

This project has been carried out under my supervision and guidance. I approve it for submission to the college for the award of Master of Science in Applied Physics.

MR. D.V.Chaure
Department Of Applied Physics
Delhi College of Engineering
Bawana Road
Delhi - 110042

ACKNOWLEDGEMENT

No student has enough good ideas for a project such as this or the perseverance to see it through without help from other is the story of this project and the people who helped me turn it into reality.

First, I would heartily thank to **Prof. K.L.Deori, Professor and Head**, Department of Applied Physics, Delhi College of Engineering, Delhi for her inspiration to this project. It is my privilege to record my deep sense of gratitude and unbound reverence to **Mr. D.V. Chaure**, Department of Applied Physics, DCE, Delhi for his constant encouragement and inspiration throughout the project. I am greatly indebted to **Dr. R. K. Sinha** and **Dr. A.K. Jha**, Faculties, Department of Applied Physics, DCE, Delhi for their valuable advices.

I would like to thank **Mrs. Rajeshwari Pandey**, Lecturer, Electronics Dept., DCE, and **Mr. Anil Rathi**, Sylvania Limited, Gurgoan, without his help in preparation of board, and writing program, project would not have take the shape. I would also like to thank all the departmental laboratory staff of Applied physics and Electronics Dept. particularly **Mr. I. D. Kapoor** and **Mr. Rajender**.

Finally I want to express my heart-felt gratitude to my Parents and brother for their cheerful encouragement, immeasurable patience and inspiration.

Sanjeev Kumar

CONTENTS

Acknowledgement
Contents
List of Tables
List of Figures

Chapter 1: 8-bit Microcontroller with 8K Bytes In-System Programmable Flash AT89S52.

1.1 Description
1.2 Pin Configurations
1.3 Block Diagram
1.4 Pin Description
1.5 Special Function Registers
1.6 Memory Organization
1.6.1 Program Memory
1.6.2 Data Memory
1.7 Watchdog Timer
1.7.1 Using the WDT
1.7.2 WDT During Power-down and Idle
1.8 UART
1.9 Timer 0 and 1
1.10 Timer 2
1.10.1 Capture Mode
1.10.2 Auto-reload
1.11. Baud Rate Generator
1.12. Programmable Clock Out
1.13. Interrupts
1.14. Oscillator Characteristics
1.15. Idle Mode
1.16. Power-down Mode
1.17. Program Memory Lock Bits
1.18. Programming the Flash – Parallel Mode
1.19. Programming the Flash – Serial Mode
1.20. Serial Programming Algorithm

Chapter 2: Capture and Layout.

- 2.1. Introduction
- 2.2. Designs and Schematics
- 2.3. Navigating Designs
- 2.4. Editing a schematic page
- 2.5. Making Connections
- 2.6. Editing properties
- 2.7. Managing Parts of Libraries
- 2.8. Making parts
- 2.9. Processing our design
- 2.10. Introduction of Layout
- 2.11. Creating a new design
- 2.12. Autorotating
- 2.13. Finishing the design
- 2.14. Using AutoECO
- 2.15. Autoplacement

Chapter 3: Description of Designing and Fabrication of AT89S52 & AT89S8252 Based Kit.

- 3.1. Circuit Description
- 3.2. MAX 232
- 3.3. Connections for MAX232:
- 3.4. Testing Procedure:
- 3.5. Description of different Software
 - 3.5.1. Atmel Microcontroller ISP Software
 - 3.5.2. Description of ASEM – 51
- 3.6. Application**
 - 3.6.1. Seven Segment LED Display
 - 3.6.2. Types of LEDs
 - 3.6.3. Driving it with a micro-controller

List of Figure

- Figure 1.1: Pin Configurations of AT89S52 Chip
- Figure 1.2: Block Diagram of AT89S52
- Figure 1.3: Timer in Capture Mode
- Figure 1.4: Timer 2 in Baud Rate Generator Mode
- Figure 1.5: Timer 2 in Clock-Out Mode
- Figure 1.6: Interrupt Sources
- Figure 1.7: Oscillator Connections
- Figure 1.8: External Clock Drive Configuration
- Figure 2.1: Capture's session frame
- Figure 2.2: New project manager window
- Figure 2.3: File tab
- Figure 2.4: Hierarchy tab
- Figure 2.5: The Edit Obstacle dialog box
- Figure 2.6: A board prior to component placement
- Figure 2.7: density graph of Layout
- Figure 2.8: Link Footprint to Component
- Figure 2.9: Component placed in autoplacement
- Figure 2.10: The Edit Net dialog box
- Figure 3.1: Circuit Diagram of AT89S52 & AT89S8252 Based Micro-Controller
- Figure 3.2: Components layout for the PCB
- Figure 3.3: MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit
- Figure 3.4: MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit
- Figure 3.5: Interfacing of PC to MAX-232 pin out on a DB-9 pin used for Asynchronous Data
- Figure 3.6: Configuration of DB-9 pin

List of Table

- Table 1.1: Port Pin Alternate Functions
- Table 1.2: Port Pin Alternate Functions
- Table 1.3: AT89S52 SFR Map and Reset Values
- Table 1.4: T2CON – Timer/Counter 2 Control Register
- Table 1.5: T2MOD – Timer 2 Mode Control Register
- Table 1.6: AUXR: Auxiliary Register
- Table 1.7: AUXR1: Auxiliary Register 1
- Table 1.8: Timer 2 Operating Modes
- Table 1.9: Interrupt Enable (IE) Register
- Table 1.10: Status of External Pins during Idle and Power-down Modes
- Table 1.11: Lock Bit Protection Modes
- Table 1.12: Flash Programming Modes

Chapter – 1

8-bit Microcontroller with 8K Bytes In-System Programmable Flash AT89S52

1.1 Description

The AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The device is Atmel's high-density nonvolatile memory technology and is compatible with the standard 80C51 instruction set and Pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, The AT89S52 provides the following standard features:

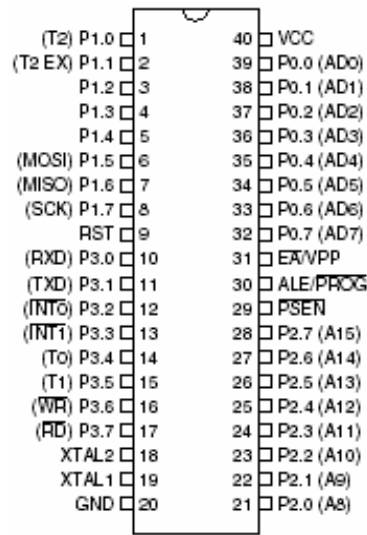
- 8K bytes of Flash,
- 256 bytes of RAM,
- 32 I/O lines,
- Watchdog timer,
- 5 two data pointers,
- three 16-bit timer/counters,
- a six-vector two-level interrupt architecture,
- a full duplex serial port,
- on-chip oscillator,
- and clock circuitry.

In addition, the AT89S52 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes.

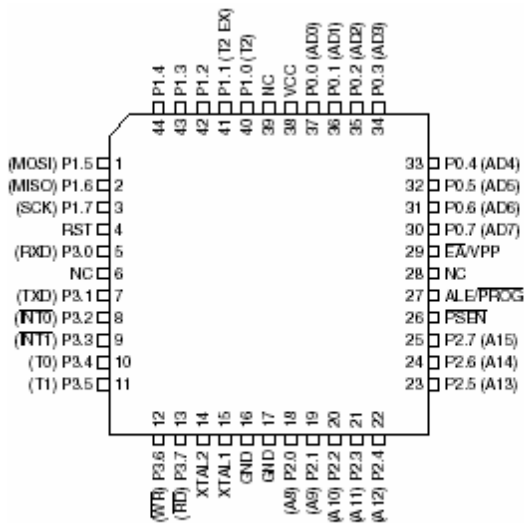
- The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning.
- The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

1.2. Pin Configurations

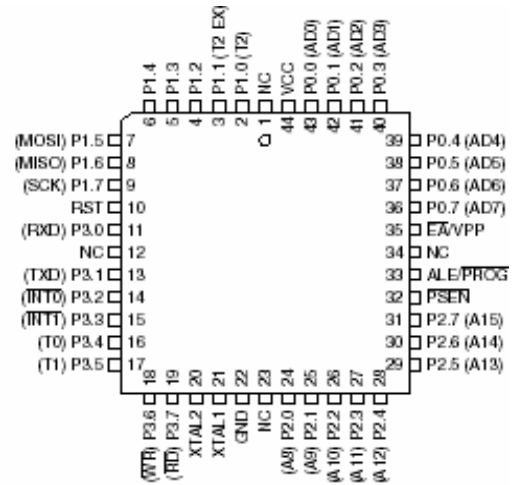
1. 40-lead PDIP



2. 42-lead PDIP



3. 44-lead PLCC



4. 44-lead TOFP

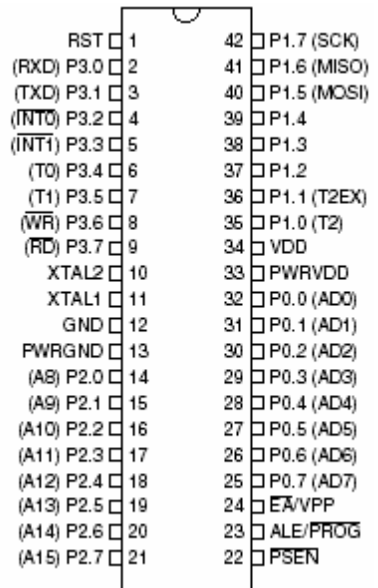


Figure 1.1 : Pin Configurations of AT89S52 Chip

1.3. Block Diagram

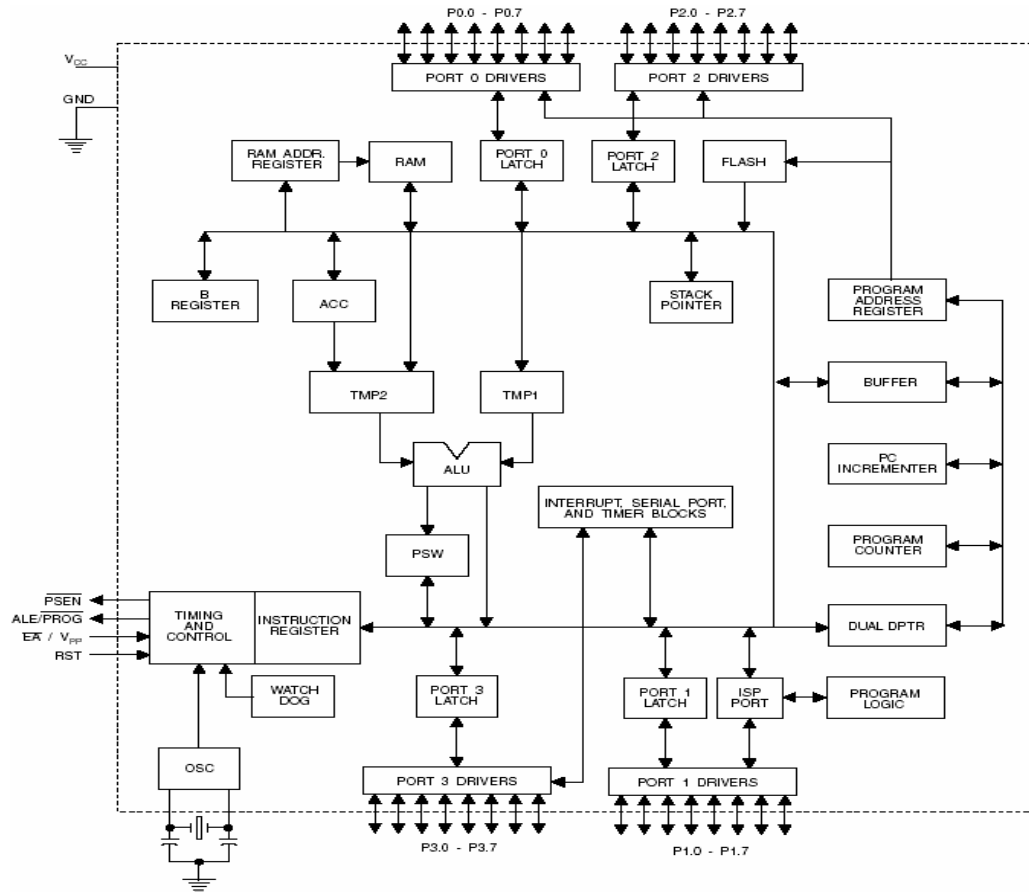


Figure 1.2 : Block Diagram of AT89S52

1.4. Pin Description

1. VCC:

Supply voltage.

2. GND:

Ground.

3. Port 0:

Port 0 is an 8-bit bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs. Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups. Port0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. **External pull-ups are required during program verification.**

4. Port 1:

Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. When 1s are written to Port 1 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups. Port 1 also receives the low-order address bytes during Flash programming and verification.

P1.0 T2	(external count input to Timer/Counter 2), clock-out
P1.1 T2EX	(Timer/Counter 2 capture/reload trigger and direction control)
P1.5 MOSI	(used for In-System Programming)
P1.6 MISO	(used for In-System Programming)
P1.7 SCK	(used for In-System Programming)

Table 1.1 : Port Pin Alternate Functions

5. Port 2:

Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the 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). Port 2 emits the contents of the P2 Special Function Register. Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

6. Port 3:

Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. When 1s are written to Port 3 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pull-ups. Port 3 receives some control signals for Flash programming and verification. Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table 2.

P3.0 RXD	(serial input port)
P3.1 TXD	(serial output port)
P3.2 INT0	(external interrupt 0)
P3.3 INT1	(external interrupt 1)
P3.4 T0	(timer 0 external input)
P3.5 T1	(timer 1 external input)
P3.6 WR	(external data memory writes strobe)
P3.7 RD	(external data memory read strobe)

Table 1.2 : Port Pin Alternate Functions

7. RST:

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives high for 98 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

8. ALE/PROG:

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming. If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

9. PSEN:

Program Store Enable (PSEN) is the read strobe to external program memory. When the AT89S52 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

10. EA/VPP:

External Access Enable. EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset. EA should be strapped to VCC for internal program executions. This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming.

11. XTAL1:

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

12. XTAL2:

Output from the inverting oscillator amplifier. Capture/Reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

1.5. Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in (Table 1.3)

0FBH								0FFH
0F0H	00000000							0F7H
0EBH								0E5H
0E0H	ACC 00000000							0E7H
0DBH								0DFH
0D0H	PSW 00000000							0D7H
0CBH	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0BBH	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0ABH	IE 0X000000							0AFH
0A0H	P2 11111111		AUXR1 XXXXXXXX0				WDTRST XXXXXXXXX	0A7H
9BH	SCON 00000000	SBUF XXXXXXXXX						9FH
90H	P1 11111111							97H
8BH	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXXX00XX0	8FH
80H	P0 11111111	SP 00001111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	PCON 00000000	87H

Table -1. 3. AT89S52 SFR Map and Reset Values

*** Here not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

Timer 2 Registers: Control and status bits are contained in registers T2CON (shown in Table -1.4) and T2MOD (shown in Table – 1.5) for Timer 2. The register pair (RCAP2H, RCAP2L) are the Capture/Reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

T2CON Address = 0C8H				Reset Value = 0000 0000B				
Bit Addressable								
Bit	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	$\overline{C/T2}$	$\overline{CP/RL2}$
	7	6	5	4	3	2	1	0
Symbol	Function							
TF2	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.							
EXF2	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).							
RCLK	Receive clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.							
TCLK	Transmit clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.							
EXEN2	Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.							
TR2	Start/Stop control for Timer 2. TR2 = 1 starts the timer.							
$\overline{C/T2}$	Timer or counter select for Timer 2. $\overline{C/T2}$ = 0 for timer function. $\overline{C/T2}$ = 1 for external event counter (falling edge triggered).							
$\overline{CP/RL2}$	Capture/Reload select. $\overline{CP/RL2}$ = 1 causes captures to occur on negative transitions at T2EX if EXEN2 = 1. $\overline{CP/RL2}$ = 0 causes automatic reloads to occur when Timer 2 overflows or negative transitions occur at T2EX when EXEN2 = 1. When either RCLK or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.							

Table 1.4. T2CON – Timer/Counter 2 Control Register

T2MOD Address = 0C9H				Reset Value = XXXX XX00B				
Not Bit Addressable								
Bit	-	-	-	-	-	-	T2OE	DCEN
	7	6	5	4	3	2	1	0
Symbol	Function							
-	Not implemented, reserved for future							
T2OE	Timer 2 Output Enable bit							
DCEN	When set, this bit allows Timer 2 to be configured as an up/down counter							

Table 1.5. T2MOD – Timer 2 Mode Control Register

Interrupt Registers: The individual interrupt enable bits are in the IE register. Two priorities can be set for each of the six interrupt sources in the IP register.

Dual Data Pointer Registers: Accessing both internal and external data memory, two banks of 16-bit Data Pointer Registers are provided: DP0 at SFR address locations 82H-83H and DP1 at 84H-85H. Bit DPS = 0 in SFR AUXR1 selects DP0 and DPS = 1 selects DP1.

AUXR	Address = 8EH	Reset Value = XXX00XX0B							
	Not Bit Addressable								
		–	–	–	WDIDLE	DISRTO	–	–	DISALE
Bit		7	6	5	4	3	2	1	0
–	Reserved for future expansion								
DISALE	Disable/Enable ALE								
	DISALE	Operating Mode							
	0	ALE is emitted at a constant rate of 1/6 the oscillator frequency							
	1	ALE is active only during a MOVX or MOVC instruction							
DISRTO	Disable/Enable Reset out								
	DISRTO								
	0	Reset pin is driven High after WDT times out							
	1	Reset pin is input only							
WDIDLE	Disable/Enable WDT in IDLE mode								
	WDIDLE								
	0	WDT continues to count in IDLE mode							
	1	WDT halts counting in IDLE mode							

Table 1.6. AUXR: Auxiliary Register

Power Off Flag: The Power Off Flag (POF) is located at bit 4 (PCON.4) in the PCON SFR. POF is set to “1” during power up. It can be set and reset under software control and is not affected by reset.

AUXR1	Address = A2H	Reset Value = XXXXXXX0B							
	Not Bit Addressable								
		–	–	–	–	–	–	–	DPS
Bit		7	6	5	4	3	2	1	0
–	Reserved for future expansion								
DPS	Data Pointer Register Select								
	DPS								
	0	Selects DPTR Registers DP0L, DP0H							
	1	Selects DPTR Registers DP1L, DP1H							

Table 1.7. AUXR1: Auxiliary Register 1

1.6. Memory Organization

MCS-51 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

1.6.1 Program Memory

If the EA pin is connected to GND, all program fetches are directed to external memory. On the AT89S52, if EA is connected to VCC, program fetches to addresses 0000H through 1FFFH are directed to internal memory and fetches to addresses 2000H through FFFFH are to external memory.

1.6.2 Data Memory

The AT89S52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions which use direct addressing access the SFR space.

1.7. Watchdog Timer (One-time Enabled with Reset-out)

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets.

1.7.1 Using the WDT

To enable the WDT, we write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, we need to service it by writing 01EH and 0E1H to WDTRST to avoid a WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH), and this will reset the device. When the WDT is enabled, it will increment every machine cycle while the oscillator is running. This means we must reset the WDT at least every 16383 machine cycles. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write-only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST pin.

1.7.2 WDT During Power-down and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power down mode, we do not need to service the WDT. There are two methods of exiting Power-down mode:

1. By hardware reset or via a level-activated external interrupt which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S52 is reset.
2. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode.

To ensure that the WDT does not overflow within a few states of exiting Power-down, it is best to reset the WDT just before entering Power-down mode. Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S52 while in IDLE mode, we should always set up a timer that will periodically exit IDLE, service the WDT, and reenter IDLE mode. With WDIDLE bit enabled, the WDT will stop to count in IDLE mode and resumes the count upon exit from IDLE.

1.8. UART

The UART in the AT89S52 operates the same way as the UART in the AT89C51 and AT89C52.

1.9. Timer 0 and 1

Timer 0 and Timer 1 in the AT89S52 operate the same way as Timer 0 and Timer 1 in the AT89C52.

1.10. Timer 2

Timer 2 is a 16-bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/T2 in the SFR T2CON (shown in Table 4). Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON, as shown in Table 8. Timer 2 consists of two 8-bit registers, TH2 and TL2. In the Timer function, the TL2 register is incremented every machine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

RCLK +TCLK	CP/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)

Table 1.8. Timer 2 Operating Modes

In the Counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since two machine cycles (24 oscillator periods) are required to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency. To ensure that a given level is sampled at least once before it changes, the level should be held for at least one full machine cycle.

1.10.1 Capture Mode

In the capture mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16-bit timer or counter which upon overflow sets bit TF2 in T2CON. This bit can then be used to generate an interrupt. If EXEN2 = 1, Timer 2 performs the same operation, but a 1-to-0 transition at external input T2EX also causes the current value in TH2 and TL2 to be captured into RCAP2H and RCAP2L, respectively.

In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. The EXF2 bit, like TF2, can generate an interrupt.

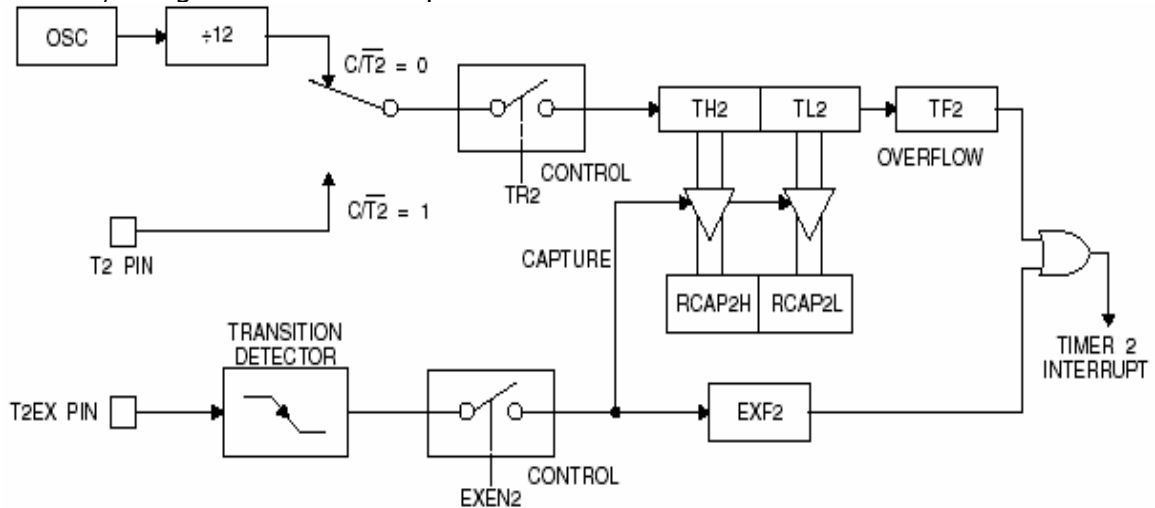


Figure 1.3. Timer in Capture Mode

1.10.2 Auto-reload (Up or Down Counter)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (in Table 10-2). Upon reset, the DCEN bit is set to 0 so that timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down, depending on the value of the T2EX pin.

1.11. Baud Rate Generator

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 1.4). Baud rates for transmit and receive can be different if Timer 2 is used for the receiver or transmitter and Timer 1 is used for the other function. The baud rate generator mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate as following Equation.

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

Normally, as a timer, it increments every machine cycle (at 1/12 the oscillator frequency). As a baud rate generator, however, it increments every state time (at 1/2 the oscillator frequency). The baud rate formula is given below.

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

Where (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer. Timer 2 as a baud rate generator is shown in Figure 4.

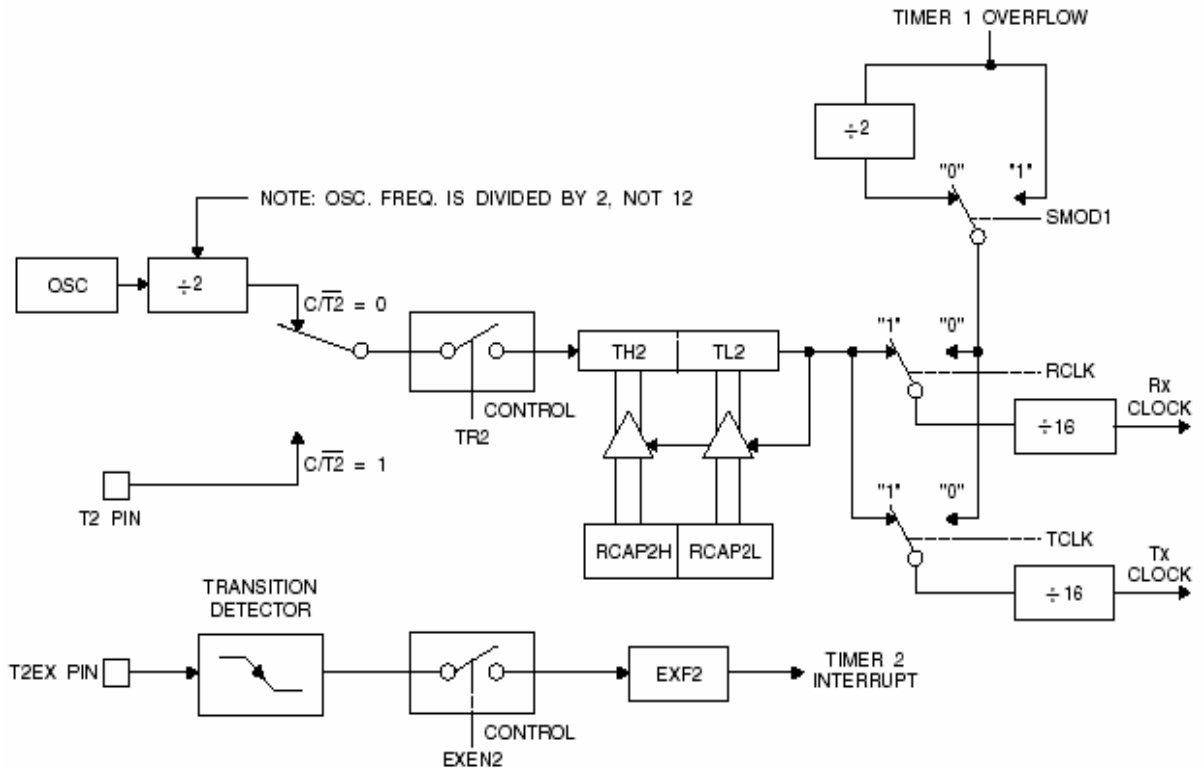


Figure 1.4. Timer 2 in Baud Rate Generator Mode

1.12. Programmable Clock Out

A 50% duty cycle clock can be programmed to come out on P1.0, as shown in Figure 1.5. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed to input the external clock for Timer/Counter 2 or to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz (for a 16-MHz operating frequency). To configure the Timer/Counter 2 as a clock generator, bit C/T2 (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer. The clock-out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L), as following equation.

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

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

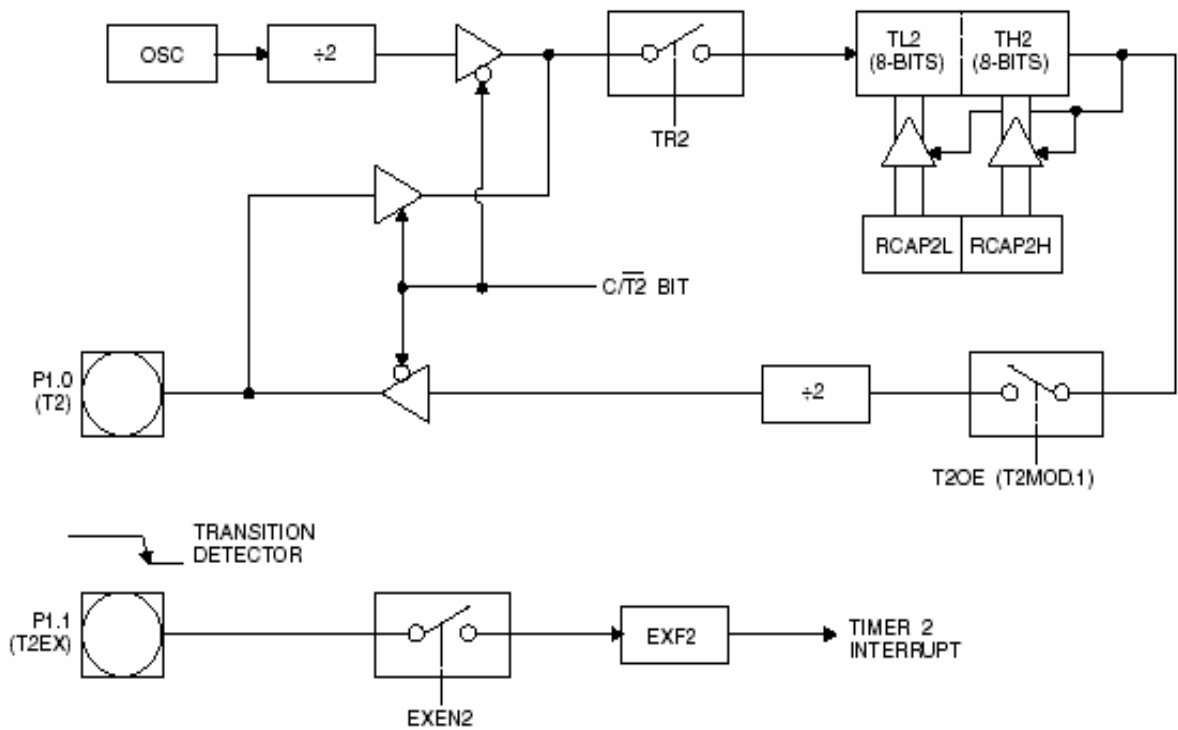
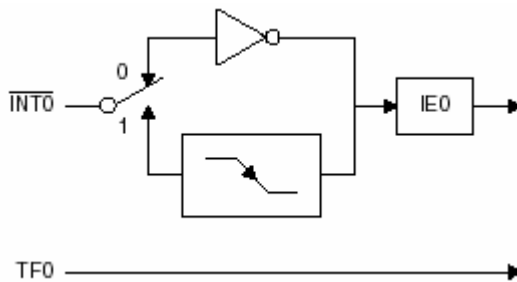


Figure 1.5. Timer 2 in Clock-Out Mode

1.13. Interrupts

The AT89S52 has a total of six interrupt vectors: two external interrupts (INT0 and INT1), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. These interrupts are all shown in Figure 1.6. Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once. Timer 2 interrupt is generated by the logical OR of bits TF2 and EXF2 in register T2CON. 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 that bit will have to be cleared in software. The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.



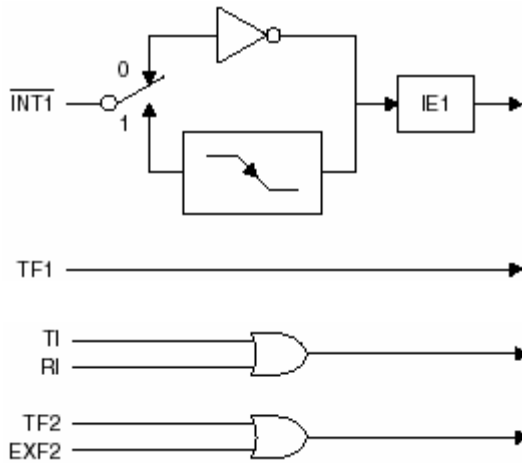


Figure 1.6. Interrupt Sources

(MSB)		(LSB)					
EA	-	ET2	ES	ET1	EX1	ET0	EX0
Enable Bit = 1 enables the interrupt.							
Enable Bit = 0 disables the interrupt.							
Symbol	Position	Function					
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.					
-	IE.6	Reserved.					
ET2	IE.5	Timer 2 interrupt enable bit.					
ES	IE.4	Serial Port interrupt enable bit.					
ET1	IE.3	Timer 1 interrupt enable bit.					
EX1	IE.2	External interrupt 1 enable bit.					
ET0	IE.1	Timer 0 interrupt enable bit.					
EX0	IE.0	External interrupt 0 enable bit.					
User software should never write 1s to reserved bits, because they may be used in future AT89 products.							

Table 1.9. Interrupt Enable (IE) Register

1.14. Oscillator Characteristics

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as in Figure 1.7. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 8. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two

flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

1.15. Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

When 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 to a port pin when idle mode is terminated by a reset, the instruction following the one that invokes idle mode should not write to a port pin or to external memory.

1.16. Power-down Mode

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by an enabled external interrupt. Reset redefines the SFRs but does not change the on-chip RAM.

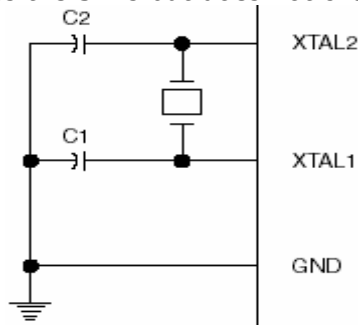


Figure 1.7. Oscillator Connections

Note: 1. C1, C2 = 30 pF □ 10 pF for Crystals
 = 40 pF □ 10 pF for Ceramic Resonators

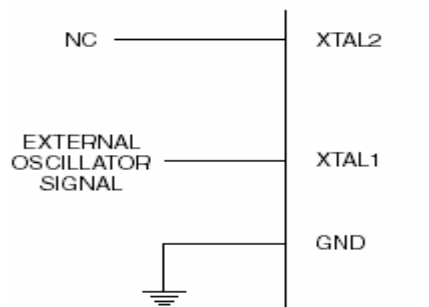


Figure 1.8. External Clock Drive Configuration

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
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

Table 10. Status of External Pins during Idle and Power-down Modes

1.17. Program Memory Lock Bits

The AT89S52 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in Table 1.11.

Program Lock Bits				Protection Type
	LB1	LB2	LB3	
1	U	U	U	No program lock features
2	P	U	U	MOV _C instructions executed from external program memory are disabled from fetching code bytes from internal memory, \overline{EA} is sampled and latched on reset, and further programming of the Flash memory is disabled
3	P	P	U	Same as mode 2, but verify is also disabled
4	P	P	P	Same as mode 3, but external execution is also disabled

Table 1.11. Lock Bit Protection Modes

When lock bit 1 is programmed, the logic level at the EA pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value and holds that value until reset is activated. The latched value of EA must agree with the current logic level at that pin in order for the device to function properly.

1.18. Programming the Flash – Parallel Mode

The AT89S52 is shipped with the on-chip Flash memory array ready to be programmed. The programming interface needs a high-voltage (12-volt) program enable signal and is compatible with conventional third-party Flash or EPROM programmers. The AT89S52 code memory array is programmed byte-by-byte.

Programming Algorithm: when we programming the AT89S52, the address, data, and control signals should be set up according to the “Flash Programming Modes” (Table 12). To program the AT89S52, we are following the steps:

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.

3. Activate the correct combination of control signals.
4. Raise EA/VPP to 12V.
5. Pulse ALE/PROG once to program a byte in the Flash array or the lock bits. The bytewrite cycle is self-timed and typically takes no more than 50 μ s. Repeat steps 1 through 5, changing the address and data for the entire.

Data Polling: The AT89S52 features Data Polling to indicate the end of a byte write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P0.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. Data Polling may begin any time after a write cycle has been initiated.

Ready/Busy: The progress of byte programming can also be monitored by the RDY/BSY output signal. P3.0 is pulled low after ALE goes high during programming to indicate BUSY. P3.0 is pulled high again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification.

The status of the individual lock bits can be verified directly by reading them back.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 000H, 100H, and 200H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

(000H) = 1EH indicates manufactured by Atmel

(100H) = 52H indicates AT89S52

(200H) = 06H

Chip Erase: In the parallel programming mode, a chip erase operation is initiated by using the proper combination of control signals and by pulsing ALE/PROG low for duration of 200 ns - 500 ns.

In the serial programming mode, a chip erase operation is initiated by issuing the Chip Erase instruction. During chip erase, a serial read from any address location will return 00H at the data output.

1.19. Programming the Flash – Serial Mode

The Code memory array can be programmed using the serial ISP interface while RST is pulled to VCC. The serial interface consists of pins SCK, MOSI (input) and MISO (output). After RST is set high, the Programming Enable instruction needs to be executed first before other operations can be executed. Before a reprogramming sequence can occur, a Chip Erase operation is required. The Chip Erase operation turns the content of every memory location in the Code array into FFH. Either an external system clock can be supplied at pin XTAL1 or a crystal needs to be connected across pins XTAL1 and XTAL2. The maximum serial clock (SCK) frequency should be less than 1/16 of the crystal frequency. With a 33 MHz oscillator clock, the maximum SCK frequency is 2 MHz.

1.20. Serial Programming Algorithm

To program and verify the AT89S52 in the serial programming mode, the following sequence is recommended:

1. Power-up sequence:

- a. Apply power between VCC and GND pins.
- b. Set RST pin to "H".

If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 33 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.

2. Enable serial programming by sending the Programming Enable serial instruction to pin MOSI/P1.5. The frequency of the shift clock supplied at pin SCK/P1.7 needs to be less

Than the CPU clock at XTAL1 divided by 16.

3. The Code array is programmed one byte at a time in either the Byte or Page mode. The write cycle is self-timed and typically takes less than 0.5 ms at 5V.

4. Any memory location can be verified by using the Read instruction which returns the Content at the selected address at serial output MISO/P1.6.

5. At the end of a programming session, RST can be set low to commence normal device Operation.

Power-off sequence (if needed):

1. Set XTAL1 to "L" (if a crystal is not used).
2. Set RST to "L".
3. Turn VCC power off.

Data Polling: The Data Polling feature is also available in the serial mode. In this mode, during a write cycle an attempted read of the last byte written will result in the complement of the MSB of the serial output byte on MISO.






Mode	V _{CC}	RST	PSEN	ALE/ PROG	EA/ V _{PP}	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.4-0	P1.7-0
												Address	
Write Code Data	5V	H	L	 ⁽²⁾	12V	L	H	H	H	H	D _{IN}	A12-8	A7-0
Read Code Data	5V	H	L	H	H	L	L	L	H	H	D _{OUT}	A12-8	A7-0
Write Lock Bit 1	5V	H	L	 ⁽³⁾	12V	H	H	H	H	H	X	X	X
Write Lock Bit 2	5V	H	L	 ⁽³⁾	12V	H	H	H	L	L	X	X	X
Write Lock Bit 3	5V	H	L	 ⁽³⁾	12V	H	L	H	H	L	X	X	X
Read Lock Bits 1, 2, 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
Chip Erase	5V	H	L	 ⁽¹⁾	12V	H	L	H	L	L	X	X	X
Read Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	X 0000	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	52H	X 0001	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	06H	X 0010	00H

Table 1.12. Flash Programming Modes

- Notes:
1. Each PROG pulse is 200 ns - 500 ns for Chip Erase.
 2. Each PROG pulse is 200 ns - 500 ns for Write Code Data.
 3. Each PROG pulse is 200 ns - 500 ns for Write Lock Bits.
 4. RDY/BSY signal is output on P3.0 during programming.
 5. X = don't care.

Chapter 2

Capture and Layout

2.1. Introduction

Orcad family products offer a total solution for our core design tasks: schematic and VHDL-based design entry; digital, analog, and mixed-signal simulation; and printed circuit board layout. Orcad family products are a suite of applications built around an engineer's design flow-not just a collection of independently developed point tools.

Capture is a versatile design entry product you can use to create schematics for analog or mixed signal designs, printed circuit board layout designs, and programmable logic designs. First, create your flat or hierarchical design in the schematic page editor, and then use Capture's tools to Quickly annotate it and prepare it for the next stage of development.



Figure 2.1. Capture's session frame

. When we start capture, the session frame appears. We initially open or create a design or library from the session frame, the minimized session log appears log contains a log of events or errors that occur during the Capture session, If we update part references or properties, create a netlist, or check design rules, information is recorded in the session log. Click the restore button to open the window. After the session log is reviewed, it can be minimized again. The session log continues to record events even when it's minimized.

5. *** Information that appears in the status bar when we work on a schematic page.

2.2. Designs and Schematics

Capture stores all of a design's schematics, schematic pages, and parts in a single file. This makes it easy to handle our designs. When we open a capture design, it displays in a project manager window.

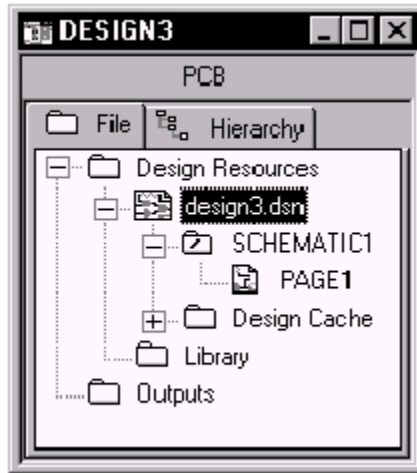


Figure 2.2. New project manager window

A Capture design contains one or more schematic folders, and each schematic folder contains one or more schematic pages. So even the simplest design contains one schematic folder with one schematic page, and a complicated design could many schematic pages. Designs include a design cache, which serves as an embedded archive of each unique part and symbol used in the design. We can set up your schematic page to be any size that our printer or plotter can handle. A schematic folder is simply a collection of schematic pages, which are logically connected by off-page connectors or hierarchical ports and pins. A design is a collection of related schematic folders. It contains hierarchy and back annotation information, as well as the design cache. Wire segments and off-page connectors on the same schematic page are electrically connected if they have the same name.

*** Design Rules Check process warns we a schematic folder contains two unconnected objects with the same name.

2.3. Navigating Designs

Capture's project manager provides a familiar graphical view of our design structure. We easily locate a single schematic page and see how it fits into the whole design. We can display our design structure in two different ways: as a file view or as hierarchy view.

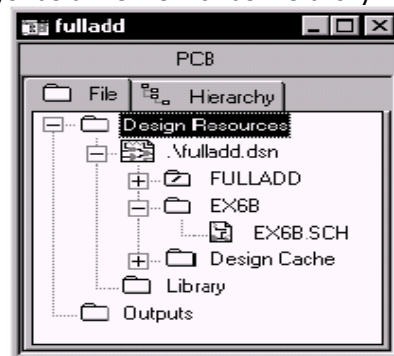


Figure 2.3. File tab

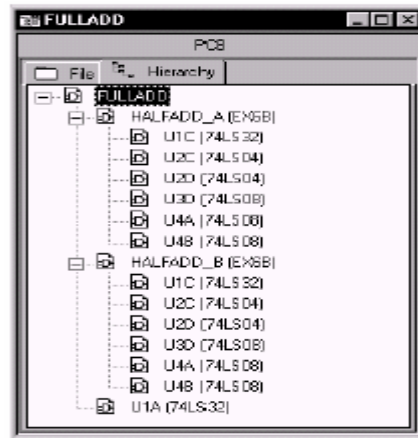


Figure 2.4. *Hierarchy tab*

Clicking the File tab displays the file view, which shows a list of all project files organized in folders. Clicking the Hierarchy tab displays the hierarchy view, which shows a hierarchical relationship among the project components. For example, this hierarchical design includes the FULLADD and HALFADD schematic folders. The file view of the design shows each schematic folder and its schematic pages, but doesn't show any hierarchical relationships. Either double-click on a schematic folder or click on the plus sign next to the schematic folder to show or hide its schematic pages. Double-click on a schematic page to open it in the schematic page editor. For a hierarchical design, you're asked which schematic page occurrence to open. The hierarchy view of the design shows the relationship of the schematic pages in a hierarchical design. Now you can see that there are actually two instances of the HALFADD schematic page in this design. Double-clicking on a schematic page or clicking on its plus sign shows or hides the parts used on the schematic page. Select a part in the hierarchy view, and then click Edit from the pop-up menu. The HALFADD schematic page opens and the part is selected. Any changes made on one instance of HALFADD are reflected on every instance of HALFADD in the design.

From the project manager, we can selectively browse objects on one or more selected schematic pages. We can sort the list various ways by clicking one of the buttons at the top of the columns. If we double-click on an item's part reference, the schematic page containing the item opens and the item is selected. We can also use the browse results to edit multiple parts in one easy step by using the Edit Properties dialog box. Use the standard Windows selection methods—including the SHIFT and CTRL keys—to select the parts we want to edit. From the Edit menu, click Properties (CTRL+ E) to display the Browse Spreadsheet dialog box. The Browse Spreadsheet dialog box lists the objects selected in the browse window. As an example, let's specify a PCB footprint for all the selected parts. Click the PCB Footprint cell for any one of the parts, type the footprint name, and press ENTER. Double-click on the cell again, and then click the Copy button. This sets it for applying in the next step. Click the PCB Footprint button at the top of the column, and then click the Paste button. All the cells in the column get the pasted text. To add a property to the parts in the spreadsheet, click the New button and enter a name and value. Click OK to record our changes.

Capture adds the new property and its property value to all the parts listed in the Browse Spreadsheet dialog box. We can also browse nets, hierarchical ports, off-page connectors, titleblocks, bookmarks, and DRC markers.

2.4. Editing a schematic page: Capture's pulldown menus and shortcut keys make editing a schematic page an easy task-and the right mouse button makes it even easier. The right mouse button provides fast access to all the editing functions for the selected objects. We select an objects by clicking the select tool from the tool palette, placing the pointer on a part, then either clicking the left mouse button or pressing the SPACE BAR. Now click the right mouse button to display a pop-up menu of editing commands for the selected object. For example, the selected part be mirrored, rotated 90 degrees, or deleted. We can edit the part's properties and zoom in or out. We can add to the selection set by holding down the CTRL key and clicking on additional objects.

Or hold the CTRL key down and click on a selected object to remove it from the selection set. To select a block of objects, click the select tool on the palette and enclose the objects in a selection rectangle. Hold down the left mouse button, drag the pointer diagonally across the objects you want to select and release the mouse button. To move a wire or bus segment place the pointer on it and hold down the left mouse button to select it.

Hold the SHIFT key while moving the mouse to 'rubberband' any connecting wires or buses. You can separate a segment from the rest of the wire by holding down the ATL key while you drag the wire. Now the wire segment is separated from its original connections. We can move several objects at the same time by enclosing them within a selection rectangle, and clicking Group from the Edit menu (ATL, E, G). Now we can move the group as a single object. Ungroup it when we finish moving it by clicking Ungroup from the Edit menu. Capture uses the Windows Clipboard to support standard cut, copy, and paste actions. Select an object and press CTR+C to copy it, then select another schematic page and press CTRL+V to paste the object. Or open another Windows application and paste the object into the appropriate location.

2.5. Making Connections

Capture makes it easy to place the parts for our design. After we place the parts, we electrically connect them. All the electrical information is stored in the design, making it easy to progress from placing parts to creating a Netlist. When we place a part on a schematic page, the part can be placed without a referenced designator assigned to it, or it has one automatically assigned. To specify either automatic reference designators, click Preferences from the Options menu. The Preferences dialog box appears. Click the Miscellaneous tab, then either select or clear the checkbox for the automatically reference place parts option and click OK. To place a part click the Place part tool on the tool palette, or click Part from the Place menu. The parts for all selected libraries display in the list box. To exclude the parts in a selected library from being listed, press CTRL as we select the library again to deselect the library. Click the Add Library button to add the contents of another part library. We can type a selective search string to instantly locate a specific part or set of parts. We need to edit the selected part click the Edit Part button. Click OK to place the part. Click the left mouse button to place the resistor. We can continue placing resistors, then quit placing them by clicking End Mode from the

pop-up menu or by pressing ESC. Click the Place part tool on the tool palette again to place a different part. You've placed the parts, now click the Place wire tool on the tool palette to create the electrical connections between them. Place the pointer on the pin where we want to start, and then click the left mouse button. Move the pointer and click the left mouse button again to change the wire direction. We can cross wires without connecting them, or click on a wire as you intersect it to automatically place a junction. To end a wire, just click on a pin. The unconnected pin box disappears, indicating the connection with the wire has been made. We can also place junctions on previously placed wires. Click the Place junction tool on the tool palette, and then press the left mouse button over two crossing wires. Now the wires are connected. We can also remove junctions using the same tool. Just place a new junction on an existing one to delete the placed junction. We place buses just like wires. Click the Place bus tool on the tool palette and draw a bus, and then click the Place bus entry tool on the tool palette to make the connections. Just place the first bus entry by hand, now hold down the CTRL key and drag a copy of the bus entry to where we want the second one, and then click Repeat Place from the Edit menu to place additional bus entries. To finish, click the Place wire tool on the tool palette to connect the bus to individual signals. If a pin is defined as a bus pin, you can connect the bus to it directly.

The net for this wire continues on another schematic page. To connect them, click the Place off-page connector tool on the tool palette. Select an off-page connector and click OK. Now place it on the end of the wire. The off-page connector indicates that the net spans multiple schematic pages. When we create a wire, it's automatically assigned a netname. We can view the netname by double-clicking on the wire. Capture displays the Net Properties dialog box. Use the Place net alias tool on the tool palette to assign a net alias. Read the definition, and then click Continue. Enter the net alias in the dialog box. We can make the net alias text look different by clicking the Change button, then choosing a different font or size. Move the net alias to the wire, then place the pointer on the wire and click to assign the alias to that wire's net on the schematic page. Once it's placed, we can move the text where you like.

We can select an entire net on a schematic page. Just select a wire, then press the right mouse button and click Select Entire Net. Now all connected wires for the net on this schematic page are selected.

2.6. Editing properties

Almost every object in a Capture design or library is described by one or more sets of named values, which are called properties. We can change most property values as our work on a design. For example, we can customize the way Capture appears and operates by changing options and preferences. To set options that aren't design-specific, click Preferences from the Options menu. We specify objects colors, grid style, area selection, and other settings in this dialog box. Click Design Template from the Options menu to change design-specific options. Here we can set the fonts, page size, and grid references. We can also define a title block for designs. Each schematic page in a design contains properties that specify the page size and orientation.

To set these properties, click Schematic Page Properties from the Options menu. Choose from standard page sizes, or define a custom size.

A part's properties define its graphical and electrical characteristics. It's easy to change individual part reference and part value properties. For example, let's double-click on a part value. we can change its display properties by specifying a different font, display color, or orientation. we can also make the text invisible. Several part properties can be changed without editing them individually. Just double-click on the part, to display the property editor. The property editor lists all the properties and their values for the selected part. If we make a property value change to the selected part instance, the change is reflected in all occurrences of that part. However, if we make a change to a part occurrence property value, only that occurrence contains the changed value. It's easy to add new properties. For example, select a net on a schematic page and click New Column in the Property editor. Enter the property name and its value, and then click OK. We can also click the property value cell for the new property and add the value. This new value can be changed for individual part occurrences.

With this capability you can assign unique net property values, such as specific PCB track widths, depending on our design requirements. If the objects in the spreadsheet have many properties, it may be easier to read and edit if we pivot the spreadsheet. To do this, double-click the top left cell.

In this example, now the part instances and occurrences appear in columns across the top, and the part properties appear in rows. To pivot the spreadsheet back to its original orientation, just double-click the top left cell again or display the pop-up menu and click Pivot.

2.7. Managing Parts of Libraries

With Capture, we don't have to keep track of multiple libraries. All the information Capture needs about every part in our design is kept in the design itself. Designs are completely self-contained and portable.

When we open and expand a library, the project manager displays all the parts in the library. A Capture library can contain parts, packages, symbols, and schematic folders. In the library, they're all treated as parts. We can move parts from one library to another by using the cut and paste toolbar buttons, or by using drag-and-drop. We can copy a part to another library by using the copy and paste toolbar buttons, or by pressing the CTRL key while we drag the part to the library. Likewise, we can copy a schematic folder from a design to a library. This makes it easy to reuse a schematic folder in several designs. When we move a schematic folder to a library, the library cache shows all the parts contained in the schematic folder. To open a part in the part editor, select the part and click Edit Part from the pop-up menu. When we create a part, we can attach a schematic folder to it by clicking the Attach Implementation button. A part with an attached schematic folder is a nonprimitive part. And a part without an attached schematic folder is usually a primitive part. we can also attach a schematic folder to a part while we are editing it in the part editor. From the View menu, click Package, Next click Package Properties from the Options menu. Click the Attach Implementation button, and then select Schematic View from the drop-down list. Enter the schematic folder name, then the path to the library containing the schematic folder. We may want Capture to process a part as a primitive, even though it has a schematic folder attached. In the schematic editor, select the part we want to change, then press the right mouse button and click Edit Properties from the pop-up menu. In the property editor, select yes in the Primitive cell to define the part as a primitive. Now the attached schematic folder is ignored.

2.8. Making parts

We may need to create a unique part. Capture makes it easy. To create a new part in an open library, select the library in the project manager and click New Part from the Design menu. In the New Part Properties dialog box, we assign a part name, a part reference prefix, and a name for its assigned PCB footprint. To create a package, select the package type and enter the number of parts in the package, then click OK. These packages have two parts. An empty part outline appears in the part editor. To draw the part, use the Place menu commands or the tool palette drawing tools. After the part is drawn, we need to add pins. From the Place menu, click Pin. Specify the pin name and number, and whether the pin represents one signal or a bus. If we select Bus, a bus can be connected directly to the pin. Pin shape and pin type determine the pin's appearance and electrical characteristics. Now place the pins. When we place more than one pin, Capture automatically increments the pin numbers. The name of Pin 2 needs to be changed to OUT. Just double-click on the pin, then change the name and click OK. Pin 2 is now an output pin. Let's view the next part in this package. From the View menu, click Next Part. The pin names for the next part are already assigned, but we need to add pin numbers. Just double-click on a pin, then type the pin number and click OK. Use the same procedure to change the other pin number.

Select the part border and stretch it so it tightly encloses the drawn part. The part value placeholder moves along with the stretched part border. If we want the pin names to be invisible for the part we are editing, click Part Properties from the Options menu. Select the Pin Names Visible property, and then select False from the drop-down list. The change applies to all parts in a multiple part package. If we also want the pin numbers invisible, set Pin Numbers Visible to False. Click OK to close the User Properties dialog box. If we need to change the PCB footprint that's assigned to the part, click Package Properties from the Options menu. Type the new PCB footprint name and click OK. Now that the two parts are done, let's view both of them.

From the View menu, click Package. The package view shows all the parts in multiple part package. In the package view, we can double-click on any part to edit it in the editor. If the part package looks OK, click Part from the View menu to return to the part editor. From the File menu, click Save to save the finished part.

2.9. Processing our design

Once we have laid out our design and connected the parts and pages, we are ready to update the part reference, verify the design's integrity, and create a netlist. Capture puts all the tools we need for processing our design on one handy menu.

- *Annotate (update part references)
- * Bill of materials
- * Cross reference parts
- * Create a netlist
- * Design rules check
- * Back annotate

If we didn't have part reference automatically assigned, then updating the part reference in a design is easy. From the Tool menu, click Annotate, then choose whether you want to

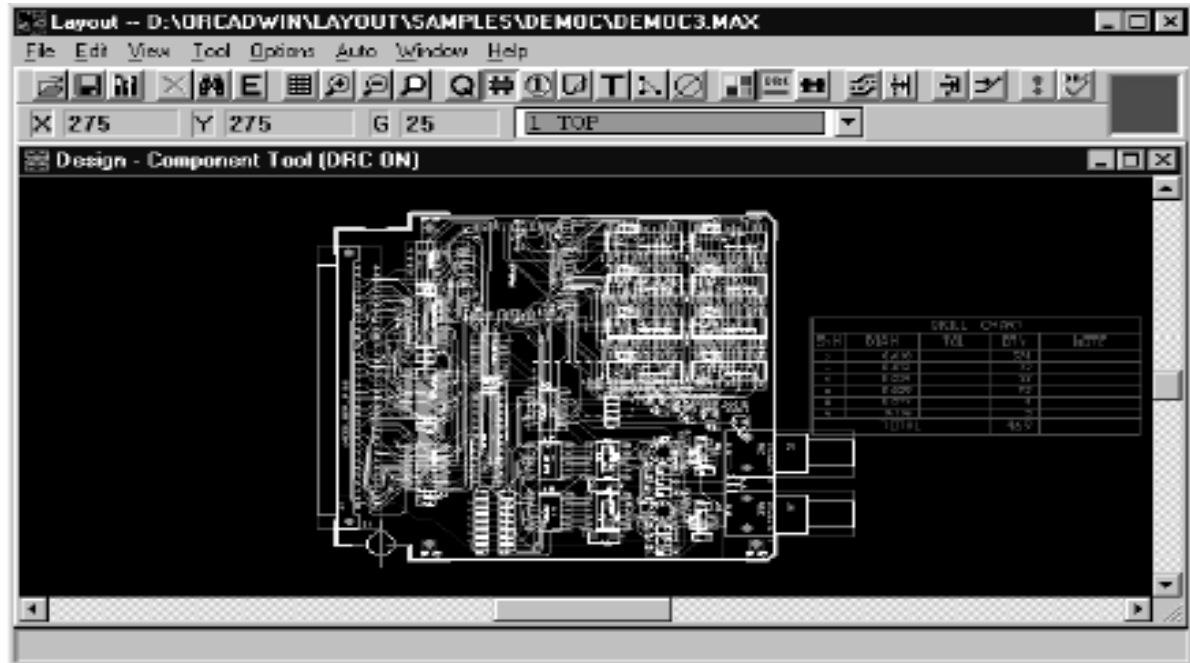
annotate part references for the entire design, or for specific schematic pages. We can also choose the type of reference update, we want to use. To check the integrity of a design, click Design Rules Check from the Tools menu. Click the ERC Matrix tab to configure test criteria for the electrical rules check. We can use the default setting, or customize the electrical rules check configuration. Capture places DRC error markers on the schematic pages, and writes related information to a report file. Use the project manager to browse for DRC errors. From the Edit menu, click Browse and then DRC Markers.

We need to create a Netlist before we produce a circuit board layout of your PCB design. Netlists are also used in digital and analog simulation. Select the design, and then click Create Netlist from the Tools menu, Click a tab to specify the type of Netlist we want to produce. Now specify the combined property strings, formatting options, and the Netlist filename. Check the View Output option to view the Netlist in an associated text editor, the Netlist is also added to the Outputs folder in the project manager. We can also view any text-based Netlist or report by double-clicking its entry in the Outputs folder. The item opens in Capture's text editor. If pin assignment, gates, or part references are changed by another tool, we can update our design. From the Tools menu, click Bake Annotate, then enter the name of the back annotation file created by we or by the other tool. Click OK to update the design. We can document our design by creating a cross-reference list of its parts. From the Tools menu, click Cross Reference, Now select processing options and specify an output filename. The report is added to the Outputs folder in the project manager. To create a bill of materials, click Bill of Materials from the Tools menu. Then select processing options and specify an output filename. The report is added to the Outputs folder in the project manager.

2.10. Introduction of Layout

OrCAD Layout (R) is a powerful circuit board layout tool that has all the automated functions we need to quickly complete our board. In Layout, we should set up the board before we begin placing components. The steps involved in the board setup process are as follow, but not all of them are necessary for every board.

- Load a template
- Create a board outline
- Set the units of measurement
- Set system grids
- Add mounting holes
- Define the layer stack
- Set global spacing
- Define Padstacks
- Define vias
- Set net properties



This chart illustrates Layout's overall design process. Layout's part in the design flow. Using a schematic capture tool, such as OrCAD Capture (R), we create a Layout-compatible netlist. The netlist contains much of the design information that Layout uses to produce the board. If we change the netlist after back annotating in Layout and Capture, Layout's Auto ECO utility automatically update the board file. The next step in the design flow is placing components. After component placement we can either manually route or autoroute the board. After routing, we can perform cross probing. Using cross probing, selecting a net in Capture highlights the corresponding track in Layout.

Conversely, selecting a track in Layout highlights the corresponding net in Capture. For post-processing, Layout produces hardcopy on printers and plotters, and also produces Gerber files, DXF files, and a wide variety of report files. We can preview and edit our Gerber files with Layout's external Gerber editor, which is known as Gerb Tool. Now let's look at Layout itself when we start Layout the session frame displays. In the session frame we can open an existing board file or library, or specify a new one. We can also import and export design files, and run a group of supplemental design tools. Let's open an existing board file. From the File menu, choose Open, then select the board files and chose the Open button. The board displays in Layout's design window. We can choose frequently used commands from menus, or from a set of toolbar buttons. A short command description, called a Tool Tip, displays when the pointer is placed over a toolbar button.

2.11. Creating a new design

In Layout we need to set up the environment when we create a new board design. We can start a new design from Layout's session frame, or from the design window. From the session frame's File menu, choose new, select a technology template for the

new design, and then choose the Open button. Read the description of a technology template, and then choose Continue. Next select the netlist that provides the connectivity information and component types for our new design, then choose the Open button. Specify a filename and directory location for the new design then choose the Save button.

Layout's AutoECO utility creates the board file. In the design window, we can load a new technology template at any time. From the File menu, choose Load, select a technology template (.TCH), and then choose the Open button. To specify the units of measurement for the design, choose System Settings from the Options menu; modify the settings in the dialog box. Then choose the OK button.

Next we can edit the board outline that was supplied by the technology template, or create a new board outline. To begin a new board outline, choose the obstacle toolbar button, or choose Obstacle, then Select Tool, from the Tool menu. Double-click the left mouse button in the design window. The Edit Obstacle dialog box displays.

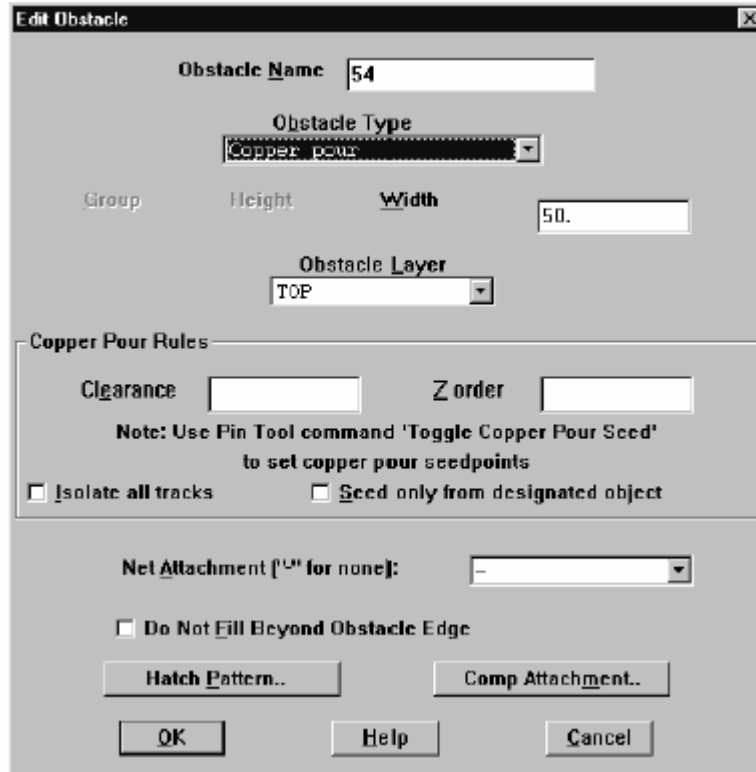


Figure 2.5. The Edit Obstacle dialog box

Modify the board outline settings, then choose the OK button and draw the board outline. To define the layer stack, choose the spreadsheet toolbar button, and then choose Layers. The Layers spreadsheet displays. Double-click in the Layer Name column of the layer we want to modify. The Edit Layer dialog box displays. Modify the settings and choose the OK button. To define grid settings for routes and vias, choose System Settings from the Options menu, modify the settings in the dialog box, and then choose the OK button. Now define global spacing rules for the pads, tracks, and vias in the

design. Choose the spreadsheet toolbar button, choose Strategy, and then choose Route Spacing. In the Route Spacing spreadsheet, double-click on a layer. The Edit Spacing dialog box displays. Modify the spacing values and choose the OK button. Padstacks for the new design are defined in the Layout footprint libraries. This point describes how to create a new padstack. Creating a new padstack

1. Choose the spreadsheet toolbar button.
2. Select Padstacks from the menu. The Padstacks spreadsheet displays.
3. Select a padstack and press the Insert key. This creates a copy of the padstack.
4. Double-click on the newly created padstack. The Edit Padstack dialog box displays.
5. Type a new name for the padstack in the Padstack text field, then edit other options.
6. Choose the OK button.

Next define the types of vias. Choose the spreadsheet toolbar button, then choose Padstacks. The Padstacks spreadsheet displays. Double-click on a Via name. The Edit Padstack dialog box displays. Configure the Via and choose the OK button. To specify settings for a free via matrix, choose Free Via Matrix Settings from the Options menu, modify the settings in the dialog box, and then choose the OK button. When we place a free via, the letters FV display on it, and it is added at the bottom of the Components spreadsheet. The last step in the setup process is setting objects or layer colors. Choose the color toolbar button to displays the Color spreadsheet, then double-click on a color in the Color spreadsheet. Select a new color in the Color dialog box, and then choose the OK button.

2.12. Autorotting

Layout's Autorouter uses sweep, shove, and interactive routing technology to provide maximum autorouting power and flexibility. With sweep technology, Layout begins autorotting in the center of the board. When the center is routed, the autorouter moves up and left, then routes the other quadrants. With shove technology, Layout finds the optimum space to route a track then checks whether it can shove other tracks or vias out of the way to make room for the track. With interactive technology, we can manually direct the routes wherever necessary, then use the autorouter.

This chart lists the steps in the autorouting process.

- Set net properties.
- Check the board outline, via definitions, and routing grid.
- Route the power and ground nets (on predominantly through-hole surface-mount boards).
- Preroute critical nets.
- Load a routing strategy file.
- Run the autorouter.
- Optimize routing using Layout's other routing tools.

There may be situations where we need to perform fanout, such as connecting an edge connector finger to a plane. Before we autoroute the board, we should manually route critical nets and lock them on the board.

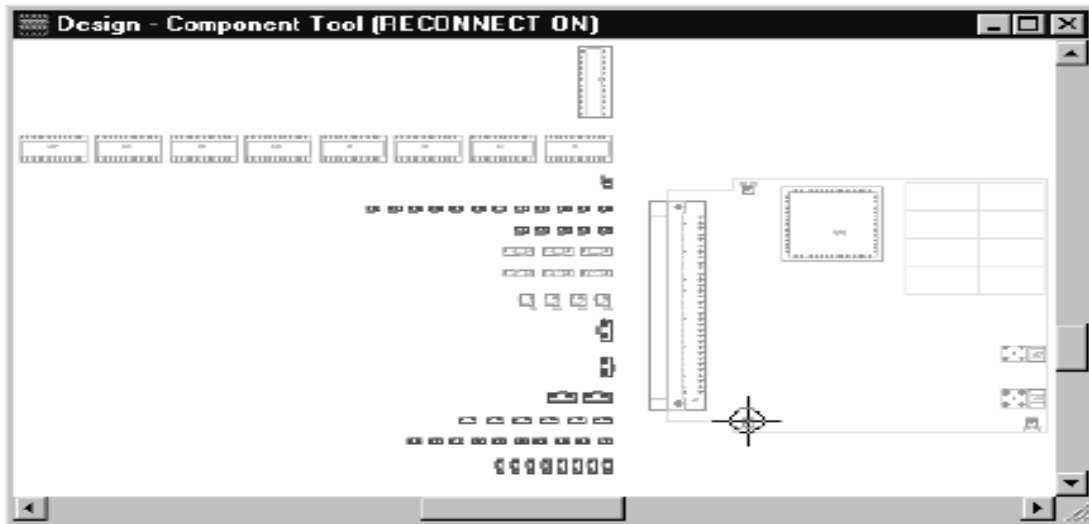


Figure 2. 6. A board prior to component placement

Load a routing strategy file by choosing Load from the File menu, selecting a file in the dialog box, then choosing the Open button, choose Autoroute, then Board. Layout performs up to six complete routing sweeps. The settings in the routing strategy file determine the actual routing parameters. If we need to manually reroute tracks, choose the shove track toolbar button. Layout automatically moves existing tracks out of the way as we route.

2.13. Finishing the design

Our board is completely routed and we are fixed any errors that were detected. Now we are ready to place the finishing touches on the design. Using the Components Renaming command, we can automatically rename component reference designators in the order of our choice. To rename the components, choose Components Renaming from the Options menu. Then Select the direction you want the renaming to occur, and then choose the OK button. From the auto menu, choose Rename Components, Layout renames the component reference designators. The next thing to do to finish our design is to place dimensions on the board. We may want to dimension the entire board, or just a single object. From the Tool menu, choose Dimension, then New. Press the right mouse button to display the pop-up menu, then choose Properties. Select Relative Dimensions or Absolute Dimensions, then specify the other options and choose the OK button. Choose the layer for the dimensions, and then press the INSERT key. Click the left mouse button where we want to begin the dimension, and then place it. Absolute dimensions are measured from the board datum. We can analyze the component connection density on our board using the density graph. From the view menu, choose Density Graph.

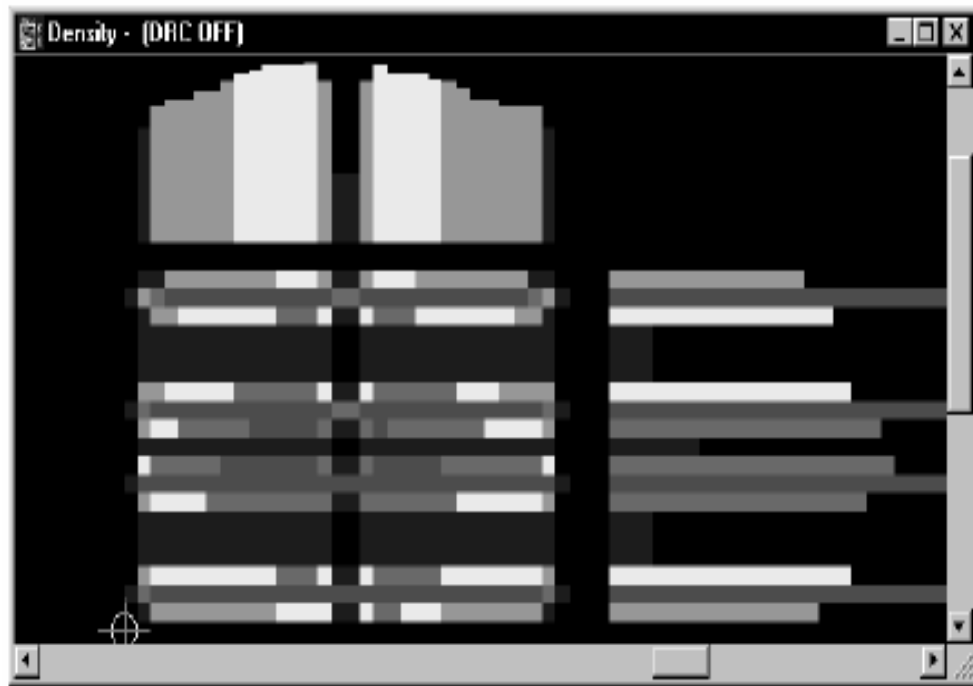


Figure 2.7. - density graph of Layout

The density graph displays a color graphical representation of the connection density. Blue and green colors indicate acceptable density; pink and red indicate high density.

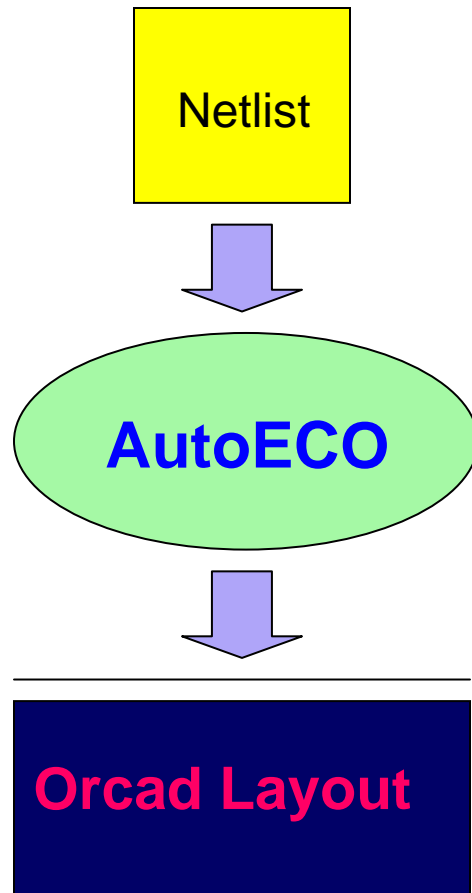
If areas of our board are too dense, indicated with dark red, you may improve the density by adjusting component placement or by trying different routing strategies.

2.14. Using AutoECO

AutoECO is Layout's Automatic Engineering change Order process that uses the information in a Layout netlist to produce a new board file, or update an existing board file.

These charts list the **AutoECO options** available in layout.

- AutoECO Adds and deletes components and nets, but doesn't override attributes. Used when Layout runs AutoECO automatically.
 - AutoEO/Override Attrs Creates a new board or merges new component and connections with an existing board. Overrides all of the attributes in an existing board.
 - AutoECO/Orerride Coords Creates a new boar or merges new components and connections with an existing board. Overrides all of the placement coordinates in an existing board.
 - AutoECO/Orerride All Creates a new board nets, but doesn't override board attributes. Doesn't delete components or nets.
- AutoECO options
- AutoEC/Override Adds and updates components and nets. Doesn't delete components or nets.



Flow Chart – 2.1

- Auto ECO/DXE Transfers obstacles and text from one board to another.
 - AutoECO/Net Attrs Transfers net attributes such as width, weight, spacing per layer, width per layer, and reconnection type from one board to another.
- When we create a new board file and load its netlist, AutoECO runs automatically.

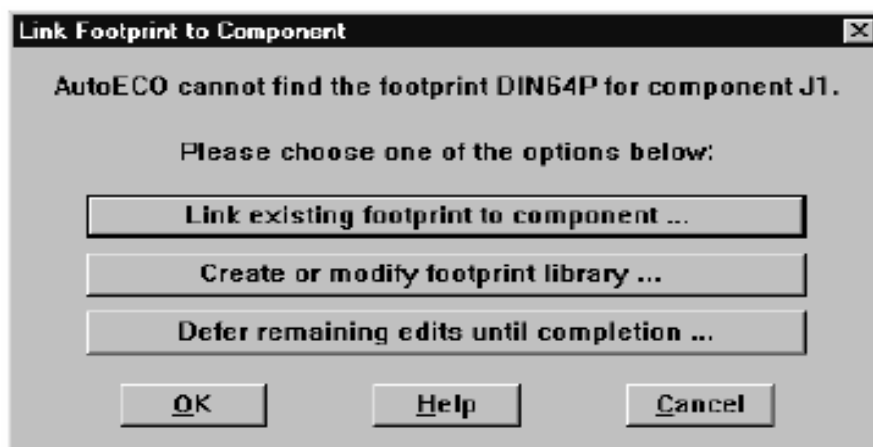


Figure 2.8: Link Footprint to Component

If AutoECO can't find a designated footprint, the Link Footprint to Component dialog box displays. AutoECO produces an ASCII report file. All errors are reported in this file. When we load an existing board files, AutoECO checks if the netlist has changed and asks we if you want to load the updated Netlist. We can also initiate AutoECO in Layout when we create a netlist in Capture. In the Layout tab in Capture's Create Netlist dialog box, select the Run ECO to Layout option before we create the netlist. If the board file is open in Layout when we create the netlist, Layout automatically asks if we want to load the new netlist and update the board file.

2.15. Autoplacement

The autoplacement utility that's available in Layout Plus can place components individually, in groups or clusters, or place the entire board.

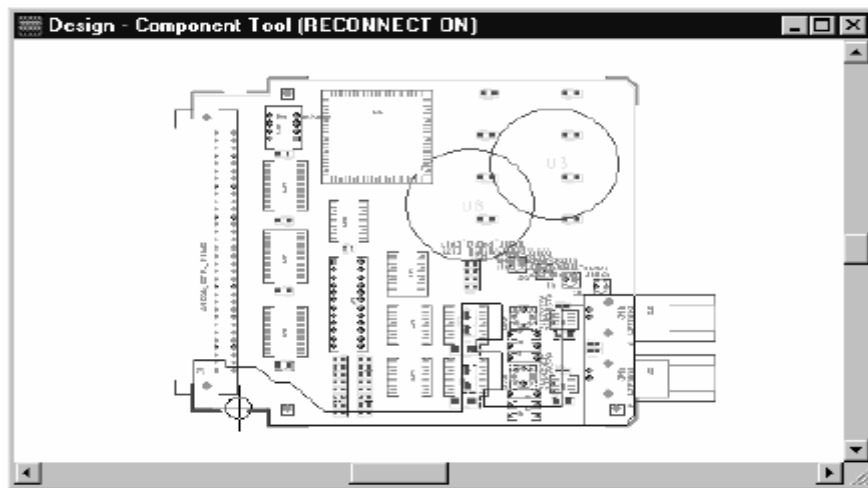


Figure2.9: Component placed in autoplacement

Layout Plus has four Autoplacement techniques as follow:

- **Instantaneous reconnect:** Displays the shortest connection possibilities in "real time" as you move components.
- **Shove component:** When we place a component, other components move out of the way to clear a spot for it.
- **Cluster placement:** Assigns each schematic group to a cluster and places the components in proximity to one another on the board.
- **Enhanced interactive placement:** Using proximity placement, swap component, and adjust component tools in combination with autoplacement.

Groups and clusters plays an important role in autoplacement. we can group components at the schematic level, or on the board using the Components spreadsheet. To create a component group, select the components we want to group. Press the right mouse button to display the pop-up menu, and then choose Properties. Specify a group number for the set of components we selected, then

choose the OK button. Close the Components spreadsheet. Since the autoplacer places components emphasizing circuit functionality, it's a good idea to disable for routing all nets attached to plane layers. Choose Selection any from the pop-up menu, specify the Group Number, then choose the OK button, from the Auto menu, choose Place, then Component(s). If we are placing new components onto an existing board, choose the component toolbar button and select a component for placement.

As we are positioning the component choose Shove from the pop-up menu. Layout moves the other components out of the way of the component being placed. A powerful feature of Layout Plus is cluster placement, which places many components at once. A cluster represent components in specify circuits. To place a cluster of components, choose the component toolbar button, then choose Select Any from the pop-up menu. In the Component Selection Criteria dialog box, type a group number of a group that was assigned in the schematic, then choose the OK button. Now move the cluster and place it on the board. If we need to move a component within the placed cluster, select the cluster and choose Break from the pop-up menu. Now we can move one of its components. To simplify the task of manually moving placed components, choose the reconnect toolbar button. This prevents the ratsnest from displaying until you move a component. The ratsnest connections are continually reconnected, so we can see the relationship between the component we are moving and the neighboring components.

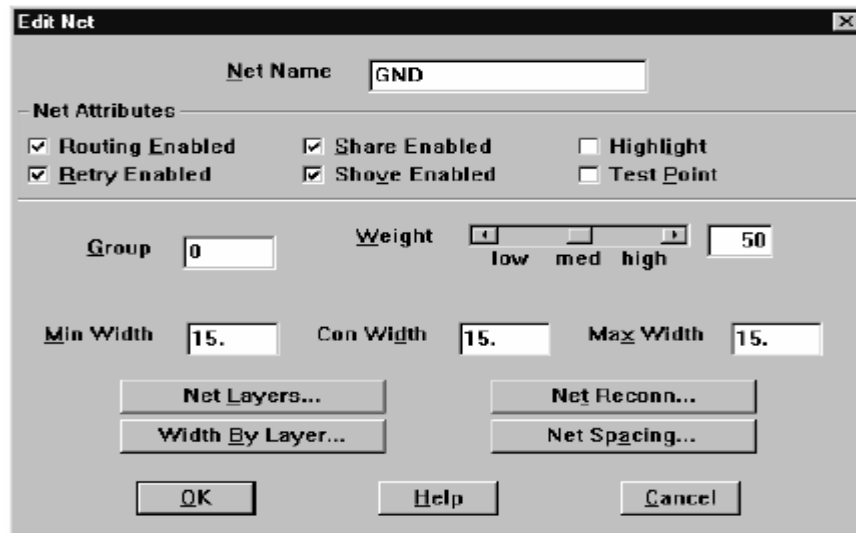


Figure 2.10: The Edit Net dialog box

A *netlist* file (.MNL) describes the interconnections of a schematic design using the names of the nets, components, and pins. A netlist contains the following:

- Footprint names
- Electrical packaging
- Component names Net names
- The component pin for each net
- Net, pin, and component property information

Chapter – 3

Description of Designing and Fabrication of AT89S52 & AT89S8252 Based Kit

In most application, a microcontroller can satisfy all the system requirements with no additional integrated circuits. Due to their low cost and a high degree of flexibility, microcontrollers are finding way into many applications that were previously accomplished by mechanical means or combinational logic. Our Kit is using Atmel AT89S52 or AT89S8252. The software for the microcontroller is written in Assembly language and using ASEM-51 assembler which creates two file one is '.hex' file another is '.lst' file or written in C language and using SDCC C- compiler, which is capable of creating a '.hex' file. The .hex file can be burnt into the microcontroller using AT89 ISP Programmer or kit.

Integrated Circuit AT89S52 & AT 89S8252 is a low-power, high-performance CMOS 8-bit microcontroller. It is manufactured using Atmel's high-density non-volatile memory technology and is compatible with the industry-standard 80C51 instruction set and pin-out. The powerful AT89S8252 microcontroller provides a highly flexible and cost-effective solution to many embedded control applications. Its main features are:

- Compatibility with MCS-51 products
- 8kB in-system reprogrammable downloadable Flash memory with SPI serial interface for program downloading and endurance of 1000 write/erase cycles
- 2kB EEPROM with endurance of 100,000 write/erase cycles
- 4V-6V operating range
- Fully static operation: 0 Hz to 24 MHz
- There-level program memory lock
- 256x8-bit internal RAM
- 32 programmable I/O lines
- There 16-bit timer/counters
- Nine interrupt sources
- Programmable UART serial channel
- SPI serial interface
- Low-power idle and power-down modes
- Interrupt recovery from power-down
- Programmable watchdog timer
- Dual data pointer
- Power-off flag

3.1. Circuit Description:

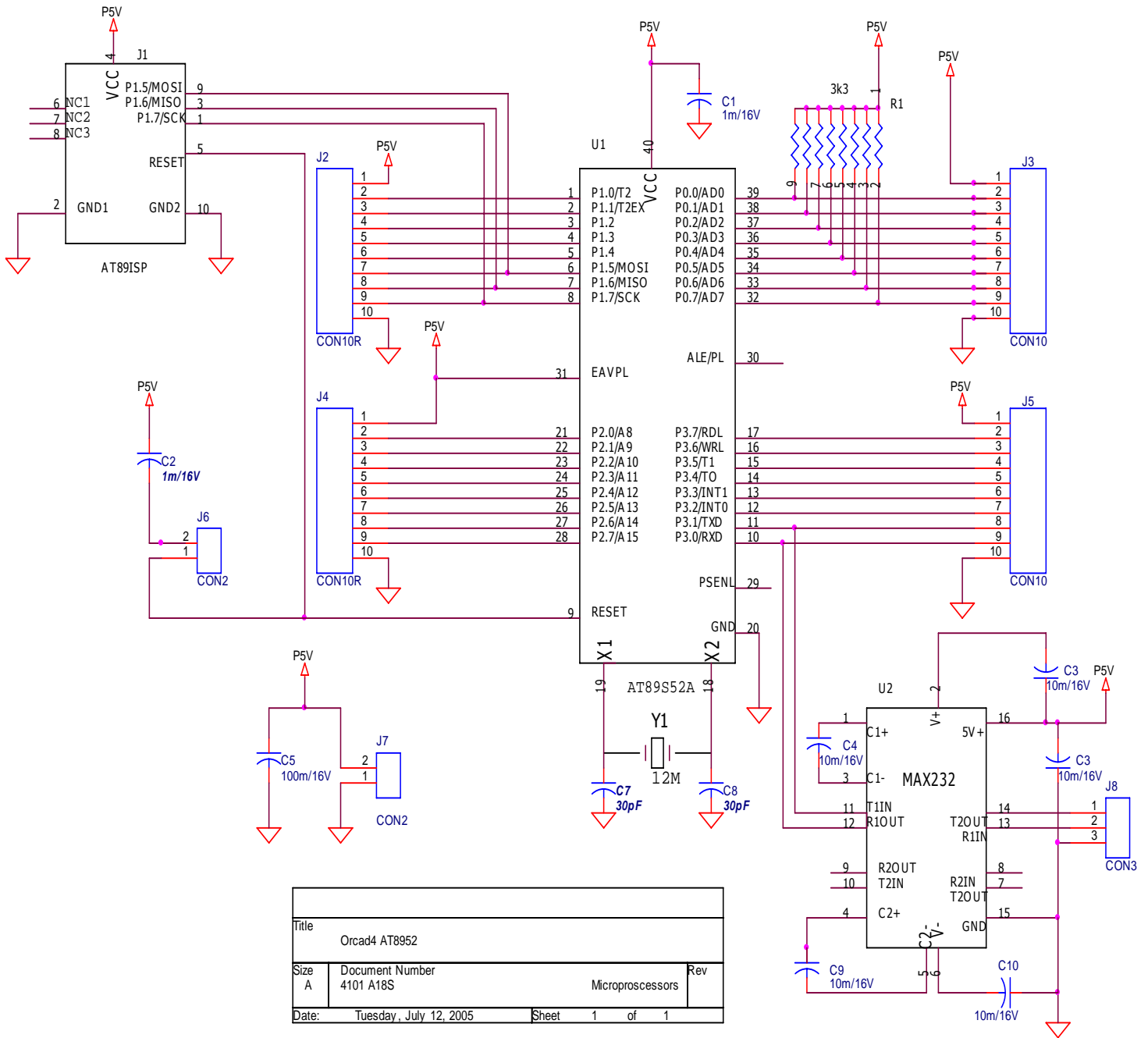


Figure 3.1 - Circuit Diagram of AT89S52 & AT89S8252 Based Micro-Controller

Figure 1 shows the Circuit Description of AT89S52 or AT89S8252 Based Micro-Controller kit, in this diagram its main component is micro controller chip it has four bidirectional 8-bit ports. Port 0 is an 8-bit open-drain bidirectional I/O port. As an output port, each pin

PARTS LIST

IC Chip

U1	—	40 Pin AT89S8252 or AT89S52 Micro controller Chip
U2	—	16 Pin MAX 232 chip.

Connector

J1	—	10 Pin connector for interfacing with AT891 SPI to PC.
J2, J3,		
J4, J5	—	10 Pin Connector, which is connected with Ports of μ C.
J6	—	Two Pin Connector, which is connected to the RESET Pin of μ C.
J7	—	Two Pin Connector, which is used for 5V DC Power Supply to the kit.
J8	—	Three Pin Connector, which is used for RXD/TXD to the μ C.

Capacitor:

C1, C2	—	1 μ F, 63V
C3, C4, C6, C9.		
C10	—	10 μ F, 63V electrolytic
C5	—	100 μ F, 63V electrolytic
C7, C8	—	30k pF, Ceramic

Resistance

R1	—	3k3
----	---	-----

Crystal

Y1	—	12MHz.
----	---	--------

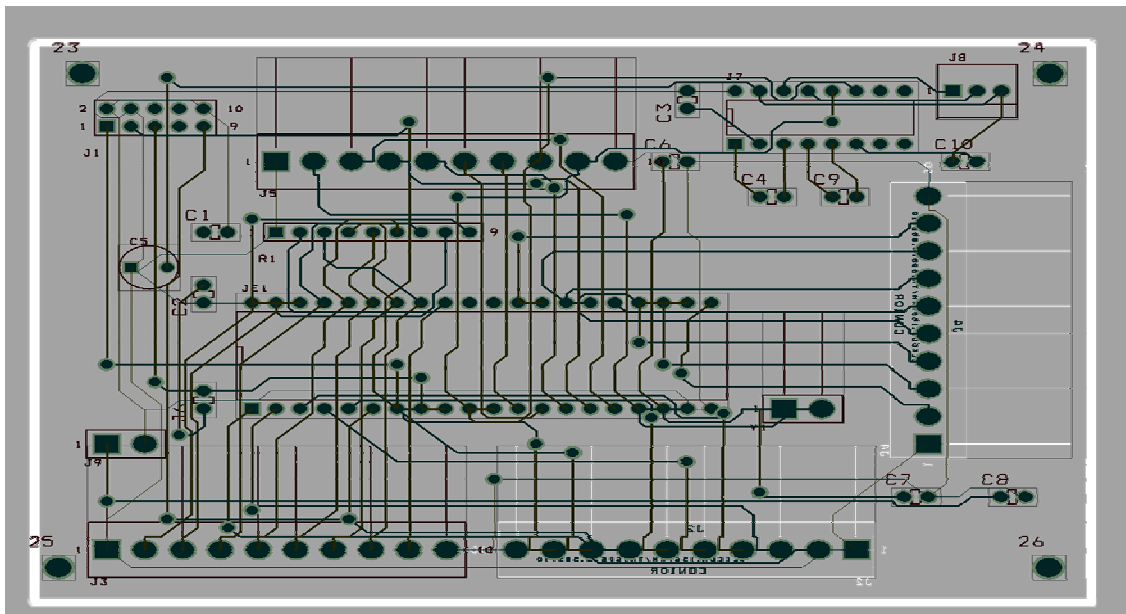
can sink eight TTL inputs. Ports 0 can also be configured as the multiplexed low-order address/data bus during accesses to the external program and data memory. External pull-ups are required during data outputs. Port 0 is used to drive the segments of all the 7-segment common-cathode displays LED. These are connected to the 10 pin connector in such a manner P0 — J3, P1 — J2, P2 — J5, & P3 — J4 respectively, here resistance R1 is also connected between port0 and connector J3, this port is used for low order data bus read/write. Next another parts 10 Pin connector, which is interface the kit with PC though AT89 ISP programmer it mainly used for burnt and erase the programs in the memory and some other function is also worked like checking memory space, read the programs in memory, run the programs etc. in IC. Its three pin1, 3 & 9 is connected to the

microcontroller in such a manner 1 — P1.7/SCK, 3 — P1.6/MISO, 9 — P1.5 /MOSI respectively.

In this diagram, a two Pin Connector (J7) is use for 4-6V DC power supply to the kit, and another two pin Connector J6 is besides the J7 pin in figure1 connected to the RESET pin of the micro-controller for manually reset the micro-controller. In this circuit diagram another next component Y1 is a 12MHz KDS crystal is used to generate the clock frequency for the micro-controller, which is connected to the Pin X1 and X2 to the micro-controller. On this board another main important IC is MAX 232, its mainly used line drivers/receivers for communication interface with three Pin Connector J8, MAX 232 chip has 16 Pin which connected to such a manner like its pin 11, 12 is connected to the port 3 of micro-controller P3.1/TXD and P3.0/RXD respectively. It's another pin 13(R1 IN) and 14(T2 OUT) is connected to the three Pin connector J8. As for example a micro-controller-to-PC communication, or another MCU's communication etc.

An actual-size of the PCB Board for AT89S8252 or AT89S52 based kit is shown in figure 2 and its components layout in figure 3.

Figure 3.2. Combined actual-size the PCB for AT89S8252 or AT89S52 based kit.



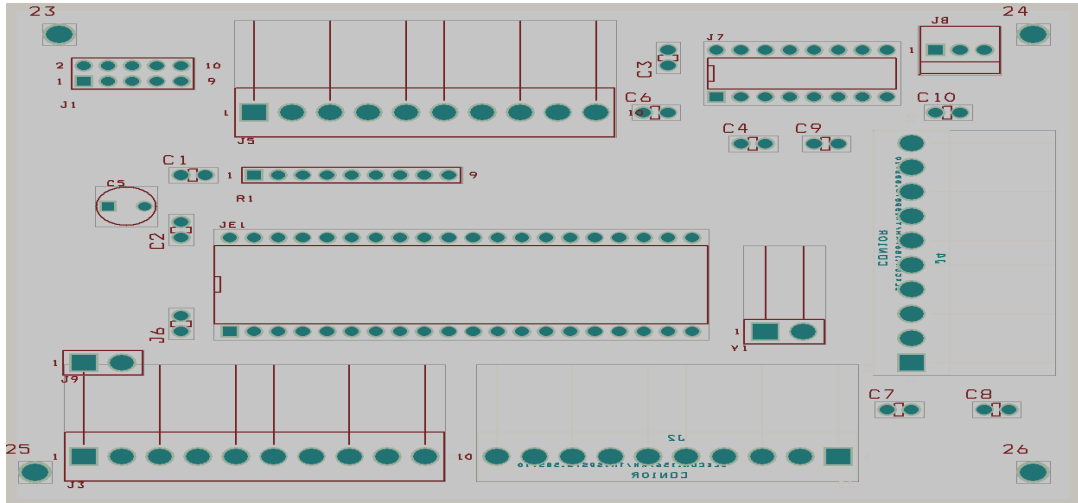


Figure 3.3. Components layout for the PCB

3.2. MAX 232:

Serial Communication simply means communicating by transmitting data bits serially may be by using the serial ports. This means that instead of sending the whole data chunk simultaneously through parallel transfer, the data bits are transmitted sequentially one by one. This mode of communication can be used to send a signal from a micro controlled circuit to a PC through its serial port using RS. 232. Although the serial mode of communication is the slower mode, it can be used for comparatively longer distances parallel data transfers are effectively restricted to one meter at most and its implementation requires less H/W and thus is much simpler. As a result, serial communication is quite popular for devices such as mouse, modem and various other small devices generally micro-controlled type of circuits.

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where $\pm 12V$ is not available. These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces Power dissipation to less than $5\mu W$.

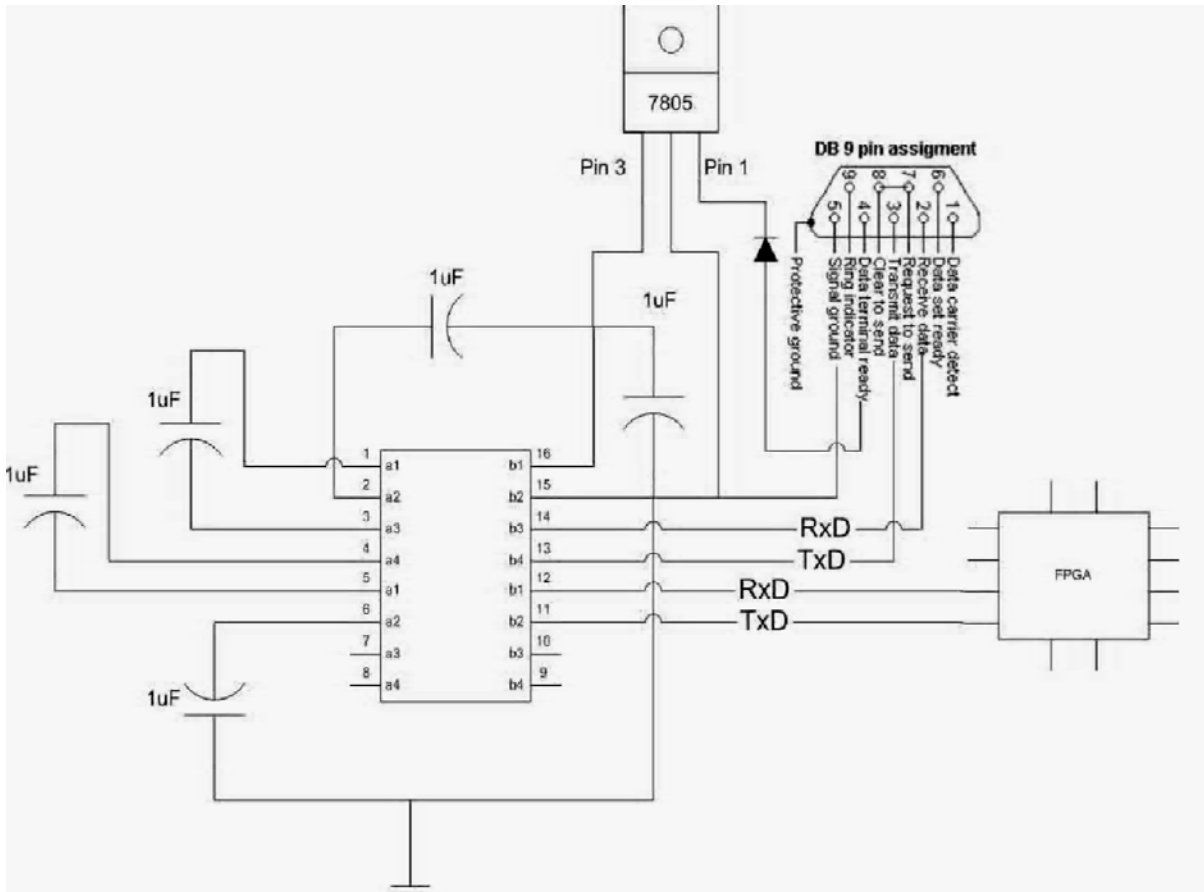


Figure – 3.5 : Interfacing of PC to MAX-232 pin out on a DB-9 pin used for Asynchronous Data.

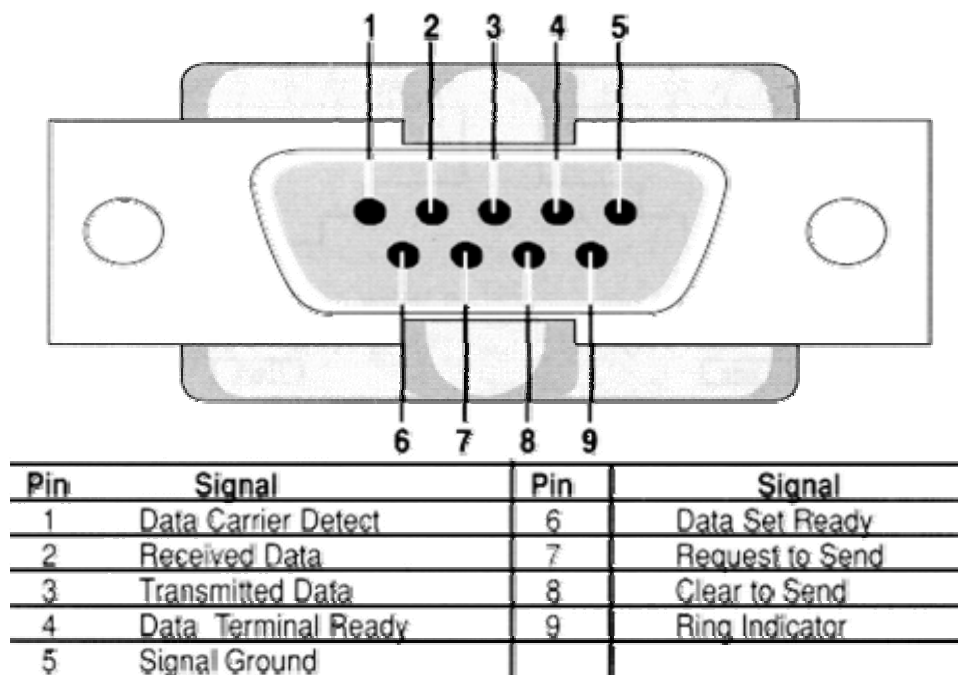


Figure 3.6: Configuration of DB-9 pin

3.4. Testing Procedure:

- After assembling the PCB, we checked the circuit connections before switching on the power supply.
- We had written a program in assembly language for Blinking the LED, as below

```
.org 0h
start: clr a
      mov p0,a
      acall delay
      dec a
      mov p0,a
      acall delay
      ajmp start
delay: mov r0,#01h
in1:   mov r1,#00h
in2:   mov r2,#00h
wait:  djnz r2,wait
      djnz r1,in2
      djnz r0,in1
      ret
      .end
```

Another Blinking Program

Program starts here.

```
$NOMOD51
```

```
$INCLUDE (89s52.MCU)
```

```
TIME EQU 1
P27IN EQU P2.7
DATAIN1 EQU 01H
DATAIN2 EQU 04H
```

```
ORG 0000H ; Reset address
```

```
NOP
```

```
AJMP 0030H ; Reset jumps to 0030H
```

```
ORG 0030H ; The program starts.
```

Initialisation of SFR etc..

```
MOV SP,#060H ;Set stack to 060H of internal RAM
```

```
MOV R0, #0H
```

```
BLINK: MOV C, P27IN ;This is IF ELSE for bit variable. Pin P2.7 is copied in Carry flag
```

```
JC FASTCOUNT ;The problem is to get DATAIN1 or DATAIN2 in "A" depending on the
```

```
MOV A,#DATAIN2 ;state of P2.7. If it is set, DATAIN1 goes to "A", else DATAIN2
```

```
FASTCOUNT: JNC FASTCOUNTEND ;goes to "A". See the positions of MOV statements, JC and JNC.
```

```
MOV A,#DATAIN1 ;In short: IF CY is SET DATAIN1 ELSE DATAIN2 goes to "A"
```

```
FASTCOUNTEND: INC R0 ;
```

```
MOV P0,R0
```

```
MOV R3,A ;wait for TIME * 125 ms (12 MHz)
```

```
MOV R2,#0
```

```
MOV R1,#0
```

```
LOOP: DJNZ R1,LOOP
```

```
DJNZ R2,LOOP
```

```
DJNZ R3,LOOP
SJMP BLINK ;This is repeated forever,
END
```

We saved this program with '.A51' in notepad editor, and using ASEM 51 Assembler which compile the program and create the .hex file. This .hex file using in AT89 ISP programmer and burnt the IC, and then after run the program with help of AT89 ISP programmer.

3.5. Description of different Software:

3.5.1. Atmel Microcontroller ISP Software

The Atmel Microcontroller ISP Software is performing in-system programming (ISP) for the Atmel devices. It interfaces for in-system programming that can be run from our personal computer. The Atmel ISP Software has a comprehensive set of features that allows we view, program, and erase data from an Atmel ISP device. The Atmel Microcontroller ISP Software also allows to load hex files containing the Code to add to the device. Using the software, we can manipulate this code, verify it against the existing code on the device, and program the code onto the device. Additionally, using the software, we can read any existing code from the device and make minor changes to the code. It is possible to update the device with our changes or save them to a hex file for use with other devices. The software also allows protecting third parties from accidentally reprogramming; the device allows we lock the device so that the code cannot be read from it. Atmel's Microcontroller ISP Software contains a variety of tools customized for Atmel ISP devices.

When we successfully install the software then we open as follow

Programs → Atmel → Microcontroller ISP Software

To select a device, as following:

1. The board was connected to the LPT port to computer the Atmel ISP cable and that the board is turned on.
2. Select the LPT port that the board is connected to by selecting **Select Port** from the **Options** menu.
3. then select a device by choosing **Select Device** from the **Options** menu.
4. Select the device that we were to program from the desired device.
5. Next select whether we want to read and write from the device in page mode or in byte mode.
6. Enter the external clock frequency (MHz) provided to the device in the text box.
7. Select **OK**.

▪ **Checking the state of the device**

We should do before attempting to program a device is to check its state. To check a device's state select **Blank Check** from the **Instructions** menu. After selecting **Blank Check**, the software will report. If the device is locked and we wish to program it we may choose **Erase Chip** from the **Options** menu.

- **Loading a buffer from a file**

Now load the program that we wish to encode onto the chip into the software. The Atmel Microcontroller ISP Programming software can load in programs that are in the Intel Hex file format.

To load in a program from disk:

1. Select **Load Buffer** from the **File** menu we will see a dialog.
2. Select the HEX file that contains the program we wish to encode onto the chip and click **Open**.
3. The buffers will then be updated with the contents of the hex file.

- **Programming the Device**

Once the program has been loaded into the buffer to program the device is to select **Auto Program** from the **Instructions** menu.

3.5.2. Description of ASEM – 51

ASEM-51 Version 1.3 (Final Release) was written by Bayreuth, in December, 2002. ASEM-51 is a two-pass macro assembler for the Intel MCS-51 family of microcontrollers. The ASEM-51 assembly language is a rich subset of the Intel standard that guarantees maximum compatibility with existing 8051 assembler sources. ASEM-51 can generate two sorts of object files: Intel-HEX format, which is required for many simulators, emulators and target debuggers. Thus ASEM-51 is suitable for small and medium MCS-51-based microcontroller.

- **Installation of ASEM-51**

An installation of ASEM-51 under MS-DOS is very simple:

- Create a new, empty scratch directory on our hard disk.
- Unpack our ASEM-51 distribution archive into this directory, or copy all files of the ASEM-51 package into it.

- **DOS Command Line Operation**

ASEM-51 provides full support of command line operation and batch capability as the best commercial development tools. The assembler is invoked by typing:

ASEM <source> [<object> [<listing>]] [<options>]

Where <source> is the 8051 assembler source, <object> is the output file, and <listing> is the assembler list file. The parameters <object> and <listing> are optional. When omitted, the file names are derived from the <source> file name, but with extensions HEX (or OMF) and LST. All file names may be specified without extensions. In these cases, the assembler adds default extensions as shown below:

File	extension
<source>	.A51
<object>	.HEX (with /OMF-51 option: .OMF)

<listing> .LST

If we want a file name to have no extension, terminate it with a '!'.

When program errors are detected, they are flagged on the console. This may look as follows:

MCS-51 Family Macro Assembler ASEM-51 V1.3

APPLICAT.A51 (14): must be known on first pass.

USERBITS.INC (6): attempt to divide by zero.

DEFINES.INC (37): symbol not defined.

APPLICAT.A51 (20): symbol not defined.

APPLICAT.A51 (27): no END statement found.

5 errors detected.

The /QUIET option suppresses all console output except error messages. When terminating, ASEM-51 returns an exit code to the operating system:

Situation	ERRORLEVEL
no errors	0
program errors detected	1
fatal runtime error	2

Note: Warnings do not influence the exit code!

*****We have use the .hex file in load in Atmel ISP Micro-controller ISP software.**

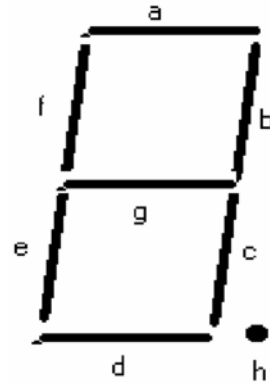
3.6. APPLICATION



3.6.1. Seven Segment LED Display

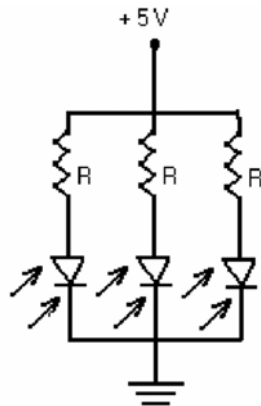
Display

- Used to display decimal digits from 0 to 9
- Used for displaying counter and timer outputs
- They are basically a collection of seven
- LED segments arranged to form the digit 8 (plus a LED for decimal point)

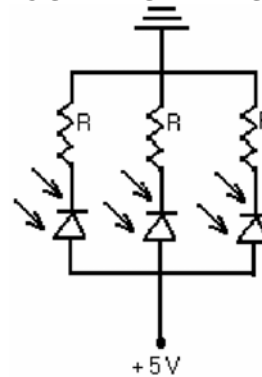


3.6.2. There are two type of LED

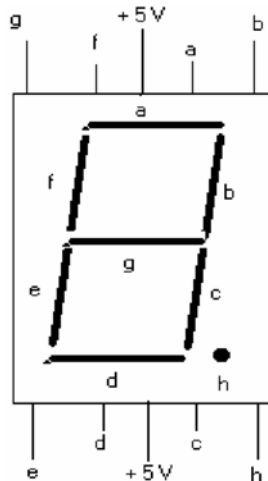
1. Common Cathode



2. Common Anode



The resistor values are commonly 330 or 440 ohms.



Pin Out of common anode seven segment display

3.6.3. Driving it with a micro-controller

- Connect a,b,c...h to a port or the micro-controller .
As for example P0.0 to a and P0.1 to b and so on
- To turn one segment on we make the corresponding pin high/low depending on whether it is common cathode or common anode.

	DECIMAL NO				A	B	C	D	E	F	G	H	PORT
0	0	0	0	0	0	0	0	0	0	0	1	1	0x03
1	0	0	0	1	1	0	0	1	1	1	1	1	0x9F
2	0	0	1	0	0	0	1	0	0	1	0	1	0x25
3	0	0	1	1	0	0	0	0	1	0	0	1	0x0D
4	0	1	0	0	1	0	0	1	1	0	0	1	0x99
5	0	1	0	1	0	1	0	0	1	0	0	1	0x49
6	0	1	1	0	0	1	0	0	0	0	0	1	0x41
7	0	1	1	1	0	0	0	1	1	1	1	1	1x1F
8	1	0	0	0	0	0	0	0	0	0	0	1	0x01
9	1	0	0	1	0	0	0	0	1	0	0	1	0x09

REFERENCES

1. **The 8051 Micro-Controller** *Second Edition* By Kenneth J: Ayala
2. **The 8051 Micro-Controller and Embedded System** by Muhammad Ali Mazidi & Janice Gillispie Mazidi
3. **The Intel Microprocessors** *Sixth Edition* By BARRY B. BREY
4. **Embeded System Design** A Unified Hardware/Software Introduction By Frank Vahid/Tony Givargis.
5. **A Embeded System Software Primer** by David E. Simon.
6. **User Manual of Orcad 9.2 Family**
7. Peatman, J. B.: "The Design of Digital Systems," chap. 3, McGraw-Hill Book Company, New York, 1971.
8. Millman, J., and H. Taub: "Pulse, Digital, and switching Waveforms," chap. 10, McGraw-Hill Book Company, New York, 1965.
9. Hoeschele, D. F. Jr. "Analog-to-digital and Digital-to-analog Conversion Techniques," John Wiley & Sons, Inc., New York, 1968.
10. **Computer as Components** Principals of Embedded Computing system Design by Wayne Wolf.
11. <http://www.atmel.com>
12. <http://www.orcad.com>
13. <http://www.signum.com/signum.html>
14. <http://plit.de/asm-51/>
15. <http://sdcc.sourceforge.net>
16. www.pirc.com/tech/8051/
17. www.raisonance.com

18. www.doc.ic.ac.uk
19. <http://www.semiconductors.philips.com>
20. <http://www.altavasta.com>
21. <http://www.digcheep.com>
22. <http://www.intersil.com/tsc>